Setup project :

Technology

1. Setup java and maven home on machine.

2. Jenkins - Install and configure jenkins,

reset the admin password.

3. Install git in mahcine

4. Install docker on machine

4. Sonar-dashboard

5. jfrog artificatory

## Step 1. Install java on machine

Installting PPA,

because we are installing java from PPA repository.

$ sudo apt install -y python-software-properties --

-- adding java repository

$ sudo add-apt-repository ppa:webupd8team/java

$ sudo apt update

-- Command to install java

$ sudo apt install -y oracle-java8-installer

## Verified java version

ubuntu@ip-172-31-10-173:~$ java -version

java version "1.8.0\_171"

Java(TM) SE Runtime Environment (build 1.8.0\_171-b11)

Java HotSpot(TM) 64-Bit Server VM (build 25.171-b11, mixed mode)

## Step 2. Download and configure Maven

$ cd /usr/local/src

$ sudo wget

http://www-us.apache.org/dist/maven/maven-3/3.5.4/binaries/apache-maven-3.5.4-bin.tar.gz

sudo tar -xf apache-maven-3.5.4-bin.tar.gz

mv apache-maven-3.5.4/ apache-maven/

sudo rm -rf apache-maven-3.5.4-bin.tar.gz

## Configure Apache-Maven Environment

Go to the '/etc/profile.d' directory and create a new configuration file 'maven.sh'.

cd /etc/profile.d/

vim maven.sh

# Apache Maven Environment Variables

# MAVEN\_HOME for Maven 1 - M2\_HOME for Maven 2

export JAVA\_HOME=/usr/lib/jvm/java-8-oracle

export M2\_HOME=/usr/local/src/apache-maven

export MAVEN\_HOME=/usr/local/src/apache-maven

export PATH=${M2\_HOME}/bin:${PATH}

## Save the file and make it executable.

chmod +x maven.sh

source maven.sh

## Verify the maven version

mvn --version

Step 3. Installting Jenkins ###

Add the repository key to the system.

wget -q -O - https://pkg.jenkins.io/debian/jenkins-ci.org.key | sudo apt-key add - -- Add jenkins repository

echo deb https://pkg.jenkins.io/debian-stable binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list -- Adding the jenkins repo to repo source.list file

sudo apt-get update -- Update the system

apt-get install jenkins -y -- Now install jenkns

## Now start jenkins ##

/etc/init.d/jenkins start |status

OR

service jenkins start | stop | status

service jenkins status

### Install docker on machine ###

apt install docker.io

service docker status

Adding user to docker group & need to restart the service:

sudo usermod -aG docker jenkins -->

# service docker stop

# service docker start

## Now access jenkins through IP and Port in browser.

http://ec2-52-221-235-166.ap-southeast-1.compute.amazonaws.com:8080/

Now put the admin password ,

cat /var/lib/jenkins/secrets/initialAdminPassword -- >

copy the password and paste it on jenkins for first time.

## Now install the suggested plug-in -- >

Save and Finish

cd /var/log/jenkins

cat jenkins.log-> to see installed plugins

## Now Create first admin user:

username id password.

Step 4. Install Git on machine .

apt-get install git

## Run docker container for jfrog ##

root@ip-172-31-10-173:~#

docker run --name artifactory -d -p 8081:8081 docker.bintray.io/jfrog/artifactory-pro:latest

$ docker ps

### Install sonarqube ###

$ docker run -d --name sonarqube -p 9000:9000 -p 9092:9092 sonarqube -- This command will run sonarqube in docker container

$ docker ps

To access Sonar:

----------------

Login into port 9000 of ec2 machine:

user: admin

passwd: admin

To access Artifactory:

-----------------------

Allow port 8081 in your SG of Ec2 mahicne.

# Access your instance on browser with port 8081,

Register for free trial.

# Copy your key for free trail and paste it to register.

# add password for jfrog repository.

user: admin

passwd: admin@123

Create a Jenkins Job:

---------------------

Goto Global Tool COnfiguration:

-------------------------------

Add JDK --> JDK name and add Java Home.

Add maven--> Maven Home

Save the file.

Install the Plugin:

-------------------

Pipeline Plugin:

-----------------

Default Pipeline plugin to verify.

Creating a JOB:

---------------

Enter item name

select pipeline project

Copy and Paste groovy scripts into Pipeline terminals in Build Triggers column.

$ docker inspect sonarid- To find the Ip of sonar= 172.17.0.1

$ docker inspect jfrog- To find the Ip of jfrog.172.17.0.2

Update the Workspace name in groovy script file.

copy and add to pipeline script.

Click Build Now.

Update the IP of sonar and Artifact in groovy scripts.

Groovy Script:

#!groovy

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\* Description :: This template is used to setup sample pipeline \*\*\*\*\*

\*\*\*\*\* Revision :: 1.0 \*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

import hudson.model.\*

import hudson.EnvVars

import groovy.json.JsonSlurperClassic

import groovy.json.JsonBuilder

import groovy.json.JsonOutput

import java.net.URL

node {

def GIT\_URL = ''

def GIT\_BRANCH = 'master'

def SONAR\_BRANCH = 'master'

def MAVEN\_GOALS = 'clean install -X'

def MAVEN\_HOME = tool 'M2\_HOME'

def JAVA\_HOME = tool 'java'

def SONAR\_URL = 'http://172.17.0.2:9000'

def SONAR\_LOGIN='admin'

def SONAR\_PASSWORD='admin'

def artifactory\_user='admin'

def artifactory\_password='admin'

def dversion='$TAG'

env.PATH = "${JAVA\_HOME}/bin:${MAVEN\_HOME}/bin:${env.PATH}"

properties([buildDiscarder(logRotator(artifactDaysToKeepStr: '5', artifactNumToKeepStr: '5', daysToKeepStr: '5', numToKeepStr: '5')), pipelineTriggers([])])

properties([parameters([string(defaultValue: 'v1', description: 'Please choose version for tomcat docker image creation; default value is v1', name: 'TAG')]), pipelineTriggers([])])

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Checkout code from GIT

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

stage('\u2780 Checkout Code from GIT')

{

echo "INFO => Checking out from URL: ${GIT\_URL} and BRANCH: ${GIT\_BRANCH}"

checkout([$class: 'GitSCM', branches: [[name: "\*/${GIT\_BRANCH}"]], doGenerateSubmoduleConfigurations: false, extensions: [[$class: 'CloneOption', noTags: false, reference: '', shallow: true]], submoduleCfg: [], userRemoteConfigs: [[credentialsId: 'git-credentials', url: "${GIT\_URL}"]]])

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Build stage to trigger the maven build

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

stage('\u2781 Compile and execute Unit test')

{

def ARTIFACT\_VERSION = getVersion()

if (ARTIFACT\_VERSION)

{

echo "INFO => Building Version: ${ARTIFACT\_VERSION}"

}

echo "INFO => MAVEN\_HOME: ${MAVEN\_HOME}"

echo "INFO => JAVA\_HOME : ${JAVA\_HOME}"

sh "mvn -B ${MAVEN\_GOALS} -DskipTests"

}

stage('\u2782 Sonar Analysis and coverage')

{

echo "INFO => Running Sonar Analysis"

sh "mvn -DBranch=${SONAR\_BRANCH} -e -B sonar:sonar -Dsonar.host.url=${SONAR\_URL} -Dsonar.login=${SONAR\_LOGIN} -Dsonar.password=${SONAR\_PASSWORD} -Dsonar.scm.disabled=true -X"

}

stage('\u2783 Upload package on artifactory')

{

def ARTIFACT\_VERSION = getVersion()

echo ARTIFACT\_VERSION

String a = ARTIFACT\_VERSION.replaceAll("-SNAPSHOT","")

/\*\*\* sh 'b=`$ARTIFACT\_VERSION| cut -f1 -d - `; echo $b' \*\*\*/

echo "INFO => Deploying package to artifactory"

sh "curl -u $artifactory\_user:$artifactory\_password -T $WORKSPACE/target/hello-world-war-${ARTIFACT\_VERSION}.war http://172.17.0.3:8081/artifactory/generic-local/hello-world-war-${ARTIFACT\_VERSION}.war"

}

stage('\u2784 Build Docker Image')

{

def ARTIFACT\_VERSION = getVersion()

echo ARTIFACT\_VERSION

String a = ARTIFACT\_VERSION.replaceAll("-SNAPSHOT","")

sh '''cd /var/lib/jenkins/workspace/project2/docker

cp $WORKSPACE/target/hello-world-war-3.0.0.war .

mv hello-world-war-3.0.0.war demo.war

docker build -t roshan/tomcat:$TAG .'''

}

}

def getVersion()

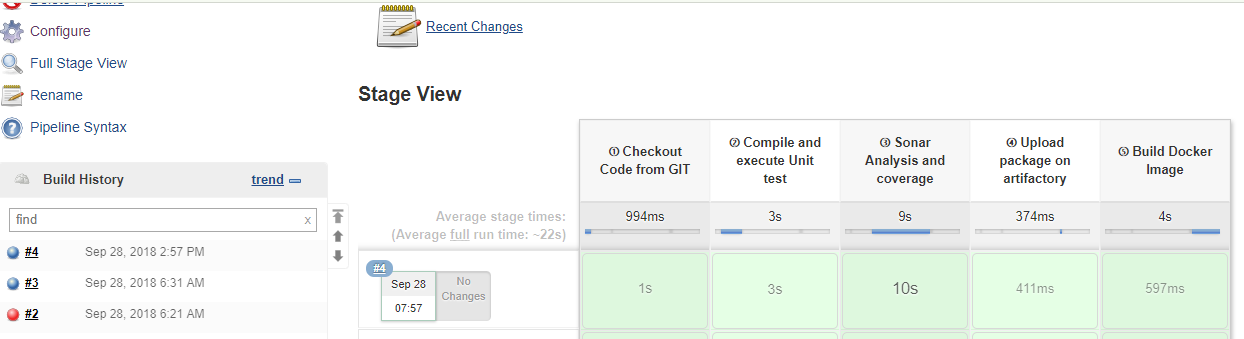
{

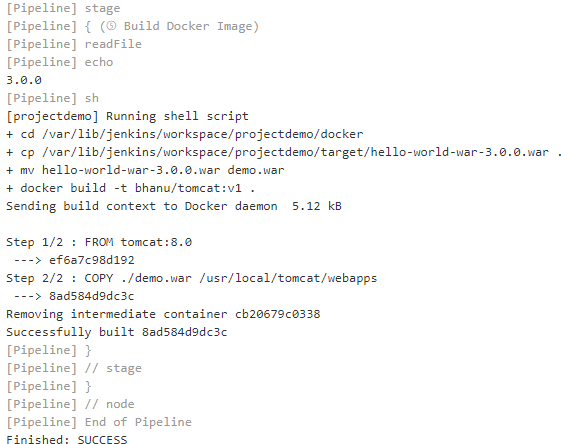
def matcher = readFile("pom.xml") =~ '<version>(.+)</version>'

matcher ? matcher[0][1] : null

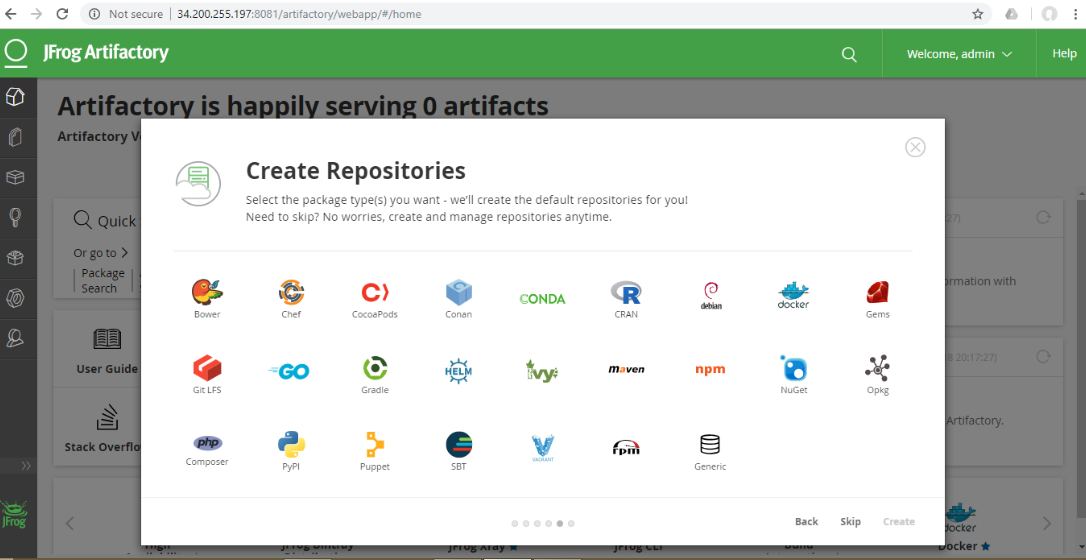
}

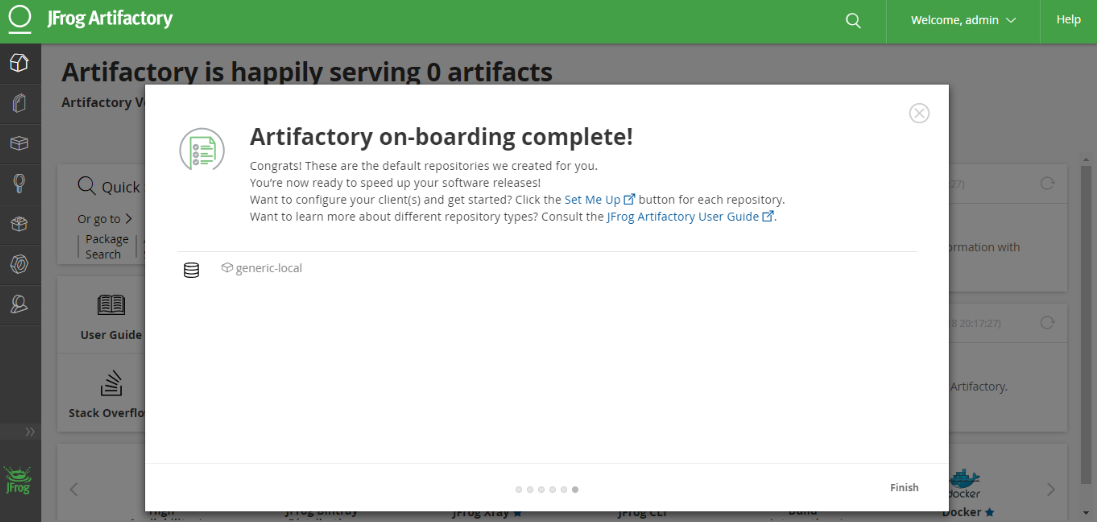
Output:

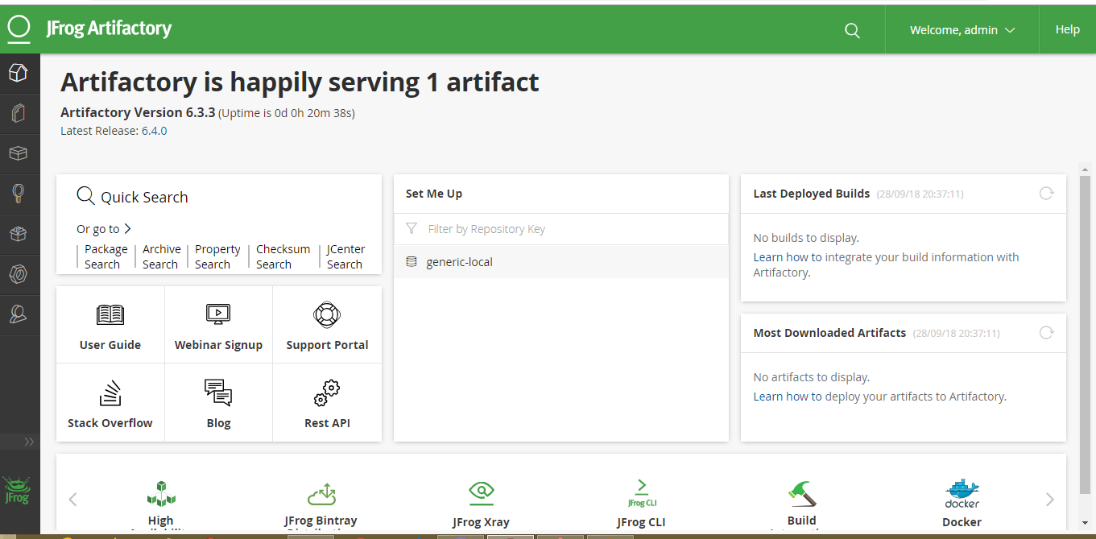


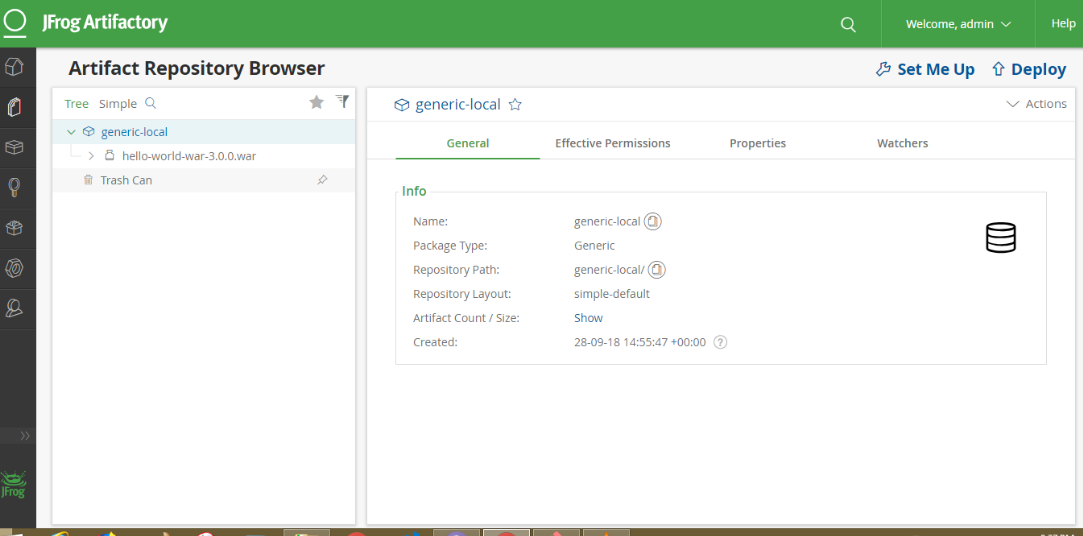


**Creating Generic Repo under trail license:**

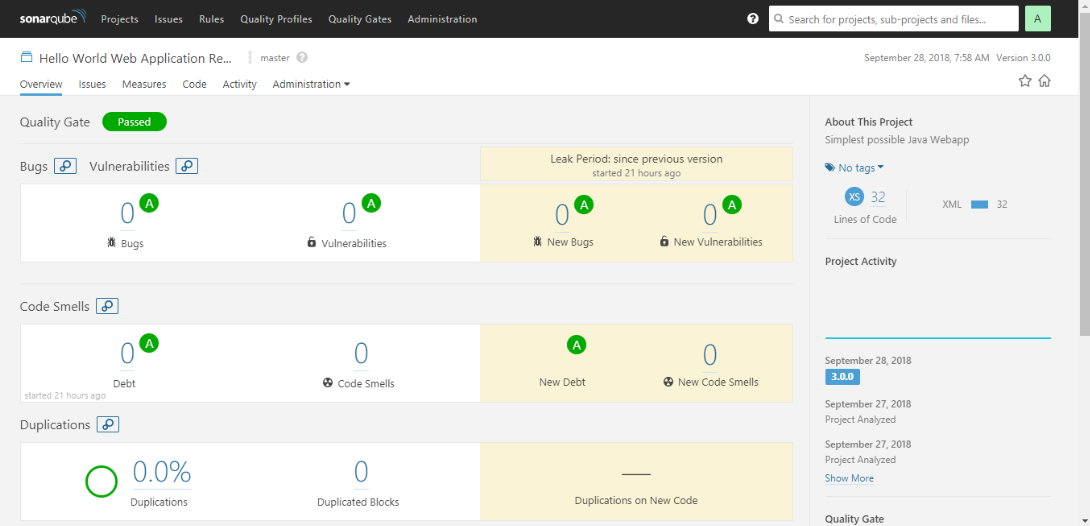


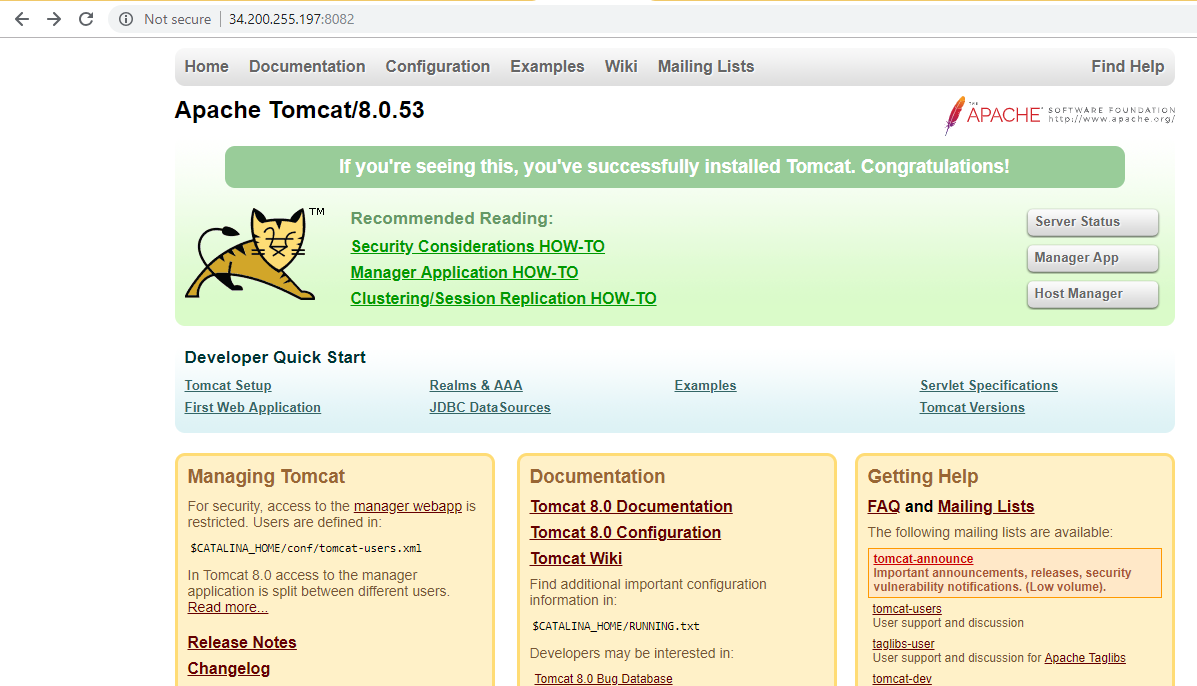






Sonar cube Output:





Deployment to Tomcat:

