****

**JENKINS-ECR-INTEGRATION**

**Step-1**

Install docker engine and jenkins in a instance(use t2.medium so that jenkins PAAC will build in quick time)

Use below script to install dockerengine and jenkins

Switch to root user using sudo -i

vi Jenkins.sh

Copy below script and paste it save and quit

Give executable permissions using chmod +x Jenkins.sh

Run the script by using command : sh Jenkins.sh

**Jenkins.sh**

#!/bin/bash

apt update -y

apt install openjdk-17-jdk -y

sleep 3

apt install maven -y

sleep 3

wget -O /usr/share/keyrings/jenkins-keyring.asc \

https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key

echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \

https://pkg.jenkins.io/debian-stable binary/ | sudo tee \

/etc/apt/sources.list.d/jenkins.list > /dev/null

sudo apt-get update

sleep 3

sudo apt-get install jenkins -y

sleep 3

systemctl restart jenkins

Check whether jenkins is installed or not using systemctl status jenkins

Get the initial password : cat /var/lib/jenkins/secrets/initialAdminPassword

Install Dockerengine using below script

Docker engine installation guide: <https://docs.docker.com/engine/install/ubuntu/>

vi Docker.sh

Copy below script and paste it save and quit

Give executable permissions using chmod +x Docker.sh

Run the script by using command : sh Docker.sh

Docker.sh

#!/bin/bash

apt update -y

apt-get update

apt-get install ca-certificates curl

install -m 0755 -d /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc

chmod a+r /etc/apt/keyrings/docker.asc

sleep 2

echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \

$(. /etc/os-release && echo "$VERSION\_CODENAME") stable" | \

tee /etc/apt/sources.list.d/docker.list > /dev/null

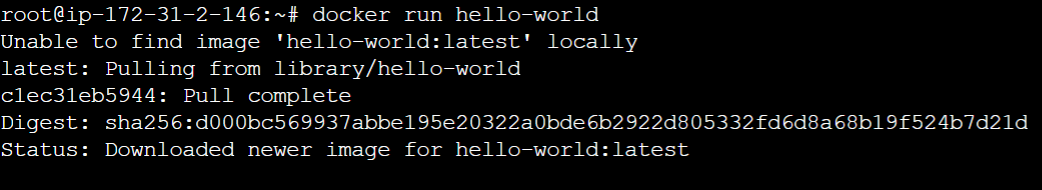
apt-get update

sleep 3

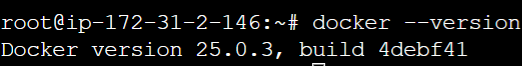
apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

Verify docker installation by running : docker run hello-world

If your installation is correct it should show as shown below

****

Or use

****

Or you can also check using systemctl status docker

Adding jenkins to docker group

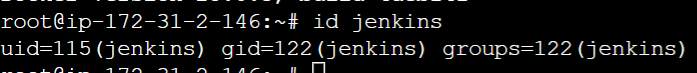
Jenkins user cannot run docker commands by default we need to add jenkins user to docker group

Verifying docker group is present or not in /etc/gshadow

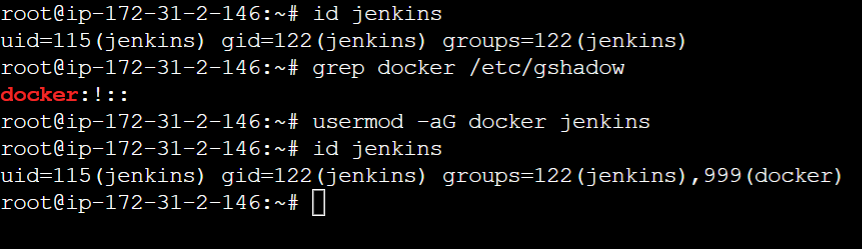
****

Add jenkins user to docker group so that jenkins user can run docker commands without failure in pipeline

Before adding

****

After adding to docker group

****

The above is mandatory otherwise you will get permission denied in Jenkins PAAC

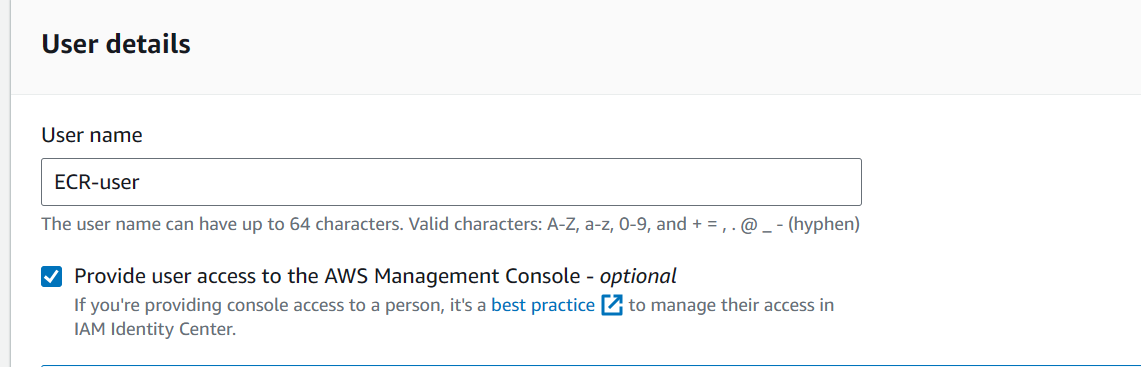
For the changes to impact

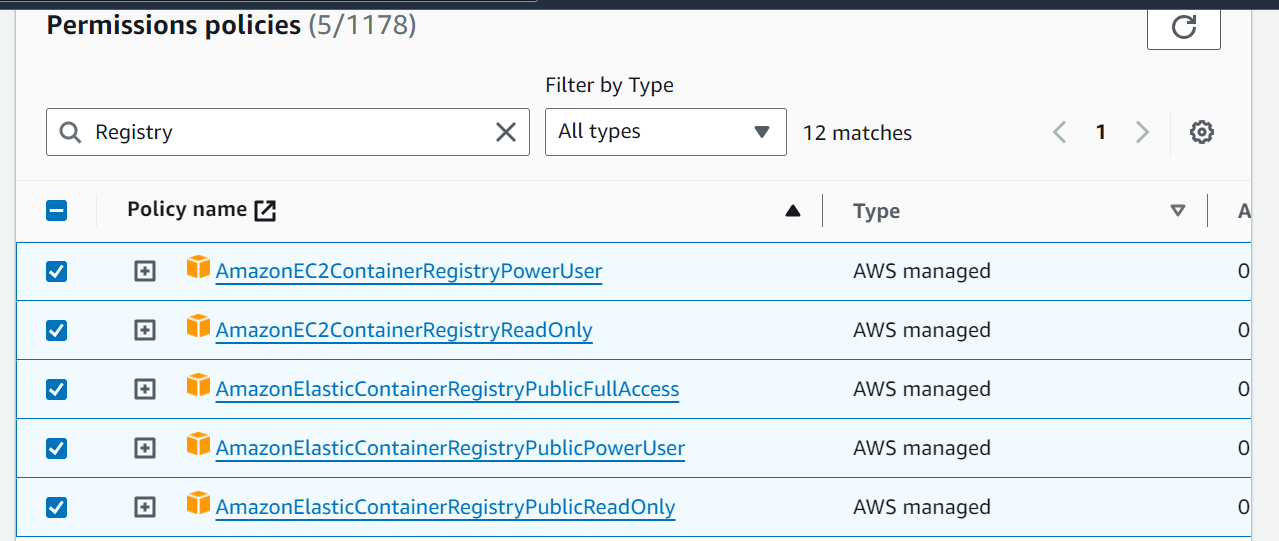
**Reboot the instance or systemctl restart jenkins** is also fine

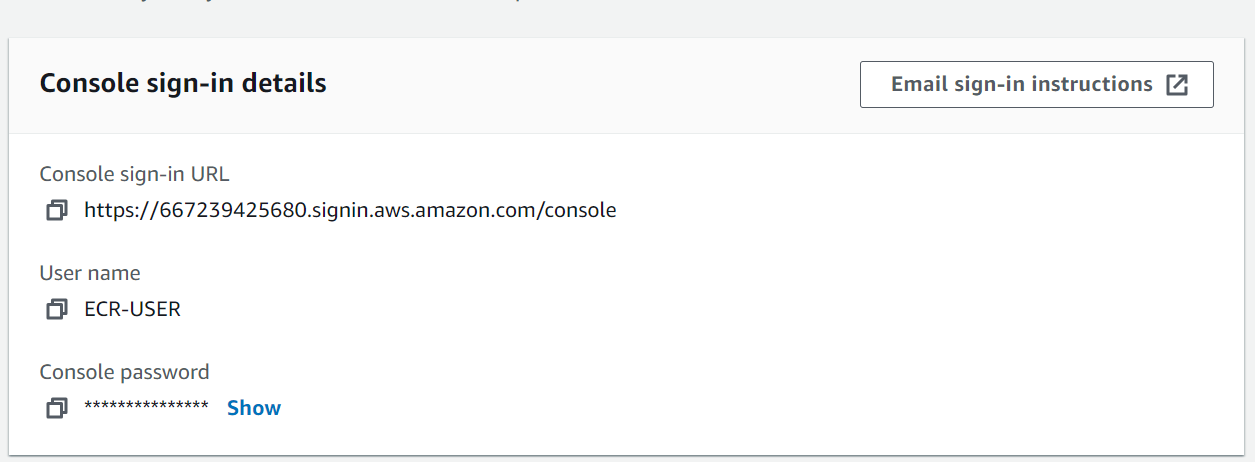
**Step-2**

**Now we need to create a IAM user who has access to ECR**

Create a user and attach below policies to user

****

****

****

[**https://667239425680.signin.aws.amazon.com/console**](https://667239425680.signin.aws.amazon.com/console)

**ACCESSKEY: AKIAZWWURU2IIEVQ2Z7O**

**SECRETKEy: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**

**Paina access key , link use chesi evadanna login avali ani chusthey anthakanna pedda vedava undadu lol**

****

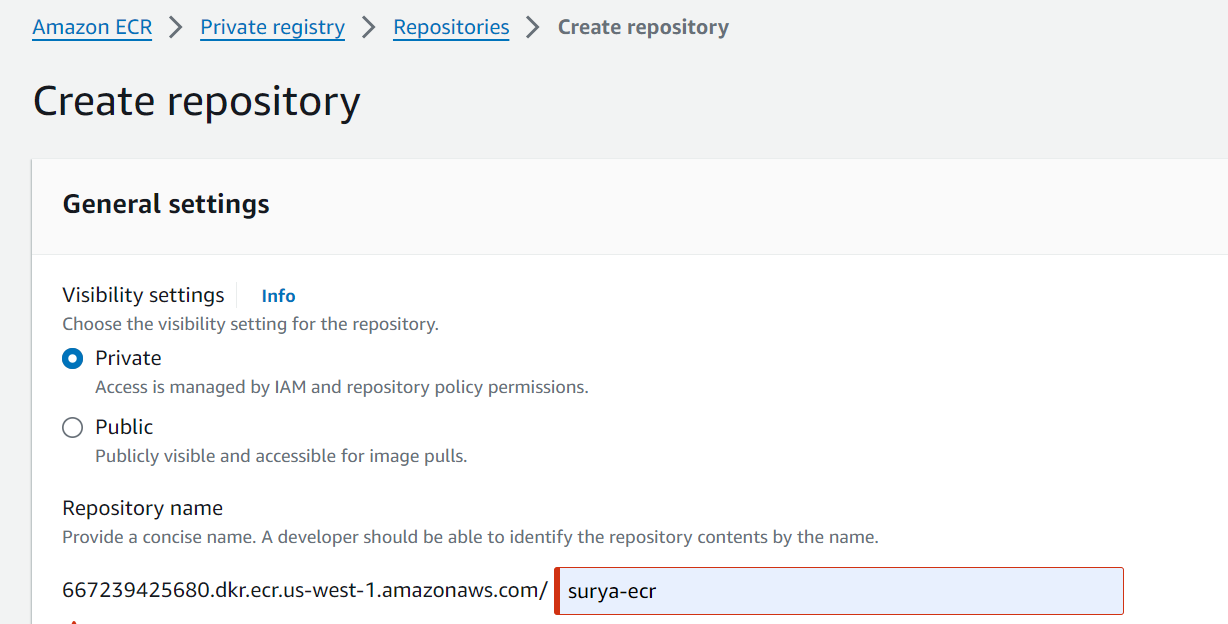
**Create accesskey and secret key which we use in Jenkins**

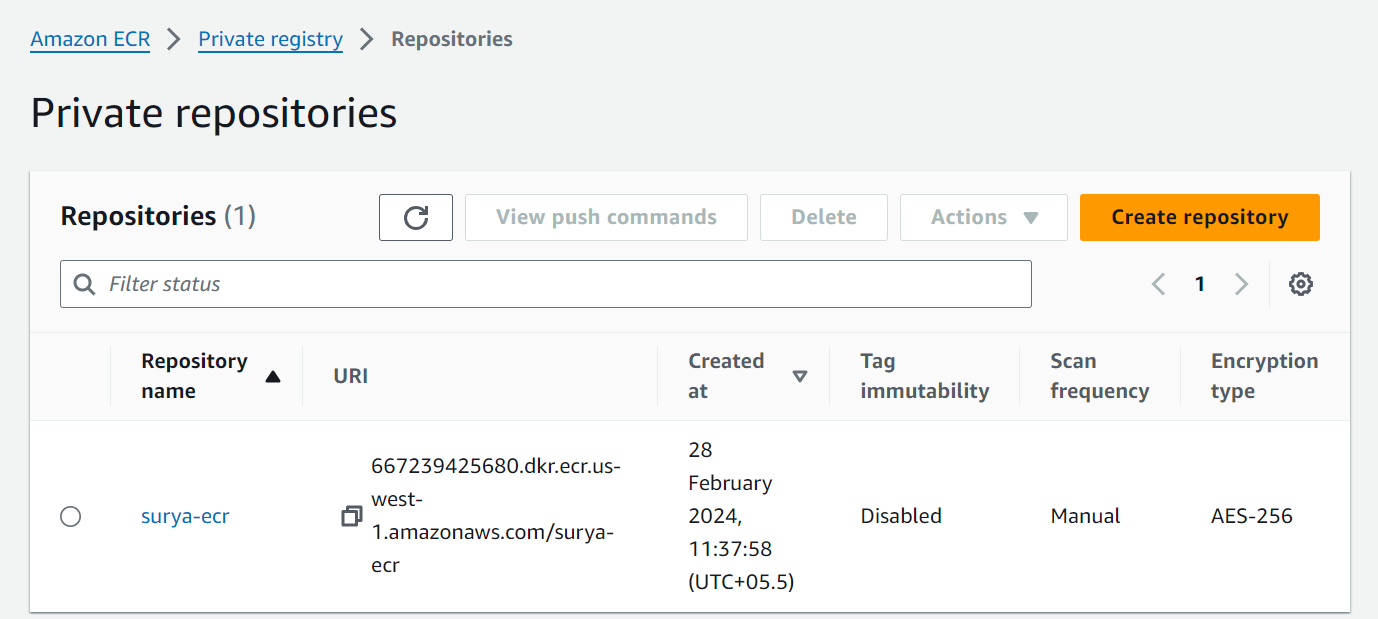
**STEP-3**

**Create a repository in ECR**

**I have created a private repository**

**Provide name of the repo**

****

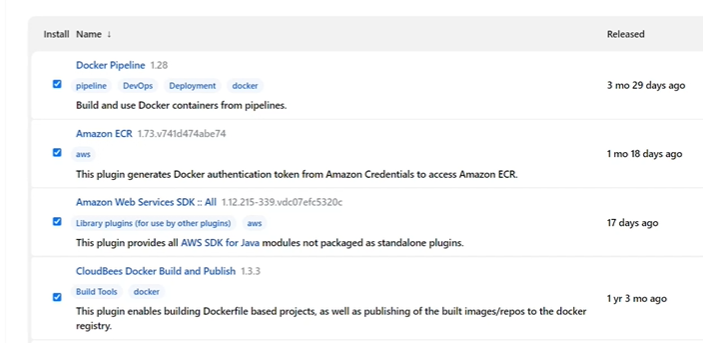
****

URL for private :667239425680.dkr.ecr.us-west-1.amazonaws.com/surya-ecr

**Step-4**

**Now login to jenkins and**

**Install below plugins**

****

****

****

****

**Jenkins file**

pipeline {

agent any

environment {

registryCredential = 'ecr:us-west-1:awscreds'

appRegistry = "667239425680.dkr.ecr.us-west-1.amazonaws.com/surya-ecr"

profileRegistry = "https://667239425680.dkr.ecr.us-west-1.amazonaws.com/surya-ecr"

}

stages {

stage('Fetch code'){

steps {

git branch: 'docker', url: 'https://github.com/devopshydclub/vprofile-project.git'

}

}

stage('Test'){

steps {

sh 'mvn test'

}

}

stage ('CODE ANALYSIS WITH CHECKSTYLE'){

steps {

sh 'mvn checkstyle:checkstyle'

}

post {

success {

echo 'Generated Analysis Result'

}

}

}

stage('Build App Image') {

steps {

script {

dockerImage = docker.build( appRegistry + ":$BUILD\_NUMBER", "./Docker-files/app/multistage/")

}

}

}

stage('Upload App Image') {

steps{

script {

docker.withRegistry( profileRegistry, registryCredential ) {

dockerImage.push("$BUILD\_NUMBER")

dockerImage.push('latest')

}

}

}

}

}

}

**Dockerfile**

FROM openjdk:11 AS BUILD\_IMAGE

RUN apt update && apt install maven -y

RUN git clone https://github.com/devopshydclub/vprofile-project.git

RUN cd vprofile-project && git checkout docker && mvn install

FROM tomcat:9-jre11

RUN rm -rf /usr/local/tomcat/webapps/\*

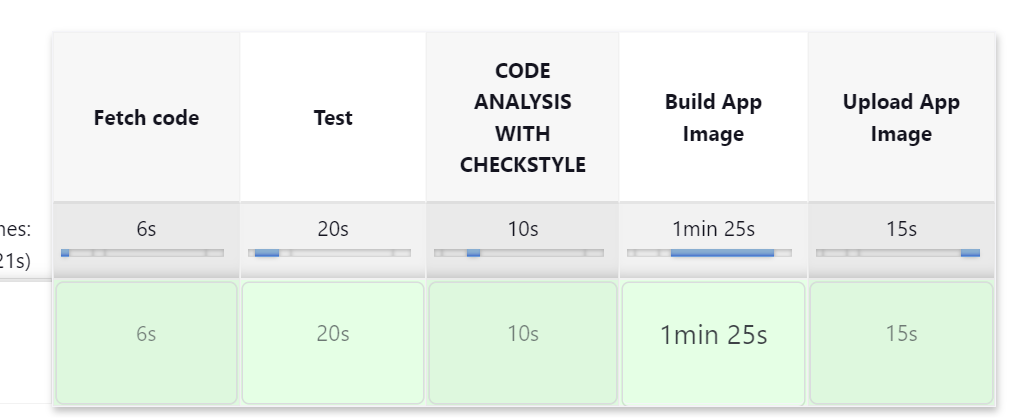
COPY --from=BUILD\_IMAGE vprofile-project/target/vprofile-v2.war /usr/local/tomcat/webapps/ROOT.war

EXPOSE 8080

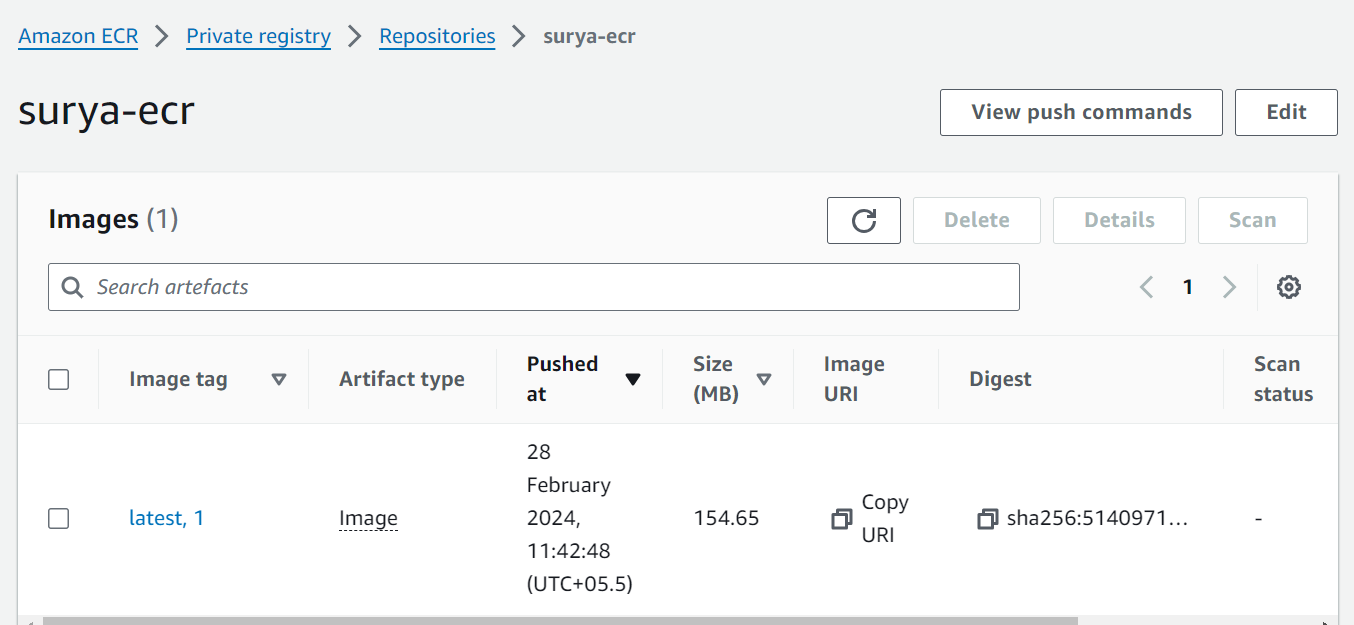
CMD ["catalina.sh", "run"]

**RUN the build**

**If build is successful**

****

**you should be able to see a image in**

****