Push Docker image to AWS ECR using Jenkins pipeline

Prerequisites:

- AWS Account with Admin Privileges
- GitHub Account

Step #1:Configuring EC2 instance in AWS

Go to the AWS dashboard and then to the EC2 services. create an instance

Step #2:Install Java on Ubuntu 22.04 LTS

After the successful SSH connection, firstly update the Linux machine. And install java using below commands:

```
sudo apt update -y
```

Now lets install java 17

```
sudo apt install openjdk-17-jre -y
```

Lets check the version of java

```
java -version
```

```
openjdk version "17.0.12" 2024-07-16
OpenJDK Runtime Environment (build 17.0.12+7-Ubuntu-1ubuntu222.04)
OpenJDK 64-Bit Server VM (build 17.0.12+7-Ubuntu-1ubuntu222.04, mixed mode, sharing)
```

Step #3:Install Jenkins on Ubuntu 22.04 LTS

Lets install jenkins using below commands

```
curl -fsSL https://pkg.jenkins.io/debian/jenkins.io-2023.key | sudo tee \
 /usr/share/keyrings/jenkins-keyring.asc > /dev/null
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
 https://pkg.jenkins.io/debian binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update -y
sudo apt-get install jenkins -y
Jenkins -version
```

Step #4:Enable and start Jenkins on Ubuntu 22.04 LTS

You can enable the Jenkins service to start at boot with the command

```
sudo systemctl enable jenkins
```

You can start the Jenkins service with the command

```
sudo systemctl start jenkins
```

You can check the status of the Jenkins service using the command

```
sudo systemctl status jenkins
```

Step #5:Install git on Ubuntu 22.04 LTS

We need to Install git using below command

```
sudo apt install git
```

Step #6:Access Jenkins on Browser

```
https//:<Instance_ip>:8080
```

After that On the browser, you should see the Jenkins interface that asks for the administrator password.

Now cat the following Jenkins file to retrieve the Administrator password and paste it to the Jenkins dashboard.

Here, create a Jenkins user

After the configuration is completed, you should see the Jenkins dashboard.

Step #7:Add AWS credentials in Jenkins

We may also set up AWS credentials in Jenkins so that it facilitates the Docker push to the ECR repository.

GO to the Manage Jenkins>>Credentials>>system>>Global credentials

Then add credentials and here add AWS username and password and account ID

Step #8:Install Docker on Ubuntu 22.04 LTS

Now here we need to Install Docker

```
sudo apt  install docker.io -Y
```
After Installing Docker we need to give some permission

```
sudo usermod -aG docker $USER
sudo chmod 666 /var/run/docker.sock
```

After installing docker lets Restart jenkins

```
sudo systemctl restart jenkins
```

Step #9:Installing plugins in Jenkins

Go to the **manage Jenkins**>>**Plugins**>>**Available Plugin**

- Docker
- Docker Pipeline
- Amazon ECR plugin

Step #10:Creating ECR Repository in AWS

Lets Create AWS ECR repository to push this image so Go to AWS ECR repository and create

Step #11:Create AmazonEC2ContainerRegistryFullAccess IAM Role in AWS

Here in this step we need to create IAM role with below permission

Attach permission policies :
AmazonEC2ContainerRegistryFullAccess

Ec2 instance - action - security - modify IAM role - add the role you created

Step #12:Install AWS CLI on Ubuntu 22.04 LTS

You can go to the official site of AWS and Install

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
“awscliv2.zip"

sudo apt install unzip
unzip awscliv2.zip
sudo ./aws/install
```

Step #13:Push Docker image to AWS ECR using Jenkins pipeline

So lets create jenkins pipeline go to the Jenkins Dashboard Click on new Item select Pipeline and paste this code

Login to aws ecr -

```
pipeline {
  agent any
  environment {
      AWS_ACCOUNT_ID="954976299686"
      AWS_DEFAULT_REGION="us-east-1"
      IMAGE_REPO_NAME="devops"
      IMAGE_TAG="latest"
      REPOSITORY_URI = "954976299686.dkr.ecr.us-
east-1.amazonaws.com/devops"
  }

  stages {

       stage('Logging into AWS ECR') {
          steps {
              script {
              sh """aws ecr get-login-password --region $
{AWS_DEFAULT_REGION} | docker login --username AWS --password-
stdin ${AWS_ACCOUNT_ID}.dkr.ecr.$
{AWS_DEFAULT_REGION}.amazonaws.com"""
```

```
            }

        }
    }

    stage('Cloning Git') {
        steps {
            checkout scmGit(branches: [[name: '*/main']], extensions: [], 
userRemoteConfigs: [[url: 
‘https://github.com/USER_ID/KubeContainerDeploy-on-EKS.git']])
        }
    }

  // Building Docker images
  stage('Building image') {
    steps{
      script {
        dockerImage = docker.build "${IMAGE_REPO_NAME}:$
{IMAGE_TAG}"
      }
    }
  }

  // Uploading Docker images into AWS ECR
  stage('Pushing to ECR') {
   steps{
       script {
            sh """docker tag ${IMAGE_REPO_NAME}:${IMAGE_TAG} $
{REPOSITORY_URI}:$IMAGE_TAG"""
            sh """docker push ${AWS_ACCOUNT_ID}.dkr.ecr.$
{AWS_DEFAULT_REGION}.amazonaws.com/${IMAGE_REPO_NAME}:$
{IMAGE_TAG}"""
       }
      }
    }
```

```
  }
}
```

Now lets Check ECR Repo our image push or not