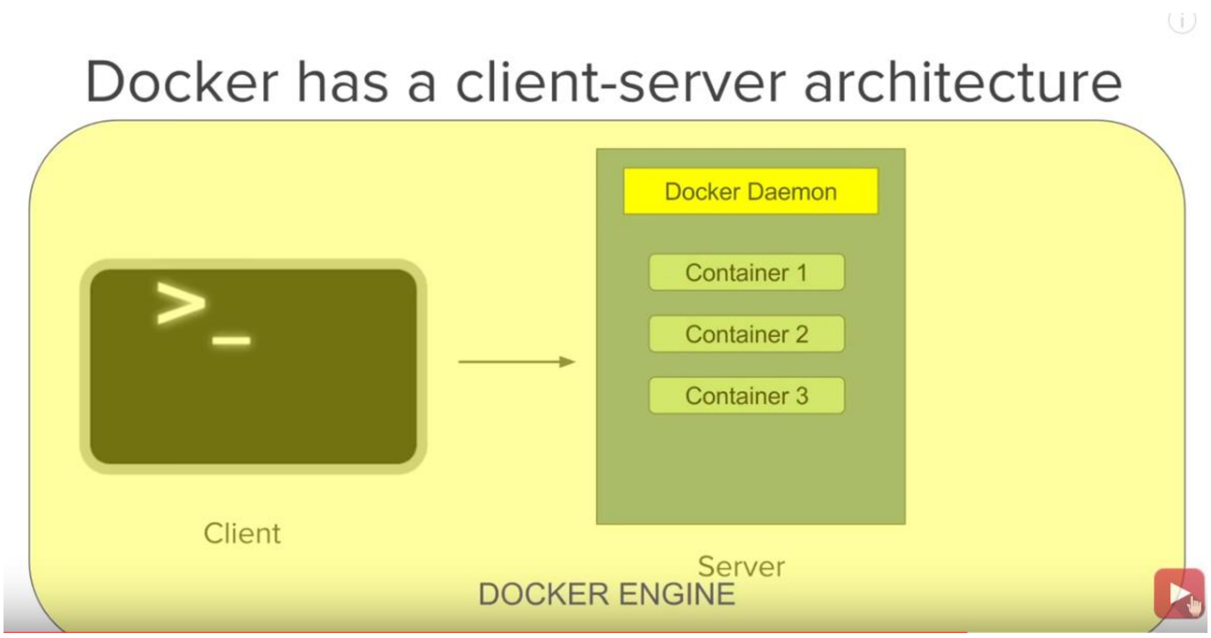
DOCKER

- Docker is a tool designed to make it easier to create, deploy, and run applications by using containers.
- Containers allow a developer to package up an application with all of the parts it
- needs, such as libraries and other dependencies, and ship it all out as one package.
- By doing so, thanks to the container, the developer can rest assured that the
- application will run on any other Linux machine regardless of any customized
- settings that machine might have that could differ from the machine used for writing
- and testing the code.
- hub.docker.com is official public repository for Docker.
- Docker containers are very lightweight.
- On single host we can run multiple containers by using changing host port ip.

Dockerfile vs Docker Image vs Docker Container:

- A Dockerfile is a recipe for creating Docker images
- A Docker image gets built by running a Docker command (which uses that Dockerfile)
- A Docker container is a running instance of a Docker image .

Docker Architecture:



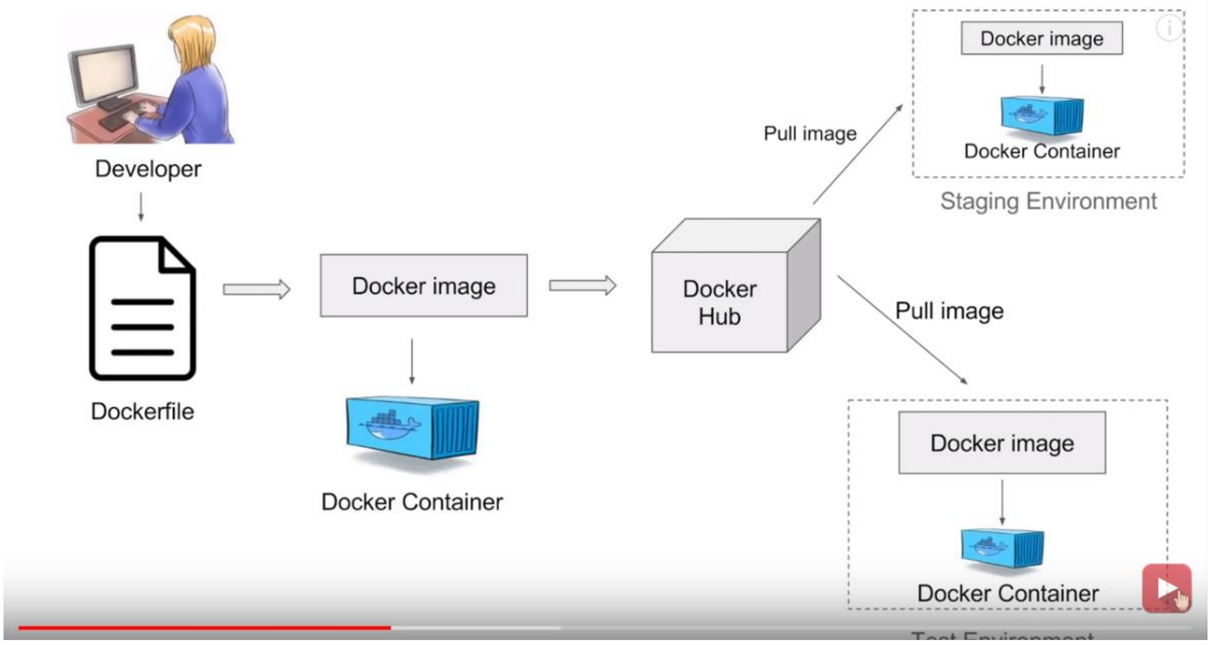
Docker follows client-server architecture. Its architecture consists mainly three parts.

1. Client: Docker provides Command Line Interface (CLI) tools to client to interact with Docker daemon. Client can build, run and stop application. Client can also interact to Docker_Host remotely.

2. Docker_Host: It contains Containers, Images, and Docker daemon. It provides complete environment to execute and run your application.

3. Docker Registry /HUB : It is global repository of images. You can access and use these images to run your application in Docker environment.

Dockerworkflow :



Installing and Configuring DOCKER on Amazon Linux Machine:

1. To run Docker on any Machine (ec2-instance), the Machine must install with Docker as a root.

2. sudo yum install docker -y

3. sudo service docker start

4. sudo chkconfig docker on

5. docker --version

Note:

1. By default Docker Pulls Images from Docker-Hub.

2. Docker Client and Docker Host both can be same machine.

1. Create one user that name is called docker3 (useradd docker3)
2. Assign password to ansible (passwd docker3)
3. edit file vi /etc/ssh/sshd_config to comment out

   Permitrootlogin =yes (remove #)

   PasswordAuthentication = Yes (remove #)
4. Add ansible user under file vi /etc/sudoers (like below)

```
ansible     ALL=(ALL)          NOPASSWD: ALL
```
5. Start ssh service

   Service sshd start
6. Switch user as docker3

   Su – docker3
7. Generate sshkey from master to nodes
   1. ssh-keygen -t rsa (enter 4 steps)

      restart the ssh service

      sudo service sshd restart

DOCKER COMMANDS:

1. sudo docker images → Shows all Available Docker Images

2. sudo docker ps → Shows all Running Containers

3. sudo docker ps -a → Shows all Containers

4. sudo docker ps -aq → Shows all Container Id's

5. docker run -d -p 8888:8080 tomcat:8.0

d = Detached mode or Demon Mode, p = Port Mapping

6. sudo docker start container id → To Start Docker Container

7. sudo docker stop container id → To Stop Docker Container

8. sudo docker rm container id → To Remove Non-Running Docker Container

Note: Before removing the container, it must be in stopped state.

9. sudo docker rm -f container id → To Remove Running Docker Container

Forcefully

10. sudo docker rmi image id → To Remove Docker Image

Note: Docker does not delete Image that used by any container

11. sudo docker rmi -f image id → To Remove Docker Image Forcefully

12. sudo docker run -d -p 2017:8080 ImageId → To Run Docker Container which

Image Present Locally

Ex: 2017 = Host Port, 8080 = Container Port

13. sudo docker exec -it container id /bin/bash → To Enter into Container

exec = Execute, it = Interactive Terminal

14. sudo docker build -t latest:0.1 . → To Build Image from Dockerfile

15. docker login → To Login to Docker Hub Account

16. sudo docker tag centos:latest archanagandla/dev1:two

17 sudo docker push archanagandla/dev1:two

Dockerfile :

A text file with instructions to build image

Automation of Docker Image Creation

Dockerfile name should be Dockerfile (capital D)

Dockerfile example :

Sudo vi Dockerfile

```
FROM docker.io/centos

MAINTAINER admin

RUN yum -y install httpd

RUN echo "Welcome to our homepage created using dockerfile" >
/var/www/html/index.html

EXPOSE 80

CMD ["/usr/sbin/httpd", "-D", "FOREGROUND"]
```

We need to write all Dockerfiles in one single location.

That is create one directory like mkdir /opt/docker

Cd /opt/docker and write all Dockerfiles here and then build docker images.

Jenkins Integrate with Docker:

Step 1: First we need to take one ec2-instance for Jenkins

Step 2: Login into Jenkins do below steps

1. Create one user that name is called ansible (useradd docker3)
2. Assign password to ansible (passwd docker3)
3. edit file vi /etc/ssh/sshd_config to comment out

   Permitrootlogin =yes (remove #)

   PasswordAuthentication = Yes (remove #)
4. Add ansible user under file vi /etc/sudoers (like below)

   Docker3   ALL=(ALL)   NOPASSWD: ALL
5. Start ssh service

   Service sshd start
6. Switch user as ansible

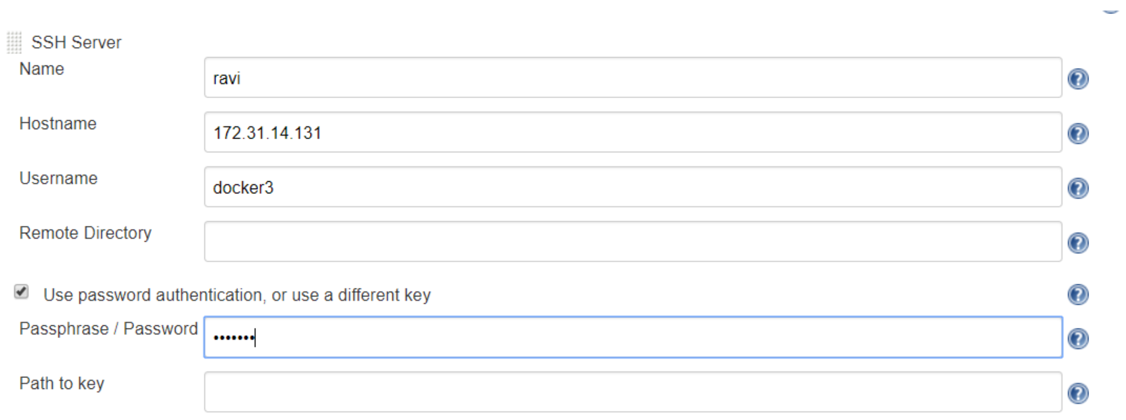
   Su – docker3
7. Generate sshkey from master to nodes
   1. ssh-keygen -t rsa (enter 4 steps)

      restart the ssh service

      sudo service sshd restart
   2. ssh-copyid docker IP

Step2: First you need to install **PublishOverssh** plugin in Jenkins

Step 3: In Jenkins go to manage Jenkins and after that click on configure system and provide ACM ansible user credentials   like below.


SSH Server

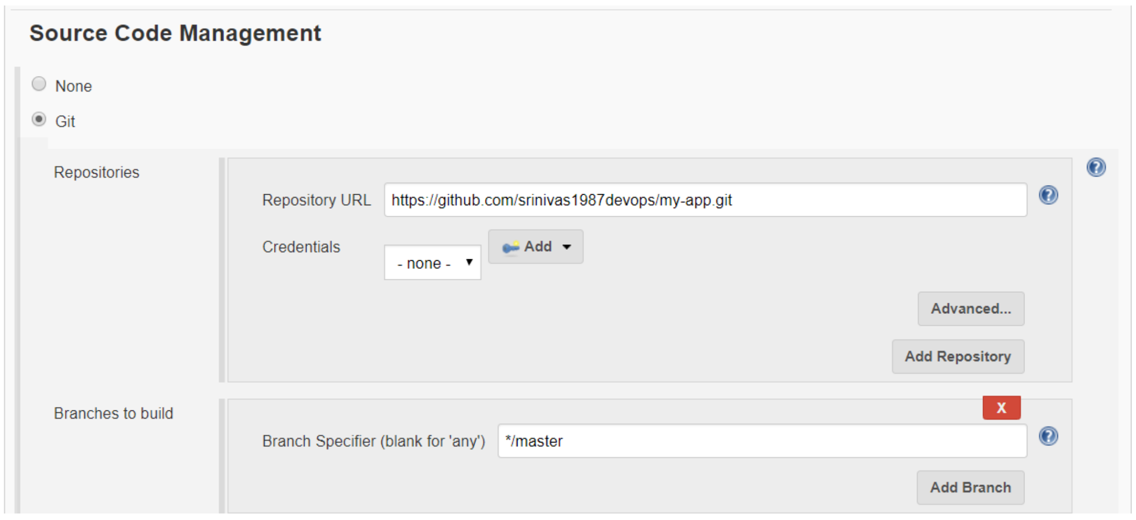
Name	ravi
Hostname	172.31.14.131
Username	docker3
Remote Directory	

☑ Use password authentication, or use a different key

Passphrase / Password	•••••••
Path to key	

Step 3: Create a Jenkins job under Freestyle like below steps

1.Under SCM section


Source Code Management

None
[✓ Git]

Repositories

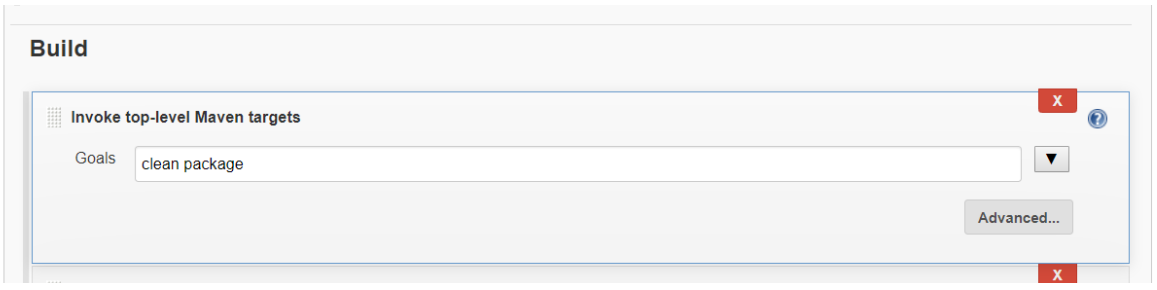
Repository URL: https://github.com/srinivas1987devops/my-app.git

Credentials: - none -

Branches to build

Branch Specifier (blank for 'any'): */master

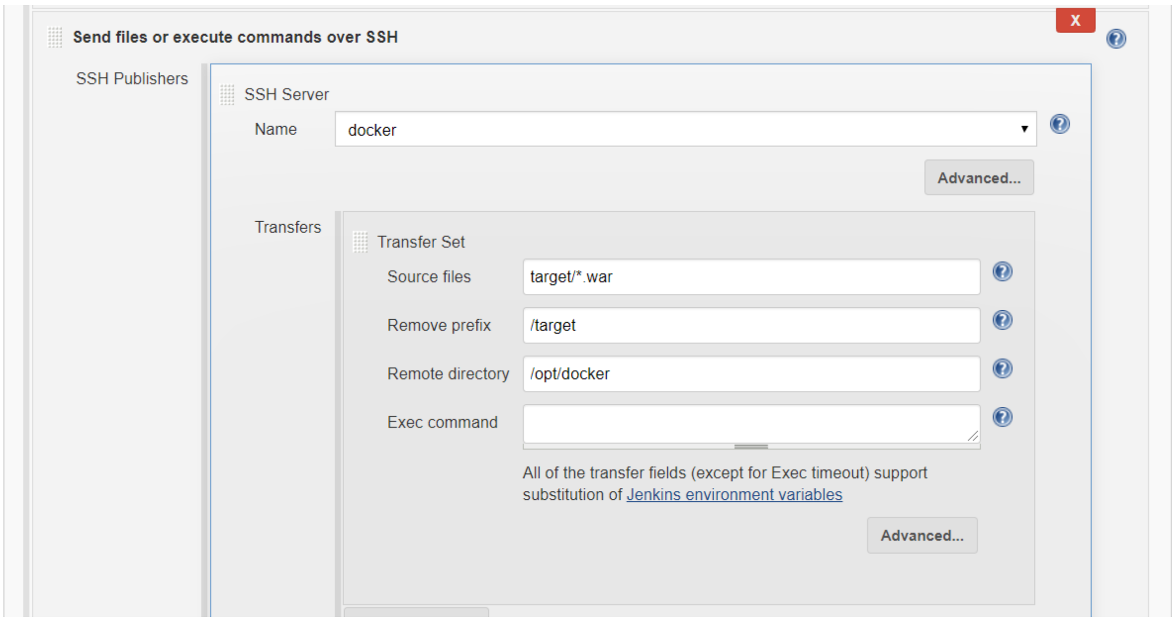
2.Under Build section


Build

Invoke top-level Maven targets

Goals: clean package

3.One more step add in Build section select send files or execute commands over ssh


Send files or execute commands over SSH

SSH Publishers

SSH Server

Name: docker

Transfers

Transfer Set

Source files: target/*.war

Remove prefix: /target

Remote directory: /opt/docker

Exec command:

All of the transfer fields (except for Exec timeout) support substitution of Jenkins environment variables

Step4 : Finally Build the Jenkins Job.

Step 5: After completion Jenkins job warfile will copied to /opt/docker

Note :This /opt/docker directory will created under docker3 user home directory

That is /home/docker3

Step 6: Go to /home/docker3 and create Docker file like below

Sudo vi Dockerfile

FROM tomcat:9

# Take the war and copy to webapps of tomcat

COPY /opt/docker/myweb-8.2.0.war /usr/local/tomcat/webapps/myweb.war

Step 7: Follow below steps

1. sudo docker build -t tomcat:1011 .
2. sudo docker images
3. sudo docker run -itd -p 2001:8080 d809f4f48185
4. sudo docker ps
5. Take public IP of docker:containerport/myweb
6. Then your application deployment is success.