CI/CD pipeline

I have booted two machines in aws one is for jenskins and other is for docker.

Jenkins installation:

#java_installation

```
sudo apt-get update
sudo apt install openjdk-8-jdk
```

#Jenkins_installation

```
wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -
sudo apt-add-repository "deb https://pkg.jenkins.io/debian-stable binary/"
sudo apt-add-repository "deb http://pkg.jenkins-ci.org/debian binary/"
sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'
sudo apt-get update
sudo apt install jenkins
```

CI/CD pipeline

Login into jenkins ==> create new job ==> Enter item name ==> select pipeline ==> Go to pipeline script

Following is the pipeline script:

step 1:

```
stage('SCM Checkout'){
git credentialsId: 'git-creds', url: 'https://github.com/javahometech/my-app'
}
```

step 2:

```
stage('Mvn Package'){
def mvnHome = tool name: 'maven-3', type: 'maven'
def mvnCMD = "${mvnHome}/bin/mvn"
sh "${mvnCMD} clean package"
}
```

set the installations of maven with maven-3 in **Snippet Generator**. **Check the generate pipeline script and set as the maven_home.**

Step 3:

```
stage('Build Docker Image'){
sh 'docker build -t ashwinkaliyappan/my-app:2.0.0 .'
}
```

Faced errors:

1.

```
+ docker build -t kammana/my-app:2.0.0 .
/var/lib/jenkins/workspace/docker-app@tmp/durable-5c210828/script.sh: 1:
/var/lib/jenkins/workspace/docker-app@tmp/durable-5c210828/script.sh: docker:
not found
```

To rectify this error: We have install docker in host machine.

2.

```
+ docker build -t kammana/my-app:2.0.0 .
Got permission denied while trying to connect to the Docker daemon socket at
unix:///var/run/docker.sock: Post http://%2Fvar%2Frun%2Fdocker.sock/v1.40/build?
buildargs=%7B%7D&cachefrom=%5B
%5D&cgroupparent=&cpuperiod=0&cpuquota=0&cpusetcpus=&cpusetmems=&cpushares=0&doc
kerfile=Dockerfile&labels=%7B
%7D&memory=0&memswap=0&networkmode=default&rm=1&session=mt1ixoqgqw2n5gzfnfo76hc9
h&shmsize=0&t=kammana%2Fmy-app%3A2.0.0&target=&ulimits=null&version=1: dial unix
/var/run/docker.sock: connect: permission denied
```

To rectify this error: we have to follow in the host machine

```
step 1: sudo groupadd docker
```
step 2: sudo usermod -aG docker $USER

step 3: chmod 777 /var/run/docker.sock

step 4:

```
stage('Push Docker Image'){
    withCredentials([string(credentialsId: 'docker-pwd', variable: 'dockerHubpwd')]) {
      sh "docker login -u ashwinkaliyappan -p ${dockerHubpwd}"
    }
    sh 'docker push ashwinkaliyappan/my-app:2.0.0'
```

For this we have to create a secret text in snippet generator,

1: select withCredentials: Bind credentials to variables

2: Type variable name “dockerHubpwd”

3: ADD jenkins

4: It will open Jenkins Credentials Provider: Jenkins

select secret text and give the password and id.And Add.

5: Run the generate Pipeline sript where we get command line.

Step 5:

```
stage('Run Container on Dev Server'){
    def dockerRun = 'docker run -p 8080:8080 -d --name my-app ashwinkaliyappan/my-app:2.0.0'
    sshagent(['dev-server']) {
      sh "ssh -o StrictHostKeyChecking=no ubuntu@18.205.27.47 ${dockerRun}"
    }
  }
```

For this we have install **ssh agent plugin** in jenkins. Then go to snippet generator.

ADD jenkins, it will lead to **Jenkins Credentials Provider: Jenkins**

Select the kind ==> **ssh username with privatekey and other credentials.**

We have install the docker in the dev-server.

Pipeline Script:

```
node{
  stage('SCM Checkout'){
     git url: 'https://github.com/javahometech/my-app'
  }
  stage('Mvn Package'){
    def mvnHome = tool name: 'maven-3', type: 'maven'
    def mvnCMD = "${mvnHome}/bin/mvn"
    sh "${mvnCMD} clean package"
  }
  stage('Build Docker Image'){
    sh 'docker build -t ashwinkaliyappan/my-app:2.0.0 .'
  }
  stage('Push Docker Image'){
    withCredentials([string(credentialsId: 'docker-pwd', variable: 'dockerHubpwd')]) {
      sh "docker login -u ashwinkaliyappan -p ${dockerHubpwd}"
    }
```

```
      sh 'docker push ashwinkaliyappan/my-app:2.0.0'
    }
    stage('Run Container on Dev Server'){
      def dockerRun = 'docker run -p 8080:8080 -d --name my-app ashwinkaliyappan/my-app:2.0.0'
      sshagent(['dev-server']) {
        sh "ssh -o StrictHostKeyChecking=no ubuntu@18.205.27.47 ${dockerRun}"
      }
    }
}
```