# Deploy-nodejs-app-to-eks-using-GitHubActions

# launch one instance t2.medium

# 1:Install eksctl CLI tool for creating EKS Clusters on AWS

**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**

# 2:Install Kubectl on Ubuntu

# Add the kubectl GPG key

**curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo gpg --dearmor -o /usr/share/keyrings/kubernetes-archive-keyring.gpg**

# Create apt repo for kubectl

**sudo touch /etc/apt/sources.list.d/kubernetes.list**

# Add Kubernetes official apt repo

**echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://packages.cloud.google.com/apt/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list > /dev/null**

# Update the system packages

**sudo apt-get update**

# Install kubectl on Ubuntu

**sudo apt-get install -y kubectl**

# 4:Install AWS CLI on Ubuntu

**sudo apt install awscli**

**aws --version**

# 5:Create AWS Access Key ID and Secret Access Key

**Aws configure**

# Create eks cluster :

**eksctl create cluster --name demo-ekscluster --region us-east-1 --version 1.27 --nodegroup-name linux-nodes --node-type t2.micro --nodes 2**

# 6:Create repository in github ex. [[deploy-nodejs-app-to-eks-using-actions](https://github.com/pawankahurke/deploy-nodejs-app-to-eks-using-actions/tree/main)]

# ->Clone this repo into local

**->ref. <https://github.com/pawankahurke/deploy-nodejs-app-to-eks-using-actions/tree/main>**

**->make changes in dockerfile and deployment.yaml according to your requirement**

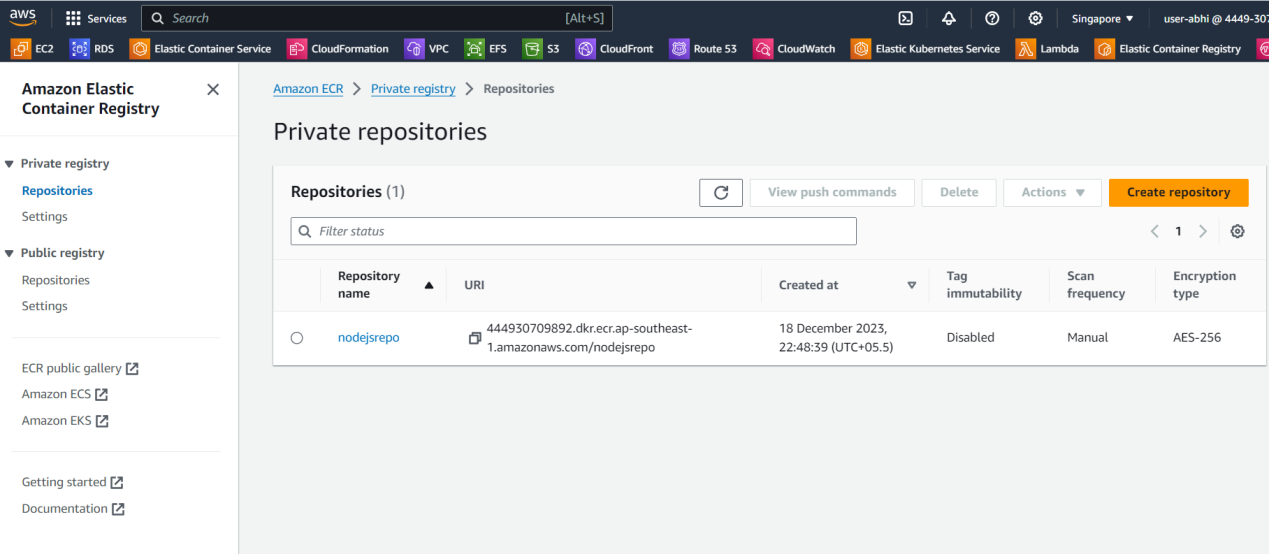
**->also create main.yaml in .github/workflows/main.yaml [default]**

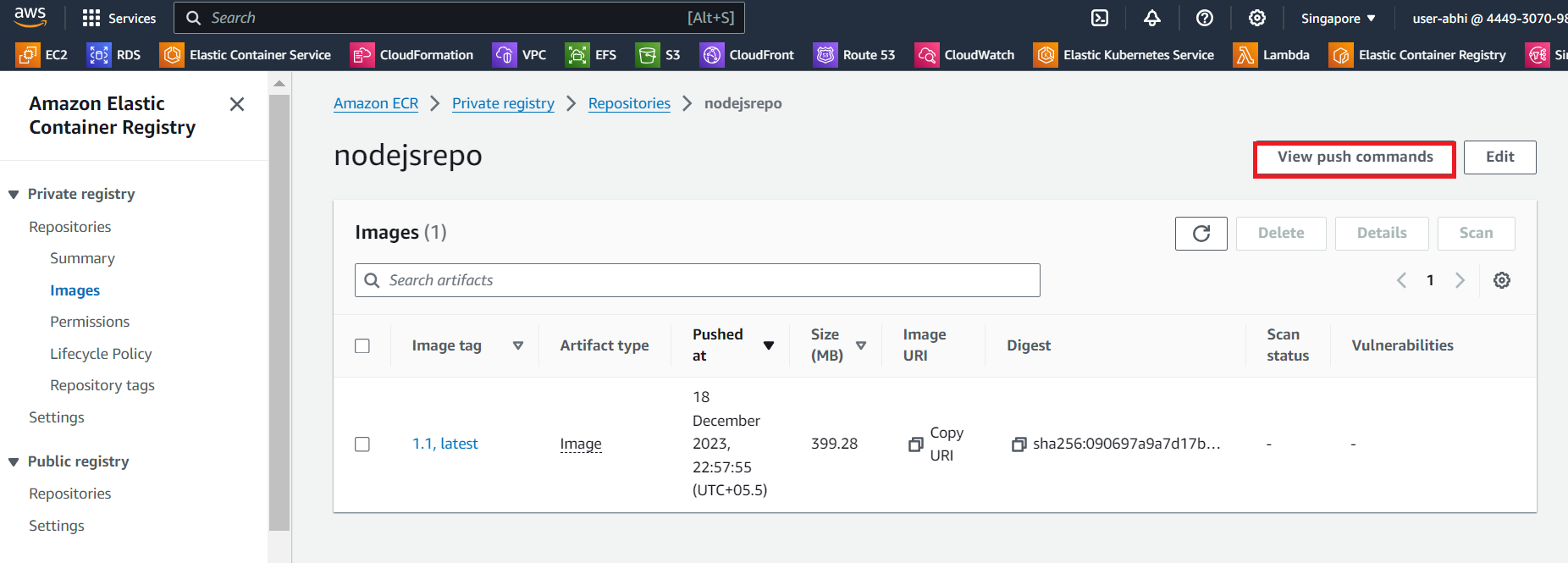
**Refer this code: <https://github.com/pawankahurke/deploy-nodejs-app-to-eks-using-actions/blob/main/.github/workflows/main.yml>**

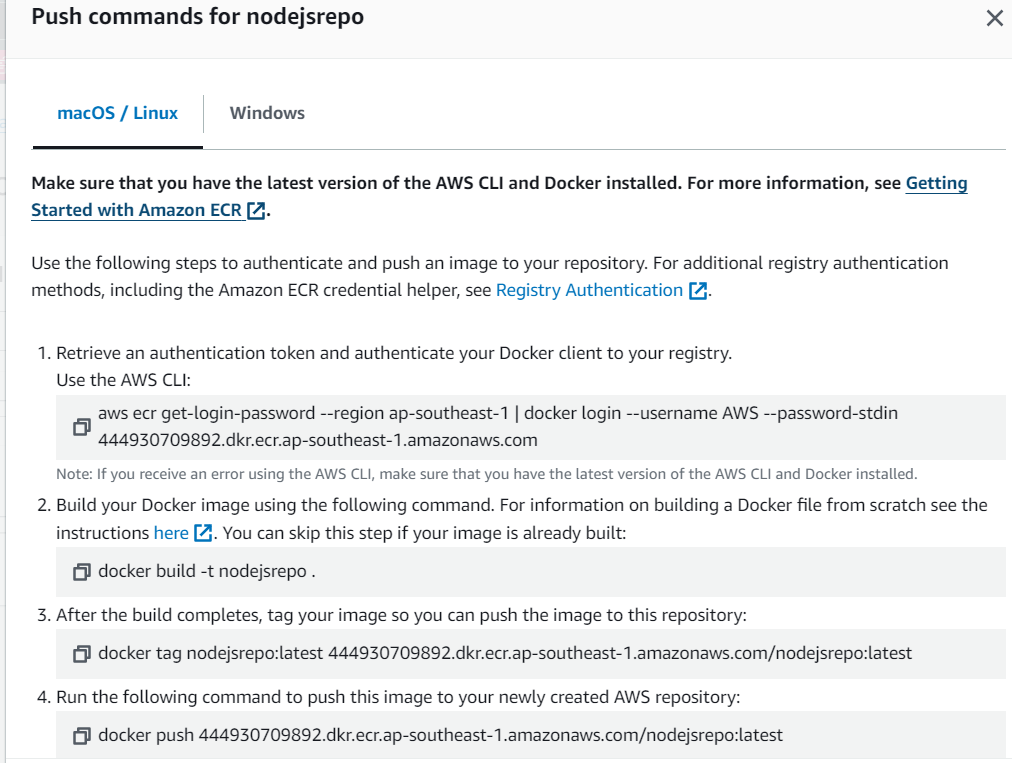
# 6:Create ecr repository private>reponame>create

**->create repo**

**->open repo and view docker push commands**







To create a Docker image and push it to Amazon Elastic Container Registry (ECR), you'll need to follow several steps. Below is a step-by-step guide assuming you already have Docker installed and configured, as well as AWS CLI configured with the necessary permissions to interact with ECR.

1. Install Docker:

**Sudo apt install docker.io**

Add docker group

**sudo usermod -aG docker ubuntu**

**newgrp docker**

**Sudo systemctl restart docker**

2. Configure AWS CLI:

Ensure that AWS CLI is installed, and configure it with the necessary credentials and region. You can configure AWS CLI by running:

**aws configure**

3. Build and Tag Docker Image:

Assuming you have a Dockerfile in your project directory, you can build and tag the Docker image with the following commands:

**# git clone ->Navigate to the directory containing your Dockerfile**

cd /path/to/your/project

# Build the Docker image

**docker build -t your-image-name:your-tag .**

# Tag the Docker image for ECR

**docker tag your-image-name:your-tag your-ecr-repository-url/your-image-name:your-tag**

Replace your-image-name, your-tag, and your-ecr-repository-url with your actual image name, tag, and ECR repository URL.

4. Authenticate Docker with ECR:

Run the following command to authenticate Docker with your ECR registry:

**aws ecr get-login-password --region your-region | docker login --username AWS --password-stdin your-ecr-repository-url**

Replace your-region and your-ecr-repository-url with your AWS region and ECR repository URL.

5. Push Docker Image to ECR:

**docker push your-ecr-repository-url/your-image-name:your-tag**

6: go to github-Actions add env and variables

->secret and access keye

->add docker secrets

7.**Edit main.yaml and run github-action pipeline**

->Go to eks cluster:

->Click service and networking

->Choose service

->**copy load balancer link and see it works or not**

