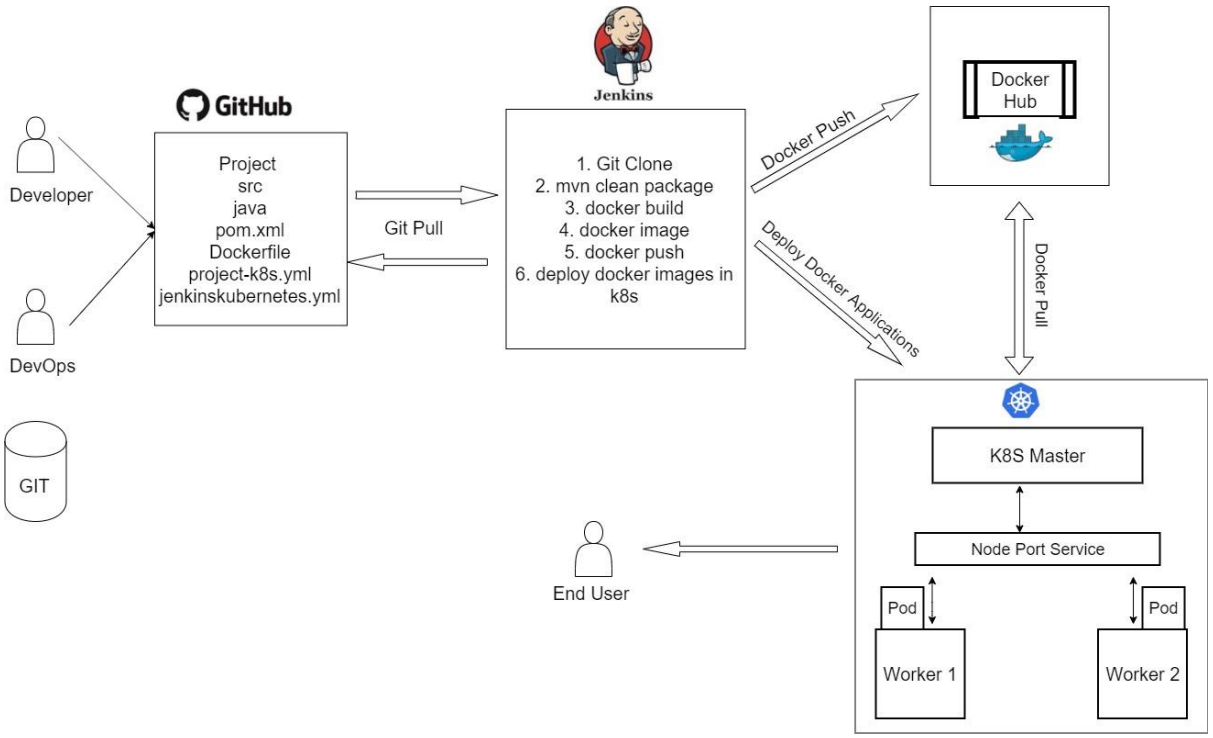
# Jenkins Kubernetes CI & CD




GitHub Code: https://github.com/prakashk0301/maven-project/tree/jenkins-ecr-eks

Reference Dockerfile: https://github.com/prakashk0301/maven-project/blob/jenkins-ecr-eks/Dockerfile

Reference Jenkinsfile for eks: https://github.com/prakashk0301/maven-project/blob/jenkins-ecr-eks/Jenkinsfile-ECR-EKS

Reference K8s manifest file: https://github.com/prakashk0301/maven-project/blob/jenkins-ecr-eks/deployment-service.yaml

Install AWS , docker and Kubernetes related plugins.

- Amazon EC2 plugin
- Amazon ECR plugin
- Docker plugin
- Docker Pipeline
- CloudBees Docker Build and Publish plugin
- Kubernetes CLI Plugin
- Pipeline: AWS Steps
- Kubernetes
- Kubernetes Credentials
- Kubernetes pipeline
- Config File Provider

**Install Docker on Jenkins Instance**

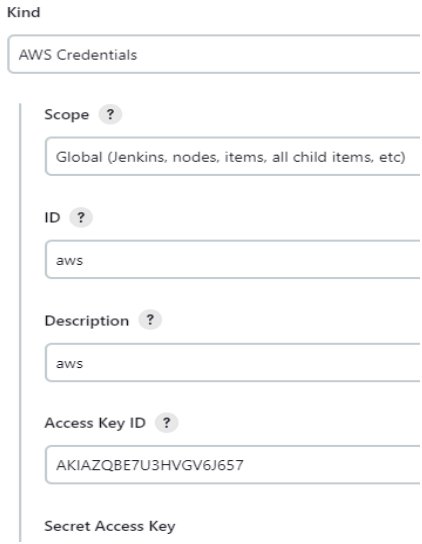
Refer docker installation document

## Create ECR repository and integrate ECR with Jenkins

Login to AWS console as IAM user ->choose ECR service -> Create a ECR repository -> give a good name -> Visibility to private -> Enable Image Scanning setting *(It detects all the layers in Docker image, checks for installed binary and library files in Docker image, detects code vulnerabilities and generate logs)* -> done

## Configure AWS Credentials

1. Generate and download IAM user's access key & secret key
2. Open the Jenkins Dashboard and click on the "Credentials"
3. Select the "System" option, and then click on the "Global credentials (unrestricted)".
4. Click on the "Add Credentials" button and select the "AWS Credentials" option.
5. Enter the AWS Access Key ID and the AWS Secret Access Key for your AWS account.
6. Provide description and select the "ID" field. This ID will be used later in the pipeline to reference the AWS credentials.

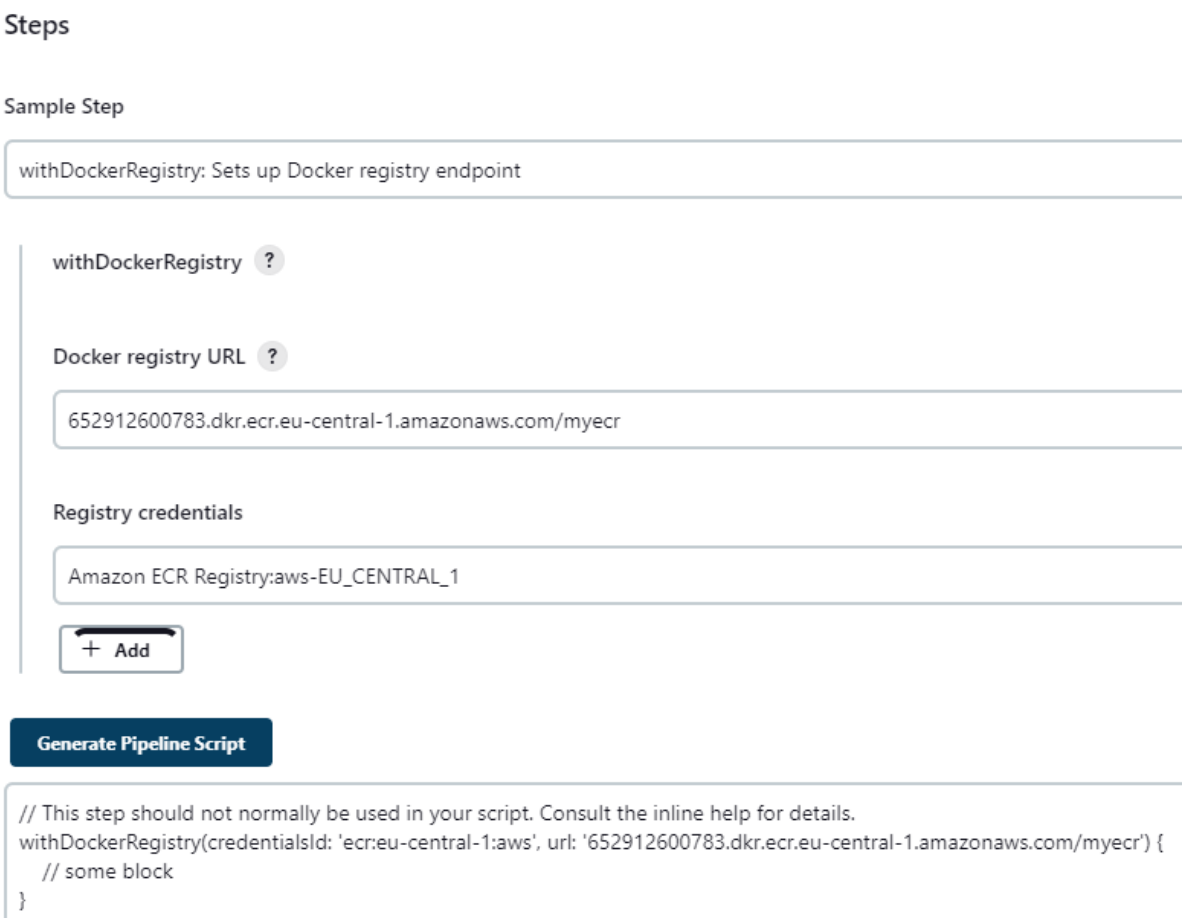



Connect to the Jenkins VM and change permission

sudo chmod 777 /var/run/docker.sock

## Update Jenkinsfile and add docker build and Push to ECR stages

Go to the syntax generator and search for **`withDockerRegistry`,** then provide **ECR URL** and provide **aws ecr registry credentials** then click on generate **syntax button**




Reference Jenkinsfile:

*pipeline {*

*agent any*

*stages {*

***stage('Checkout')***

*{ steps { git branch: 'master', url: 'https://github.com/prakashk0301/golang-jenkins-ecr-eks' } }*

***stage('Docker Image Build')***

*{ steps { sh 'docker build -t 652912600783.dkr.ecr.eu-central-1.amazonaws.com/myecr:latest .' } }*

*//commented: syntax { steps { sh 'docker build -t <ecr docker registry id or IMAGE NAME> .' } }*

```
    stage('Push Docker Image to ECR')


    { steps { withDockerRegistry(credentialsId: 'ecr:eu-central-1:aws', url:
'https://652912600783.dkr.ecr.eu-central-1.amazonaws.com/myecr')

      { sh 'aws ecr get-login-password --region eu-central-1 | docker login --username AWS --password-
stdin 652912600783.dkr.ecr.eu-central-1.amazonaws.com'

        sh 'docker push 652912600783.dkr.ecr.eu-central-1.amazonaws.com/myecr:latest'


      // commented: get ecr get login and push command from ECR

        } }

} } }
```

**Once you upload Docker image to ECR successfully, now we can add the final stage to deploy in kubernetes**

Before we deploy to EKS from jenkins, make sure you install helm and kubeclt package on Jenkins VM. https://archive.eksworkshop.com/020_prerequisites/k8stools/

sudo curl --silent --location -o /usr/local/bin/kubectl https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.5/2022-01-21/bin/linux/amd64/kubectl

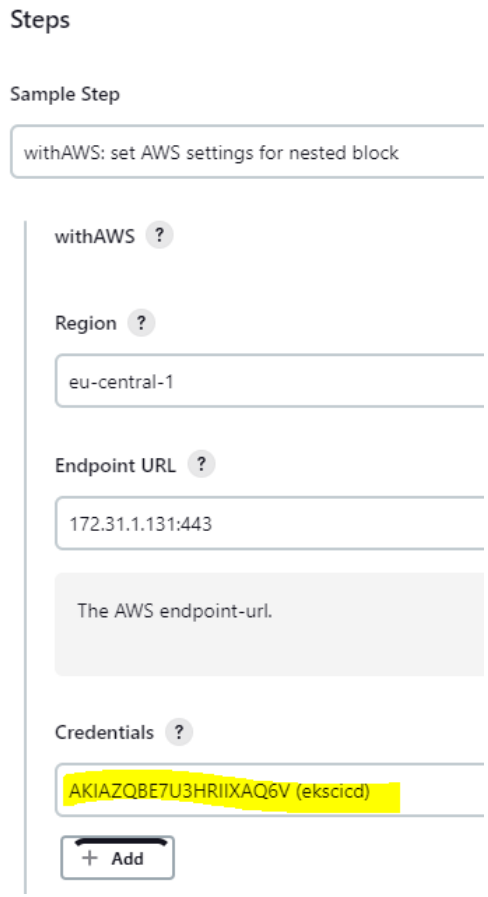
sudo chmod 777 /usr/local/bin/kubectl

if you get any error just connect to Jenkins terminal and follow below link:
https://aws.amazon.com/premiumsupport/knowledge-center/eks-api-server-unauthorized-error/

https://docs.aws.amazon.com/eks/latest/userguide/install-kubectl.html

https://archive.eksworkshop.com/020_prerequisites/k8stools/

**Add kubernetes deployment stage**: (under credentials section only add credential **without** ECR at bottom)




**to get kubernetes server endpoint connect to jenkins and run below command

aws --version

aws configure ( then provide access key, secret key & region code)

aws sts get-caller-identity

aws eks --region <**region**> update-kubeconfig --name <**cluster_name**>

ex: aws eks --region eu-central-1 update-kubeconfig --name k8s-eks

kubectl get endpoints

reference link: https://plugins.jenkins.io/kubernetes-cli/

your stage should look like this:

```
stage ('deploy to EKS kubernetes cluster')
 {steps { withAWS(credentials: 'ekscicd', endpointUrl: '172.31.1.131:443', region: 'eu-central-1')
 {
   sh 'aws eks --region eu-central-1 update-kubeconfig --name eks-test'
   sh 'kubectl apply -f deployment-service.yaml'
```

*// sh 'helm install ./<chartname> '*

*} }}*