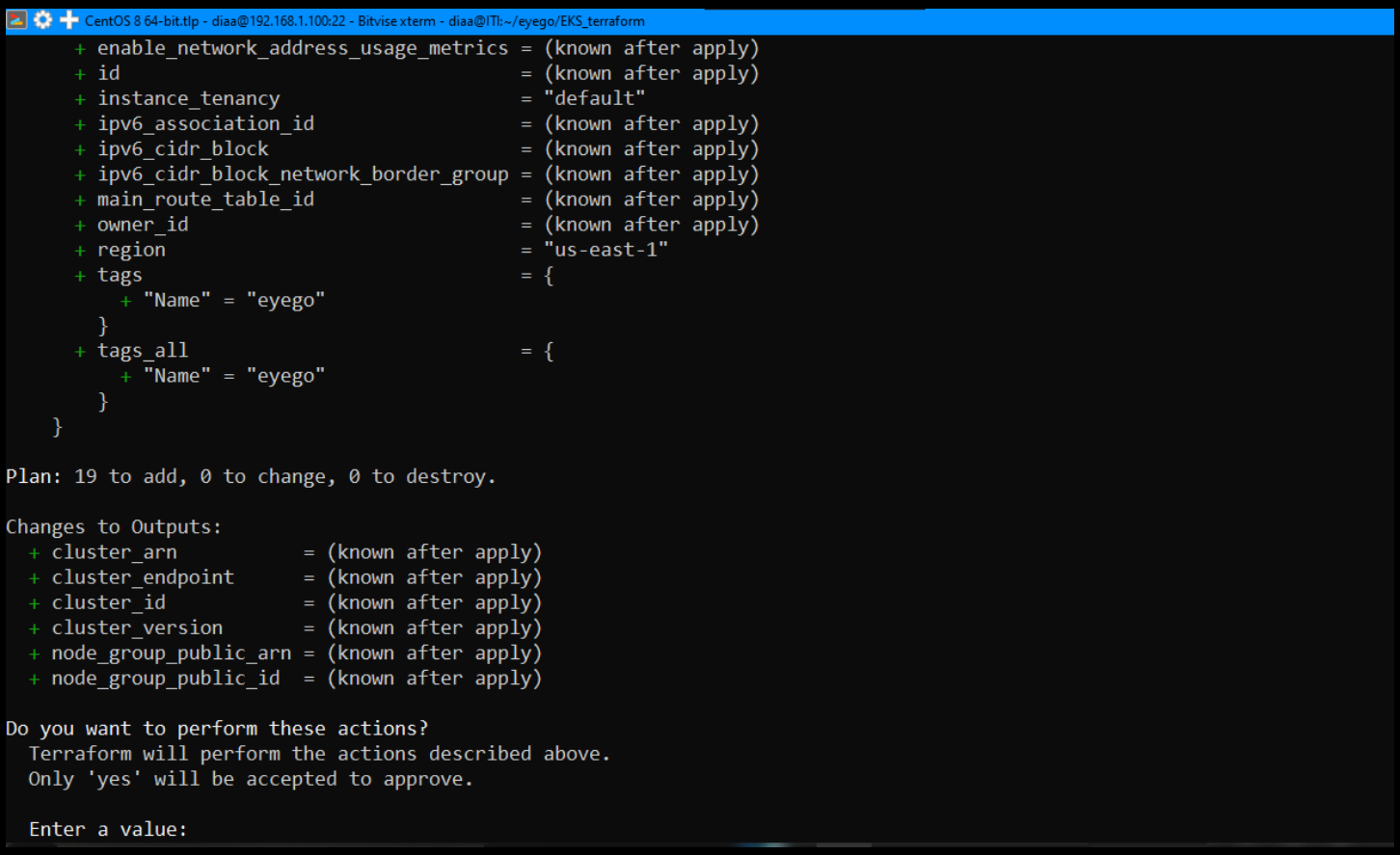
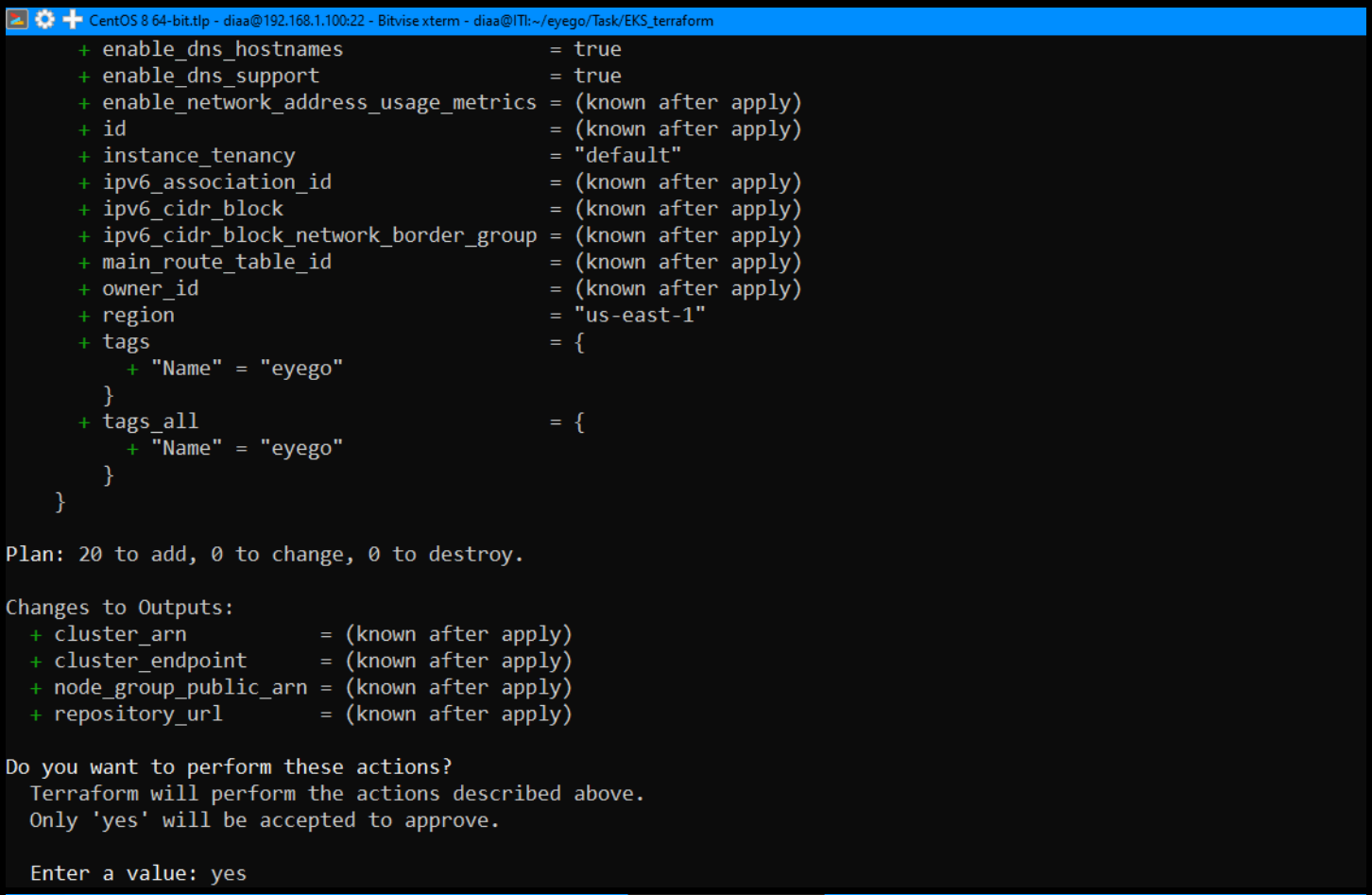
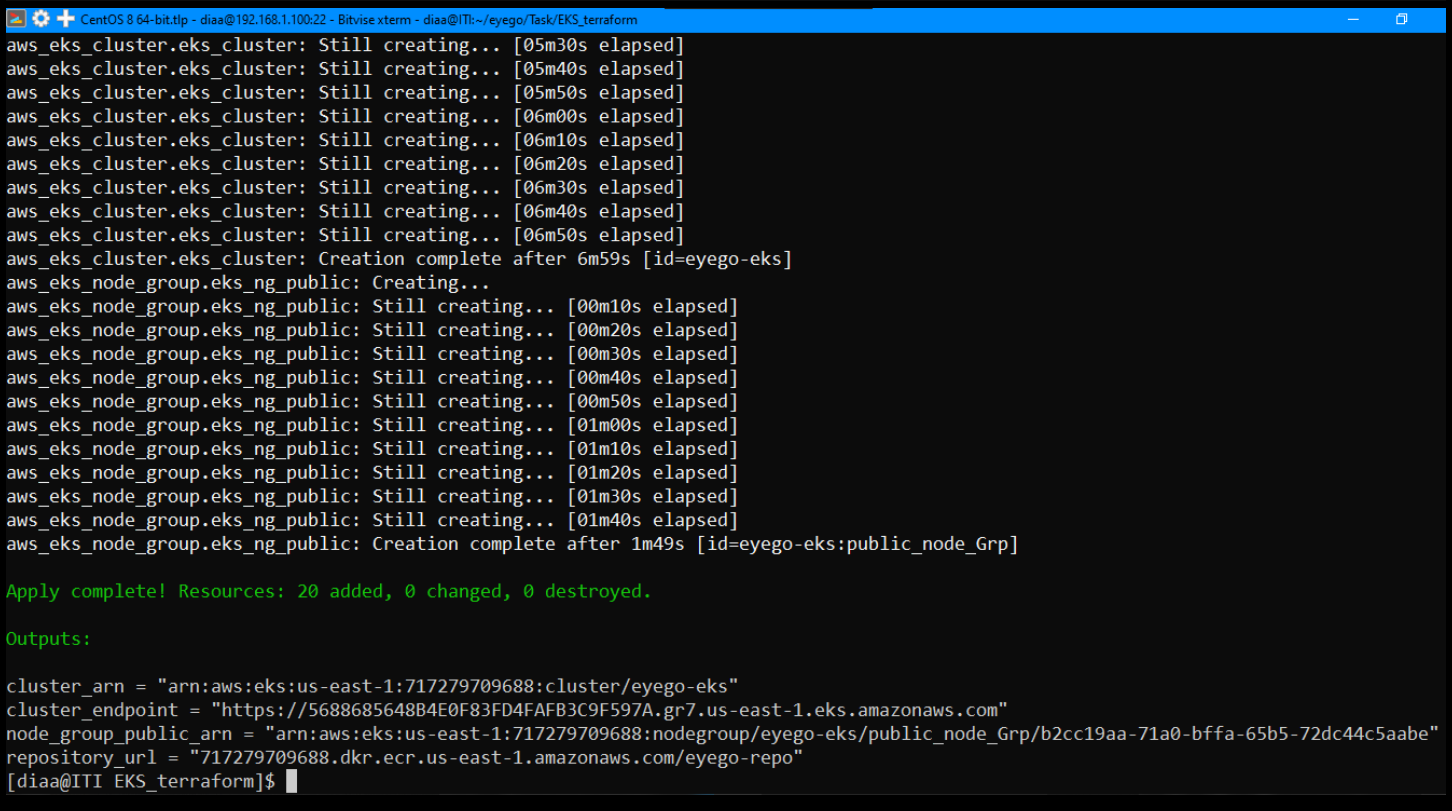
Eyego-Task

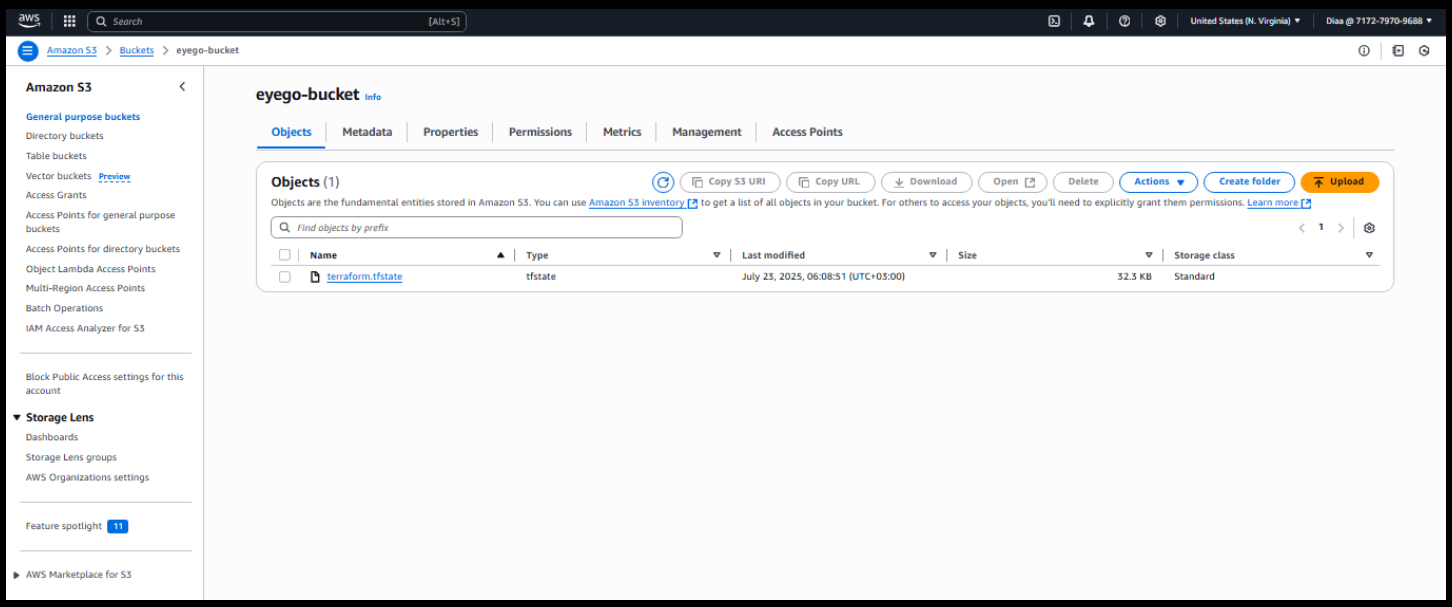
Diaa Qassem

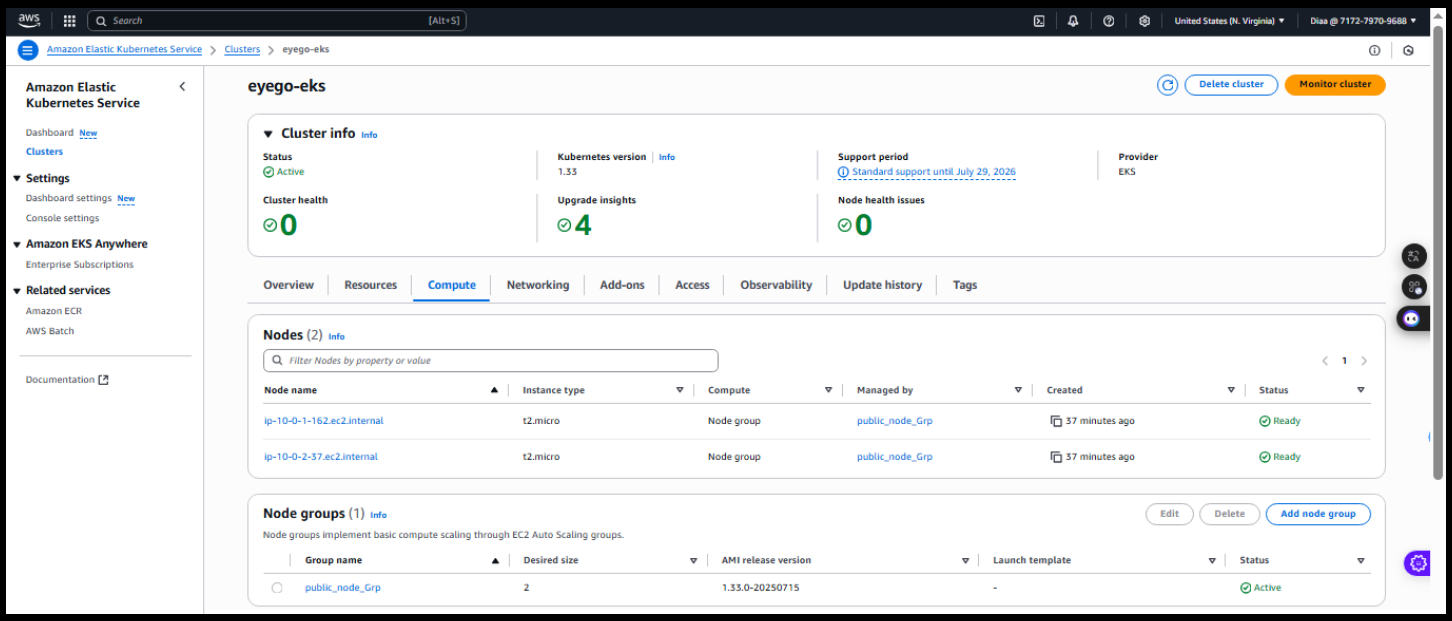
7/23/2025


CentOS 8 64-bit.tlp - diaa@192.168.1.100:22 - Bitvise xterm - diaa@ITI:~/eyego/EKS_terraform
+ enable_network_address_usage_metrics = (known after apply)
+ id = (known after apply)
+ instance_tenancy = "default"
+ ipv6_association_id = (known after apply)
+ ipv6_cidr_block = (known after apply)
+ ipv6_cidr_block_network_border_group = (known after apply)
+ main_route_table_id = (known after apply)
+ owner_id = (known after apply)
+ region = "us-east-1"
+ tags = {
+ "Name" = "eyego"
}
+ tags_all = {
+ "Name" = "eyego"
}
}
Plan: 19 to add, 0 to change, 0 to destroy.
Changes to Outputs:
+ cluster_arn = (known after apply)
+ cluster_endpoint = (known after apply)
+ cluster_id = (known after apply)
+ cluster_version = (known after apply)
+ node_group_public_arn = (known after apply)
+ node_group_public_id = (known after apply)
Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.
Enter a value:


CentOS 8 64-bit.tlp - diaa@192.168.1.100:22 - Bitvise xterm - diaa@ITI:~/eyego/Task/EKS_terraform
+ enable_dns_hostnames = true
+ enable_dns_support = true
+ enable_network_address_usage_metrics = (known after apply)
+ id = (known after apply)
+ instance_tenancy = "default"
+ ipv6_association_id = (known after apply)
+ ipv6_cidr_block = (known after apply)
+ ipv6_cidr_block_network_border_group = (known after apply)
+ main_route_table_id = (known after apply)
+ owner_id = (known after apply)
+ region = "us-east-1"
+ tags = {
+ "Name" = "eyego"
}
+ tags_all = {
+ "Name" = "eyego"
}
}
Plan: 20 to add, 0 to change, 0 to destroy.
Changes to Outputs:
+ cluster_arn = (known after apply)
+ cluster_endpoint = (known after apply)
+ node_group_public_arn = (known after apply)
+ repository_url = (known after apply)
Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.
Enter a value: yes


CentOS 8 64-bit.tlp - diaa@192.168.1.100:22 - Bitvise xterm - diaa@ITI:~/eyego/Task/EKS_terraform
aws_eks_cluster.eks_cluster: Still creating... [05m30s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [05m40s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [05m50s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [06m00s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [06m10s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [06m20s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [06m30s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [06m40s elapsed]
aws_eks_cluster.eks_cluster: Still creating... [06m50s elapsed]
aws_eks_cluster.eks_cluster: Creation complete after 6m59s [id=eyego-eks]
aws_eks_node_group.eks_ng_public: Creating...
aws_eks_node_group.eks_ng_public: Still creating... [00m10s elapsed]
aws_eks_node_group.eks_ng_public: Still creating... [00m20s elapsed]
aws_eks_node_group.eks_ng_public: Still creating... [00m30s elapsed]
aws_eks_node_group.eks_ng_public: Still creating... [00m40s elapsed]
aws_eks_node_group.eks_ng_public: Still creating... [00m50s elapsed]
aws_eks_node_group.eks_ng_public: Still creating... [01m00s elapsed]
aws_eks_node_group.eks_ng_public: Still creating... [01m10s elapsed]
aws_eks_node_group.eks_ng_public: Still creating... [01m20s elapsed]
aws_eks_node_group.eks_ng_public: Still creating... [01m30s elapsed]
aws_eks_node_group.eks_ng_public: Still creating... [01m40s elapsed]
aws_eks_node_group.eks_ng_public: Creation complete after 1m49s [id=eyego-eks:public_node_Grp]
Apply complete! Resources: 20 added, 0 changed, 0 destroyed.
Outputs:
cluster_arn = "arn:aws:eks:us-east-1:717279709688:cluster/eyego-eks"
cluster_endpoint = "https://5688685648B4E0F83FD4FAFB3C9F597A.gr7.us-east-1.eks.amazonaws.com"
node_group_public_arn = "arn:aws:eks:us-east-1:717279709688:nodegroup/eyego-eks/public_node_Grp/b2cc19aa-71a0-bffa-65b5-72dc44c5aabe"
repository_url = "717279709688.dkr.ecr.us-east-1.amazonaws.com/eyego-repo"
[diaa@ITI EKS_terraform]$





Install kubectl

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

another way to setup kubectl

curl -LO "https://dl.k8s.io/release/$(curl -Ls https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"

chmod +x kubectl

sudo mv kubectl /usr/local/bin/

aws eks describe-cluster --name eyego-eks --region us-east-1 --query "cluster.status"

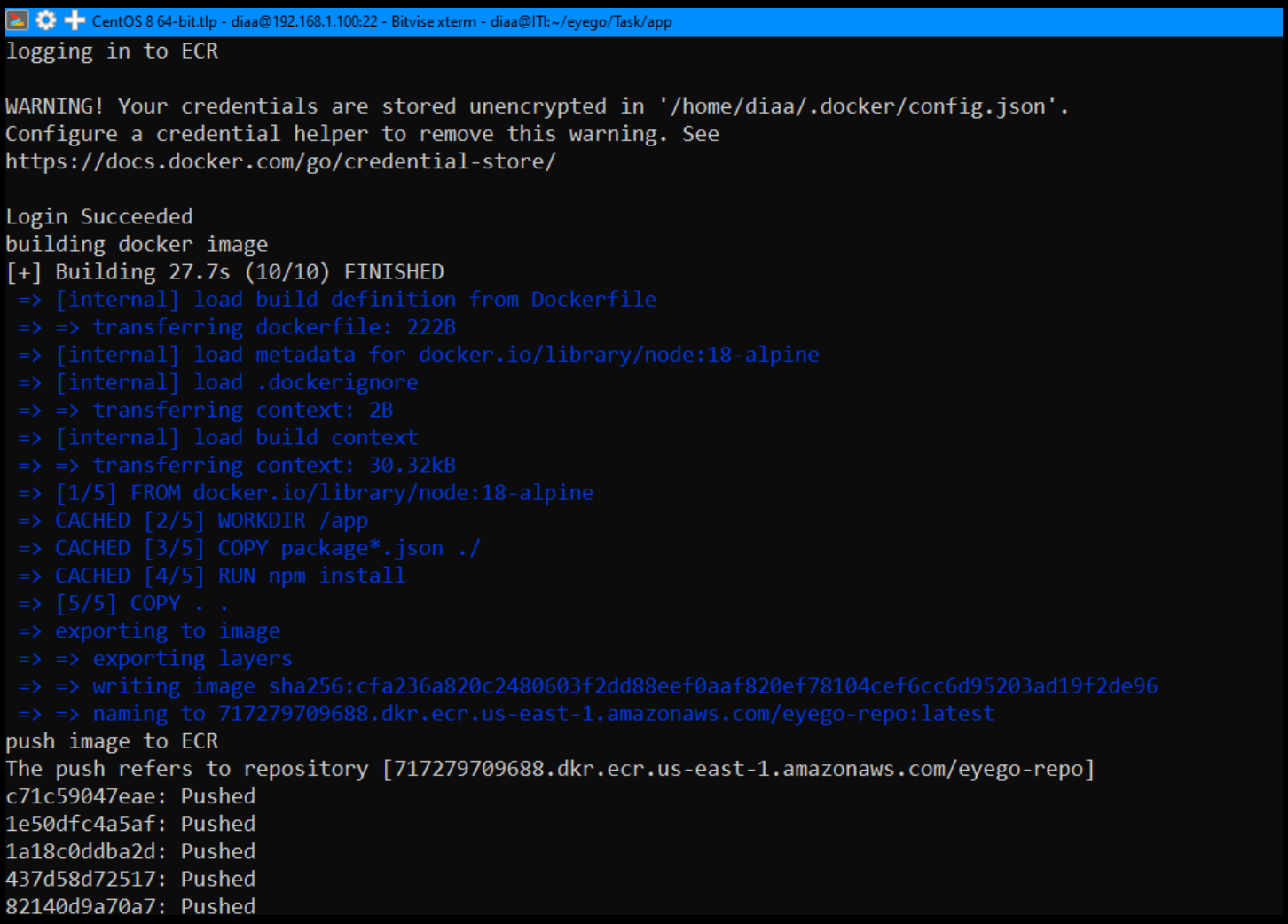
aws eks --region us-east-1 update-kubeconfig --name eyego-eks

```
[diaa@ITI EKS_terraform]$ vi ../app/push-image.sh
[diaa@ITI EKS_terraform]$ aws eks describe-cluster --name eyego-eks --region us-east-1 --query "cluster.status"
"ACTIVE"
[diaa@ITI EKS_terraform]$ aws eks --region us-east-1 update-kubeconfig --name eyego-eks
Added new context arn:aws:eks:us-east-1:717279709688:cluster/eyego-eks to /home/diaa/.kube/config
[diaa@ITI EKS_terraform]$ kubectl get nodes
NAME                         STATUS   ROLES    AGE   VERSION
ip-10-0-1-162.ec2.internal   Ready    <none>   23m   v1.33.0-eks-802817d
ip-10-0-2-37.ec2.internal    Ready    <none>   22m   v1.33.0-eks-802817d
[diaa@ITI EKS_terraform]$
```

Push image to ecr

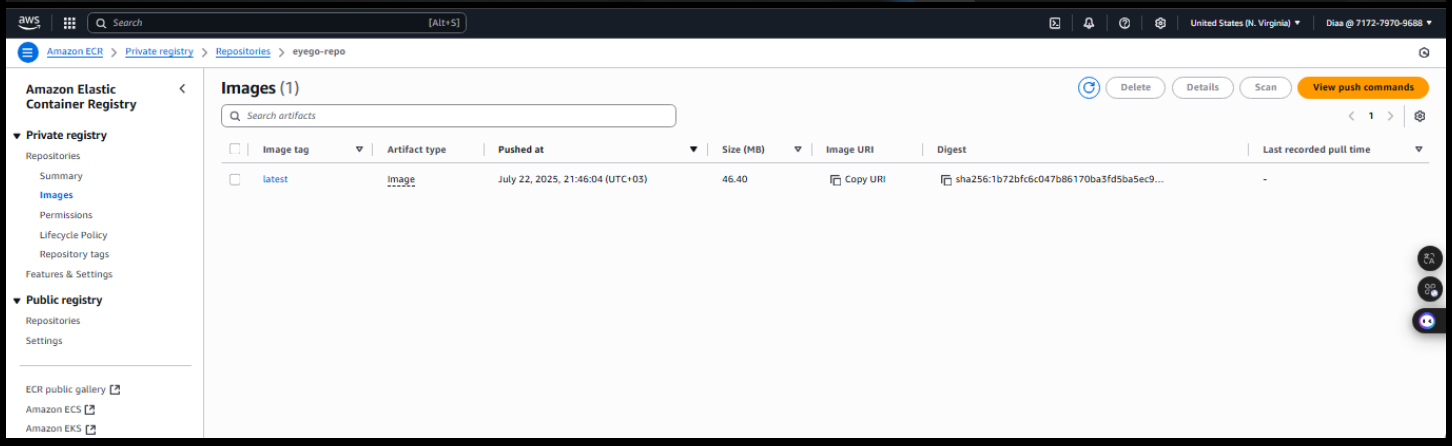
CentOS 8 64-bit.tlp - diaa@192.168.1.100:22 - Bitvise xterm - diaa@ITI:~/eyego/Task/EKS_terraform

```
#!/bin/bash
REPO_URL="717279709688.dkr.ecr.us-east-1.amazonaws.com/eyego-repo"
IMAGE_NAME="eyego-app"

echo "logging in to ECR"
aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin $REPO_URL

echo "building docker image"
docker build -t $REPO_URL:latest .

echo "push image to ECR"
docker push $REPO_URL:latest
~
```


CentOS 8 64-bit.tlp - diaa@192.168.1.100:22 - Bitvise xterm - diaa@ITI:~/eyego/Task/app
logging in to ECR
WARNING! Your credentials are stored unencrypted in '/home/diaa/.docker/config.json'.
Configure a credential helper to remove this warning. See
https://docs.docker.com/go/credential-store/
Login Succeeded
building docker image
[+] Building 27.7s (10/10) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 222B
=> [internal] load metadata for docker.io/library/node:18-alpine
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load build context
=> => transferring context: 30.32kB
=> [1/5] FROM docker.io/library/node:18-alpine
=> CACHED [2/5] WORKDIR /app
=> CACHED [3/5] COPY package*.json ./
=> CACHED [4/5] RUN npm install
=> [5/5] COPY . .
=> exporting to image
=> => exporting layers
=> => writing image sha256:cfa236a820c2480603f2dd88eef0aaf820ef78104cef6cc6d95203ad19f2de96
=> => naming to 717279709688.dkr.ecr.us-east-1.amazonaws.com/eyego-repo:latest
push image to ECR
The push refers to repository [717279709688.dkr.ecr.us-east-1.amazonaws.com/eyego-repo]
c71c59047eae: Pushed
1e50dfc4a5af: Pushed
1a18c0ddba2d: Pushed
437d58d72517: Pushed
82140d9a70a7: Pushed

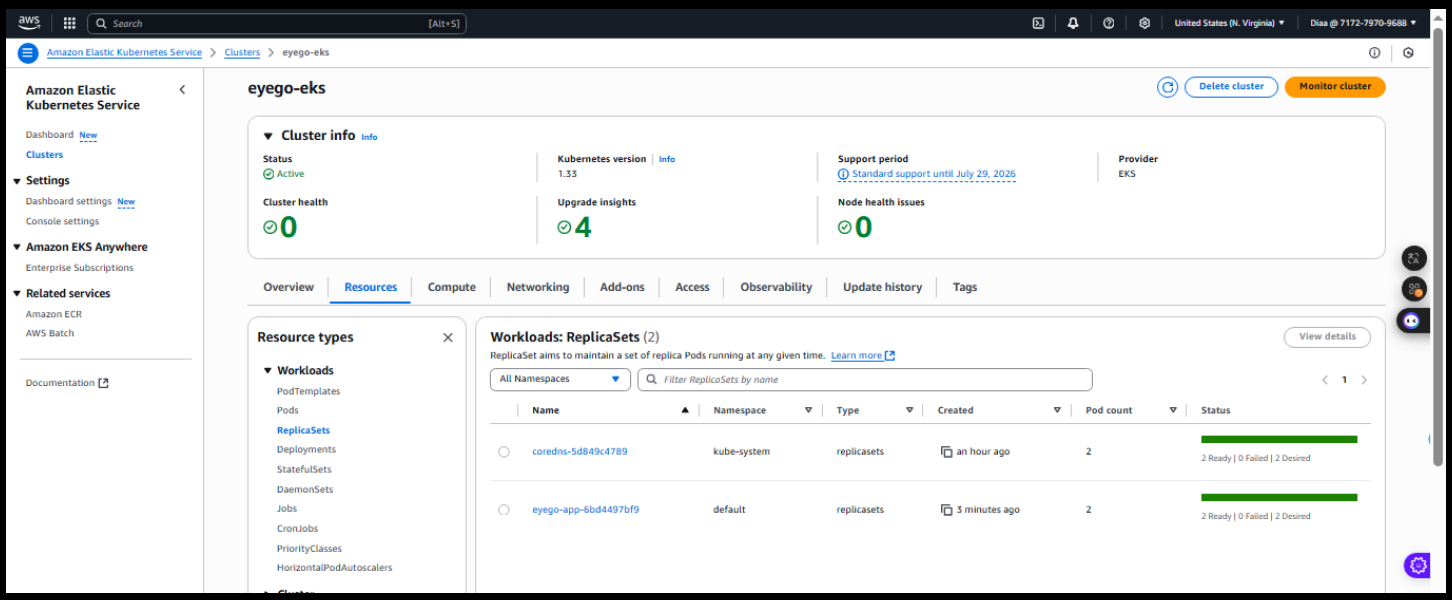
CentOS 8 64-bit.tlp - diaa@192.168.1.100:22 - Bitvise xterm - diaa@ITI:~/eyego/Task/app

```
Login Succeeded
building docker image
[+] Building 27.7s (10/10) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 222B
 => [internal] load metadata for docker.io/library/node:18-alpine
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load build context
 => => transferring context: 30.32kB
 => [1/5] FROM docker.io/library/node:18-alpine
 => CACHED [2/5] WORKDIR /app
 => CACHED [3/5] COPY package*.json ./
 => CACHED [4/5] RUN npm install
 => [5/5] COPY . .
 => exporting to image
 => => exporting layers
 => => writing image sha256:cfa236a820c2480603f2dd88eef0aaf820ef78104cef6cc6d95203ad19f2de96
 => => naming to 717279709688.dkr.ecr.us-east-1.amazonaws.com/eyego-repo:latest
push image to ECR
The push refers to repository [717279709688.dkr.ecr.us-east-1.amazonaws.com/eyego-repo]
c71c59047eae: Pushed
1e50dfc4a5af: Pushed
1a18c0ddba2d: Pushed
437d58d72517: Pushed
82140d9a70a7: Pushed
f3b40b0cdb1c: Pushed
0b1f26057bd0: Pushed
08000c18d16d: Pushed
latest: digest: sha256:1b72bfc6c047b86170ba3fd5ba5ec97e9b45d4dcad6babc203ffc88abeb69000 size: 1991
[diaa@ITI app]$
```



```
[diaa@ITI k8s]$ kubectl apply -f deployment.yml
deployment.apps/eyego-app created
[diaa@ITI k8s]$ kubectl get deployment
NAME        READY   UP-TO-DATE   AVAILABLE   AGE
eyego-app   2/2     2            2           65s
[diaa@ITI k8s]$
```

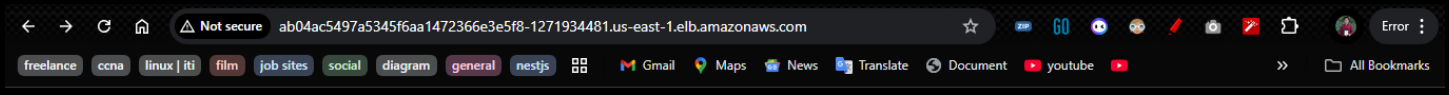
Run deployment



Create svc

```
[diaa@ITI k8s]$ vi service.yml
[diaa@ITI k8s]$ kubectl apply -f service.yml
service/eyego-service created
[diaa@ITI k8s]$ kubectl get svc
NAME           TYPE           CLUSTER-IP     EXTERNAL-IP                                                              PORT(S)
 AGE
eyego-service   LoadBalancer   172.20.124.6   ab04ac5497a5345f6aa1472366e3e5f8-1271934481.us-east-1.elb.amazonaws.com   80:30334/TCP
 11s
kubernetes      ClusterIP      172.20.0.1     <none>                                                                   443/TCP
 57m
```

```
[diaa@ITI k8s]$ curl ab04ac5497a5345f6aa1472366e3e5f8-1271934481.us-east-1.elb.amazonaws.com
<body style="background: black; color: white;"><h1><center>
    Hello Eyego
    </center></h1></body [diaa@ITI k8s]$
```



Hello Eyego

Jenkins

Install docker , kubectl, aws cli , aws configure, git,eks(kubectl) ,docker {

```
sudo yum install -y docker
sudo service docker start
sudo usermod -a -G docker ec2-user
sudo usermod -aG docker jenkins
newgrp docker
sudo systemctl restart jenkinson agent
```

For plugins docker , credentials binding pipeline

Add aws credentials to jenkins (Manage Jenkins > Global Tool Configuration > Docker Installations)

ssh -T git@github.com

```
[ec2-user@ip-172-31-47-125 ~]$ sudo fallocate -l 2G /swapfile
sudo chmod 600 /swapfile
sudo mkswap /swapfile
sudo swapon /swapfile
[ec2-user@ip-172-31-47-125 ~]$ swapon --show
[ec2-user@ip-172-31-47-125 ~]$ sudo chmod 600 /swapfile
[ec2-user@ip-172-31-47-125 ~]$ sudo mkswap /swapfile
Setting up swapspace version 1, size = 2 GiB (2147479552 bytes)
no label, UUID=9d54e25d-665b-4ca4-afd3-c544ea927a7c
[ec2-user@ip-172-31-47-125 ~]$ sudo swapon /swapfile
[ec2-user@ip-172-31-47-125 ~]$ swapon --show
NAME      TYPE SIZE USED PRIO
/swapfile file   2G   0B   -2
[ec2-user@ip-172-31-47-125 ~]$ sudo rm -rf /tmp/*
[ec2-user@ip-172-31-47-125 ~]$ echo '/swapfile none swap sw 0 0' | sudo t
ee -a /etc/fstab
/swapfile none swap sw 0 0
[ec2-user@ip-172-31-47-125 ~]$ sudo systemctl restart jenkins
```

sudo fallocate -l 2G /swapfile

sudo chmod 600 /swapfile

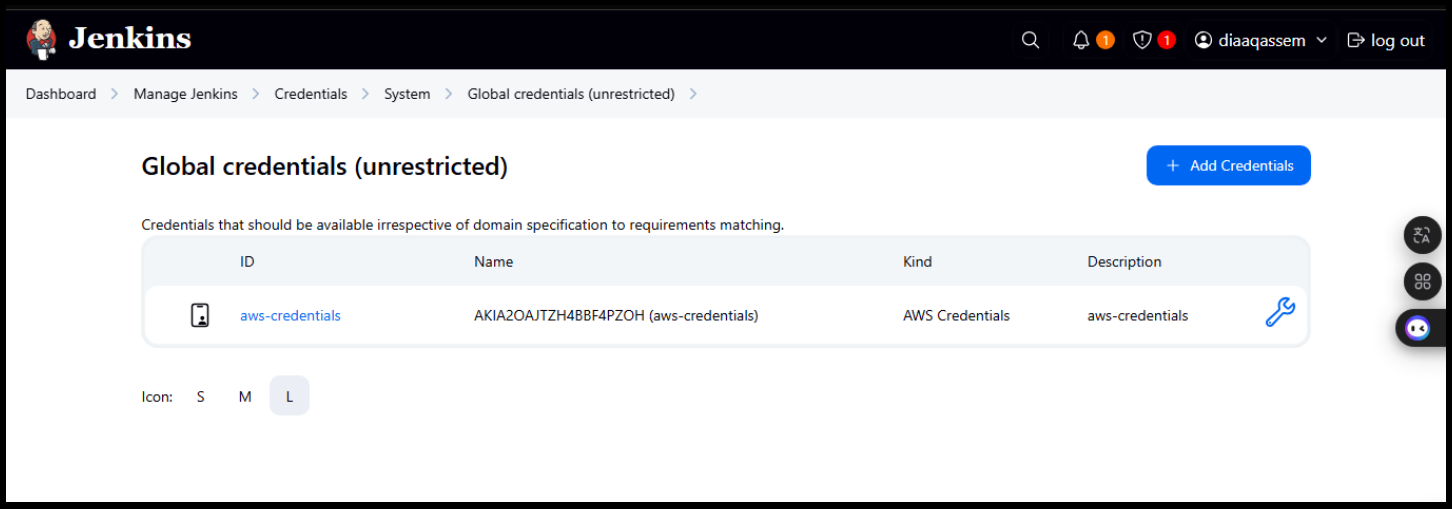
sudo mkswap /swapfile

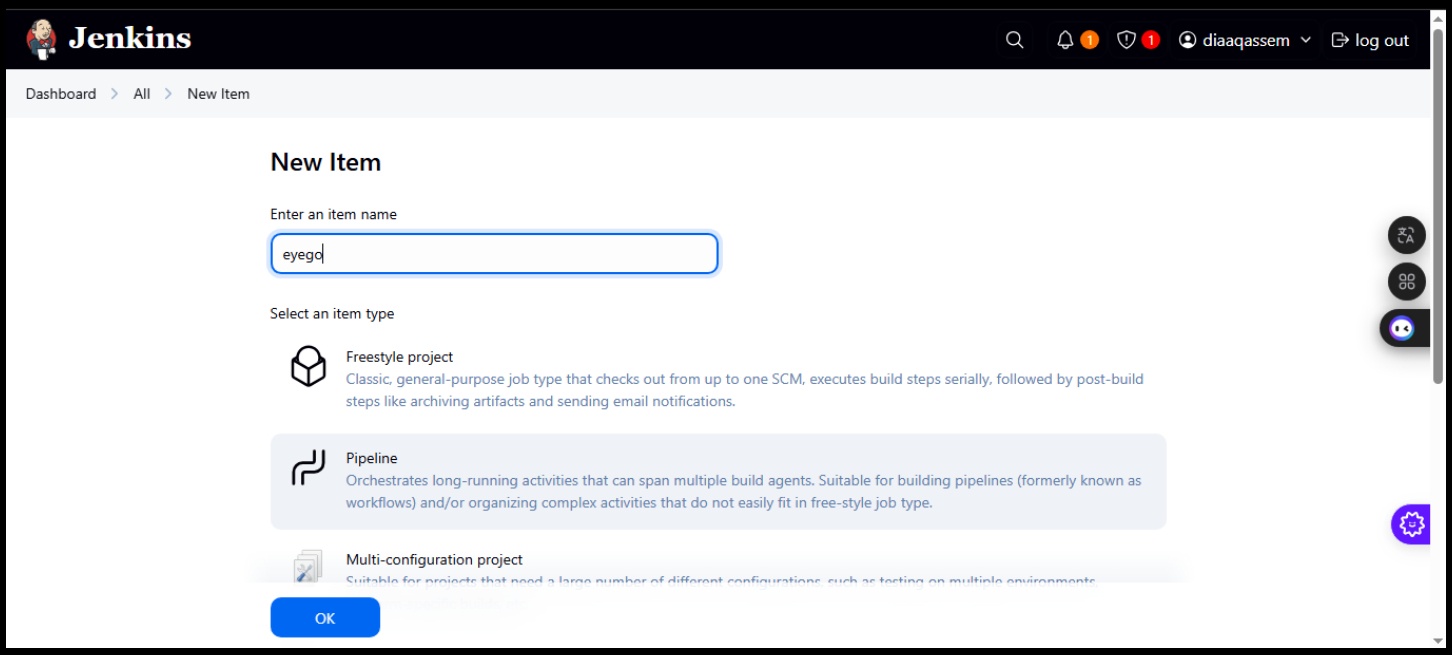
sudo swapon /swapfile

swapon --show

echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab

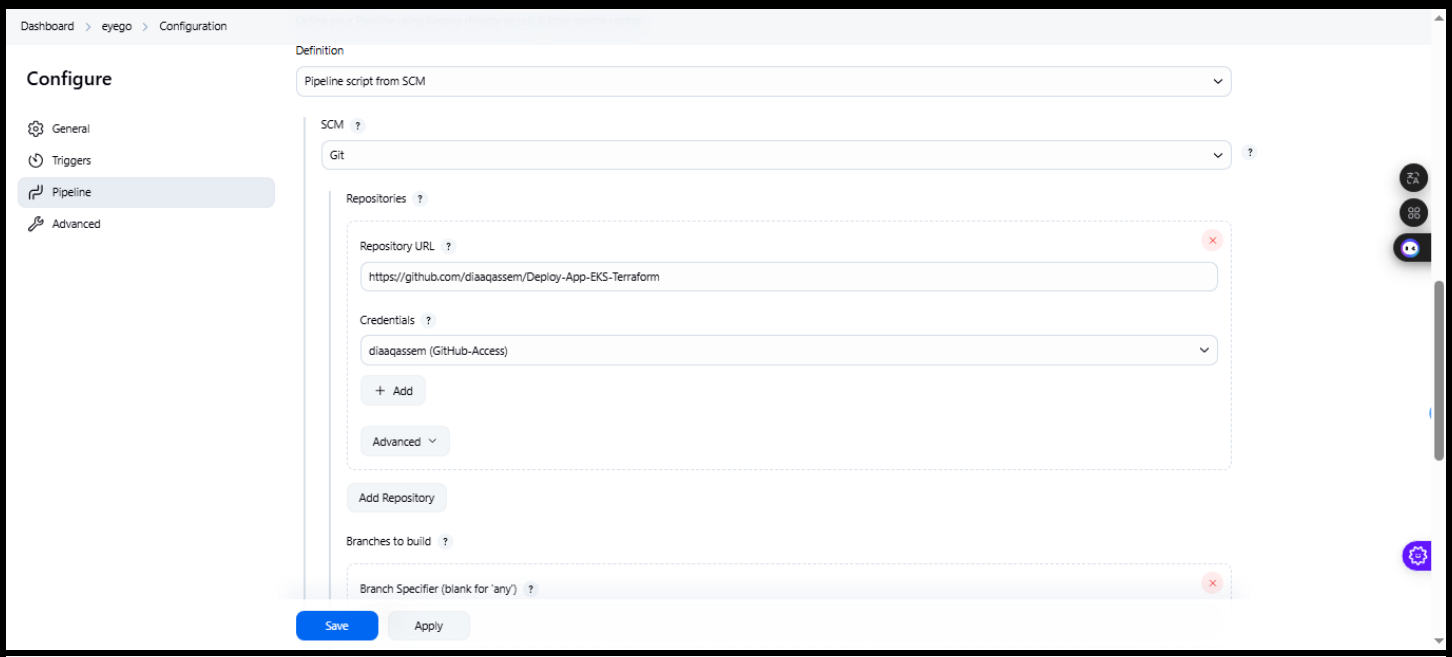
add aws credentials to jenkins

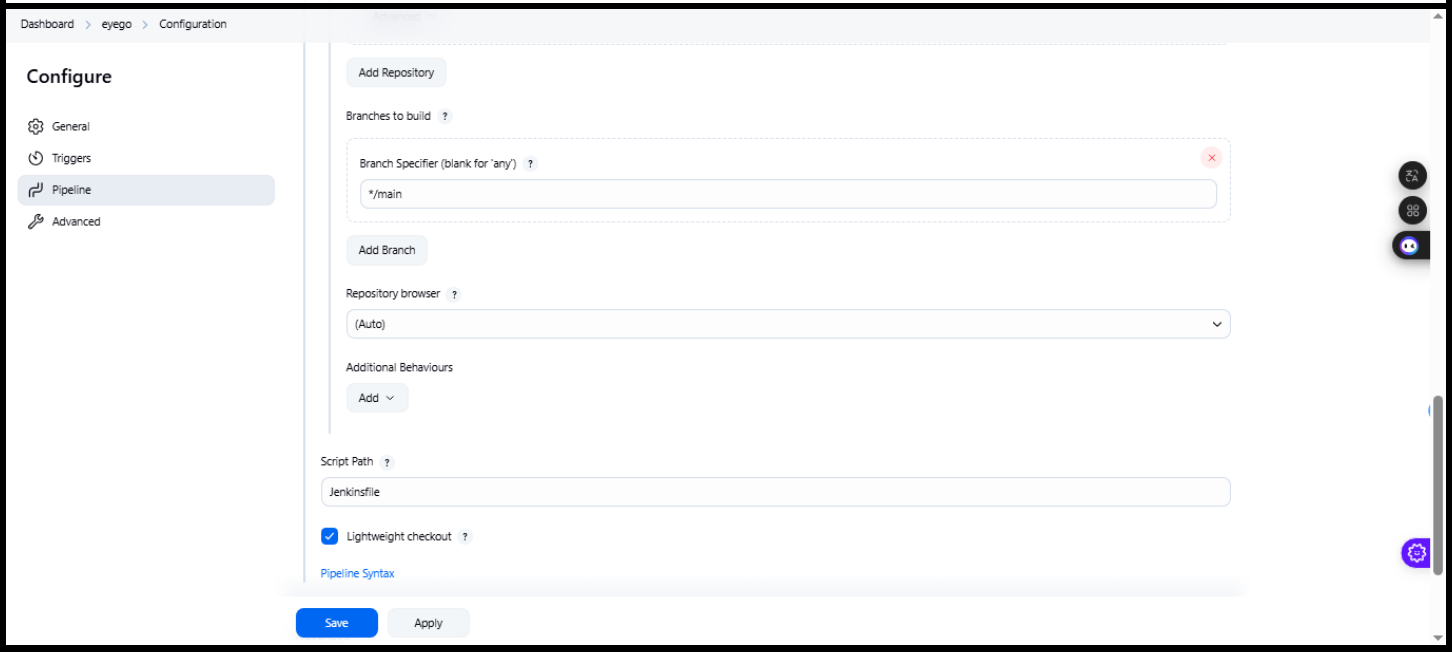

Jenkins
diaaqassem
log out
Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >
Global credentials (unrestricted)
+ Add Credentials
Credentials that should be available irrespective of domain specification to requirements matching.
ID
Name
Kind
Description
aws-credentials
AKIA2OAJTZH4BBF4PZOH (aws-credentials)
AWS Credentials
aws-credentials
Icon: S M L

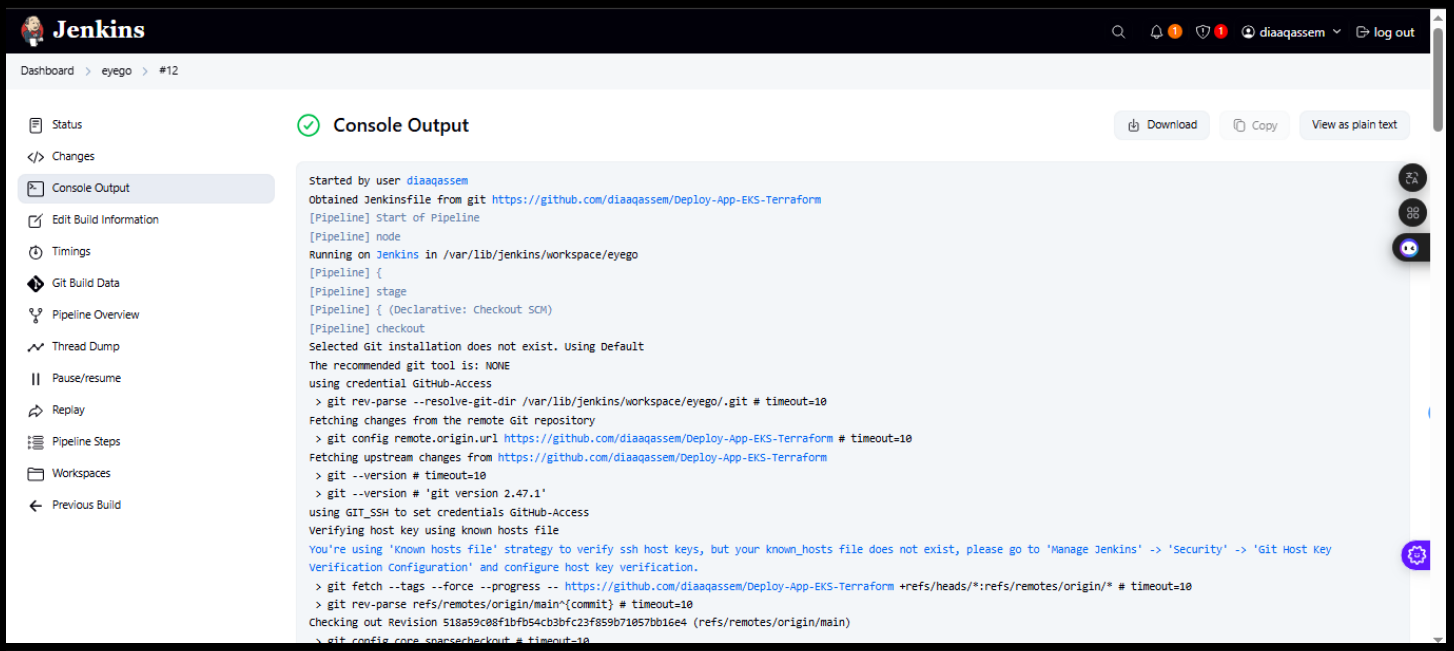

Jenkins
diaaqassem
log out
Dashboard > All > New Item
New Item
Enter an item name
eyego
Select an item type
Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
Multi-configuration project
OK

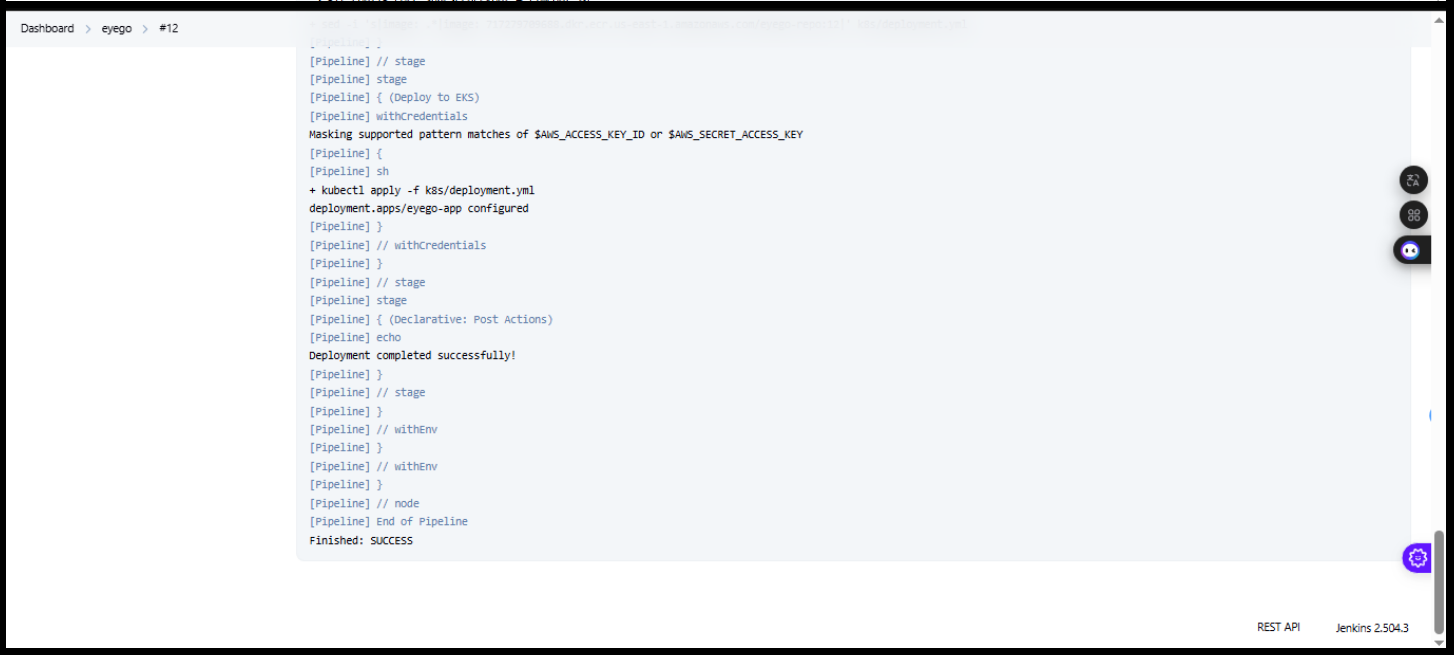
Jenkins file

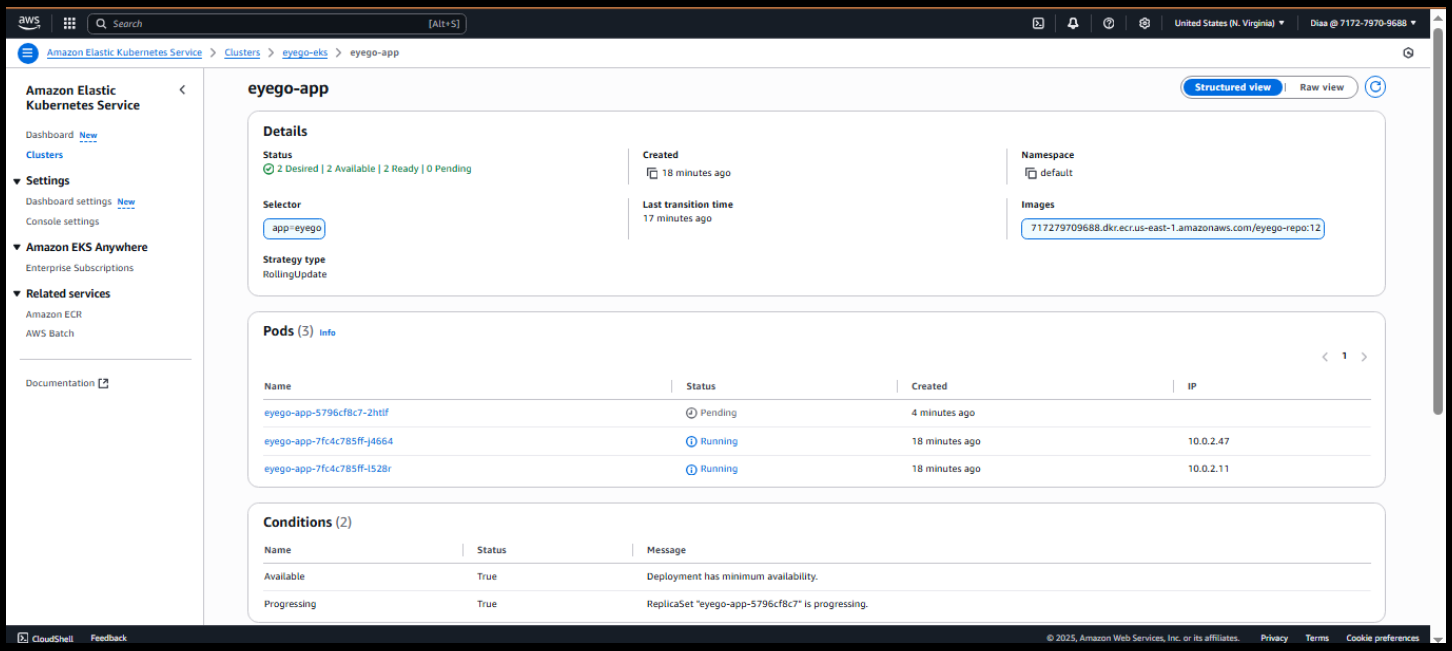
```
pipeline {
    agent any

    environment {
        AWS_REGION     = "us-east-1"
        ECR_REPO       = "717279709688.dkr.ecr.us-east-1.amazonaws.com/eyego-repo"
        IMAGE_TAG      = "${BUILD_NUMBER}"
        CLUSTER_NAME   = "eyego-eks"
        DEPLOYMENT_YML = "k8s/deployment.yml"
    }

    stages {
        stage('Checkout Code') {
            steps {
                git url: 'https://github.com/diaaqassem/Deploy-App-EKS-Terraform.git', branch: 'main'
            }
        }

        stage('Build Docker Image') {
            steps {
                sh "docker build -t $ECR_REPO:$IMAGE_TAG ./app"
            }
        }

        stage('Login to AWS ECR') {
            steps {
                withCredentials([[$class: 'AmazonWebServicesCredentialsBinding',
                                  credentialsId: 'aws-credentials']]) {
                    sh '''
                        aws ecr get-login-password --region $AWS_REGION | docker login --username AWS --password-stdin $ECR_REPO
                    '''
                }
            }
        }

        stage('Push Image to ECR') {
            steps {
                sh "docker push $ECR_REPO:$IMAGE_TAG"
            }
        }

        stage('Update Kubeconfig') {
            steps {
                withCredentials([[$class: 'AmazonWebServicesCredentialsBinding',
                                  credentialsId: 'aws-credentials']]) {
                    sh "aws eks update-kubeconfig --region $AWS_REGION --name $CLUSTER_NAME"
                }
            }
        }

        stage('Update Deployment YAML with new image') {
            steps {
                sh 'sed -i "s|image: .*|image: ${ECR_REPO}:${IMAGE_TAG}|" ${DEPLOYMENT_YML}'
            }
        }

        stage('Deploy to EKS') {
            steps {
               withCredentials([[$class: 'AmazonWebServicesCredentialsBinding',
                                  credentialsId: 'aws-credentials']]){
                     sh "kubectl apply -f $DEPLOYMENT_YML"
        }
    }
        }
    }

    post {
        success {
            echo 'Deployment completed successfully!'
        }
        failure {
            echo 'Deployment failed.'
        }
    }
}
```

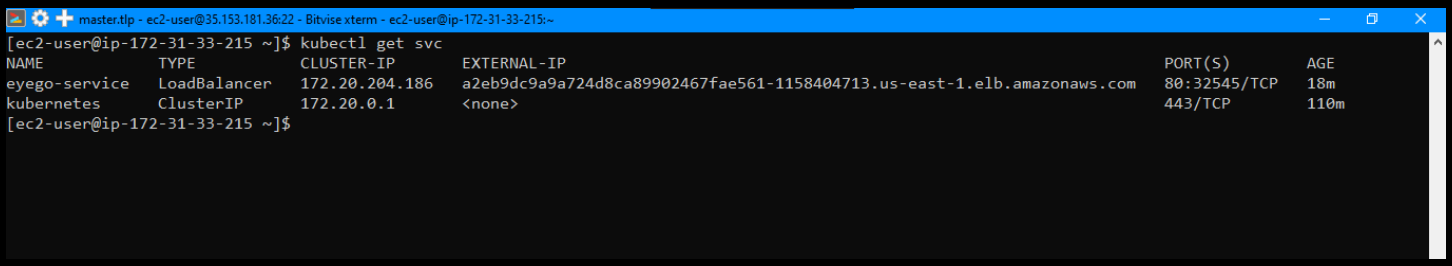

Dashboard
eyego
Configuration
Configure
General
Triggers
Pipeline
Advanced
Definition
Pipeline script from SCM
SCM
Git
Repositories
Repository URL
https://github.com/diaaqassem/Deploy-App-EKS-Terraform
Credentials
diaaqassem (GitHub-Access)
Add
Advanced
Add Repository
Branches to build
Branch Specifier (blank for 'any')
Save
Apply


Dashboard
eyego
Configuration
Configure
General
Triggers
Pipeline
Advanced
Add Repository
Branches to build
Branch Specifier (blank for 'any')
*/main
Add Branch
Repository browser
(Auto)
Additional Behaviours
Add
Script Path
Jenkinsfile
Lightweight checkout
Pipeline Syntax
Save
Apply


Jenkins
diaaqassem
log out
Dashboard
eyego
#12
Status
Changes
Console Output
Edit Build Information
Timings
Git Build Data
Pipeline Overview
Thread Dump
Pause/resume
Replay
Pipeline Steps
Workspaces
Previous Build
Console Output
Download
Copy
View as plain text
Started by user diaaqassem
Obtained Jenkinsfile from git https://github.com/diaaqassem/Deploy-App-EKS-Terraform
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/eyego
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
using credential GitHub-Access
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/eyego/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/diaaqassem/Deploy-App-EKS-Terraform # timeout=10
Fetching upstream changes from https://github.com/diaaqassem/Deploy-App-EKS-Terraform
> git --version # timeout=10
> git --version # 'git version 2.47.1'
using GIT_SSH to set credentials GitHub-Access
Verifying host key using known hosts file
You're using 'Known hosts file' strategy to verify ssh host keys, but your known_hosts file does not exist, please go to 'Manage Jenkins' -> 'Security' -> 'Git Host Key Verification Configuration' and configure host key verification.
> git fetch --tags --force --progress -- https://github.com/diaaqassem/Deploy-App-EKS-Terraform +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 518a59c08f1bfb54cb3bfc23f859b71057bb16e4 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10

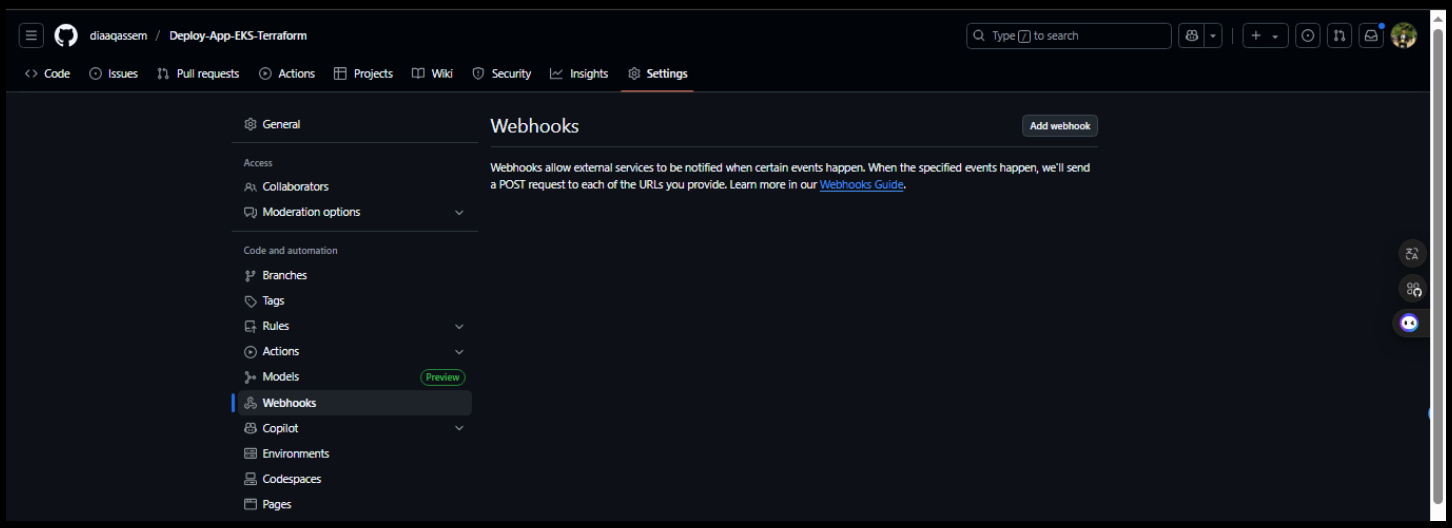

Dashboard
eyego
#12
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Deploy to EKS)
[Pipeline] withCredentials
Masking supported pattern matches of $AWS_ACCESS_KEY_ID or $AWS_SECRET_ACCESS_KEY
[Pipeline] {
[Pipeline] sh
+ kubectl apply -f k8s/deployment.yml
deployment.apps/eyego-app configured
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
Deployment completed successfully!
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
REST API
Jenkins 2.504.3

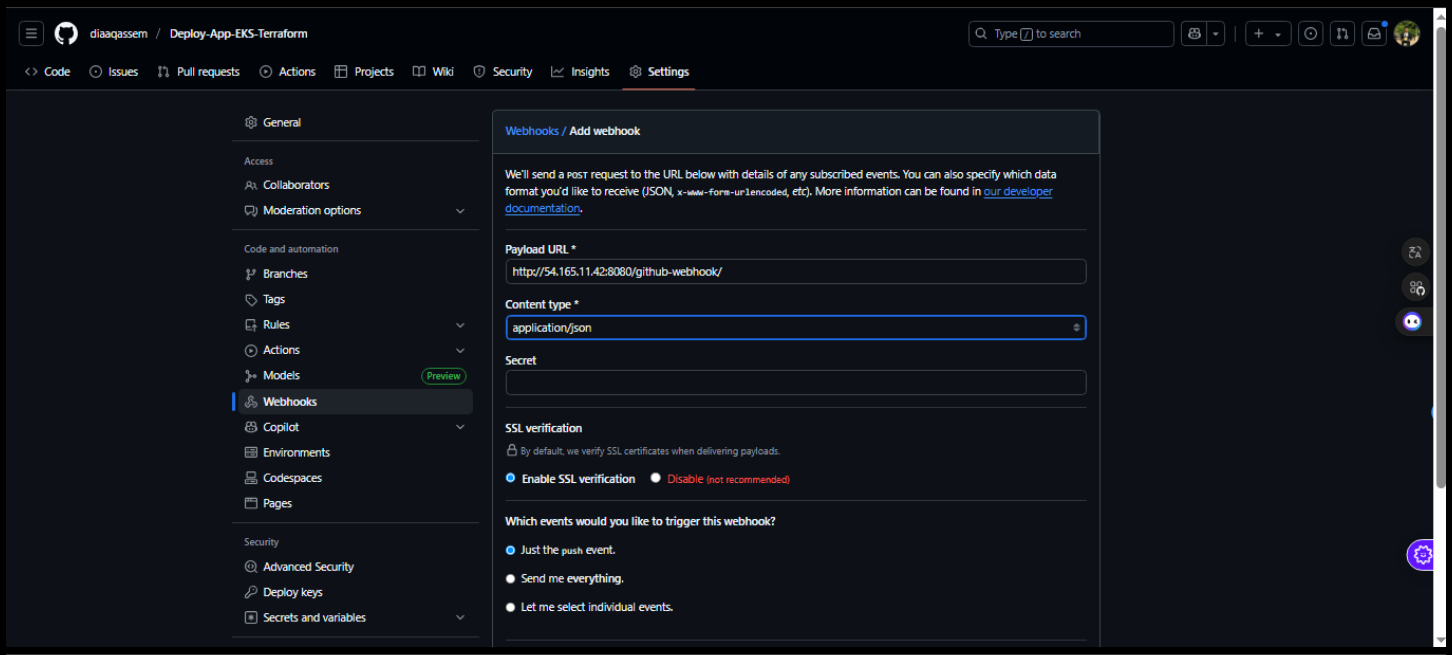


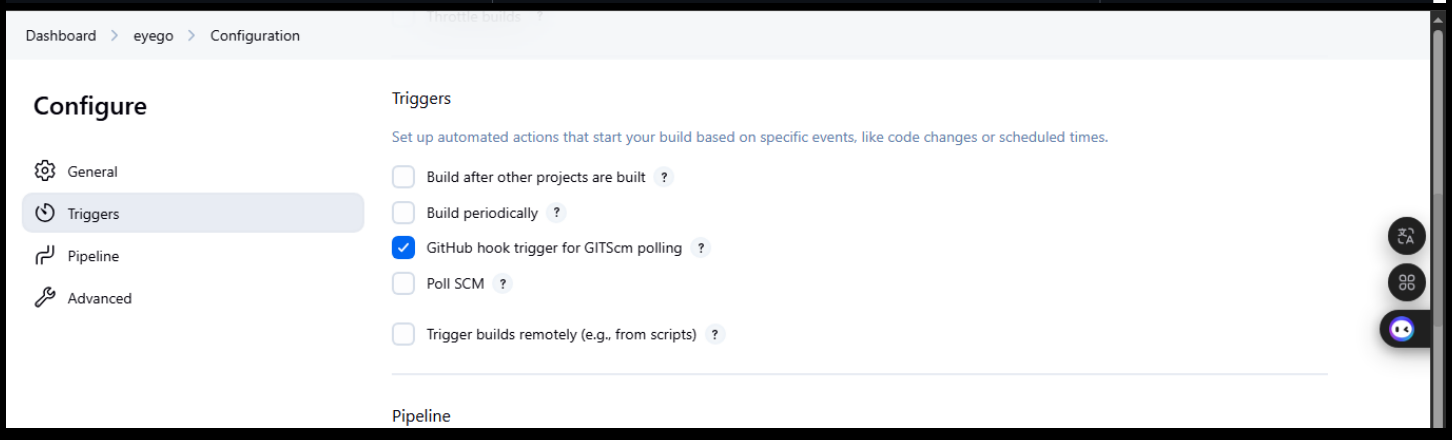


Hello Eyego

Then add trigger

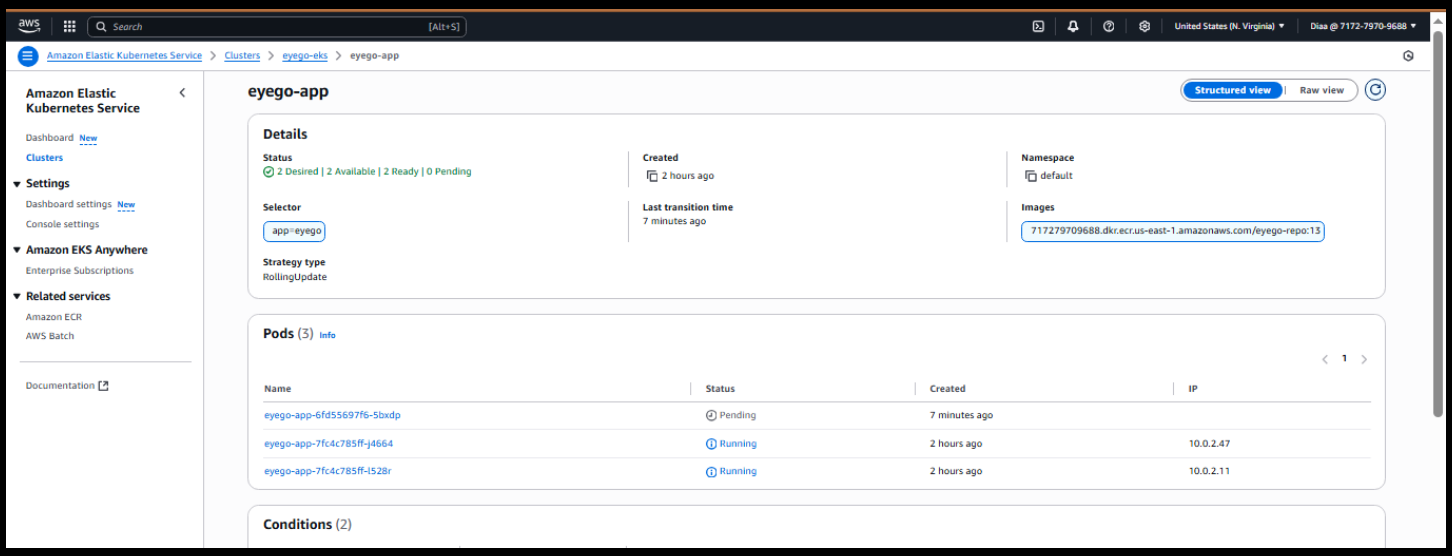




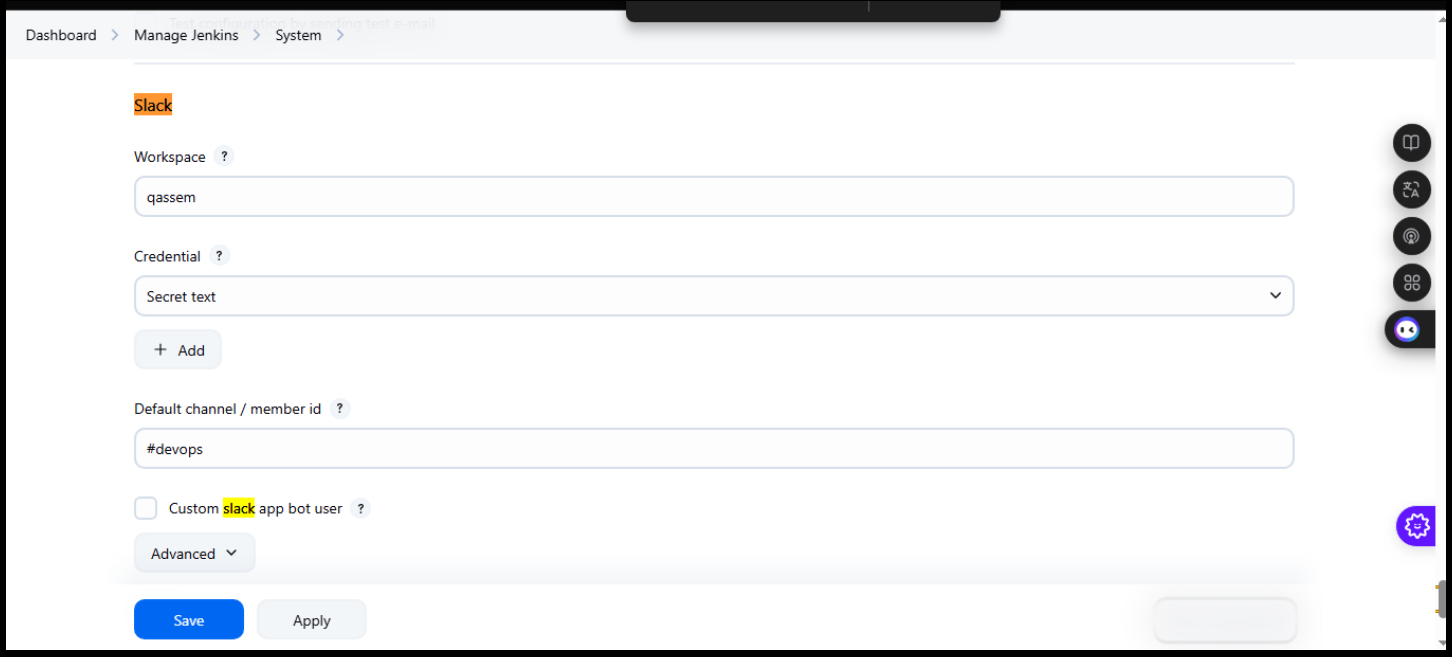


Test trigger

```
$ git push -u origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 420 bytes | 140.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:diaaqassem/Deploy-App-EKS-Terraform.git
   06fcb20..6b772a5  main -> main
branch 'main' set up to track 'origin/main'.

FreeComp@Deyaa-Qassem MINGW64 /e/Courses/ITI/task1/Task/Task (main)
$
```

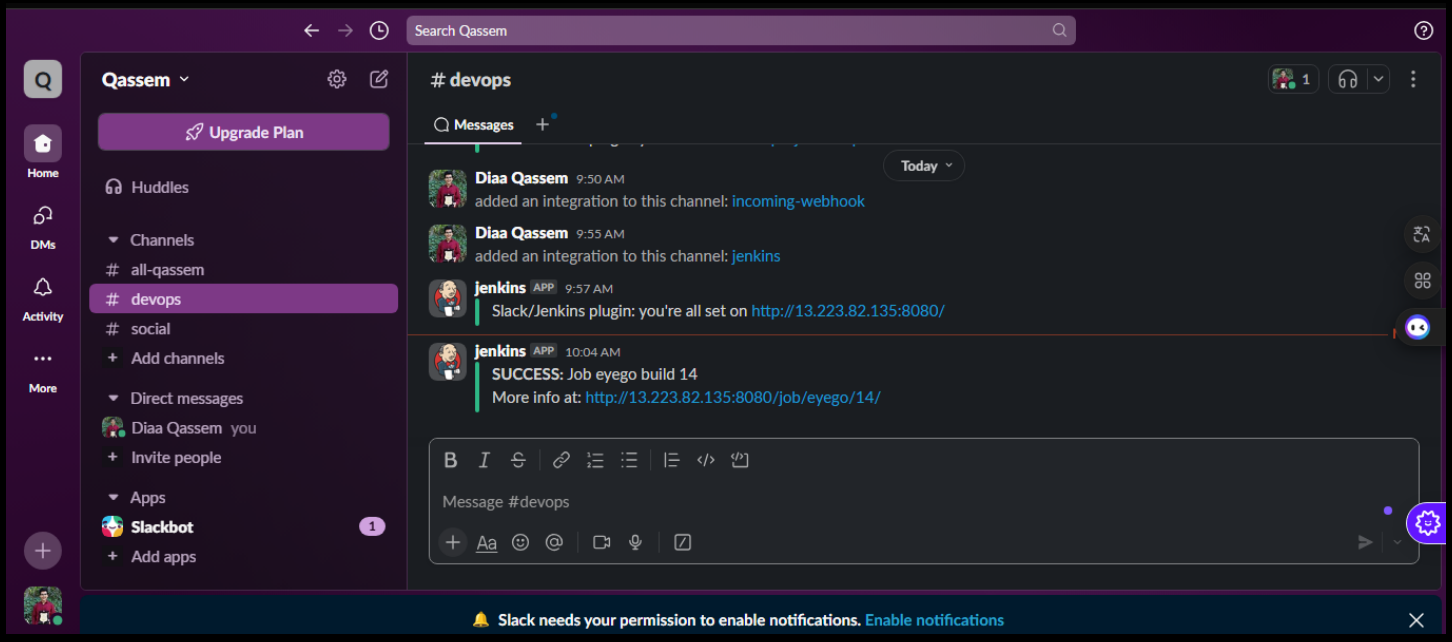


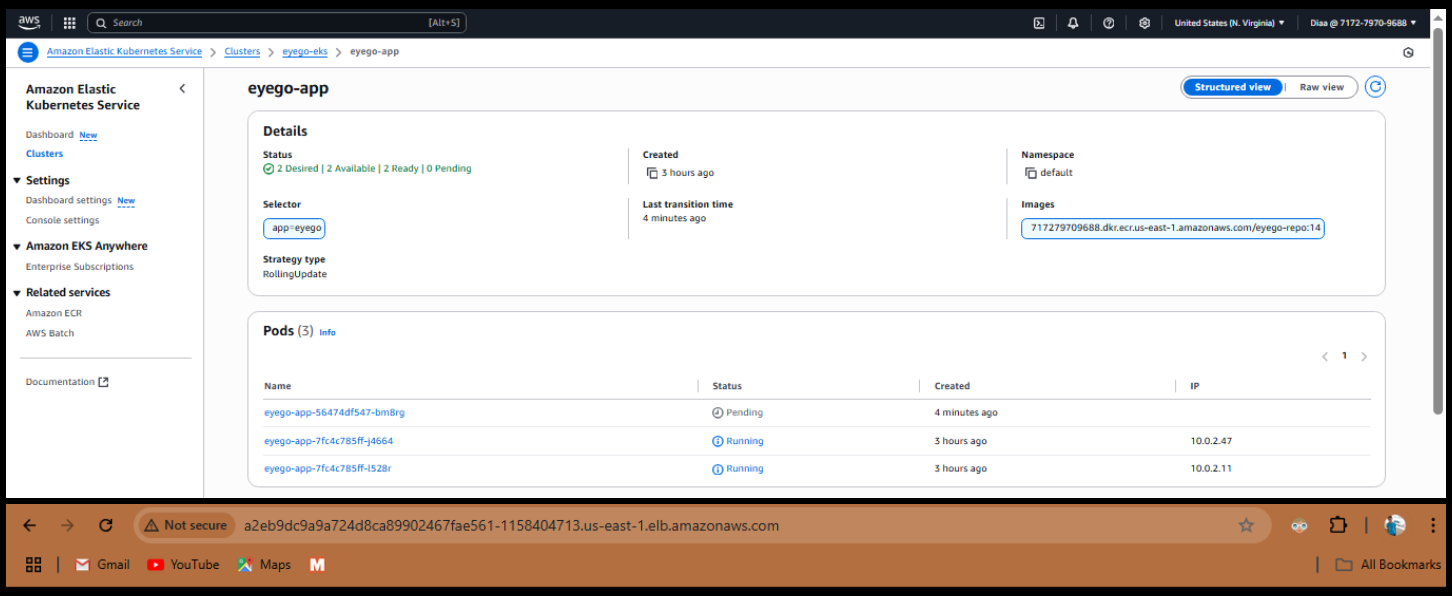
Add slack notification https://qassem.slack.com/services/B096ZCX5TSS?added=1



```
def COLOR_MAP = [
    'SUCCESS': 'good',
    'FAILURE': 'danger',
]

pipeline {
    agent any

    environment {
        AWS_REGION     = "us-east-1"
        ECR_REPO       = "717279709688.dkr.ecr.us-east-1.amazonaws.com/eyego-repo"
        IMAGE_TAG      = "${BUILD_NUMBER}"
        CLUSTER_NAME   = "eyego-eks"
        DEPLOYMENT_YML = "k8s/deployment.yml"
    }

    stages {
        stage('Checkout Code') {
            steps {
                git url: 'https://github.com/diaaqassem/Deploy-App-EKS-Terraform.git', branch: 'main'
            }
        }

        stage('Build Docker Image') {
            steps {
                sh "docker build -t $ECR_REPO:$IMAGE_TAG ./app"
            }
        }

        stage('Login to AWS ECR') {
            steps {
                withCredentials([[$class: 'AmazonWebServicesCredentialsBinding',
                                  credentialsId: 'aws-credentials']]) {
                    sh '''
                        aws ecr get-login-password --region $AWS_REGION | docker login --username AWS --password-stdin $ECR_REPO
                    '''
                }
            }
        }

        stage('Push Image to ECR') {
            steps {
                sh "docker push $ECR_REPO:$IMAGE_TAG"
            }
        }

        stage('Update Kubeconfig') {
            steps {
                withCredentials([[$class: 'AmazonWebServicesCredentialsBinding',
                                  credentialsId: 'aws-credentials']]) {
                    sh "aws eks update-kubeconfig --region $AWS_REGION --name $CLUSTER_NAME"
                }
            }
        }

        stage('Update Deployment YAML with new image') {
            steps {
                sh 'sed -i "s|image: .*|image: ${ECR_REPO}:${IMAGE_TAG}|" ${DEPLOYMENT_YML}'
            }
        }

        stage('Deploy to EKS') {
            steps {
               withCredentials([[$class: 'AmazonWebServicesCredentialsBinding',
                                  credentialsId: 'aws-credentials']]){
                     sh "kubectl apply -f $DEPLOYMENT_YML"
            }
    }
        }
    }

    post {
        always {
            echo 'Slack Notifications.'
            slackSend channel: '#devops',
                color: COLOR_MAP[currentBuild.currentResult],
                message: "*${currentBuild.currentResult}:* Job ${env.JOB_NAME} build ${env.BUILD_NUMBER} \n More info at: ${env.BUILD_URL}"
        }
        success {
            echo 'Deployment completed successfully!'
        }
        failure {
            echo 'Deployment failed.'
        }
    }
}
```

```
$ git push -u origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 538 bytes | 269.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:diaaqassem/Deploy-App-EKS-Terraform.git
   6b772a5..49c019d  main -> main
branch 'main' set up to track 'origin/main'.
```





Hello Eyego