IMPLEMENTATION OF CI/CD PIPELINE PROJECT WITH HELP OF AWS, GIT, JENKINS, MAVEN, SONARQUBE, DOCKER AND NEXUS

ABSTRACT

Few years back when agile methodology was playing a major role in the industry, software was deployed in monthly, quarterly or annual basis which was time consuming. But now it’s DevOps era! Where software can be deployed multiple times a day. In current era, delivering creative ideas in a rapid and steady manner is eminently significant for all organizations. In addition to that, organizations need to react to vigorous market requirements, faster time to market, decrease in failure rate and increase in customer interaction. This could be achieved with the help of DevOps methodology. DevOps methodology extends the agile to quickly produce software and automatically deploy them across various platforms/environment in order to gain high performance and quality assurance products. Continuous integration/Continuous deployment (CI/CD) is the backbone of DevOps environment. By automating the build, testing and deployment of software, CI/CD bridges the gap between development and operation teams. In this project, the source code(java application) will be deployed using AWS cloud service, Git, Maven, Sonarqube, Jenkins, Docker, Nexus in order to automate the entire environment.

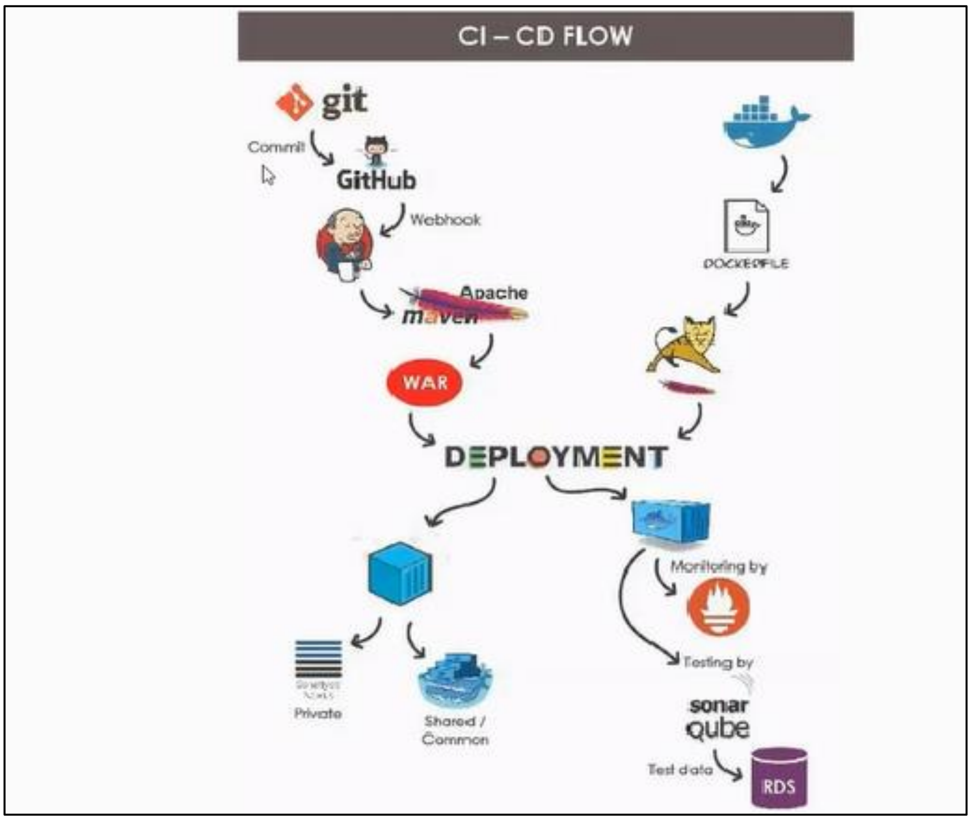
INTRODUCTION

Due to increasing competition in software industry, organizations play a major in assigning required resources to develop and deliver trustworthy and high quality products to consumers. Consumers expect to have continuous interaction with DevOps team so that they can provide their continuous feedback. DevOps is blending of two terms development and operations which aims to provide conjoin approach to industry‟s software development and operation team job in software development lifecycle. It provides a good communication between these two teams. DevOps describes the conformation of automation and programmable software development and infrastructure deployment and maintenance. Continuous integration, continuous deployment and continuous delivery are the important factors in software industry that helps organizations to constantly release new attributes and products that are trustworthy. Continuous integration focuses on integrating each developers work multiple times per day so that debugging of error is easy. Continuous delivery focuses on demoting discordance in deployment or release process and automating the build step so that code can be released securely at any time. CI/CD pipeline provides following benefits in software delivery lifecycle: obtaining rapid feedback from customers, rapid and steady release leads to have customer satisfaction and quality assured product, CD helps to automate tasks which was carried out manually.

PROPOSED METHODOLOGY:

In this project, we copy the source code from local machine to ec2 server using winscp. Using git, we commit and push the source code to the public repository called github. Jenkins automatically triggers the source code from git hub by integrating Jenkins and git hub. Maven triggers the source code from Jenkins and converts it as a war file by integrating maven and jenkins. Sonarqube automatically triggers the war file of source code for quality testing by integrating jenkins and Sonarqube. Tomcat install in docker to deploy the war file on top of it. war file converted as docker image and push to the docker hub and also private repository called nexus.

PROJECT ARCHITECTURE:



TOOLS USED IN THIS PROJECT:

Git:

Git is a version control tool used to push the code into remote repository i.e., Github.com during software development lifecycle. It is also used to monitor changes in file sets. Developers push their code to repository created in Github.com using git commands. Initially install git into the server using sudo yum install git -y command.

Maven:

Maven is project management and comprehension tool which provides complete build lifecycle framework for developers. Maven is based on Project Object Tool (POM) file. POM is used for project builds, dependency and documentation. POM is a XML file that is present in the base directory of project as pom.xml. POM file contains all the necessary information and configuration details of the project.

SonarQube:

SonarQube is an open source platform developed by sonar source for continuous inspection of code quality to perform automatic reviews with static analysis of code to detect bugs. SonarQube offers reports on duplicated code, coding standards, unit tests, code coverage, code complexity, comments, bugs and security recommendations. SonarQube provides fully automated analysis and can be integrated with building tool like maven and continuous integration tool like Jenkins.

Jenkins:

Continuous integration (CI) process is carried out using Jenkins tool. Jenkins is an open source automation server helps to automate manual work of software development related to building, testing, and deploying, facilitating continuous integration and continuous delivery. It is a server-based system that runs in serverlet containers such as Apache Tomcat.

Docker:

Docker is a containerization platform that is used to create a package containing an application and all its dependencies altogether in the form of a docker container to make sure that the application works perfectly in all environments. Docker container is a standardized unit which is created on the fly to deploy a specific application or environment. Consider a scenario where code running in one machine is not running in another machine. This is due to environmental change. To overcome this problem, Docker is used. Docker image is created.

Nexus:

Nexus Repository is an open source repository that supports many artifact formats, including Docker, Java™, and npm. With the Nexus tool integration, pipelines in your toolchain can publish and retrieve versioned apps and their dependencies by using central repositories that are accessible from other environments.

AWS EC2 Sever:

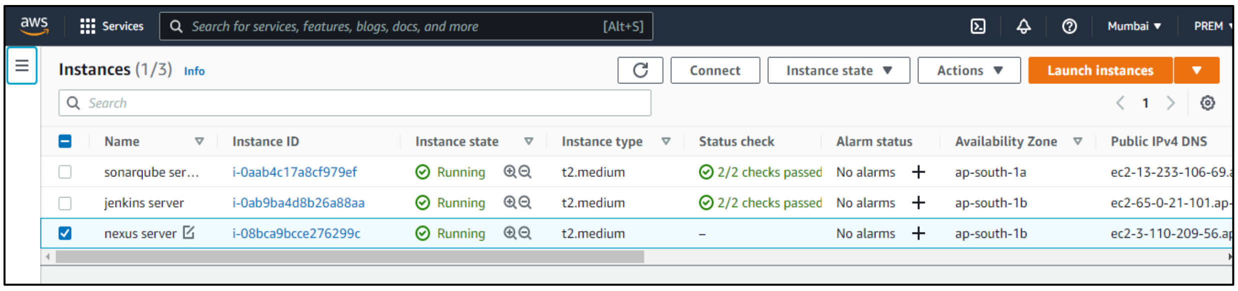
Elastic Compute Cloud is a virtual machine that represents a physical server for you to deploy your application. Ec2 allows users to build apps to automate scaling according to changing needs and peak periods, and makes it simple to deploy virtual servers and manage storage, lessening the need to invest in hardware and helping streamline development processes.

RDS:

Amazon Relational Database Service (RDS) is a collection of managed services that makes it simple to set up, operate, and scale databases in the cloud. Amazon RDS supports an array of database engines to store and organize data. It also helps with relational database management tasks, such as data migration, backup, recovery and patching.

WORKING PROCEDURE:

Create and launch three ec2 servers on any region (Mumbai) and go as root user. Install and start Git, Jenkins, maven, Docker on to the Jenkins server. Install and start SonarQube on Sonarqube server and install and start nexus on nexus server.



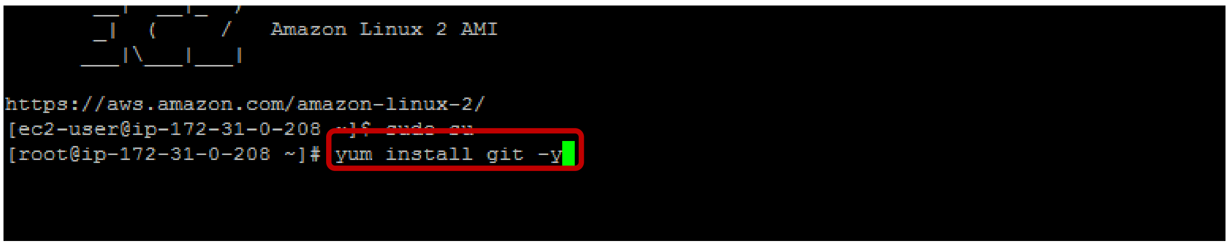
Instances (1/3) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
sonarqube ser...	i-0aab4c17a8cf979ef	Running	t2.medium	2/2 checks passed	No alarms	ap-south-1a	ec2-13-233-106-69.
jenkins server	i-0ab9ba4d8b26a88aa	Running	t2.medium	2/2 checks passed	No alarms	ap-south-1b	ec2-65-0-21-101.ap-
nexus server	i-08bca9bcce276299c	Running	t2.medium	–	No alarms	ap-south-1b	ec2-3-110-209-56.a

GIT INSTALLATION:

# Switch to root user
# Git installs on the jenkins server by following command.

- ➢ yum install git –y



```
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-0-208 ~]$ sudo su
[root@ip-172-31-0-208 ~]# yum install git -y
```

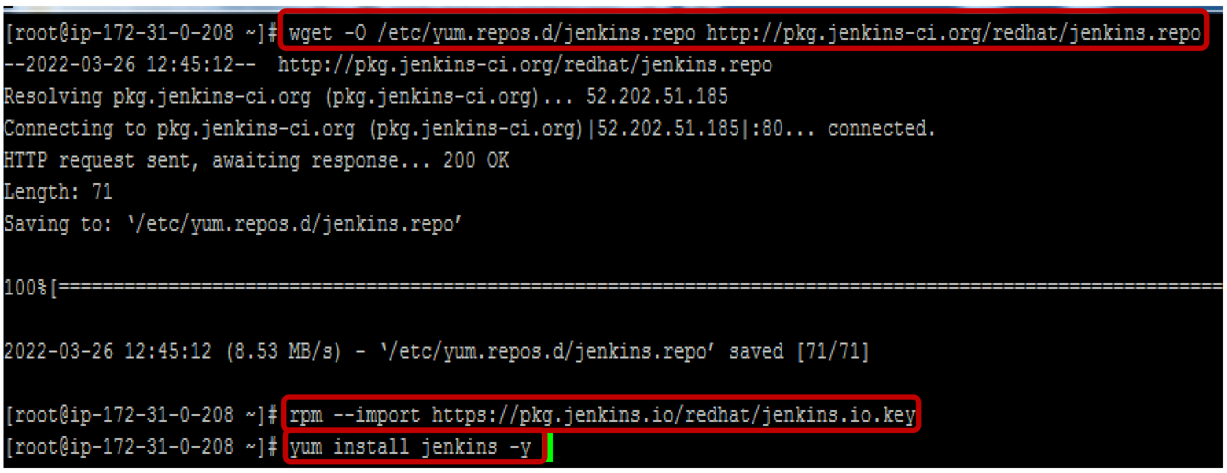
INSTALLATION OF JENKINS:

Prerequisites of Installation Jenkins:

# 256 MB of RAM is required.
# 1 GB of drive space is required(although 10 GB is a recommended minimum if running Jenkins as a Docker container.
# JDK should install before installing the jenkins.

# Jenkins installs on the jenkins ec2 server by following commands.
# Switch to root user.

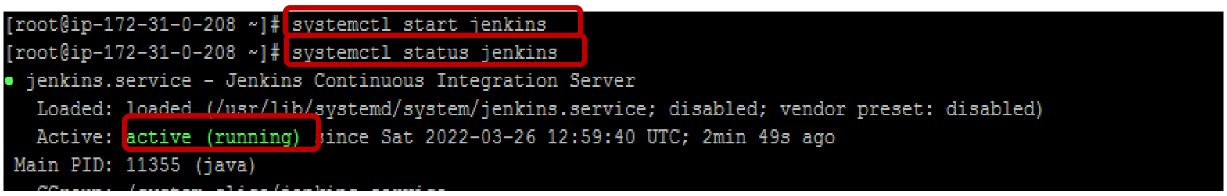
- amazon-linux-extras install epel –y
- yum update –y
- wget -O /etc/yum.repos.d/jenkins.repo \https://pkg.jenkins.io/redhat-stable/Jenkins.repo
- rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
- amazon-linux-extras install java-openjdk11
- yum install jenkins -y



```
[root@ip-172-31-0-208 ~]# wget -O /etc/yum.repos.d/jenkins.repo http://pkg.jenkins-ci.org/redhat/jenkins.repo
--2022-03-26 12:45:12--  http://pkg.jenkins-ci.org/redhat/jenkins.repo
Resolving pkg.jenkins-ci.org (pkg.jenkins-ci.org)... 52.202.51.185
Connecting to pkg.jenkins-ci.org (pkg.jenkins-ci.org)|52.202.51.185|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 71
Saving to: '/etc/yum.repos.d/jenkins.repo'

100%[==========================================================================================

2022-03-26 12:45:12 (8.53 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [71/71]

[root@ip-172-31-0-208 ~]# rpm --import https://pkg.jenkins.io/redhat/jenkins.io.key
[root@ip-172-31-0-208 ~]# yum install jenkins -y
```

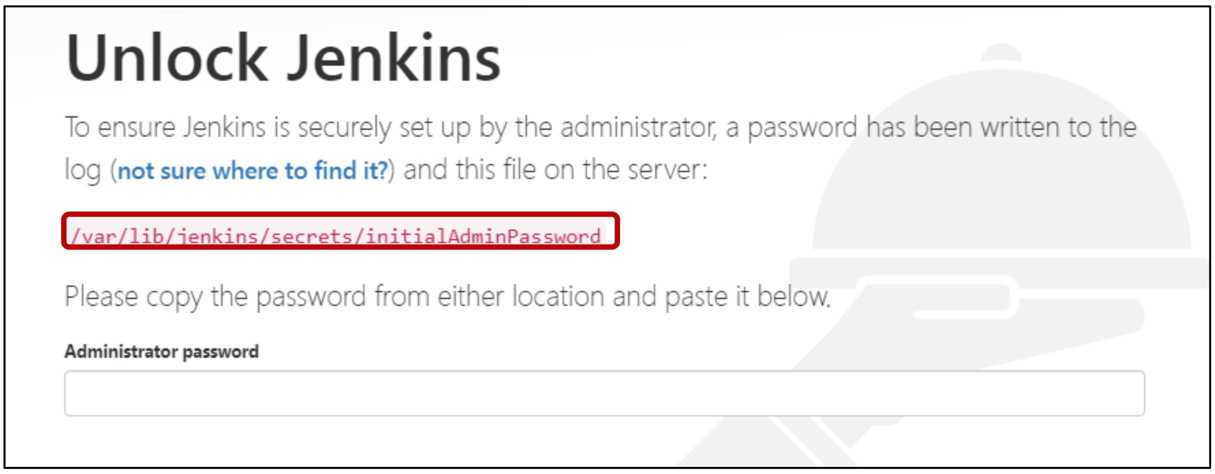
# Jenkins Start and know the status of jenkins by following commands.

- systemctl start jenkins
- systemctl status jenkins



```
[root@ip-172-31-0-208 ~]# systemctl start jenkins
[root@ip-172-31-0-208 ~]# systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; disabled; vendor preset: disabled)
   Active: active (running) since Sat 2022-03-26 12:59:40 UTC; 2min 49s ago
 Main PID: 11355 (java)
```

# Hit the browser by public ip address of jenkins server with 8080(jenkins port number) to open jenkins console.



Unlock Jenkins

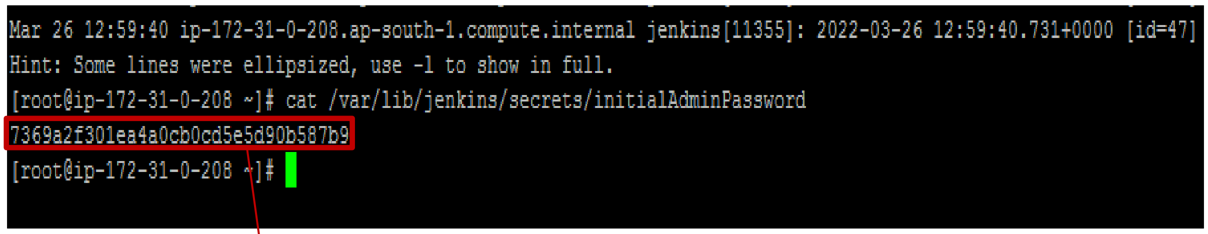
To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

/var/lib/jenkins/secrets/initialAdminPassword

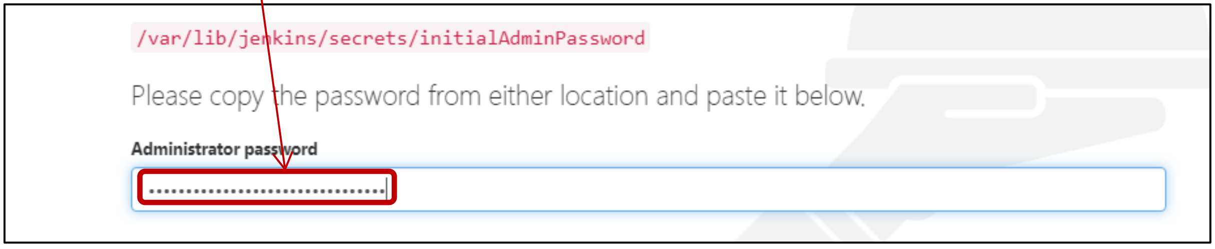
Please copy the password from either location and paste it below.

Administrator password

# Get the Administrator password from jenkins servers /var/lib/jenkins/secrets/initialAdminPassword path and paste it onto jenkins consoles Administrator password place to unlock the jenkins console.



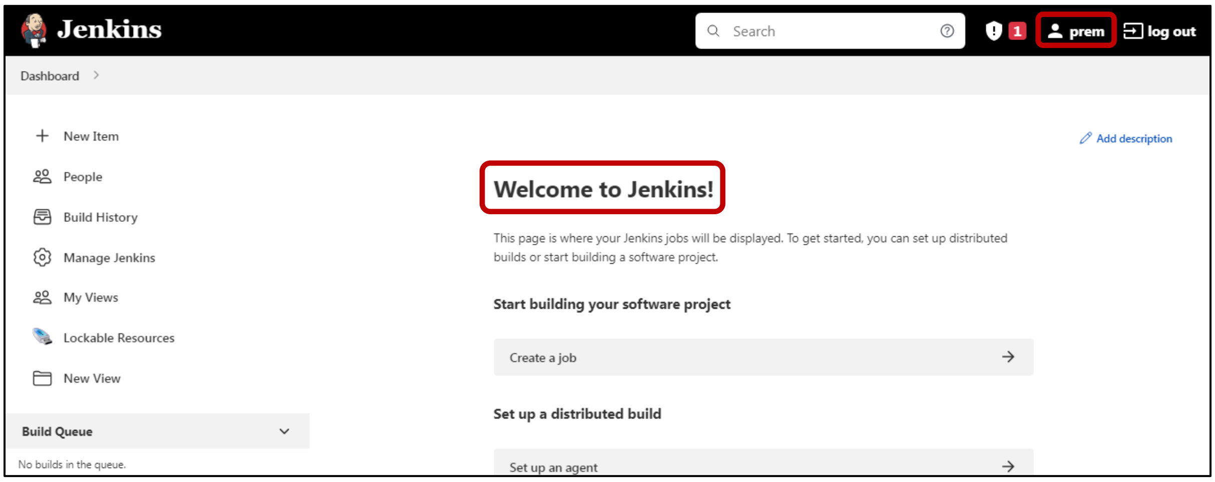
```
Mar 26 12:59:40 ip-172-31-0-208.ap-south-1.compute.internal jenkins[11355]: 2022-03-26 12:59:40.731+0000 [id=47]
Hint: Some lines were ellipsized, use -l to show in full.
[root@ip-172-31-0-208 ~]# cat /var/lib/jenkins/secrets/initialAdminPassword
7369a2f301ea4a0cb0cd5e5d90b587b9
[root@ip-172-31-0-208 ~]#
```



/var/lib/jenkins/secrets/initialAdminPassword

Please copy the password from either location and paste it below.

Administrator password



No builds in the queue.

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

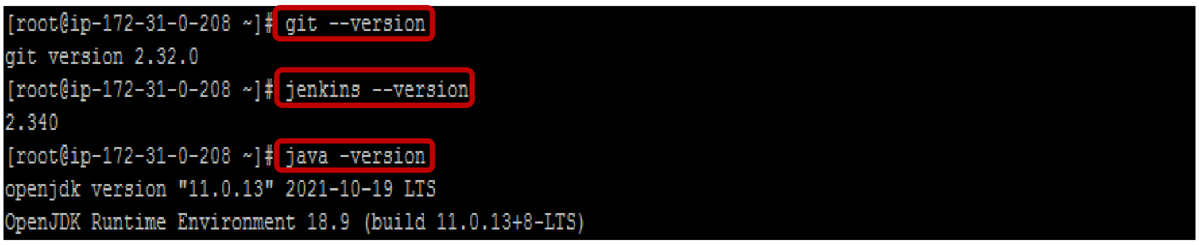
Start building your software project

Set up a distributed build

# Check the version of git, jenkins and java by the following commands.

- Git –version
- Jenkins –version

- Java --version

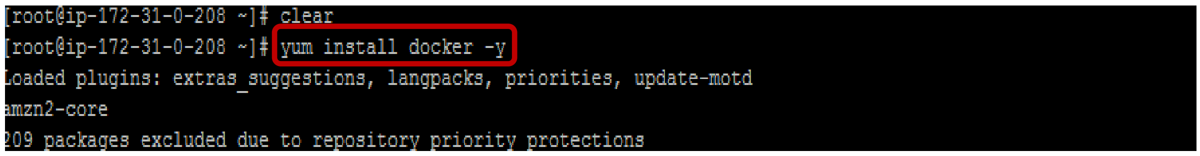


```
[root@ip-172-31-0-208 ~]# git --version
git version 2.32.0
[root@ip-172-31-0-208 ~]# jenkins --version
2.340
[root@ip-172-31-0-208 ~]# java -version
openjdk version "11.0.13" 2021-10-19 LTS
OpenJDK Runtime Environment 18.9 (build 11.0.13+8-LTS)
```

DOCKER INSTALLATION:

# Docker installs on server by following command.

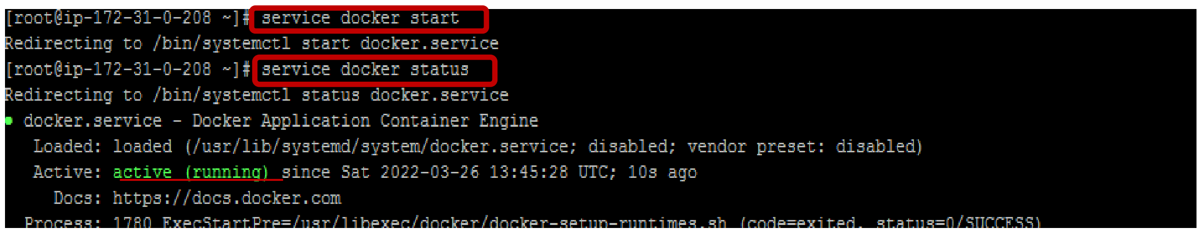
- yum install docker –y



```
[root@ip-172-31-0-208 ~]# clear
[root@ip-172-31-0-208 ~]# yum install docker -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
209 packages excluded due to repository priority protections
```

# docker start and know the status of docker by the following command.

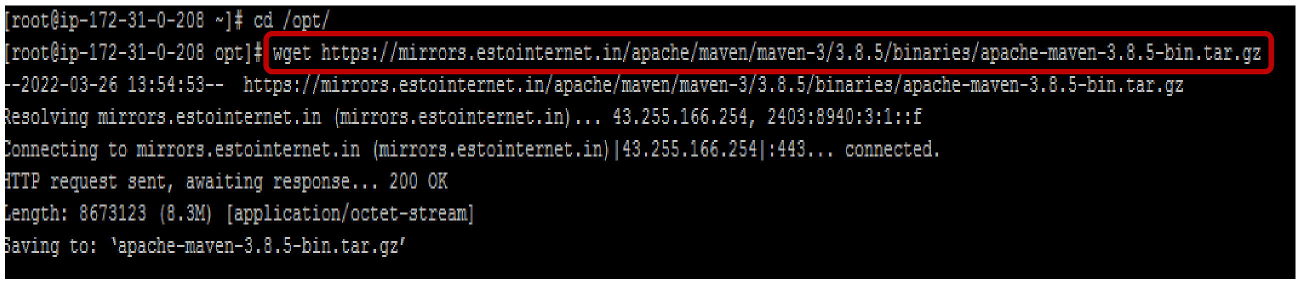
- service docker start.
- Service docker status



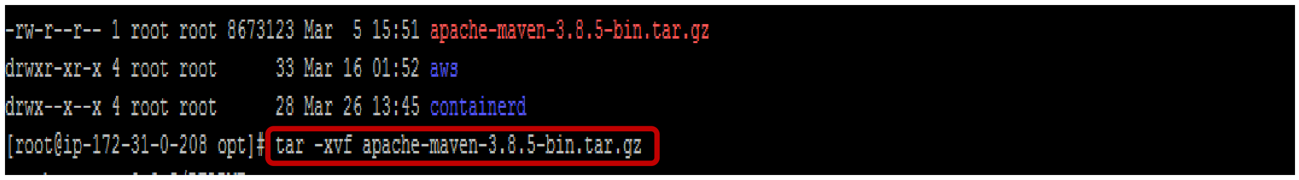
```
[root@ip-172-31-0-208 ~]# service docker start
Redirecting to /bin/systemctl start docker.service
[root@ip-172-31-0-208 ~]# service docker status
Redirecting to /bin/systemctl status docker.service
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; vendor preset: disabled)
   Active: active (running) since Sat 2022-03-26 13:45:28 UTC; 10s ago
     Docs: https://docs.docker.com
  Process: 1780 ExecStartPre=/usr/libexec/docker/docker-setup-runtimes.sh (code=exited, status=0/SUCCESS)
```

MAVEN INSTALLATION:
# Maven installs on server's opt directory.
cd /opt/
# Maven installs on the server by following command.

- wget https://mirrors.estointernet.in/apache/maven/maven3/3.8.5/binaries/apache-maven-3.8.5-bin.tar.gz
- tar –xvf apache-maven-3.8.5



```
[root@ip-172-31-0-208 ~]# cd /opt/
[root@ip-172-31-0-208 opt]# wget https://mirrors.estointernet.in/apache/maven/maven-3/3.8.5/binaries/apache-maven-3.8.5-bin.tar.gz
--2022-03-26 13:54:53--  https://mirrors.estointernet.in/apache/maven/maven-3/3.8.5/binaries/apache-maven-3.8.5-bin.tar.gz
Resolving mirrors.estointernet.in (mirrors.estointernet.in)... 43.255.166.254, 2403:8940:3:1::f
Connecting to mirrors.estointernet.in (mirrors.estointernet.in)|43.255.166.254|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 8673123 (8.3M) [application/octet-stream]
Saving to: 'apache-maven-3.8.5-bin.tar.gz'
```



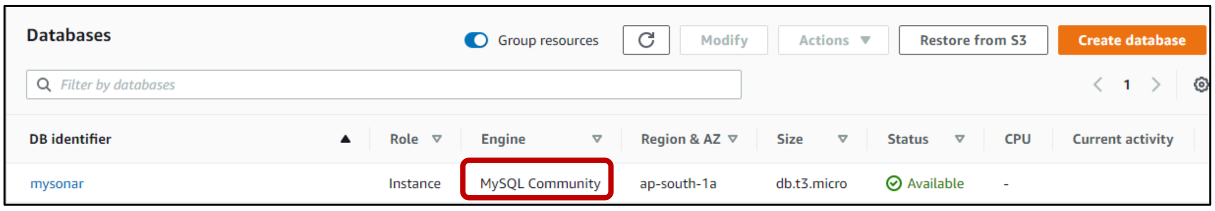
```
-rw-r--r-- 1 root root 8673123 Mar  5 15:51 apache-maven-3.8.5-bin.tar.gz
drwxr-xr-x 4 root root      33 Mar 16 01:52 aws
drwx--x--x 4 root root      28 Mar 26 13:45 containerd
[root@ip-172-31-0-208 opt]# tar -xvf apache-maven-3.8.5-bin.tar.gz
```

INSTALLATION OF SONARQUBE:

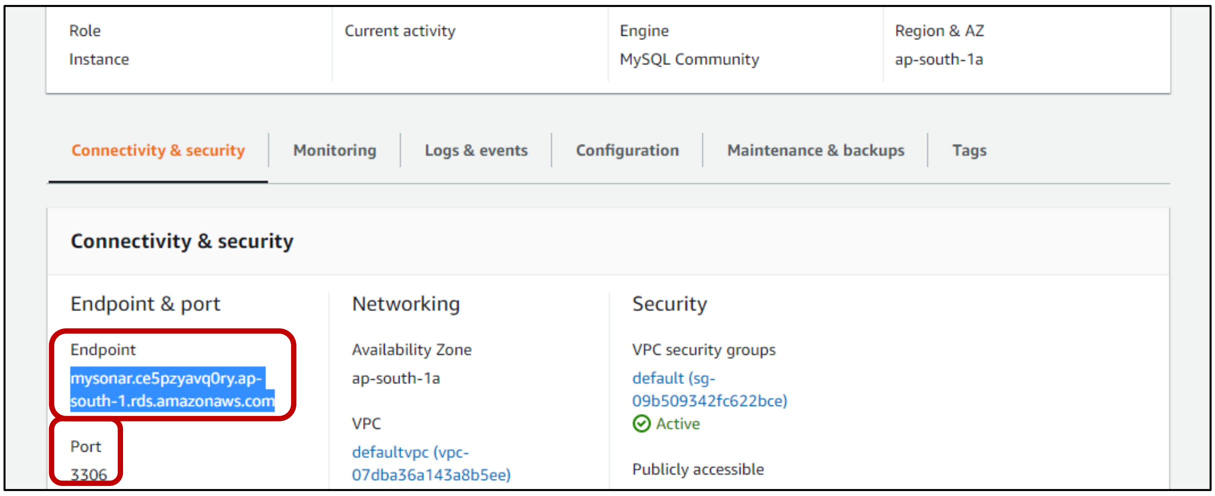
Prerequisite of Sonarqube:

- 3gb ram machine is required.
- Java open-jdk is necessary.
- RDS Data base server (mysql) is required
- Creation of local user and remote user and permission access in data base server.
- Sonarqube should not start with root user

# Creation of RDS database server


Databases

DB identifier	Role	Engine	Region & AZ	Size	Status	CPU	Current activity
mysonar	Instance	MySQL Community	ap-south-1a	db.t3.micro	Available	-	


Role: Instance
Current activity:
Engine: MySQL Community
Region & AZ: ap-south-1a

Connectivity & security

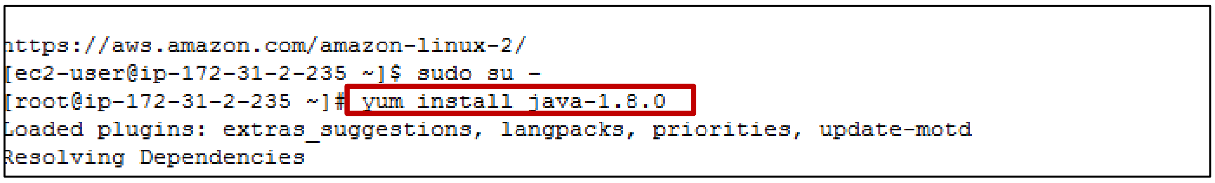
Endpoint & port
Endpoint: mysonar.ce5pzyavq0ry.ap-south-1.rds.amazonaws.com
Port: 3306

Networking
Availability Zone: ap-south-1a
VPC: defaultvpc (vpc-07dba36a143a8b5ee)

Security
VPC security groups: default (sg-09b509342fc622bce) Active
Publicly accessible

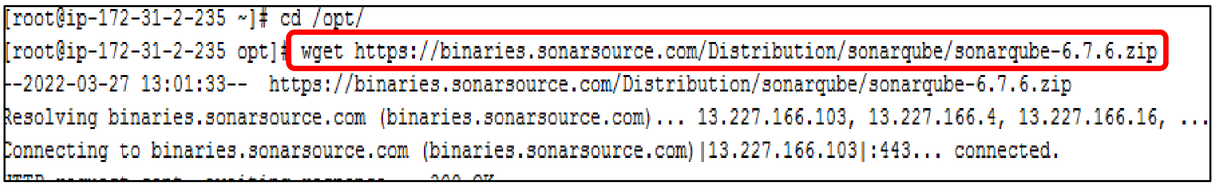
# Java installation on the sonarqube server using the following command.

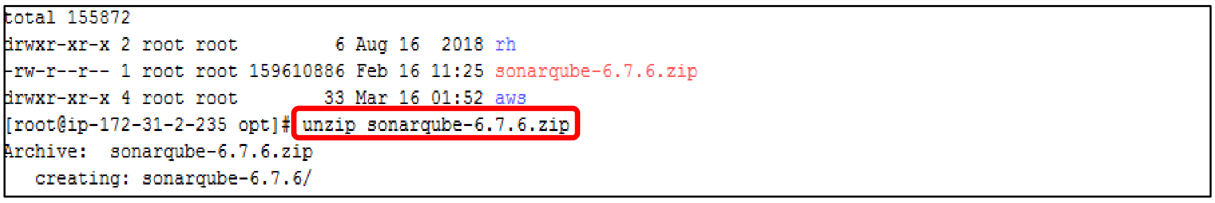
- yum install java-1.8.0


```
https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-2-235 ~]$ sudo su -
[root@ip-172-31-2-235 ~]# yum install java-1.8.0
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
```

# Installation of sonarqube by the following command.
# SonarQube installs on server's opt directory.
cd /opt

- wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-6.7.6.zip
- unzip sonarqube-6.7.6.zip

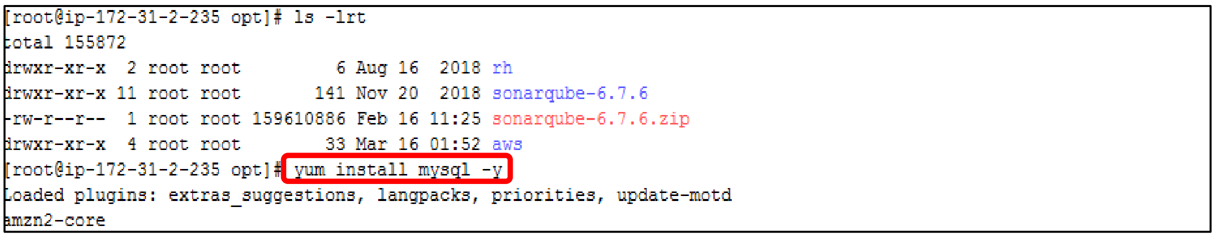

```
[root@ip-172-31-2-235 ~]# cd /opt/
[root@ip-172-31-2-235 opt]# wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-6.7.6.zip
--2022-03-27 13:01:33--  https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-6.7.6.zip
Resolving binaries.sonarsource.com (binaries.sonarsource.com)... 13.227.166.103, 13.227.166.4, 13.227.166.16, ...
Connecting to binaries.sonarsource.com (binaries.sonarsource.com)|13.227.166.103|:443... connected.
```



```
total 155872
drwxr-xr-x 2 root root         6 Aug 16  2018 rh
-rw-r--r-- 1 root root 159610886 Feb 16 11:25 sonarqube-6.7.6.zip
drwxr-xr-x 4 root root        33 Mar 16 01:52 aws
[root@ip-172-31-2-235 opt]# unzip sonarqube-6.7.6.zip
Archive:  sonarqube-6.7.6.zip
   creating: sonarqube-6.7.6/
```

# Install mysql on Sonarqube server by following command.

- ➢ yum install mysql -y

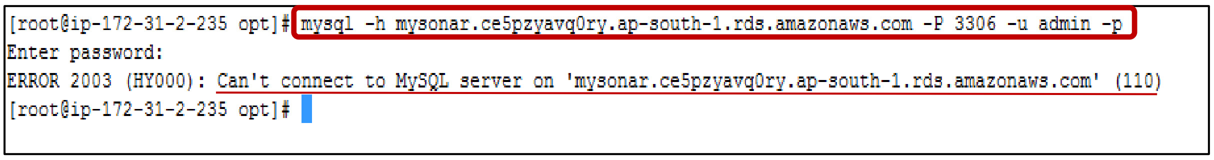


```
[root@ip-172-31-2-235 opt]# ls -lrt
total 155872
drwxr-xr-x  2 root root         6 Aug 16  2018 rh
drwxr-xr-x 11 root root       141 Nov 20  2018 sonarqube-6.7.6
-rw-r--r--  1 root root 159610886 Feb 16 11:25 sonarqube-6.7.6.zip
drwxr-xr-x  4 root root        33 Mar 16 01:52 aws
[root@ip-172-31-2-235 opt]# yum install mysql -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
```

# Connect RDS endpoint to the mysql database by following command.

- ➢ Mysql –h endpoint mysonar.ce5pzyavq0ry.ap-south-1.rds.amazonaws.com –P 3306 –u admin –p

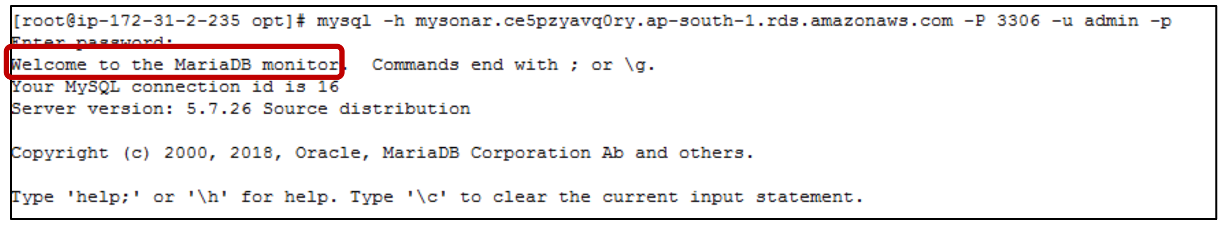
ERROR 1:



```
[root@ip-172-31-2-235 opt]# mysql -h mysonar.ce5pzyavq0ry.ap-south-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
ERROR 2003 (HY000): Can't connect to MySQL server on 'mysonar.ce5pzyavq0ry.ap-south-1.rds.amazonaws.com' (110)
[root@ip-172-31-2-235 opt]#
```

TROBLESHOOT:

1. Go to EC2 Dashboard
2. Go to Security Groups tab
3. Select and only select the RDS database security group. You'll see the security group detail at the bottom
4. Click Inbound tab
5. Click Edit button
6. Add Type:MYSQL/Aurora;Protocol:TCP;Range:3306;Source:0.0.0.0/0



```
[root@ip-172-31-2-235 opt]# mysql -h mysonar.ce5pzyavq0ry.ap-south-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 16
Server version: 5.7.26 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

# Create local and remote user and grant mysql database access to the user..
# cd /opt/
# Create Database by following command.

- ➢ CREATE DATABASE sonar CHARACTER SET utf8 COLLATE utf8_general_ci;

# Create a local and a remote user by following command.

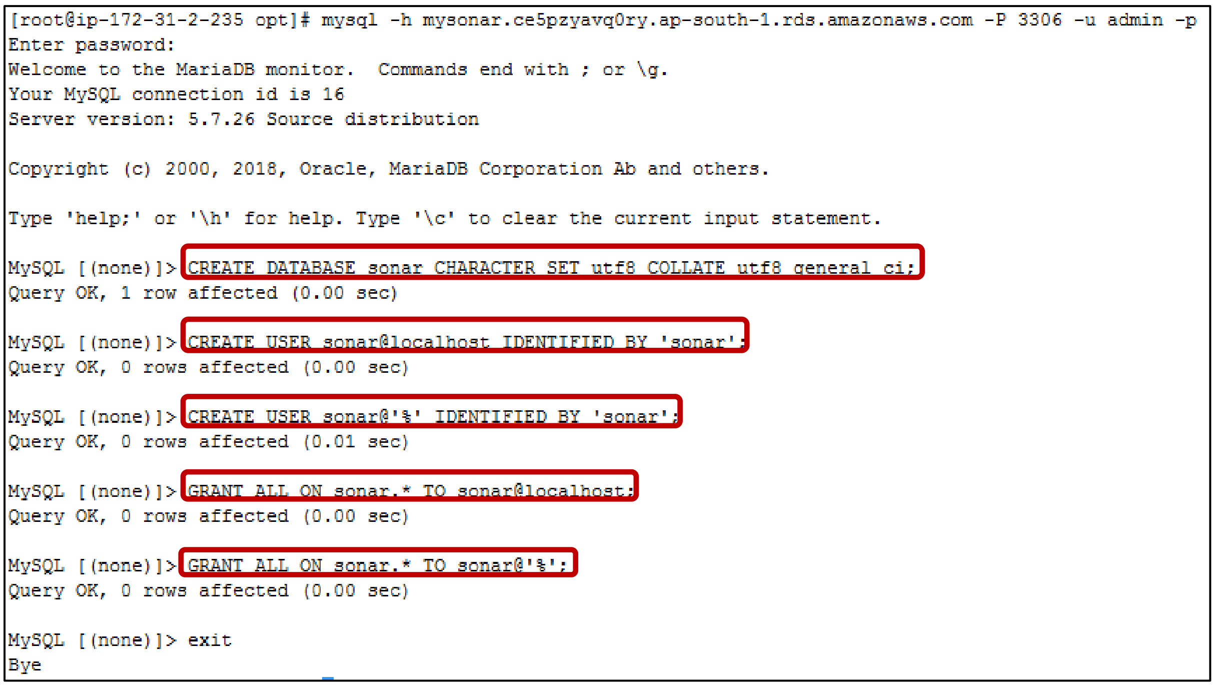
- ➢ CREATE USER sonar@localhost IDENTIFIED BY 'sonar';
- ➢ CREATE USER sonar@'%' IDENTIFIED BY 'sonar';

# Grant database access permissions to users by following command

- ➢ GRANT ALL ON sonar.* TO sonar@localhost;

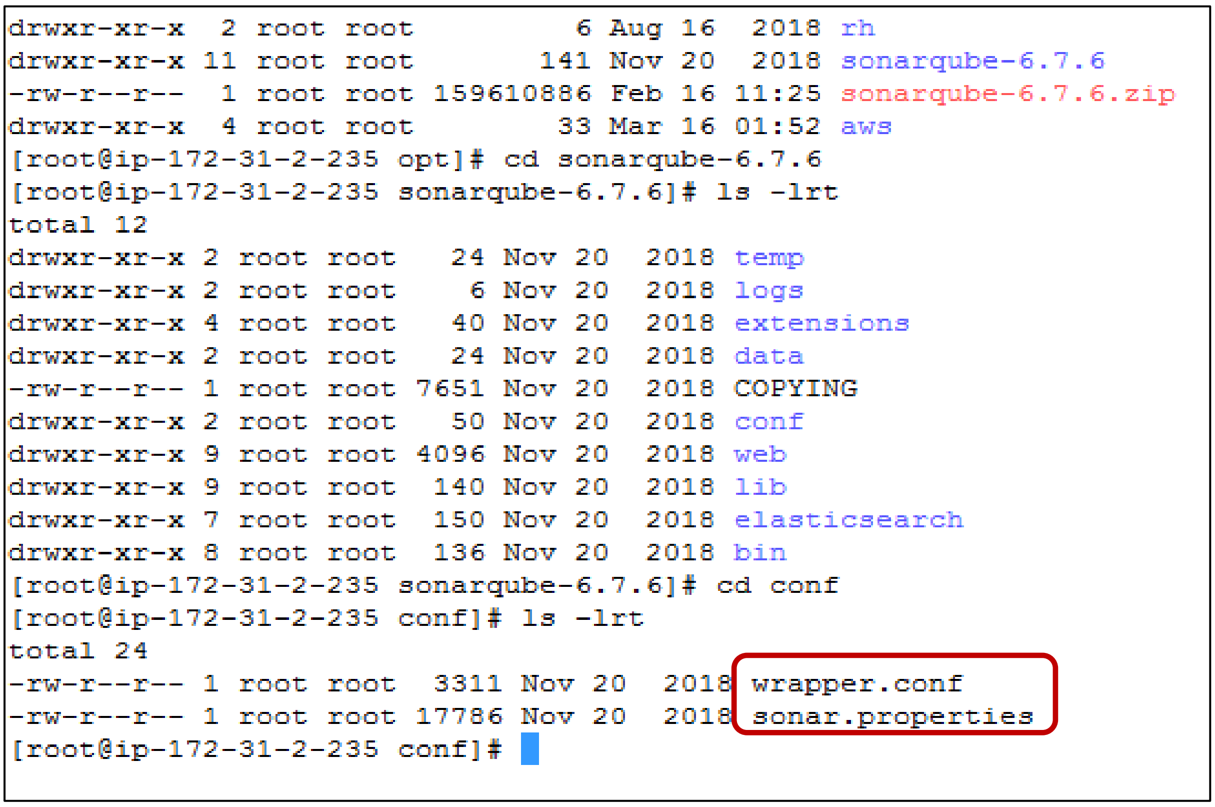
- GRANT ALL ON sonar.* TO sonar@'%';

# Exit.



```
[root@ip-172-31-2-235 opt]# mysql -h mysonar.ce5pzyavq0ry.ap-south-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 16
Server version: 5.7.26 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> CREATE DATABASE sonar CHARACTER SET utf8 COLLATE utf8_general_ci;
Query OK, 1 row affected (0.00 sec)

MySQL [(none)]> CREATE USER sonar@localhost IDENTIFIED BY 'sonar';
Query OK, 0 rows affected (0.00 sec)

MySQL [(none)]> CREATE USER sonar@'%' IDENTIFIED BY 'sonar';
Query OK, 0 rows affected (0.01 sec)

MySQL [(none)]> GRANT ALL ON sonar.* TO sonar@localhost;
Query OK, 0 rows affected (0.00 sec)

MySQL [(none)]> GRANT ALL ON sonar.* TO sonar@'%';
Query OK, 0 rows affected (0.00 sec)

MySQL [(none)]> exit
Bye
```

# cd /opt/Sonarqube-6..7.6/conf

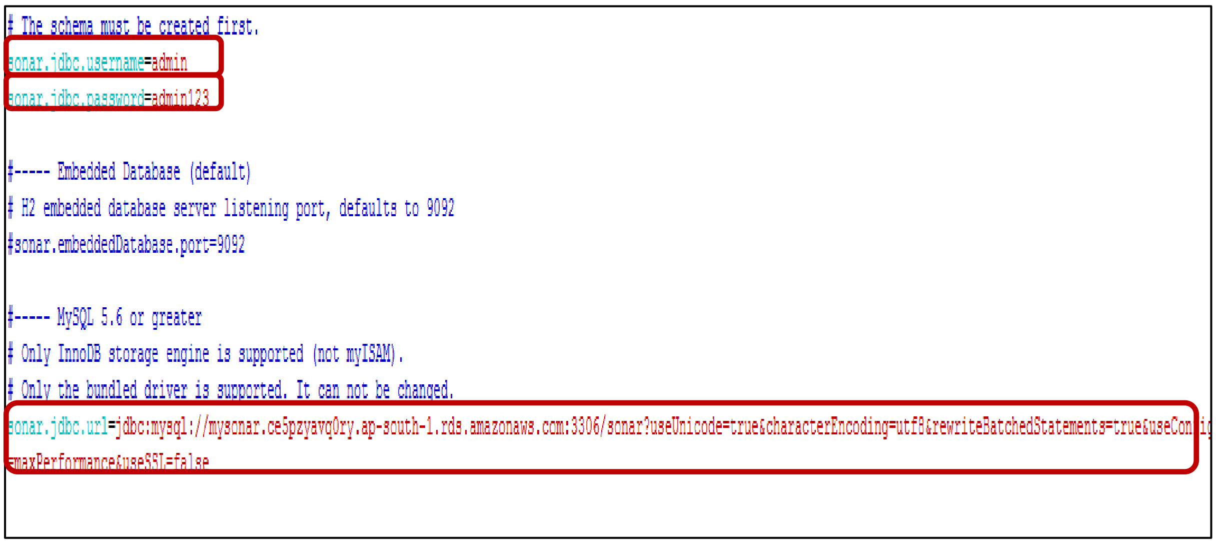


```
drwxr-xr-x  2 root root         6 Aug 16  2018 rh
drwxr-xr-x 11 root root       141 Nov 20  2018 sonarqube-6.7.6
-rw-r--r--  1 root root 159610886 Feb 16 11:25 sonarqube-6.7.6.zip
drwxr-xr-x  4 root root        33 Mar 16 01:52 aws
[root@ip-172-31-2-235 opt]# cd sonarqube-6.7.6
[root@ip-172-31-2-235 sonarqube-6.7.6]# ls -lrt
total 12
drwxr-xr-x 2 root root   24 Nov 20  2018 temp
drwxr-xr-x 2 root root    6 Nov 20  2018 logs
drwxr-xr-x 4 root root   40 Nov 20  2018 extensions
drwxr-xr-x 2 root root   24 Nov 20  2018 data
-rw-r--r-- 1 root root 7651 Nov 20  2018 COPYING
drwxr-xr-x 2 root root   50 Nov 20  2018 conf
drwxr-xr-x 9 root root 4096 Nov 20  2018 web
drwxr-xr-x 9 root root  140 Nov 20  2018 lib
drwxr-xr-x 7 root root  150 Nov 20  2018 elasticsearch
drwxr-xr-x 8 root root  136 Nov 20  2018 bin
[root@ip-172-31-2-235 sonarqube-6.7.6]# cd conf
[root@ip-172-31-2-235 conf]# ls -lrt
total 24
-rw-r--r-- 1 root root  3311 Nov 20  2018 wrapper.conf
-rw-r--r-- 1 root root 17786 Nov 20  2018 sonar.properties
[root@ip-172-31-2-235 conf]#
```

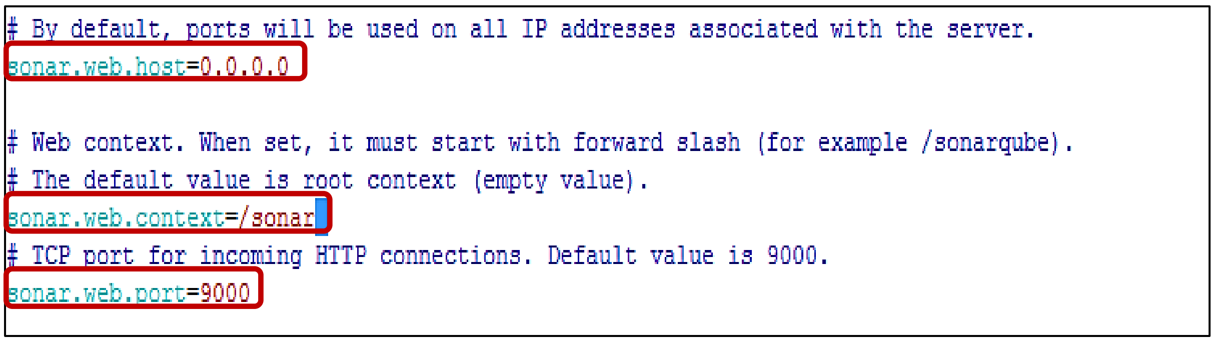
# Edit sonar.properties file to uncomment and provide required information for below properties.

# File Name: /opt/sonar/conf/sonar.properties

- sonar.jdbc.username=sonar
- sonar.jdbc.password=sonar
- sonar.jdbc.url=jdbc:mysql mysonar.ce5pzyavq0ry.ap-south-1.rds.amazonaws.com :3306/sonar?useUnicode=true&characterEncoding=utf8&rewriteBatchedStatements=true&useConfigs =maxPerformance&useSSL=false
- sonar.web.host=0.0.0.0
- sonar.web.context=/sonar

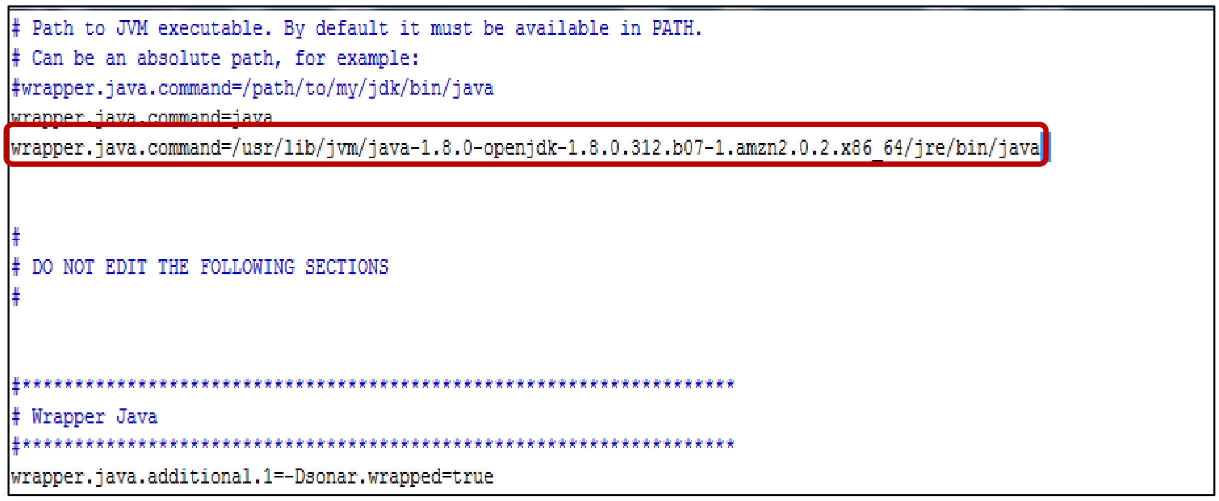


```
# The schema must be created first.
sonar.jdbc.username=admin
sonar.jdbc.password=admin123

#----- Embedded Database (default)
# H2 embedded database server listening port, defaults to 9092
#sonar.embeddedDatabase.port=9092

#----- MySQL 5.6 or greater
# Only InnoDB storage engine is supported (not myISAM).
# Only the bundled driver is supported. It can not be changed.
sonar.jdbc.url=jdbc:mysql://mysonar.ce5pzyavq0ry.ap-south-1.rds.amazonaws.com:3306/sonar?useUnicode=true&characterEncoding=utf8&rewriteBatchedStatements=true&useConfig
=maxPerformance&useSSL=false
```



```
# By default, ports will be used on all IP addresses associated with the server.
sonar.web.host=0.0.0.0

# Web context. When set, it must start with forward slash (for example /sonarqube).
# The default value is root context (empty value).
sonar.web.context=/sonar
# TCP port for incoming HTTP connections. Default value is 9000.
sonar.web.port=9000
```

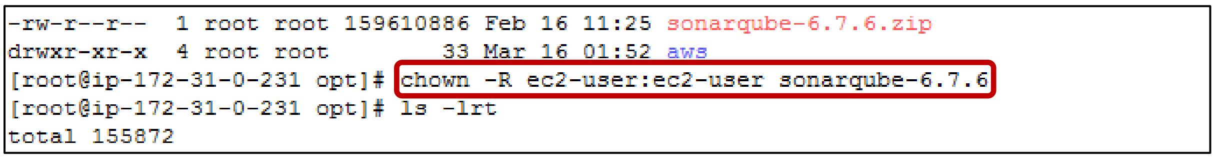
# Do changes in wrapper.conf file
# Filename: /opt/sonar/conf/wrapper.conf
Wrapper.java.command=/usr/lib/java-1.8.0-openjdk-1.8.0.312.b07-1.amzn2.0.2.x86_64/jre/bin/java



```
# Path to JVM executable. By default it must be available in PATH.
# Can be an absolute path, for example:
#wrapper.java.command=/path/to/my/jdk/bin/java
wrapper.java.command=java
wrapper.java.command=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.312.b07-1.amzn2.0.2.x86_64/jre/bin/java


#
# DO NOT EDIT THE FOLLOWING SECTIONS
#


#********************************************************************
# Wrapper Java
#********************************************************************
wrapper.java.additional.1=-Dsonar.wrapped=true
```

# Sonarqube should not start with root user.
# cd /opt/
# following command is used to convert the sonarqube-6.7.6 directory as a ec2-user from root user.

- ➢ chown –R ec2-user:ec2-user sonarqube-6.7.6

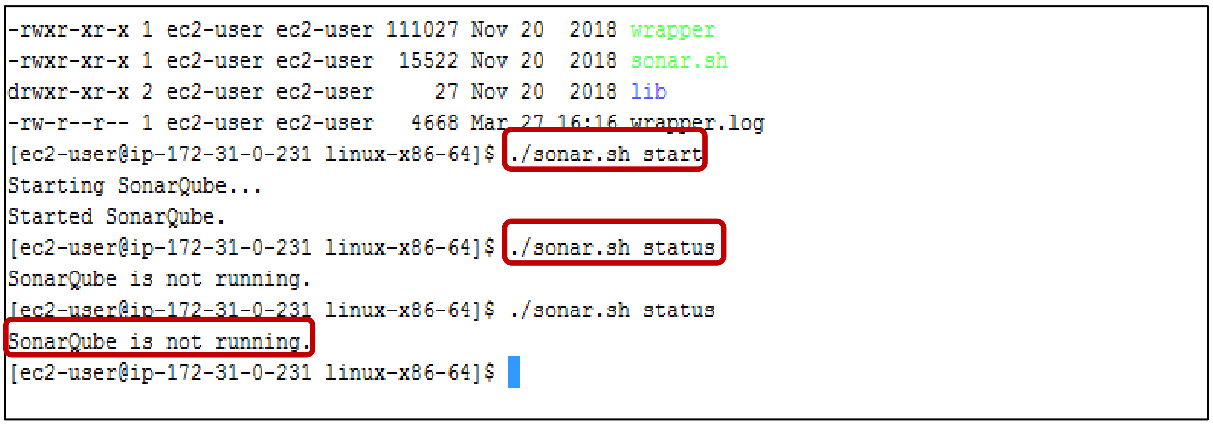


```
-rw-r--r--  1 root root 159610886 Feb 16 11:25 sonarqube-6.7.6.zip
drwxr-xr-x  4 root root        33 Mar 16 01:52 aws
[root@ip-172-31-0-231 opt]# chown -R ec2-user:ec2-user sonarqube-6.7.6
[root@ip-172-31-0-231 opt]# ls -lrt
total 155872
```

# cd /sonarqube-6.7.6/bin/linux-x86-64
# SonarQube Start and know the status of the Sonarqube by following command.

- ➢ ./sonar.sh start
- ➢ ./sonar.sh.status

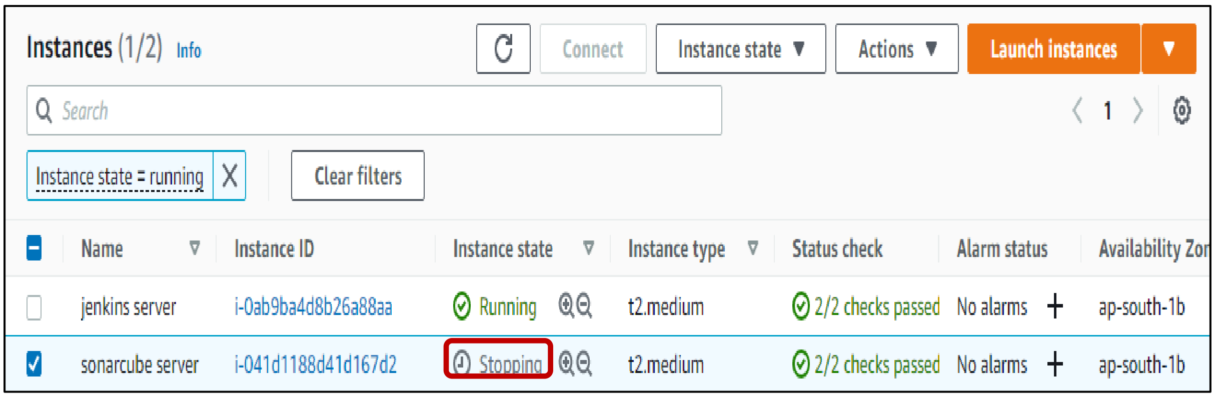
ERROR 2:



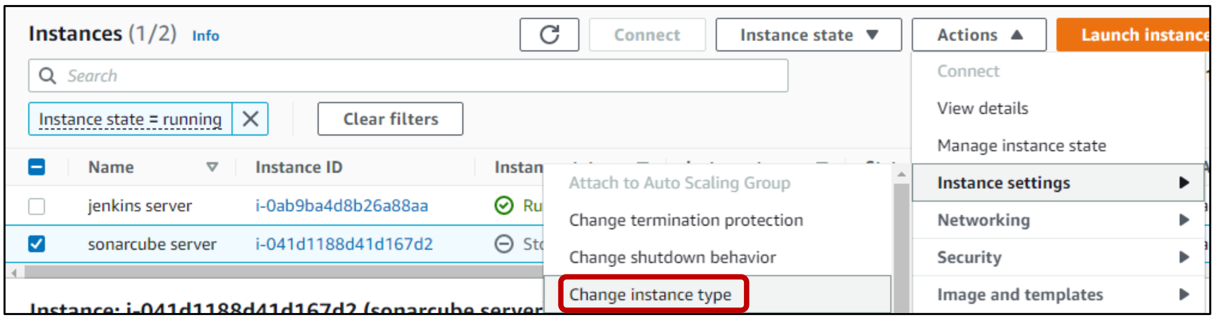
```
-rwxr-xr-x 1 ec2-user ec2-user 111027 Nov 20  2018 wrapper
-rwxr-xr-x 1 ec2-user ec2-user  15522 Nov 20  2018 sonar.sh
drwxr-xr-x 2 ec2-user ec2-user     27 Nov 20  2018 lib
-rw-r--r-- 1 ec2-user ec2-user   4668 Mar 27 16:16 wrapper.log
[ec2-user@ip-172-31-0-231 linux-x86-64]$ ./sonar.sh start
Starting SonarQube...
Started SonarQube.
[ec2-user@ip-172-31-0-231 linux-x86-64]$ ./sonar.sh status
SonarQube is not running.
[ec2-user@ip-172-31-0-231 linux-x86-64]$ ./sonar.sh status
SonarQube is not running.
[ec2-user@ip-172-31-0-231 linux-x86-64]$
```

TROUBLESHOOT:

# Change the instance type from t2.mediam to t2.large and restart the Sonarqube.

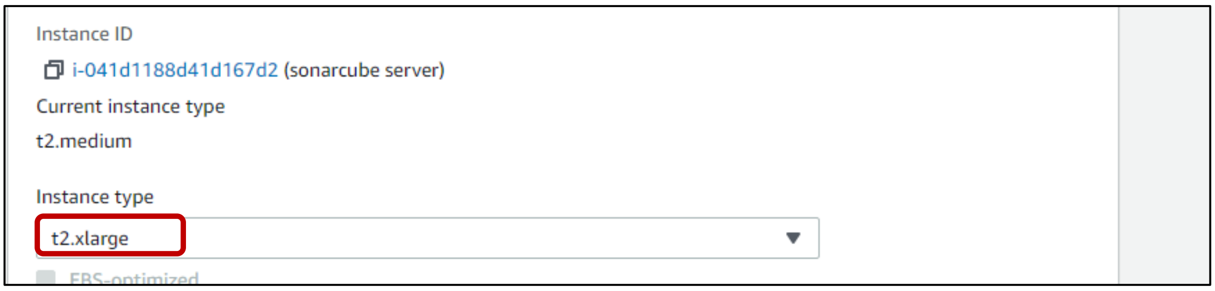


Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zor
jenkins server	i-0ab9ba4d8b26a88aa	Running	t2.medium	2/2 checks passed	No alarms +	ap-south-1b
sonarcube server	i-041d1188d41d167d2	Stopping	t2.medium	2/2 checks passed	No alarms +	ap-south-1b

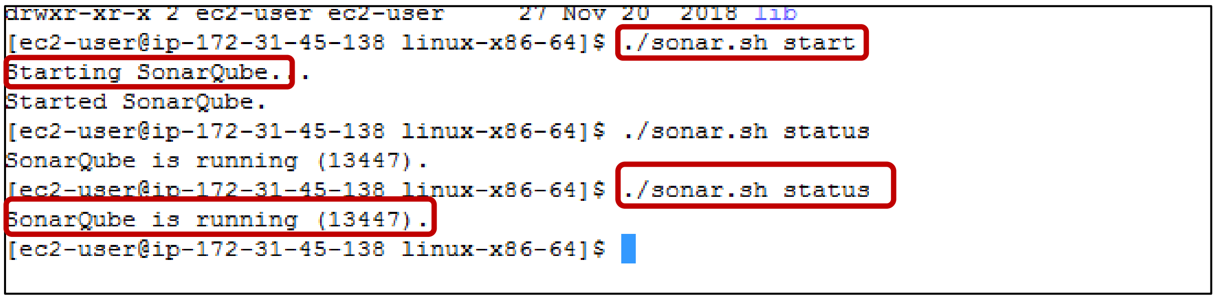

Instances (1/2) Info

Instance state = running

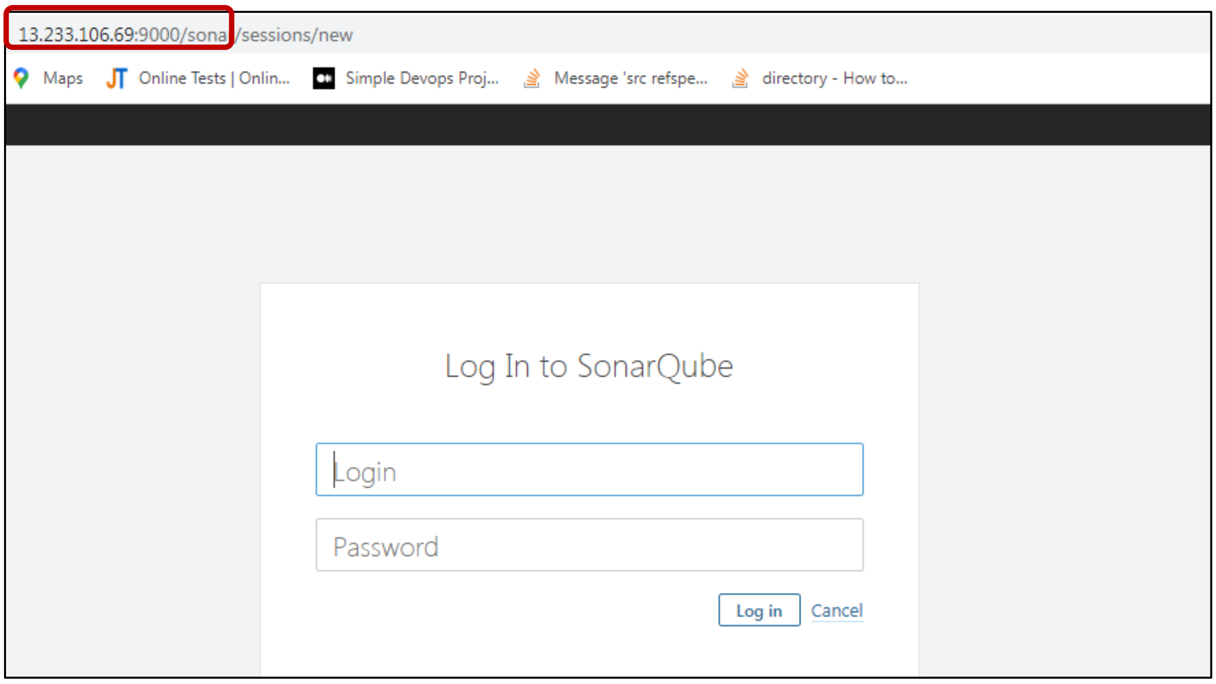
Name	Instance ID	Instan
jenkins server	i-0ab9ba4d8b26a88aa	Ru
sonarcube server	i-041d1188d41d167d2	Sto


Instance ID
i-041d1188d41d167d2 (sonarcube server)
Current instance type
t2.medium

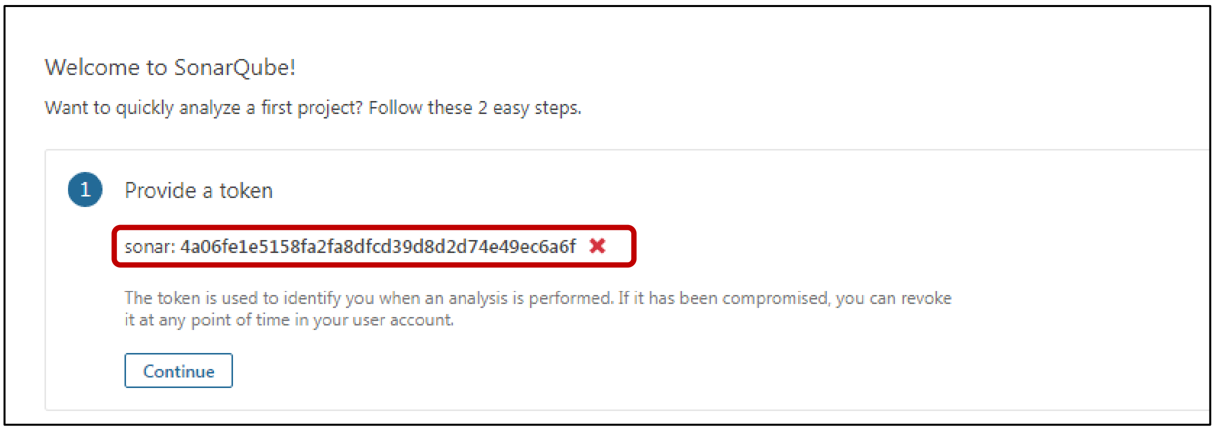
Instance type
t2.xlarge


```
drwxr-xr-x 2 ec2-user ec2-user    27 Nov 20  2018 lib
[ec2-user@ip-172-31-45-138 linux-x86-64]$ ./sonar.sh start
Starting SonarQube...
Started SonarQube.
[ec2-user@ip-172-31-45-138 linux-x86-64]$ ./sonar.sh status
SonarQube is running (13447).
[ec2-user@ip-172-31-45-138 linux-x86-64]$ ./sonar.sh status
SonarQube is running (13447).
[ec2-user@ip-172-31-45-138 linux-x86-64]$
```

# Access the sonarqube console with 9000 port and create a token


13.233.106.69:9000/sonar/sessions/new

# This token is to communicate jenkins and Sonarqube.


Welcome to SonarQube!

Want to quickly analyze a first project? Follow these 2 easy steps.

1 Provide a token

sonar: 4a06fe1e5158fa2fa8dfcd39d8d2d74e49ec6a6f ✖

The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point of time in your user account.

Continue

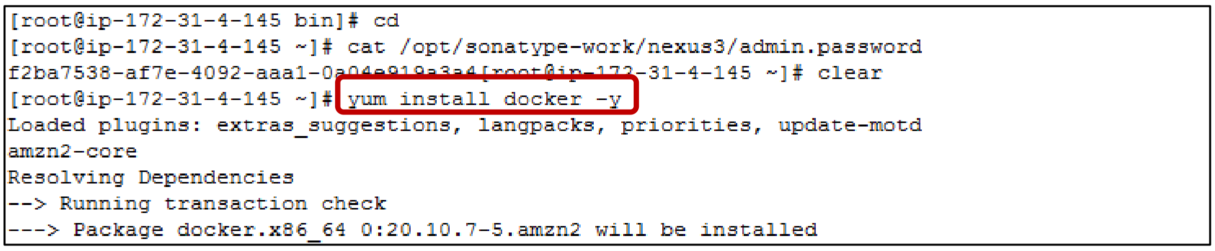
INSTALLATION OF NEXUS:

Prerequisite of Nexus:

- ➢ Minimum 3bb ram machine is required.
- ➢ Java open-jdk is also required

# Docker installs on the nexus server by following command.

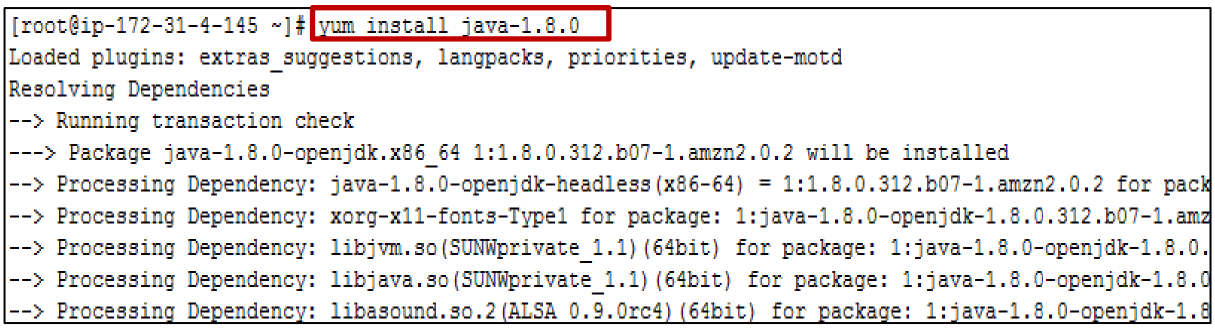
- ➢ yum install docker -y



```
[root@ip-172-31-4-145 bin]# cd
[root@ip-172-31-4-145 ~]# cat /opt/sonatype-work/nexus3/admin.password
f2ba7538-af7e-4092-aaa1-0a04e919a3a4[root@ip-172-31-4-145 ~]# clear
[root@ip-172-31-4-145 ~]# yum install docker -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
Resolving Dependencies
--> Running transaction check
---> Package docker.x86_64 0:20.10.7-5.amzn2 will be installed
```

# Java installs on the nexus server by the following command.

- ➢ yum install java-1.8.0

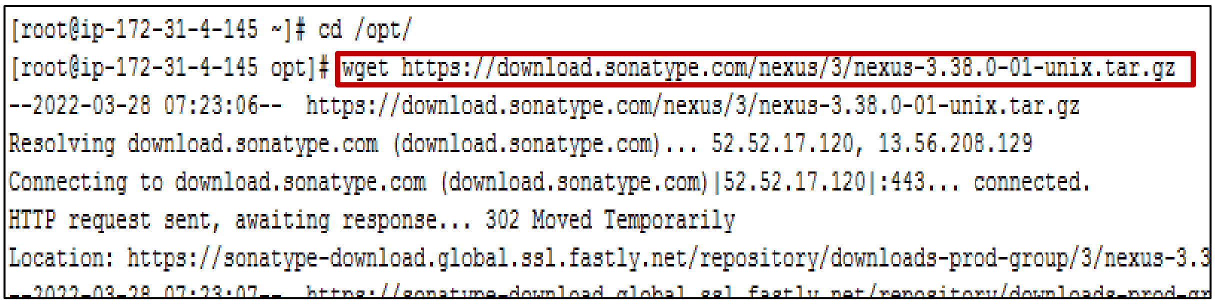


```
[root@ip-172-31-4-145 ~]# yum install java-1.8.0
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
--> Running transaction check
---> Package java-1.8.0-openjdk.x86_64 1:1.8.0.312.b07-1.amzn2.0.2 will be installed
--> Processing Dependency: java-1.8.0-openjdk-headless(x86-64) = 1:1.8.0.312.b07-1.amzn2.0.2 for pack
--> Processing Dependency: xorg-x11-fonts-Type1 for package: 1:java-1.8.0-openjdk-1.8.0.312.b07-1.amz
--> Processing Dependency: libjvm.so(SUNWprivate_1.1)(64bit) for package: 1:java-1.8.0-openjdk-1.8.0.
--> Processing Dependency: libjava.so(SUNWprivate_1.1)(64bit) for package: 1:java-1.8.0-openjdk-1.8.0
--> Processing Dependency: libasound.so.2(ALSA_0.9.0rc4)(64bit) for package: 1:java-1.8.0-openjdk-1.8
```

# Install nexus on opt diecrtory

# cd /opt

- ➢ wget https://download.sonatype.com/nexus/3/nexus-3.38.0-01-unix.tar.gz



```
[root@ip-172-31-4-145 ~]# cd /opt/
[root@ip-172-31-4-145 opt]# wget https://download.sonatype.com/nexus/3/nexus-3.38.0-01-unix.tar.gz
--2022-03-28 07:23:06--  https://download.sonatype.com/nexus/3/nexus-3.38.0-01-unix.tar.gz
Resolving download.sonatype.com (download.sonatype.com)... 52.52.17.120, 13.56.208.129
Connecting to download.sonatype.com (download.sonatype.com)|52.52.17.120|:443... connected.
HTTP request sent, awaiting response... 302 Moved Temporarily
Location: https://sonatype-download.global.ssl.fastly.net/repository/downloads-prod-group/3/nexus-3.3
```

# untar the nexus-3.38.0-01-unix.tar.gz using following command

- ➢ tar –xvf nexus-3.38.0-01-unix.tar.gz

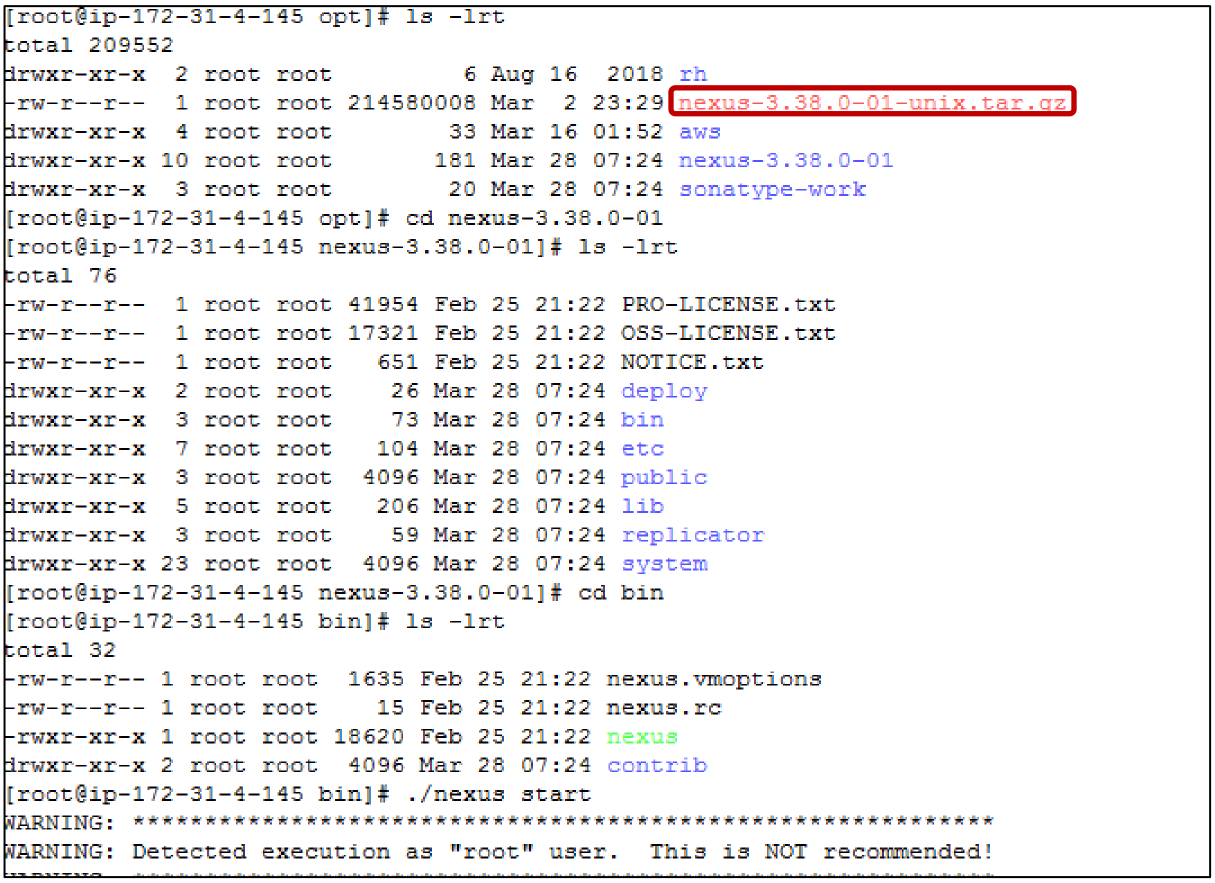
# Start the nexus from bin using the following command.

# cd /opt/ nexus-3.38.0-01/bin

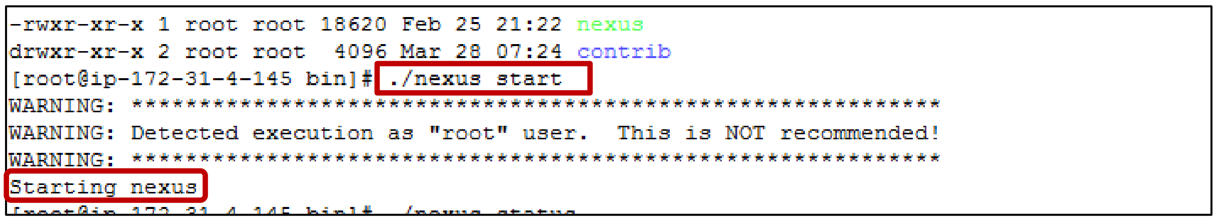
- ➢ ./nexus start

# Know the status of nexus using following command.

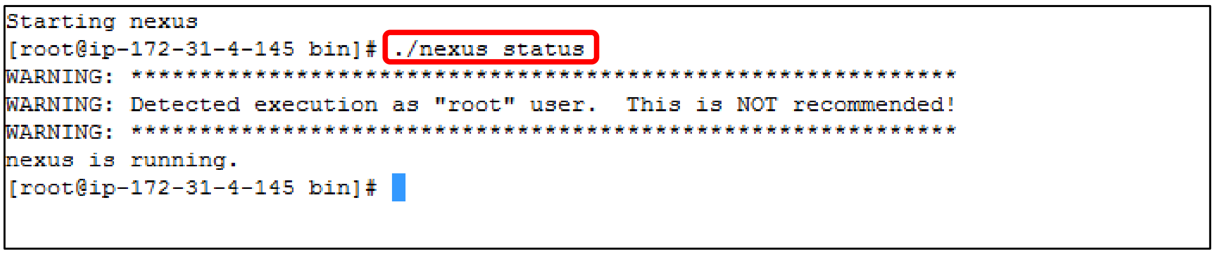
- ➢ ./ nexus status



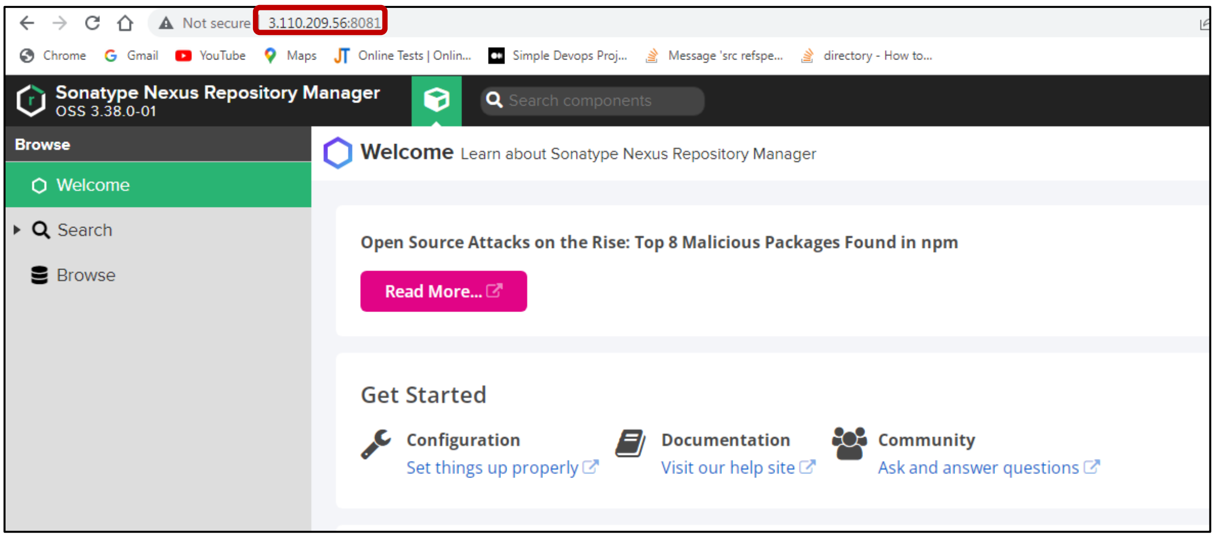
```
[root@ip-172-31-4-145 opt]# ls -lrt
total 209552
drwxr-xr-x  2 root root         6 Aug 16  2018 rh
-rw-r--r--  1 root root 214580008 Mar  2 23:29 nexus-3.38.0-01-unix.tar.gz
drwxr-xr-x  4 root root        33 Mar 16 01:52 aws
drwxr-xr-x 10 root root       181 Mar 28 07:24 nexus-3.38.0-01
drwxr-xr-x  3 root root        20 Mar 28 07:24 sonatype-work
[root@ip-172-31-4-145 opt]# cd nexus-3.38.0-01
[root@ip-172-31-4-145 nexus-3.38.0-01]# ls -lrt
total 76
-rw-r--r--  1 root root 41954 Feb 25 21:22 PRO-LICENSE.txt
-rw-r--r--  1 root root 17321 Feb 25 21:22 OSS-LICENSE.txt
-rw-r--r--  1 root root   651 Feb 25 21:22 NOTICE.txt
drwxr-xr-x  2 root root    26 Mar 28 07:24 deploy
drwxr-xr-x  3 root root    73 Mar 28 07:24 bin
drwxr-xr-x  7 root root   104 Mar 28 07:24 etc
drwxr-xr-x  3 root root  4096 Mar 28 07:24 public
drwxr-xr-x  5 root root   206 Mar 28 07:24 lib
drwxr-xr-x  3 root root    59 Mar 28 07:24 replicator
drwxr-xr-x 23 root root  4096 Mar 28 07:24 system
[root@ip-172-31-4-145 nexus-3.38.0-01]# cd bin
[root@ip-172-31-4-145 bin]# ls -lrt
total 32
-rw-r--r-- 1 root root  1635 Feb 25 21:22 nexus.vmoptions
-rw-r--r-- 1 root root    15 Feb 25 21:22 nexus.rc
-rwxr-xr-x 1 root root 18620 Feb 25 21:22 nexus
drwxr-xr-x 2 root root  4096 Mar 28 07:24 contrib
[root@ip-172-31-4-145 bin]# ./nexus start
WARNING: ************************************************************
WARNING: Detected execution as "root" user.  This is NOT recommended!
```

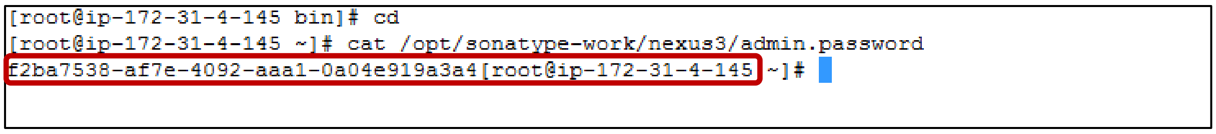


```
-rwxr-xr-x 1 root root 18620 Feb 25 21:22 nexus
drwxr-xr-x 2 root root  4096 Mar 28 07:24 contrib
[root@ip-172-31-4-145 bin]# ./nexus start
WARNING: ************************************************************
WARNING: Detected execution as "root" user.  This is NOT recommended!
WARNING: ************************************************************
Starting nexus
```

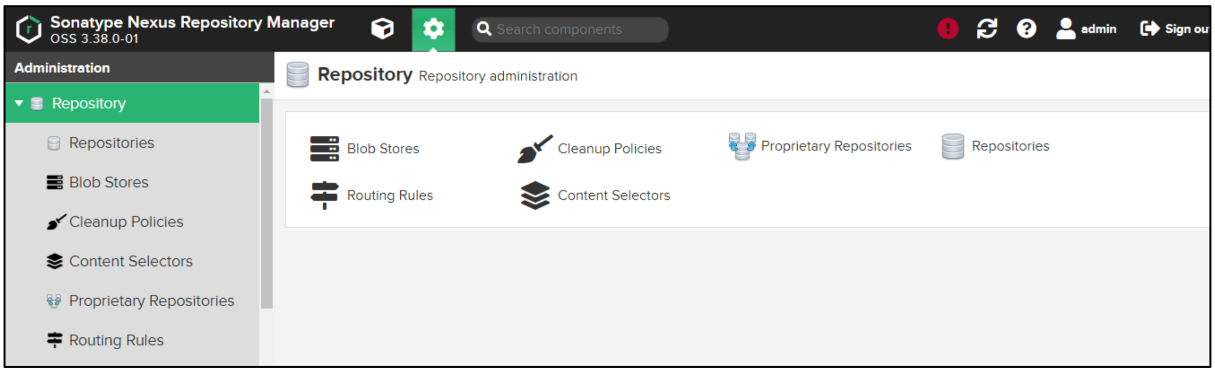

```
Starting nexus
[root@ip-172-31-4-145 bin]# ./nexus status
WARNING: ************************************************************
WARNING: Detected execution as "root" user.  This is NOT recommended!
WARNING: ************************************************************
nexus is running.
[root@ip-172-31-4-145 bin]#
```

# Access the nexus console with port 8081 and get the password from cat /opt/sonatype-work/nexus3/admin.password/


Not secure 3.110.209.56:8081
Chrome Gmail YouTube Maps Online Tests | Onlin... Simple Devops Proj... Message 'src refspe... directory - How to...
Sonatype Nexus Repository Manager
OSS 3.38.0-01
Search components
Browse
Welcome
Search
Browse
Welcome Learn about Sonatype Nexus Repository Manager
Open Source Attacks on the Rise: Top 8 Malicious Packages Found in npm
Read More...
Get Started
Configuration
Set things up properly
Documentation
Visit our help site
Community
Ask and answer questions

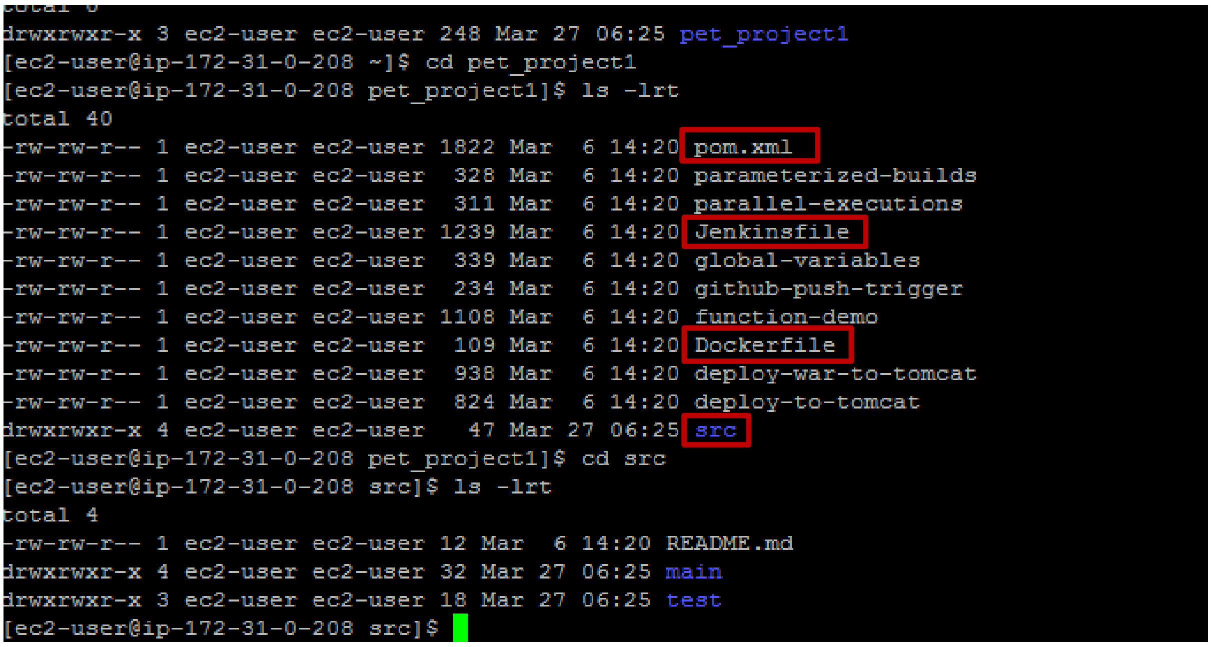

```
[root@ip-172-31-4-145 bin]# cd
[root@ip-172-31-4-145 ~]# cat /opt/sonatype-work/nexus3/admin.password
f2ba7538-af7e-4092-aaa1-0a04e919a3a4[root@ip-172-31-4-145 ~]#
```

# Copy above password and paste it to nexus console to open the sonatype nexus repository manager.

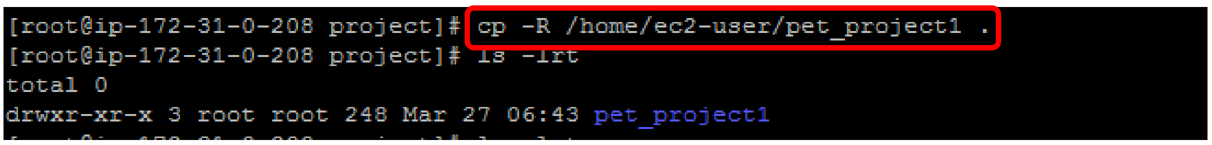


PUSH THE SOURCE CODE TO GITHUB:

# Copy the source code from local system to ec2 server using winscp.



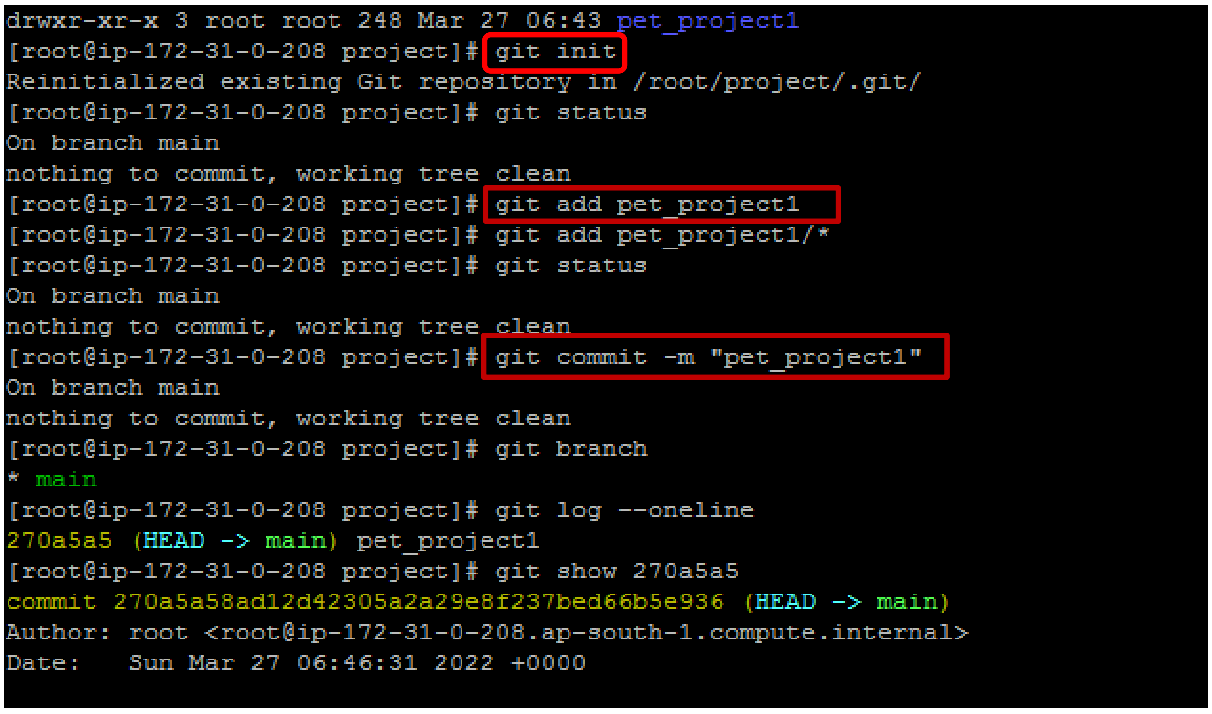
```
drwxrwxr-x 3 ec2-user ec2-user 248 Mar 27 06:25 pet_project1
[ec2-user@ip-172-31-0-208 ~]$ cd pet_project1
[ec2-user@ip-172-31-0-208 pet_project1]$ ls -lrt
total 40
-rw-rw-r-- 1 ec2-user ec2-user 1822 Mar  6 14:20 pom.xml
-rw-rw-r-- 1 ec2-user ec2-user  328 Mar  6 14:20 parameterized-builds
-rw-rw-r-- 1 ec2-user ec2-user  311 Mar  6 14:20 parallel-executions
-rw-rw-r-- 1 ec2-user ec2-user 1239 Mar  6 14:20 Jenkinsfile
-rw-rw-r-- 1 ec2-user ec2-user  339 Mar  6 14:20 global-variables
-rw-rw-r-- 1 ec2-user ec2-user  234 Mar  6 14:20 github-push-trigger
-rw-rw-r-- 1 ec2-user ec2-user 1108 Mar  6 14:20 function-demo
-rw-rw-r-- 1 ec2-user ec2-user  109 Mar  6 14:20 Dockerfile
-rw-rw-r-- 1 ec2-user ec2-user  938 Mar  6 14:20 deploy-war-to-tomcat
-rw-rw-r-- 1 ec2-user ec2-user  824 Mar  6 14:20 deploy-to-tomcat
drwxrwxr-x 4 ec2-user ec2-user   47 Mar 27 06:25 src
[ec2-user@ip-172-31-0-208 pet_project1]$ cd src
[ec2-user@ip-172-31-0-208 src]$ ls -lrt
total 4
-rw-rw-r-- 1 ec2-user ec2-user 12 Mar  6 14:20 README.md
drwxrwxr-x 4 ec2-user ec2-user 32 Mar 27 06:25 main
drwxrwxr-x 3 ec2-user ec2-user 18 Mar 27 06:25 test
[ec2-user@ip-172-31-0-208 src]$
```



```
[root@ip-172-31-0-208 project]# cp -R /home/ec2-user/pet_project1 .
[root@ip-172-31-0-208 project]# ls -lrt
total 0
drwxr-xr-x 3 root root 248 Mar 27 06:43 pet_project1
```

# Commit the source code from working directory to local server by the following commands.

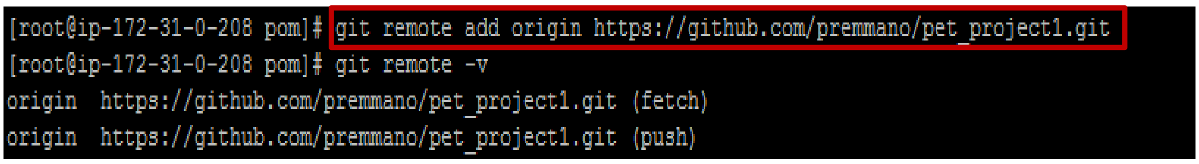
- git init
- git add pet_project1/*
- git commit –m “pet_project”



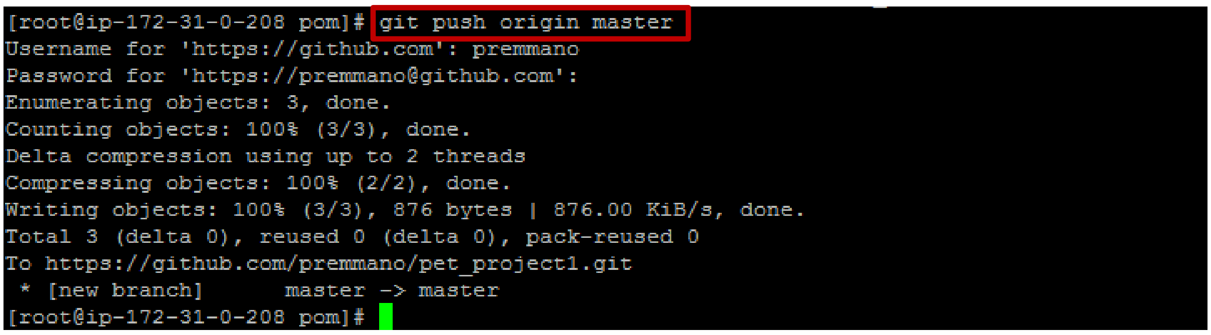
```
drwxr-xr-x 3 root root 248 Mar 27 06:43 pet_project1
[root@ip-172-31-0-208 project]# git init
Reinitialized existing Git repository in /root/project/.git/
[root@ip-172-31-0-208 project]# git status
On branch main
nothing to commit, working tree clean
[root@ip-172-31-0-208 project]# git add pet_project1
[root@ip-172-31-0-208 project]# git add pet_project1/*
[root@ip-172-31-0-208 project]# git status
On branch main
nothing to commit, working tree clean
[root@ip-172-31-0-208 project]# git commit -m "pet_project1"
On branch main
nothing to commit, working tree clean
[root@ip-172-31-0-208 project]# git branch
* main
[root@ip-172-31-0-208 project]# git log --oneline
270a5a5 (HEAD -> main) pet_project1
[root@ip-172-31-0-208 project]# git show 270a5a5
commit 270a5a58ad12d42305a2a29e8f237bed66b5e936 (HEAD -> main)
Author: root <root@ip-172-31-0-208.ap-south-1.compute.internal>
Date:   Sun Mar 27 06:46:31 2022 +0000
```

# Committed source code push to remote repository from local repository by the following commands.

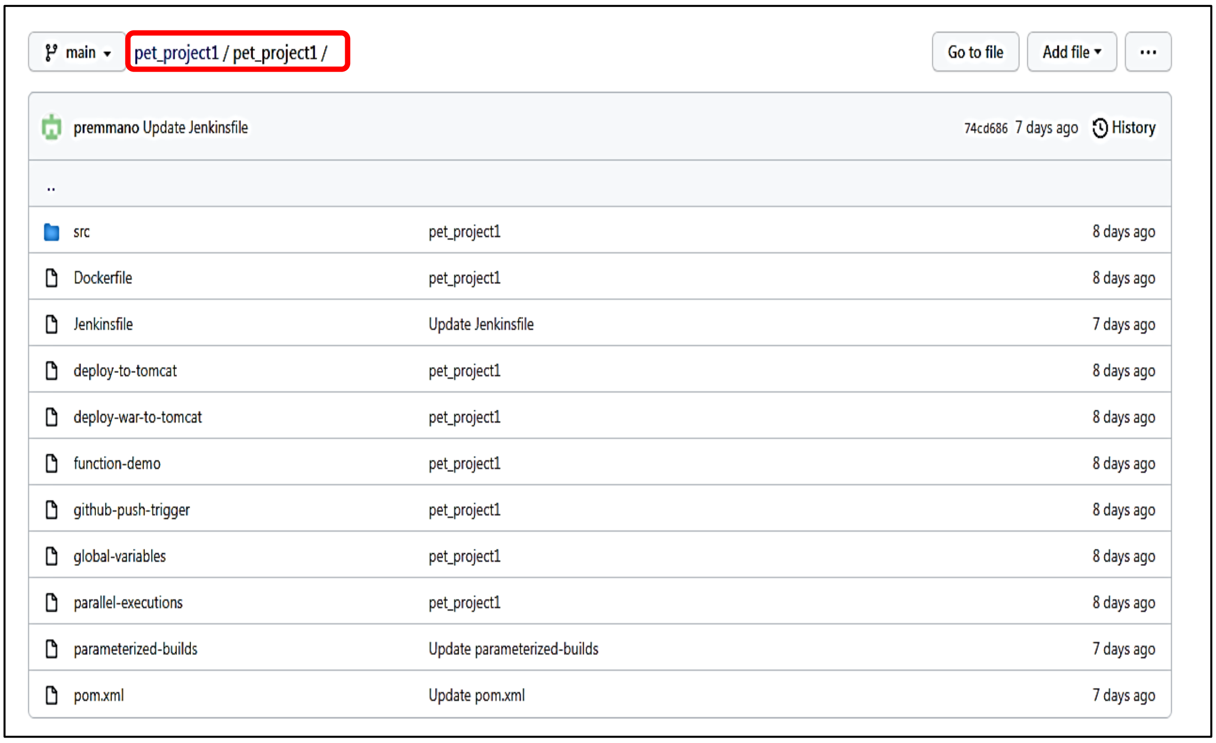
- git remote add origin https://github.com/premmano/pet_project1.git
- git push origin master



```
[root@ip-172-31-0-208 pom]# git remote add origin https://github.com/premmano/pet_project1.git
[root@ip-172-31-0-208 pom]# git remote -v
origin  https://github.com/premmano/pet_project1.git (fetch)
origin  https://github.com/premmano/pet_project1.git (push)
```



```
[root@ip-172-31-0-208 pom]# git push origin master
Username for 'https://github.com': premmano
Password for 'https://premmano@github.com':
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 2 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 876 bytes | 876.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/premmano/pet_project1.git
 * [new branch]      master -> master
[root@ip-172-31-0-208 pom]#
```



main   pet_project1 / pet_project1 /   Go to file   Add file   ...

premmano Update Jenkinsfile   74cd686 7 days ago   History

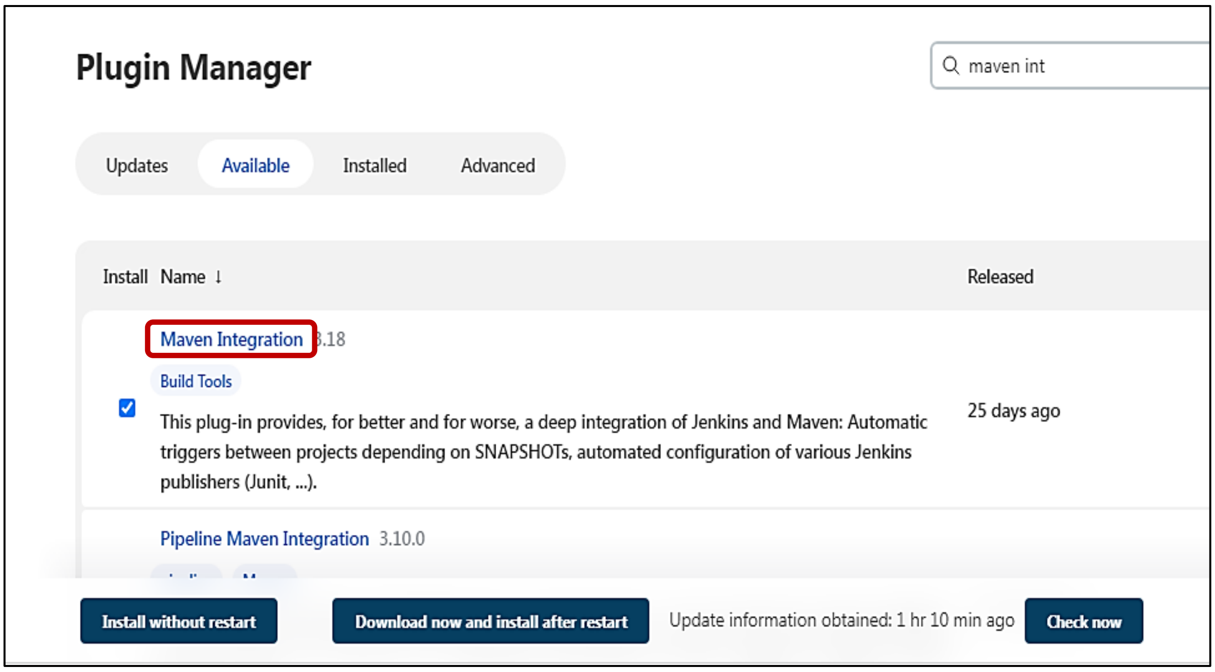
..

src	pet_project1	8 days ago
Dockerfile	pet_project1	8 days ago
Jenkinsfile	Update Jenkinsfile	7 days ago
deploy-to-tomcat	pet_project1	8 days ago
deploy-war-to-tomcat	pet_project1	8 days ago
function-demo	pet_project1	8 days ago
github-push-trigger	pet_project1	8 days ago
global-variables	pet_project1	8 days ago
parallel-executions	pet_project1	8 days ago
parameterized-builds	Update parameterized-builds	7 days ago
pom.xml	Update pom.xml	7 days ago

INTEGRATING AND CONFIGURING TOOLS WITH JENKINS:

INTEGRATION OF JENKINS AND MAVEN:

# Integrate the jenkins and maven using **mavan integration** plugin.
# Go to Manage jenkins – manage plugin – available – maven integration – install without restart.



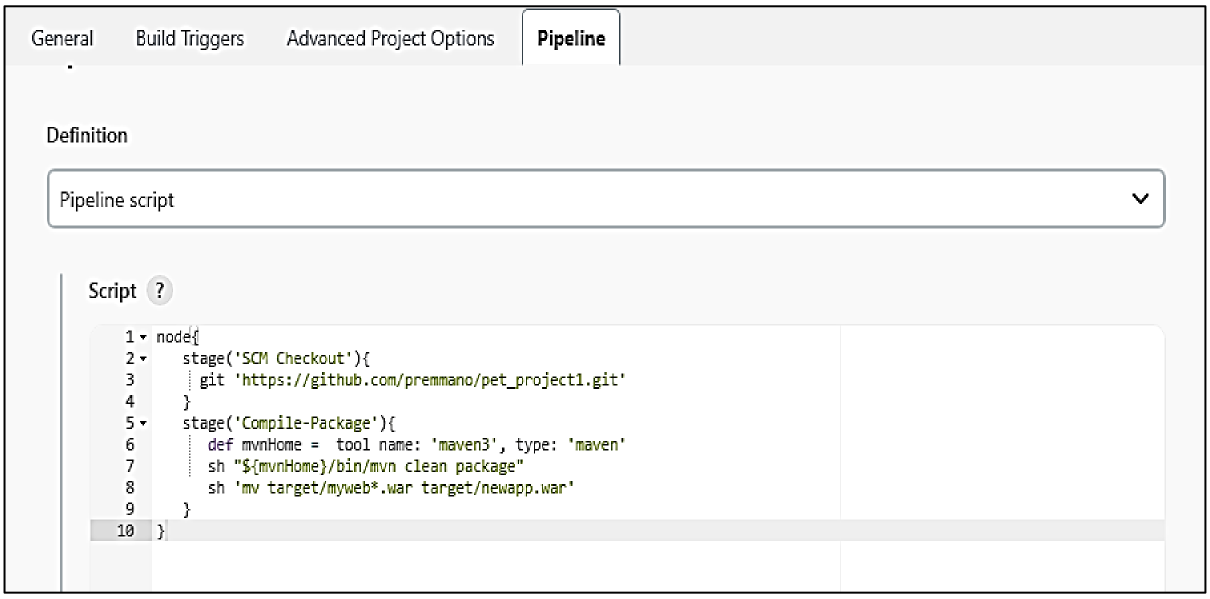
Maven Integration 3.18
Build Tools
This plug-in provides, for better and for worse, a deep integration of Jenkins and Maven: Automatic triggers between projects depending on SNAPSHOTs, automated configuration of various Jenkins publishers (Junit, ...).
25 days ago

Pipeline Maven Integration 3.10.0

Install without restart
Download now and install after restart
Update information obtained: 1 hr 10 min ago
Check now

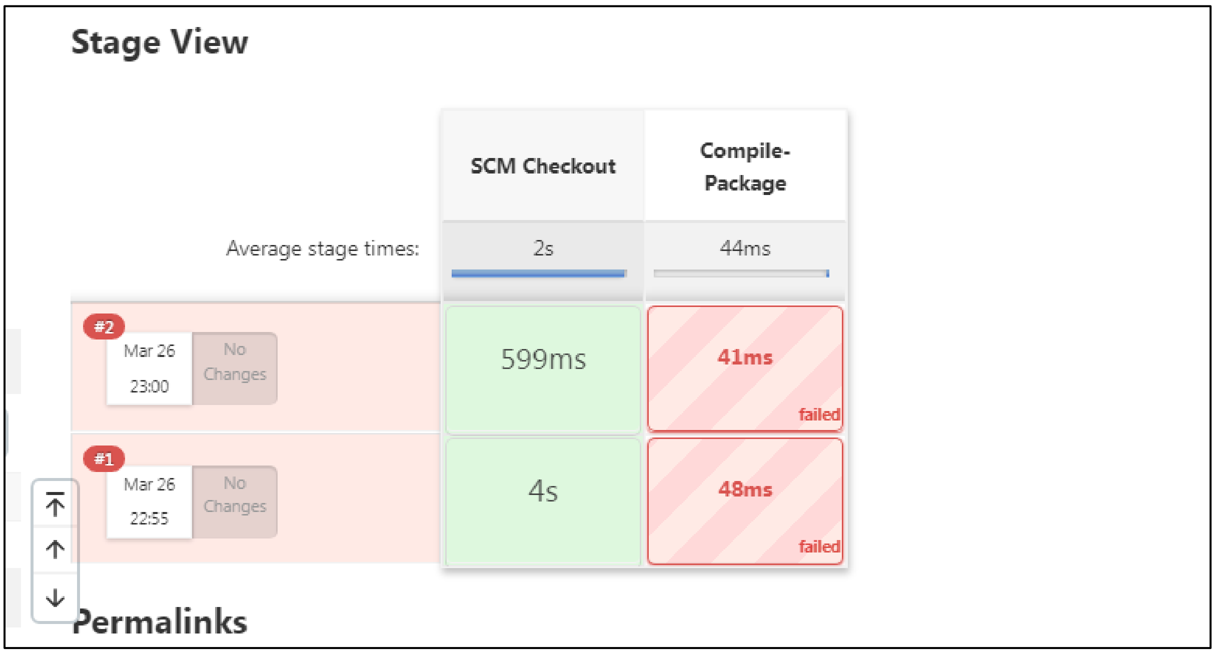
# Put the jenkins file script to pipeline.
# Go to New – projectCICD – pipeline – pipeline script

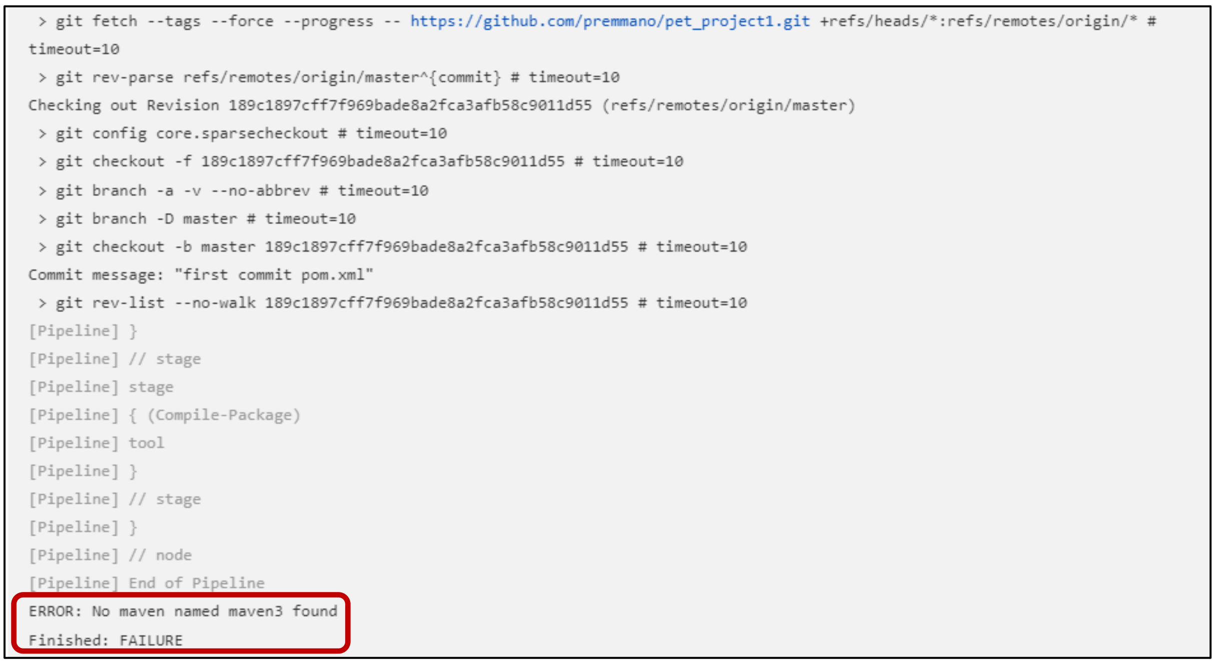
```
node{
   stage('SCM Checkout'){
     git 'https://github.com/damodaranj/my-app.git'
   }
   stage('Compile-Package'){
      def mvnHome =  tool name: 'maven3', type: 'maven'
      sh "${mvnHome}/bin/mvn clean package"
                sh 'mv target/myweb*.war target/newapp.war'
   }
```



```
node{
    stage('SCM Checkout'){
      git 'https://github.com/premmano/pet_project1.git'
    }
    stage('Compile-Package'){
      def mvnHome =  tool name: 'maven3', type: 'maven'
      sh "${mvnHome}/bin/mvn clean package"
      sh 'mv target/myweb*.war target/newapp.war'
    }
}
```

ERROR 3:





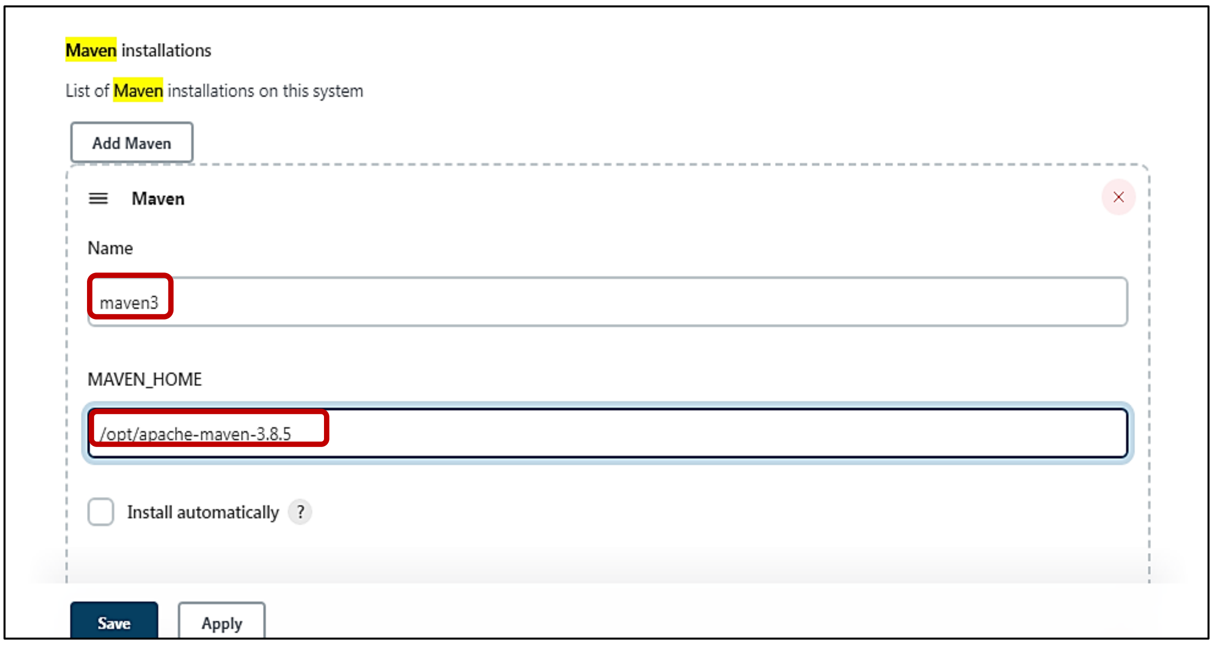
```
 > git fetch --tags --force --progress -- https://github.com/premmano/pet_project1.git +refs/heads/*:refs/remotes/origin/* #
timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 189c1897cff7f969bade8a2fca3afb58c9011d55 (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 189c1897cff7f969bade8a2fca3afb58c9011d55 # timeout=10
 > git branch -a -v --no-abbrev # timeout=10
 > git branch -D master # timeout=10
 > git checkout -b master 189c1897cff7f969bade8a2fca3afb58c9011d55 # timeout=10
Commit message: "first commit pom.xml"
 > git rev-list --no-walk 189c1897cff7f969bade8a2fca3afb58c9011d55 # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Compile-Package)
[Pipeline] tool
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
ERROR: No maven named maven3 found
Finished: FAILURE
```

TROUBLESHOOT:

# It is required to specify the maven path on the jenkins console to fix the error.

# Go to jenkins console - Manage jenkins – global tool configuration - maven

Maven path specify:


Maven installations

List of Maven installations on this system

Add Maven

Maven

Name

maven3

MAVEN_HOME

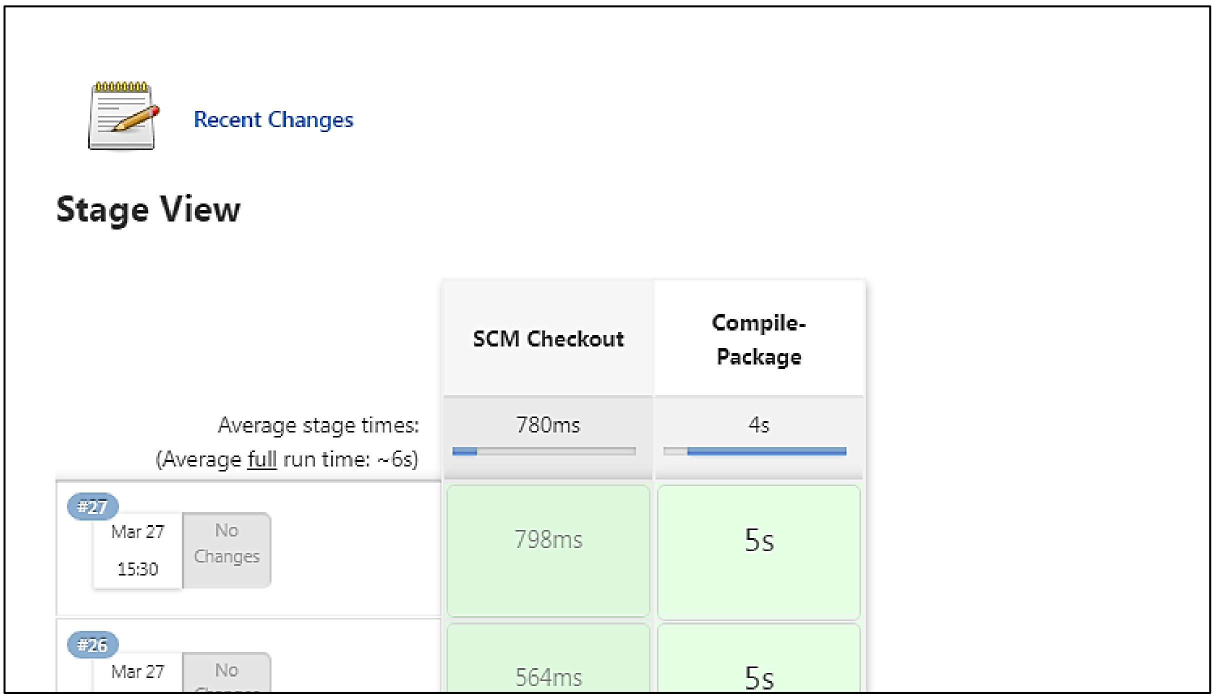
/opt/apache-maven-3.8.5

Install automatically

Save Apply

Run the script:

# Go to New – projectCICD – pipeline – pipeline script – add the script - save

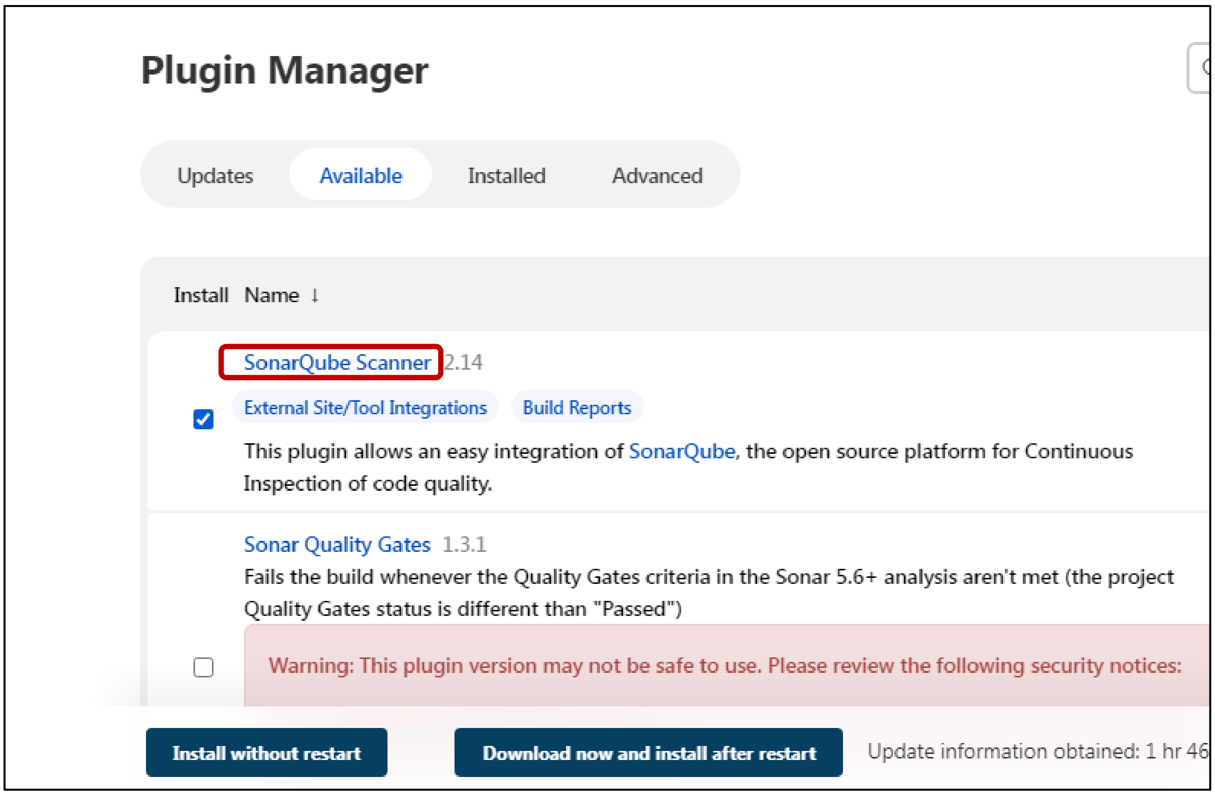

Recent Changes

Stage View

	SCM Checkout	Compile-Package
Average stage times: (Average full run time: ~6s)	780ms	4s
#27 Mar 27 15:30 No Changes	798ms	5s
#26 Mar 27 No	564ms	5s

INTEGRATION OF JENKINS AND SONARQUBE:

# Open jenkins console.
# Integrate jenkins and sonarqube by **SonarQube Scanner plugin.**
# Go to Jenkins console - Manage jenkins – manage plugins – available – SonarQube sscanner – install without restart.


Plugin Manager

Updates | Available | Installed | Advanced

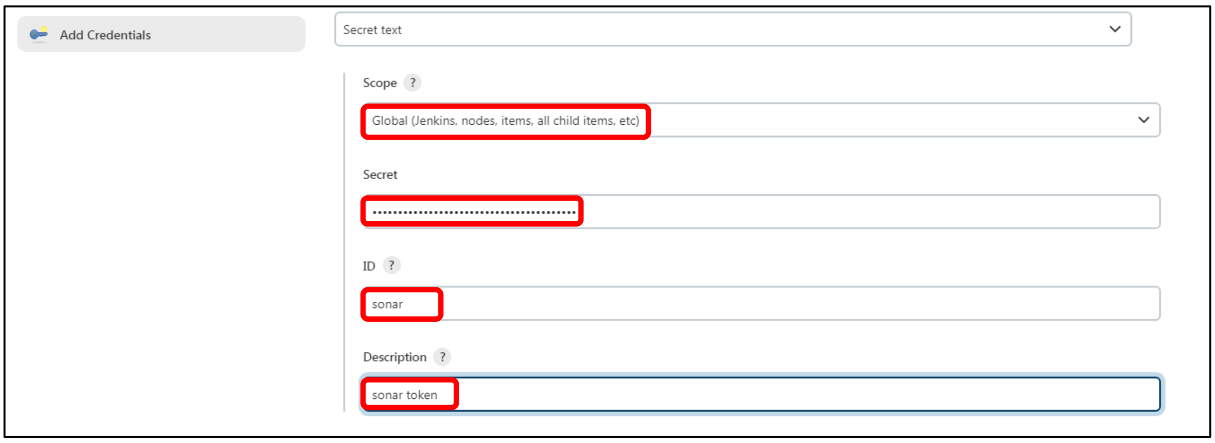
Install Name ↓

- [✓] SonarQube Scanner 2.14
  External Site/Tool Integrations | Build Reports
  This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.
- [ ] Sonar Quality Gates 1.3.1
  Fails the build whenever the Quality Gates criteria in the Sonar 5.6+ analysis aren't met (the project Quality Gates status is different than "Passed")
  Warning: This plugin version may not be safe to use. Please review the following security notices:

Install without restart | Download now and install after restart | Update information obtained: 1 hr 46

# SonarQube credentials should add on jenkins server.

# Go to jenkins console - Manage jenkins – manage credentials – jenkins – Global credentials – add credentials
# Scope – Global
# Secret – Paste sonarqube token
# ID – sonar
# Description – sonar token


Add Credentials

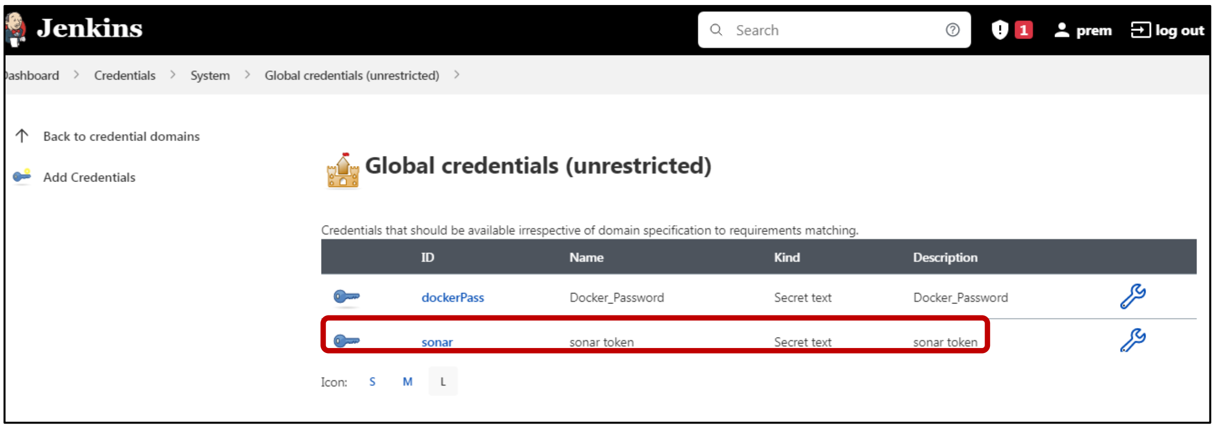
Secret text

Scope: Global (Jenkins, nodes, items, all child items, etc)

Secret: ··

ID: sonar

Description: sonar token



Jenkins
Search
1
prem
log out
ashboard > Credentials > System > Global credentials (unrestricted) >
Back to credential domains
Add Credentials

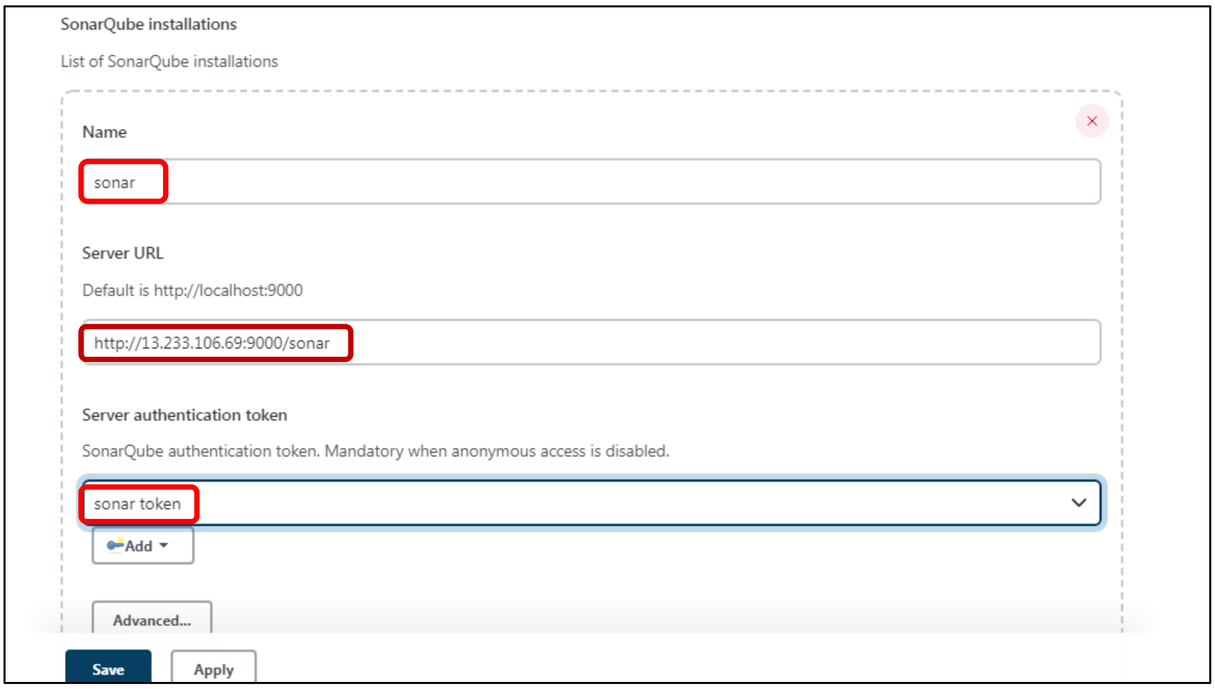
Global credentials (unrestricted)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
dockerPass	Docker_Password	Secret text	Docker_Password
sonar	sonar token	Secret text	sonar token

Icon: S M L

# Go to Jenkins console - Manage jenkins – configure system – Sonarqube servers – sonar installations.
# Name – sonar
# Server url – http://sonarqube server ip:9000/sonar (http://13.233.106.69:9000/sonar)
# Server authentication token – sonar token



SonarQube installations

List of SonarQube installations

Name: sonar

Server URL

Default is http://localhost:9000

http://13.233.106.69:9000/sonar

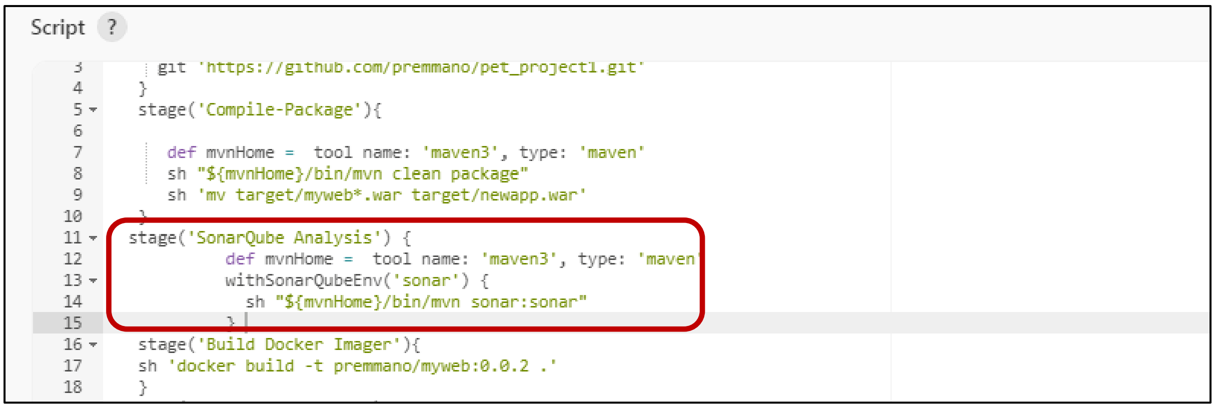
Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

sonar token

# run the script

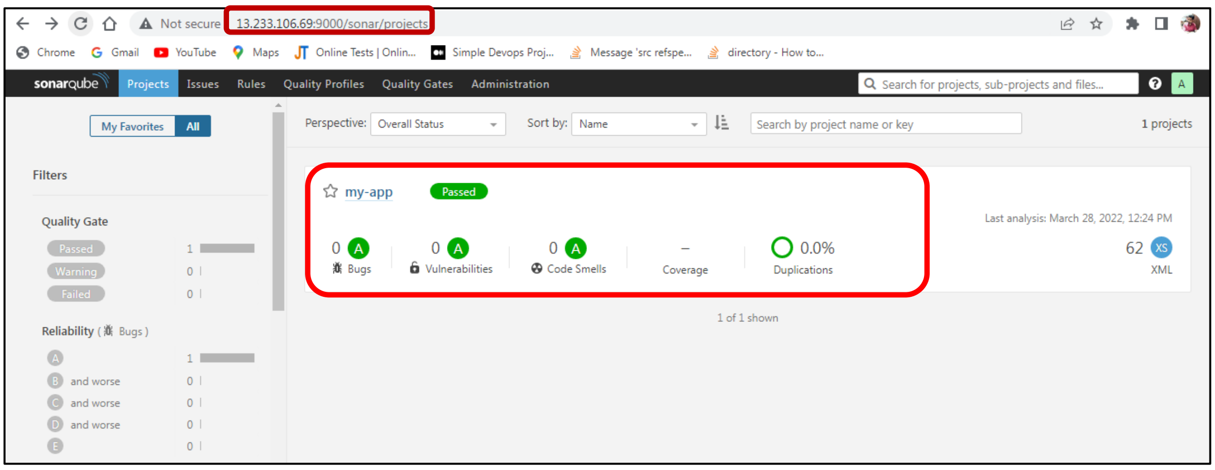
# Go to New – projectCICD – pipeline – pipeline script – add the script – save.



Script ?

```
       git 'https://github.com/premmano/pet_project1.git'
    }
    stage('Compile-Package'){

      def mvnHome =  tool name: 'maven3', type: 'maven'
      sh "${mvnHome}/bin/mvn clean package"
      sh 'mv target/myweb*.war target/newapp.war'
    }
   stage('SonarQube Analysis') {
            def mvnHome =  tool name: 'maven3', type: 'maven
            withSonarQubeEnv('sonar') {
              sh "${mvnHome}/bin/mvn sonar:sonar"
            }
    stage('Build Docker Imager'){
    sh 'docker build -t premmano/myweb:0.0.2 .'
    }
```

# Go to sonarqube console with port no 9000.



Not secure 13.233.106.69:9000/sonar/projects

Perspective: Overall Status Sort by: Name Search by project name or key 1 projects

Filters

Quality Gate
Passed 1
Warning 0
Failed 0

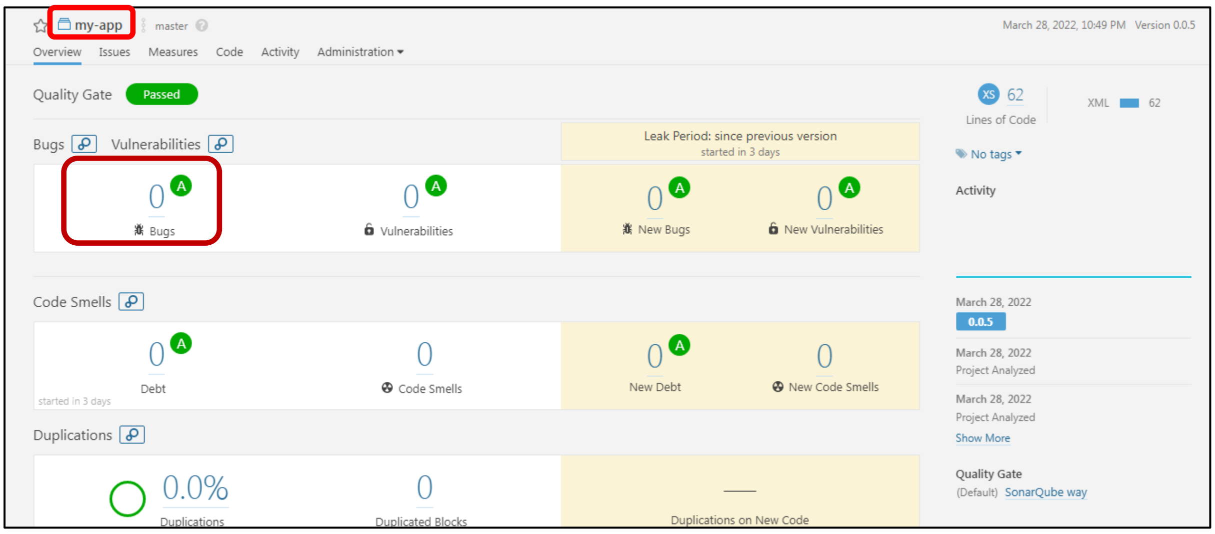
Reliability (Bugs)
A 1
B and worse 0
C and worse 0
D and worse 0
E 0

my-app Passed

0 A Bugs | 0 A Vulnerabilities | 0 A Code Smells | – Coverage | 0.0% Duplications

Last analysis: March 28, 2022, 12:24 PM
62 XS XML

1 of 1 shown



my-app master March 28, 2022, 10:49 PM Version 0.0.5

Quality Gate Passed

Bugs Vulnerabilities
Leak Period: since previous version
started in 3 days

0 A Bugs | 0 A Vulnerabilities | 0 A New Bugs | 0 A New Vulnerabilities

Code Smells
0 A Debt | 0 Code Smells | 0 A New Debt | 0 New Code Smells
started in 3 days

Duplications
0.0% Duplications | 0 Duplicated Blocks | — Duplications on New Code

XS 62 Lines of Code XML 62
No tags
Activity
March 28, 2022 0.0.5
March 28, 2022 Project Analyzed
March 28, 2022 Project Analyzed
Show More
Quality Gate
(Default) SonarQube way

DOCKER BUIlD:

Docker File:

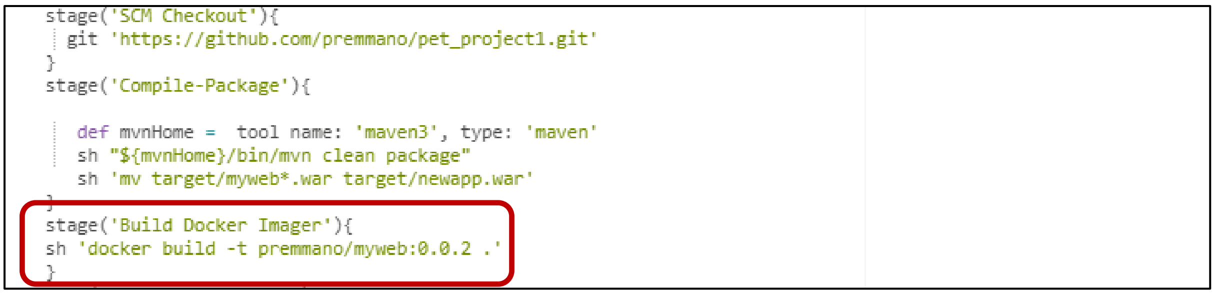
FROM tomcat:8
# Take the war and copy to webapps of tomcat
COPY target/newapp.war /usr/local/tomcat/webapps/

# Run the script.

# Go to New – projectCICD – pipeline – pipeline script – add the script - save

stage('Build Docker Imager'){
   sh 'docker build -t premmano/myweb:0.0.2 .'
   }

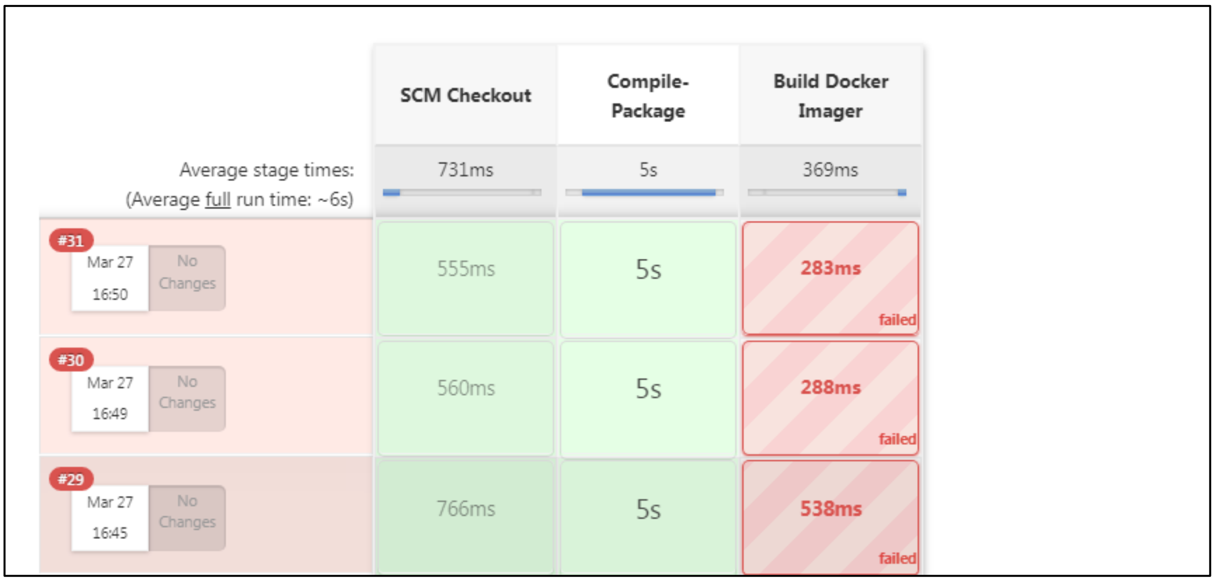
# This script converts the docker file into a docker image called as premmano/myweb.



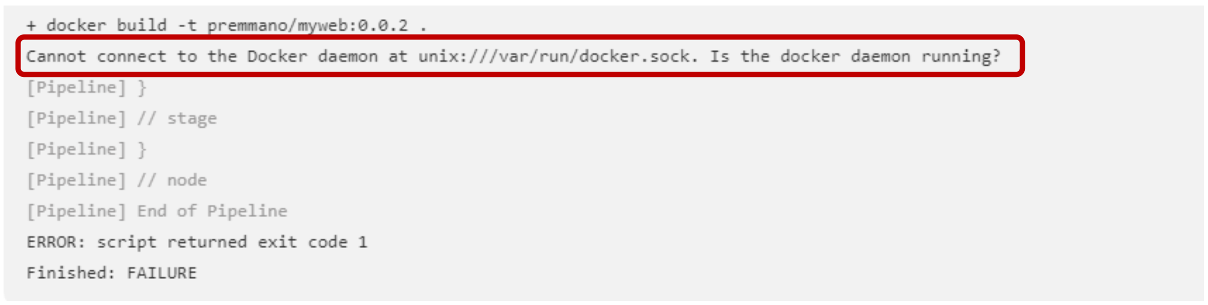
```
stage('SCM Checkout'){
  git 'https://github.com/premmano/pet_project1.git'
}
stage('Compile-Package'){

   def mvnHome =  tool name: 'maven3', type: 'maven'
   sh "${mvnHome}/bin/mvn clean package"
   sh 'mv target/myweb*.war target/newapp.war'
}
stage('Build Docker Imager'){
sh 'docker build -t premmano/myweb:0.0.2 .'
}
```

ERROR 4:

# Can not connect to the docker daemon at unix:///var/run/docker.sock.



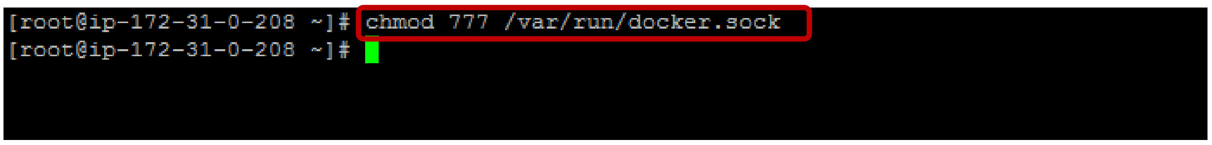
	SCM Checkout	Compile-Package	Build Docker Imager
Average stage times: (Average full run time: ~6s)	731ms	5s	369ms
#31 Mar 27 16:50 No Changes	555ms	5s	283ms failed
#30 Mar 27 16:49 No Changes	560ms	5s	288ms failed
#29 Mar 27 16:45 No Changes	766ms	5s	538ms failed



```
+ docker build -t premmano/myweb:0.0.2 .
Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
ERROR: script returned exit code 1
Finished: FAILURE
```

TROUBLE SHOOT:

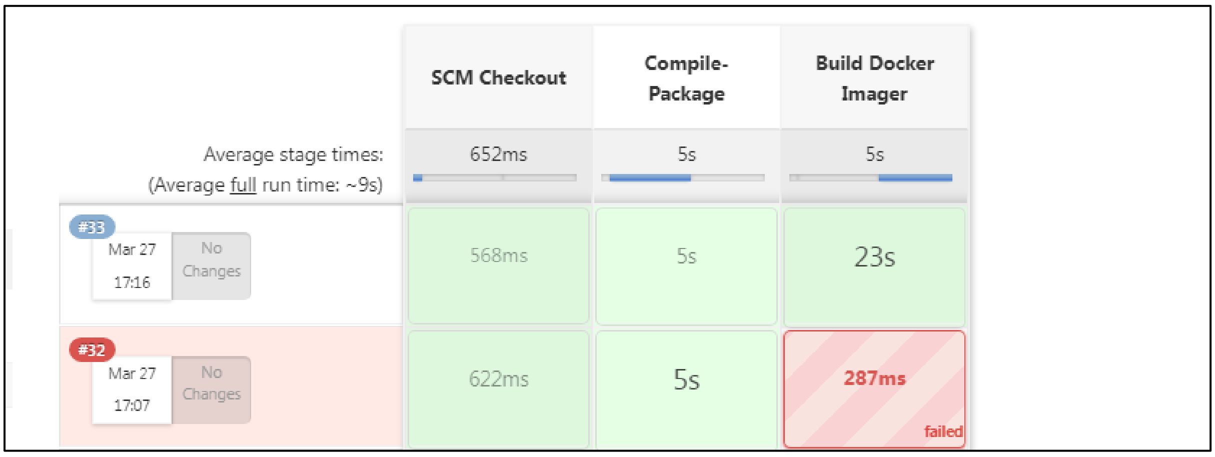
Give full permission to fix the error.



```
[root@ip-172-31-0-208 ~]# chmod 777 /var/run/docker.sock
[root@ip-172-31-0-208 ~]#
```

# Run the jenkins file script.

# Go to New – project_CICD – pipeline – pipeline script – add the script - save



	SCM Checkout	Compile-Package	Build Docker Imager
Average stage times: (Average full run time: ~9s)	652ms	5s	5s
#33 Mar 27 17:16 No Changes	568ms	5s	23s
#32 Mar 27 17:07 No Changes	622ms	5s	287ms failed

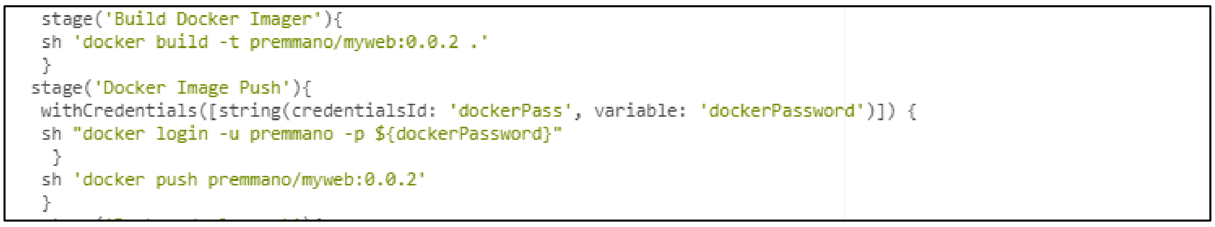
DOCKER PUSH:

# This script is to push the docker image to dockerhub. The Docker hubs user name and password are essential to push the image to the docker hub.

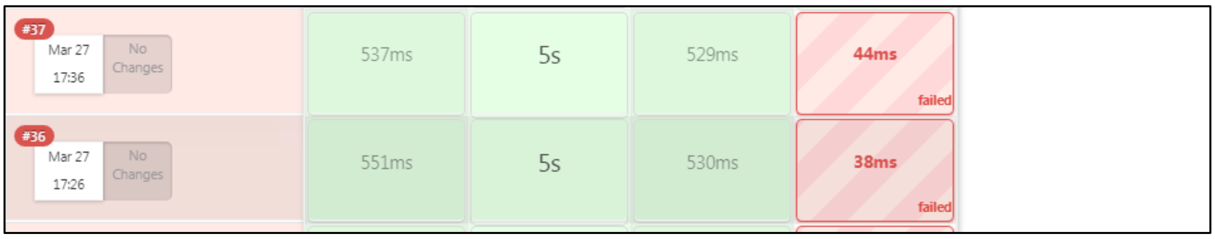
```
stage('Docker Image Push'){
  withCredentials([string(credentialsId: 'dockerPass', variable: 'dockerPassword')]) {
  sh "docker login -u premmano -p ${dockerPassword}"
   }
  sh 'docker push premmano/myweb:0.0.2'
  }
```

# Run the script.

# Go to new – projectCICD – pipeline – pipeline script – add the script - save



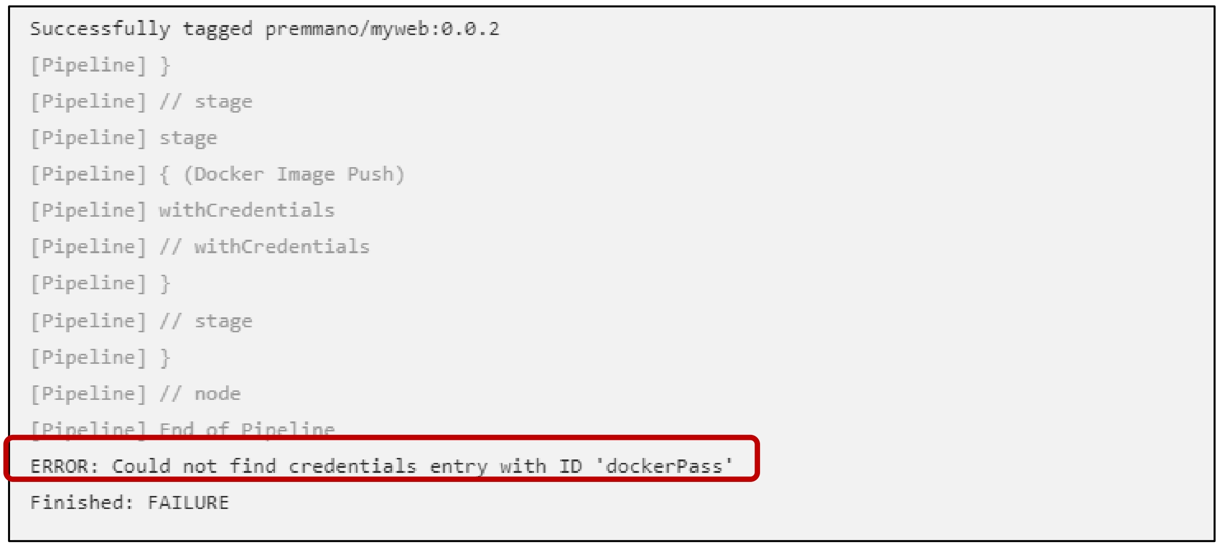
```
   stage('Build Docker Imager'){
   sh 'docker build -t premmano/myweb:0.0.2 .'
   }
  stage('Docker Image Push'){
   withCredentials([string(credentialsId: 'dockerPass', variable: 'dockerPassword')]) {
   sh "docker login -u premmano -p ${dockerPassword}"
    }
   sh 'docker push premmano/myweb:0.0.2'
   }
```



#37 Mar 27 17:36 No Changes	537ms	5s	529ms	44ms failed
#36 Mar 27 17:26 No Changes	551ms	5s	530ms	38ms failed

ERROR 5:

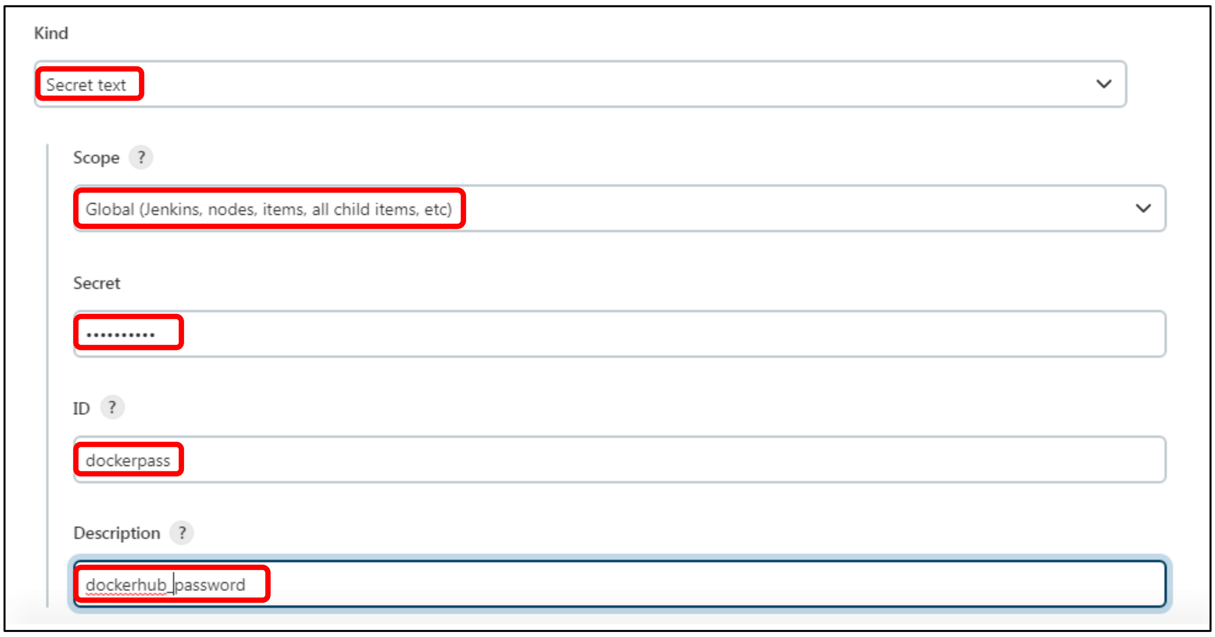
# Could not find any credentials entry with ID “dockerPass” on jenkins server.



```
Successfully tagged premmano/myweb:0.0.2
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Docker Image Push)
[Pipeline] withCredentials
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
ERROR: Could not find credentials entry with ID 'dockerPass'
Finished: FAILURE
```

TROUBLE SHOOT:

# Set the docker credential id on jenkins server to integrate jenkins and docker.
# Go to configuration – manage jenkins – manage credentials – jenkins – global credentials – add credentials.
# kind - secret text
# Scope – Global
# Secret – Dockerhub password
# ID – dockerpass
# Description – dockerhub_password



Kind

Secret text

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

Secret

••••••••••

ID ?

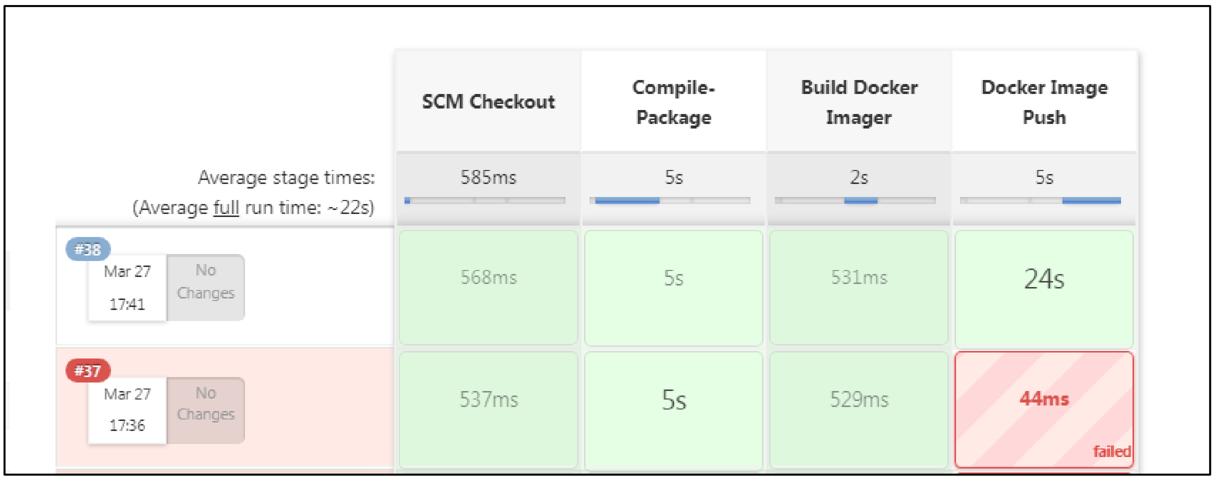
dockerpass

Description ?

dockerhub_password

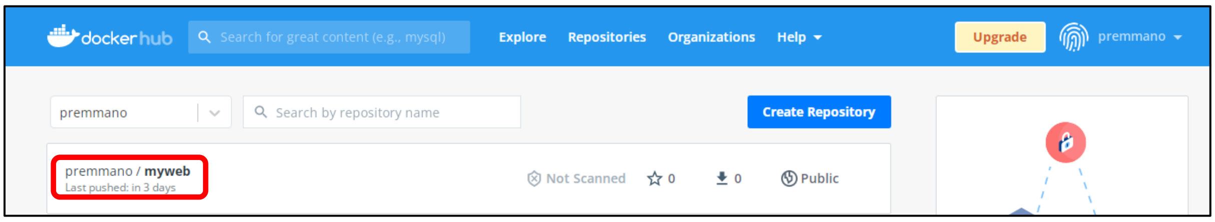
# Run the script.

# Go to New – projectCICD – pipeline – pipeline script – add the script - save

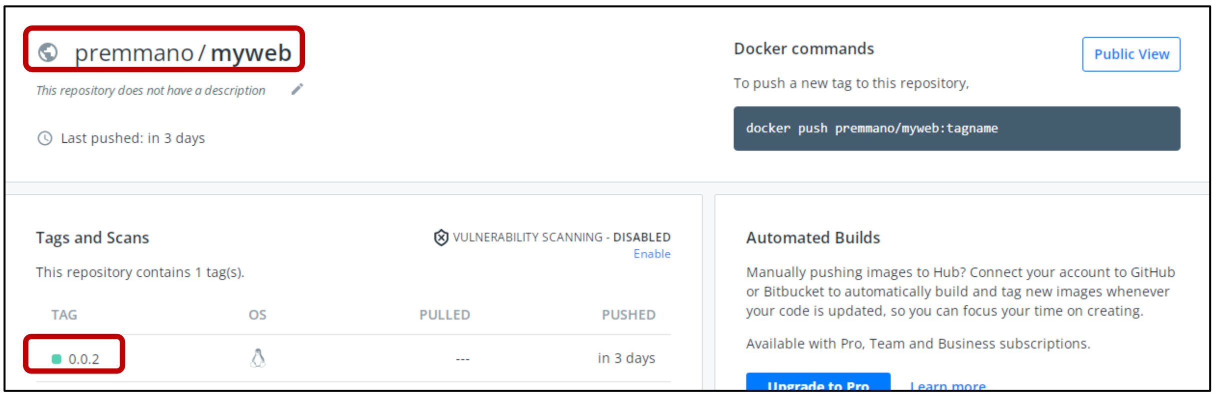


	SCM Checkout	Compile-Package	Build Docker Imager	Docker Image Push
Average stage times: (Average full run time: ~22s)	585ms	5s	2s	5s
#38 Mar 27 17:41 No Changes	568ms	5s	531ms	24s
#37 Mar 27 17:36 No Changes	537ms	5s	529ms	44ms failed

# Go to docker hub and check the image.



premmano / myweb
Last pushed: in 3 days
Not Scanned 0 0 Public



premmano / myweb
This repository does not have a description
Last pushed: in 3 days

Docker commands
To push a new tag to this repository,

```
docker push premmano/myweb:tagname
```

Tags and Scans
VULNERABILITY SCANNING - DISABLED
This repository contains 1 tag(s).

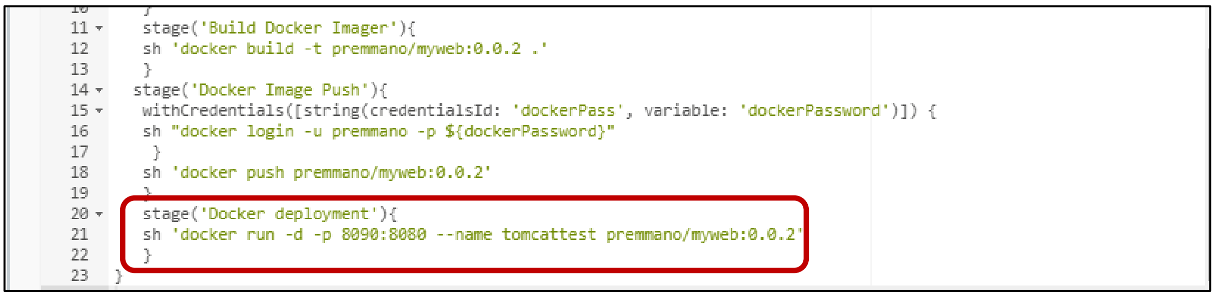
TAG	OS	PULLED	PUSHED
0.0.2		---	in 3 days

Automated Builds
Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.
Available with Pro, Team and Business subscriptions.

DOCKER DEPOYMENT:

```
stage('Docker deployment'){
  sh 'docker run -d -p 8090:8080 --name tomcattest premmano/myweb:0.0.2'
  }
```

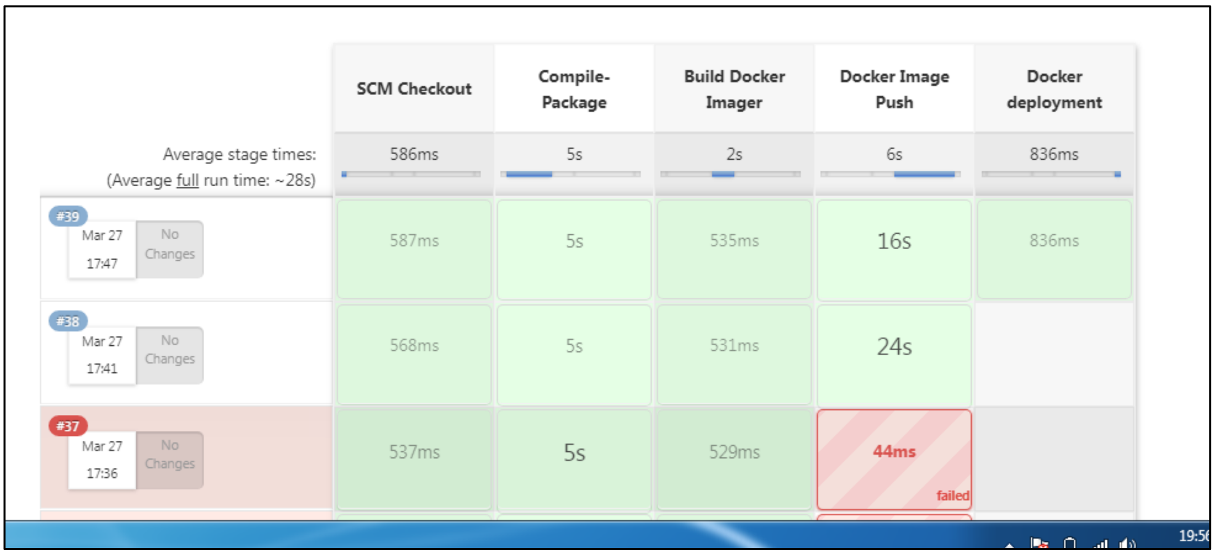
# This script is to launch the container called tamcattest.



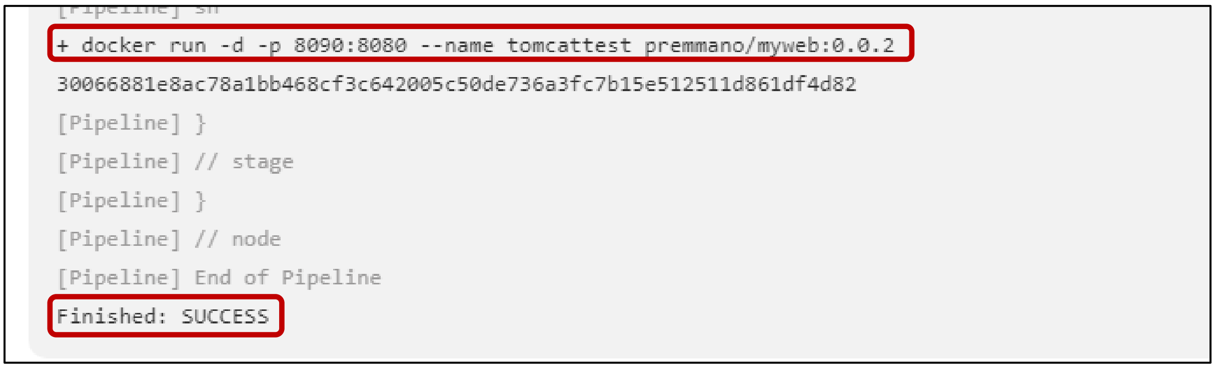
```
   }
   stage('Build Docker Imager'){
   sh 'docker build -t premmano/myweb:0.0.2 .'
   }
  stage('Docker Image Push'){
   withCredentials([string(credentialsId: 'dockerPass', variable: 'dockerPassword')]) {
   sh "docker login -u premmano -p ${dockerPassword}"
    }
   sh 'docker push premmano/myweb:0.0.2'
   }
   stage('Docker deployment'){
   sh 'docker run -d -p 8090:8080 --name tomcattest premmano/myweb:0.0.2'
   }
}
```

# Run the script.

# New – projectCICD – pipeline – pipeline script – add the script - save



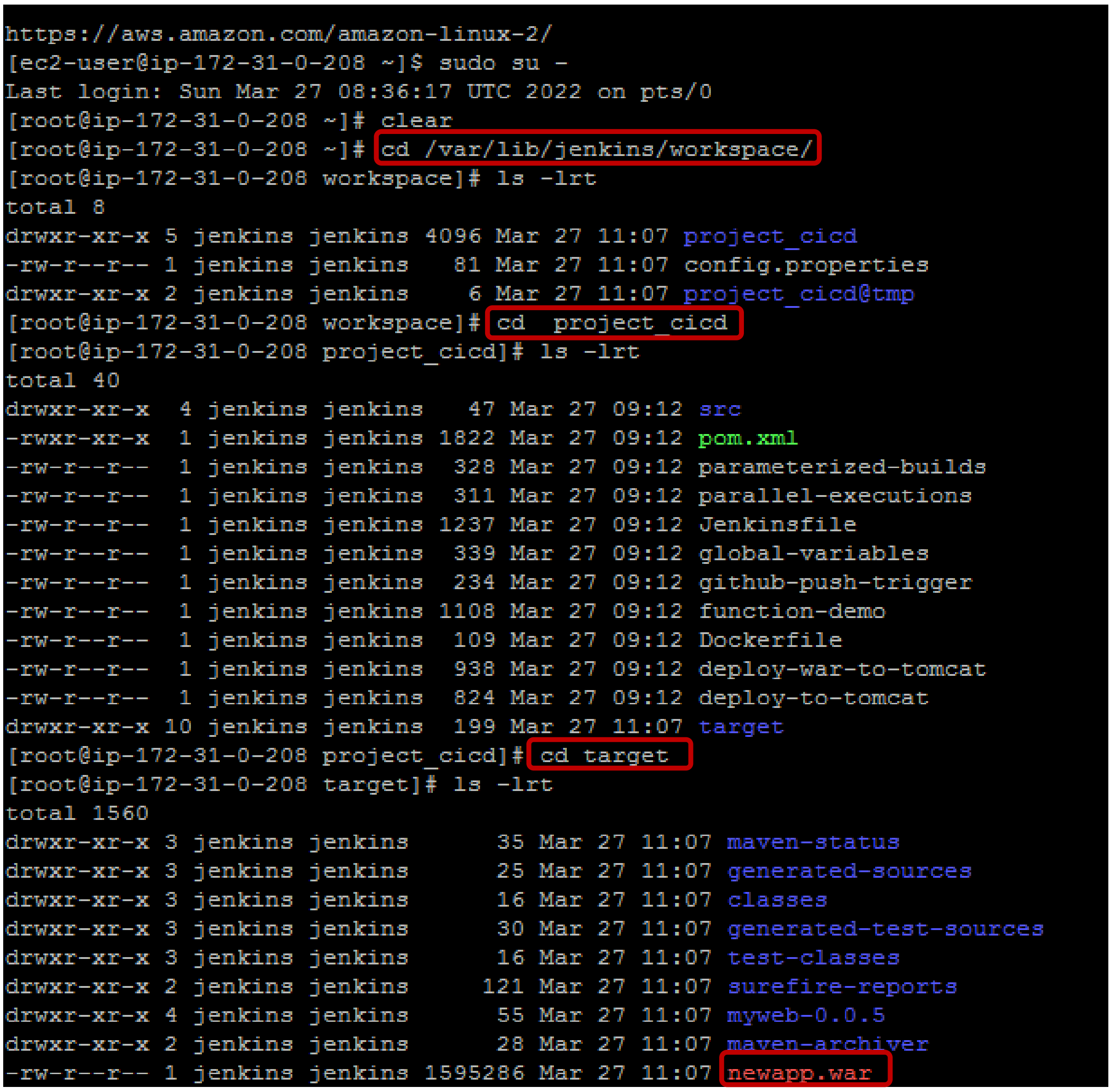
	SCM Checkout	Compile-Package	Build Docker Imager	Docker Image Push	Docker deployment
Average stage times: (Average full run time: ~28s)	586ms	5s	2s	6s	836ms
#39 Mar 27 17:47 No Changes	587ms	5s	535ms	16s	836ms
#38 Mar 27 17:41 No Changes	568ms	5s	531ms	24s	
#37 Mar 27 17:36 No Changes	537ms	5s	529ms	44ms failed	



```
+ docker run -d -p 8090:8080 --name tomcattest premmano/myweb:0.0.2
30066881e8ac78a1bb468cf3c642005c50de736a3fc7b15e512511d861df4d82
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

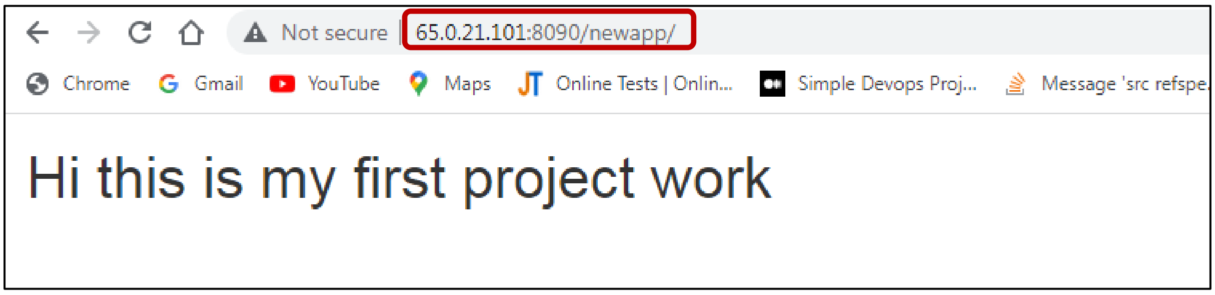
# Go to filename name: newapp.war

# cd /var/lib/jenkins/workspace/cicd_project/target/newapp.war



```
https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-0-208 ~]$ sudo su -
Last login: Sun Mar 27 08:36:17 UTC 2022 on pts/0
[root@ip-172-31-0-208 ~]# clear
[root@ip-172-31-0-208 ~]# cd /var/lib/jenkins/workspace/
[root@ip-172-31-0-208 workspace]# ls -lrt
total 8
drwxr-xr-x 5 jenkins jenkins 4096 Mar 27 11:07 project_cicd
-rw-r--r-- 1 jenkins jenkins   81 Mar 27 11:07 config.properties
drwxr-xr-x 2 jenkins jenkins    6 Mar 27 11:07 project_cicd@tmp
[root@ip-172-31-0-208 workspace]# cd  project_cicd
[root@ip-172-31-0-208 project_cicd]# ls -lrt
total 40
drwxr-xr-x  4 jenkins jenkins   47 Mar 27 09:12 src
-rwxr-xr-x  1 jenkins jenkins 1822 Mar 27 09:12 pom.xml
-rw-r--r--  1 jenkins jenkins  328 Mar 27 09:12 parameterized-builds
-rw-r--r--  1 jenkins jenkins  311 Mar 27 09:12 parallel-executions
-rw-r--r--  1 jenkins jenkins 1237 Mar 27 09:12 Jenkinsfile
-rw-r--r--  1 jenkins jenkins  339 Mar 27 09:12 global-variables
-rw-r--r--  1 jenkins jenkins  234 Mar 27 09:12 github-push-trigger
-rw-r--r--  1 jenkins jenkins 1108 Mar 27 09:12 function-demo
-rw-r--r--  1 jenkins jenkins  109 Mar 27 09:12 Dockerfile
-rw-r--r--  1 jenkins jenkins  938 Mar 27 09:12 deploy-war-to-tomcat
-rw-r--r--  1 jenkins jenkins  824 Mar 27 09:12 deploy-to-tomcat
drwxr-xr-x 10 jenkins jenkins  199 Mar 27 11:07 target
[root@ip-172-31-0-208 project_cicd]# cd target
[root@ip-172-31-0-208 target]# ls -lrt
total 1560
drwxr-xr-x 3 jenkins jenkins      35 Mar 27 11:07 maven-status
drwxr-xr-x 3 jenkins jenkins      25 Mar 27 11:07 generated-sources
drwxr-xr-x 3 jenkins jenkins      16 Mar 27 11:07 classes
drwxr-xr-x 3 jenkins jenkins      30 Mar 27 11:07 generated-test-sources
drwxr-xr-x 3 jenkins jenkins      16 Mar 27 11:07 test-classes
drwxr-xr-x 2 jenkins jenkins     121 Mar 27 11:07 surefire-reports
drwxr-xr-x 4 jenkins jenkins      55 Mar 27 11:07 myweb-0.0.5
drwxr-xr-x 2 jenkins jenkins      28 Mar 27 11:07 maven-archiver
-rw-r--r-- 1 jenkins jenkins 1595286 Mar 27 11:07 newapp.war
```

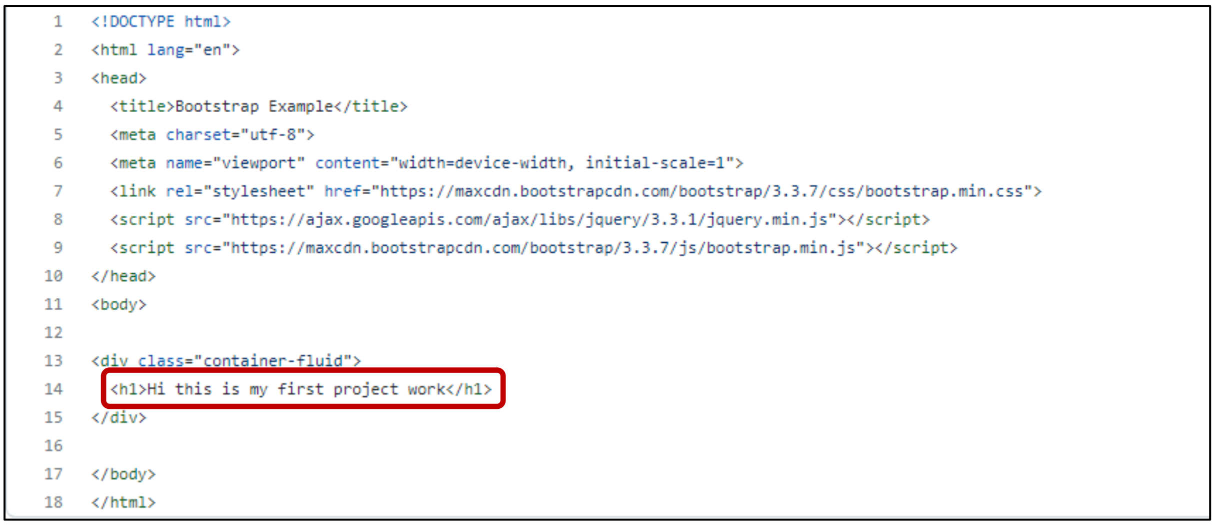
# Check: jenkins server ip:8090/newapp/



Hi this is my first project work

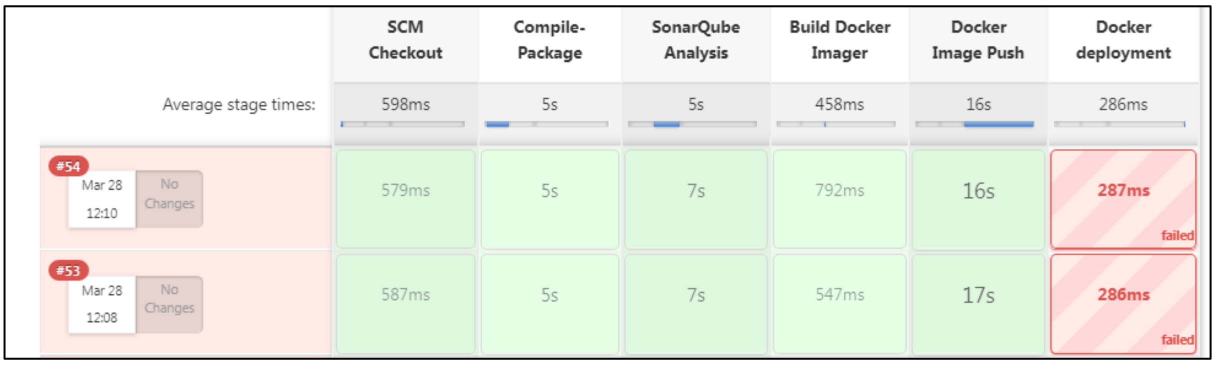
# Go to Github – Src – main – webapp - index.html

# In the index.html file mention “Hi this is my first project work”.



```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Bootstrap Example</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
</head>
<body>

<div class="container-fluid">
  <h1>Hi this is my first project work</h1>
</div>

</body>
</html>
```

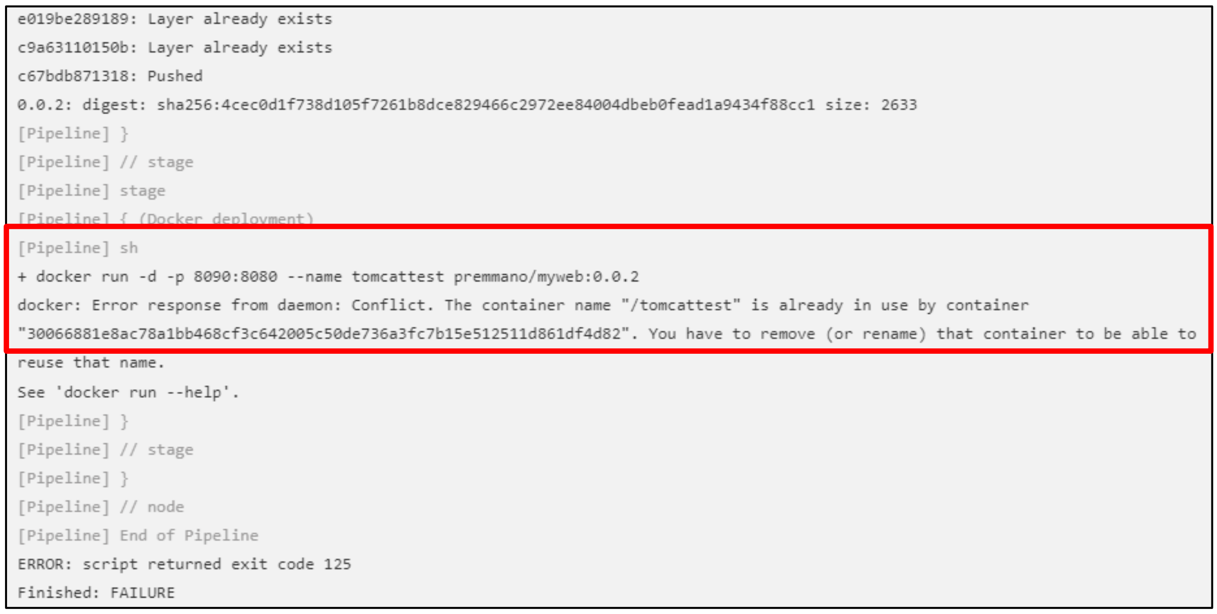
# run the script again.



	SCM Checkout	Compile-Package	SonarQube Analysis	Build Docker Imager	Docker Image Push	Docker deployment
Average stage times:	598ms	5s	5s	458ms	16s	286ms
#54 Mar 28 12:10 No Changes	579ms	5s	7s	792ms	16s	287ms failed
#53 Mar 28 12:08 No Changes	587ms	5s	7s	547ms	17s	286ms failed

ERROR 6:

# The container name “tomcattest” is already used by a existing container. So that container must be removed (or renamed) in order to reuse that name.

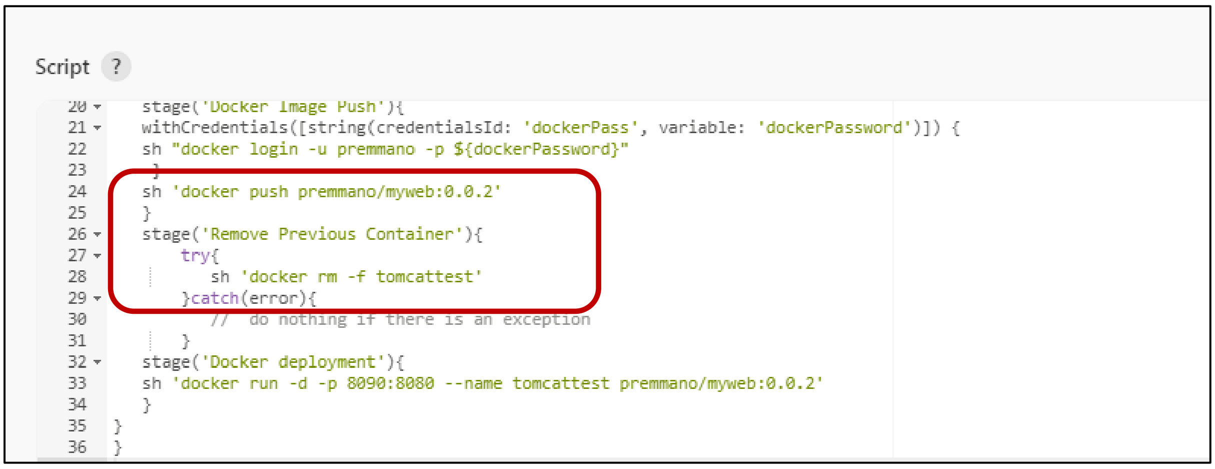


```
e019be289189: Layer already exists
c9a63110150b: Layer already exists
c67bdb871318: Pushed
0.0.2: digest: sha256:4cec0d1f738d105f7261b8dce829466c2972ee84004dbeb0fead1a9434f88cc1 size: 2633
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Docker deployment)
[Pipeline] sh
+ docker run -d -p 8090:8080 --name tomcattest premmano/myweb:0.0.2
docker: Error response from daemon: Conflict. The container name "/tomcattest" is already in use by container
"30066881e8ac78a1bb468cf3c642005c50de736a3fc7b15e512511d861df4d82". You have to remove (or rename) that container to be able to
reuse that name.
See 'docker run --help'.
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
ERROR: script returned exit code 125
Finished: FAILURE
```

TROUBLESHOOT:

# The following script is used to remove (or rename) that container for reuse that name.

```
stage('Remove Previous Container'){
        try{
                sh 'docker rm -f tomcattest'
        }catch(error){
                //  do nothing if there is an exception
        }
```

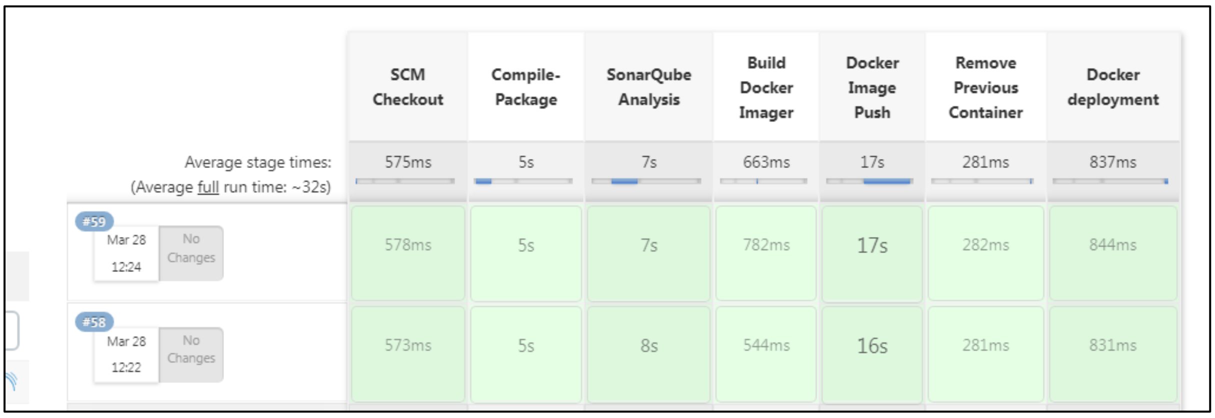


Script ?

```
    stage('Docker Image Push'){
    withCredentials([string(credentialsId: 'dockerPass', variable: 'dockerPassword')]) {
    sh "docker login -u premmano -p ${dockerPassword}"
     }
    sh 'docker push premmano/myweb:0.0.2'
    }
    stage('Remove Previous Container'){
        try{
            sh 'docker rm -f tomcattest'
        }catch(error){
            //  do nothing if there is an exception
        }
    stage('Docker deployment'){
    sh 'docker run -d -p 8090:8080 --name tomcattest premmano/myweb:0.0.2'
    }
}
}
```

# Run the pipeline

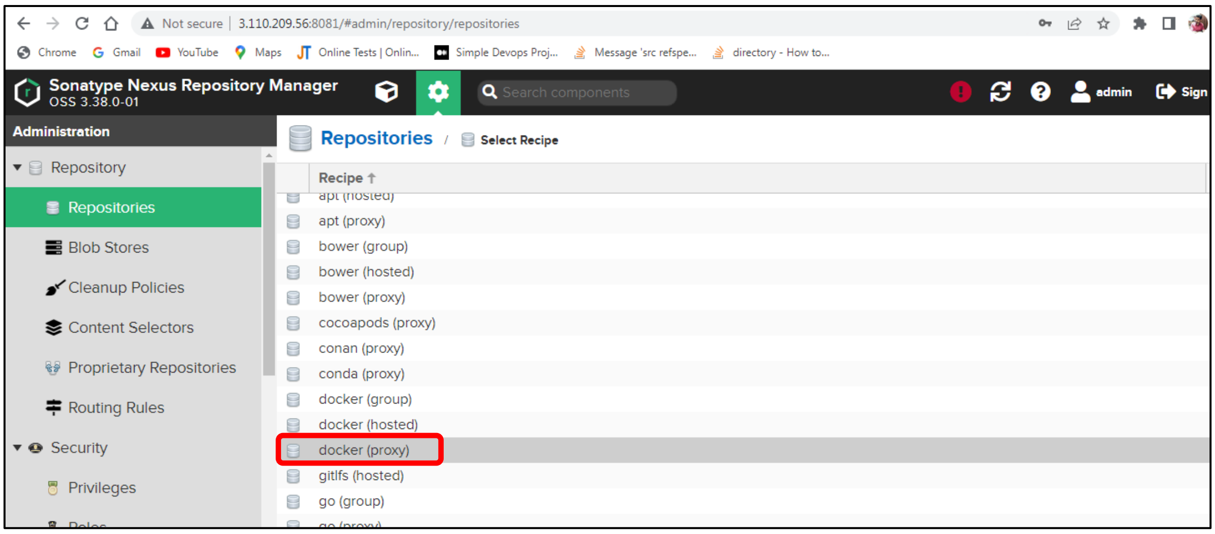
# New – projectCICD – pipeline – pipeline script – add the script - save



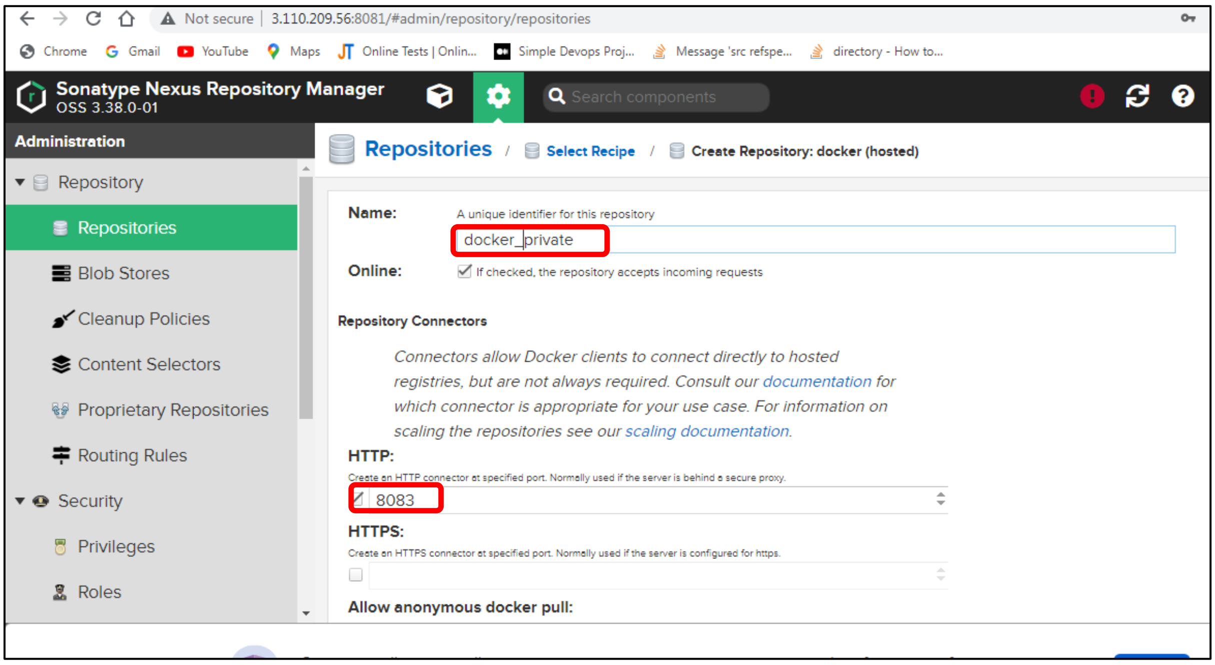
	SCM Checkout	Compile-Package	SonarQube Analysis	Build Docker Imager	Docker Image Push	Remove Previous Container	Docker deployment
Average stage times: (Average full run time: ~32s)	575ms	5s	7s	663ms	17s	281ms	837ms
#59 Mar 28 12:24 No Changes	578ms	5s	7s	782ms	17s	282ms	844ms
#58 Mar 28 12:22 No Changes	573ms	5s	8s	544ms	16s	281ms	831ms

NEXUS:

# Go to nexus console – repositories – docker(proxy) – docker(hosted).
# Name : docker private
# http: 8083
# select docker VIAP1


Repositories / Select Recipe

Recipe
apt (hosted)
apt (proxy)
bower (group)
bower (hosted)
bower (proxy)
cocoapods (proxy)
conan (proxy)
conda (proxy)
docker (group)
docker (hosted)
docker (proxy)
gitlfs (hosted)
go (group)
go (proxy)


Repositories / Select Recipe / Create Repository: docker (hosted)

Name: A unique identifier for this repository
docker_private

Online: ✓ If checked, the repository accepts incoming requests

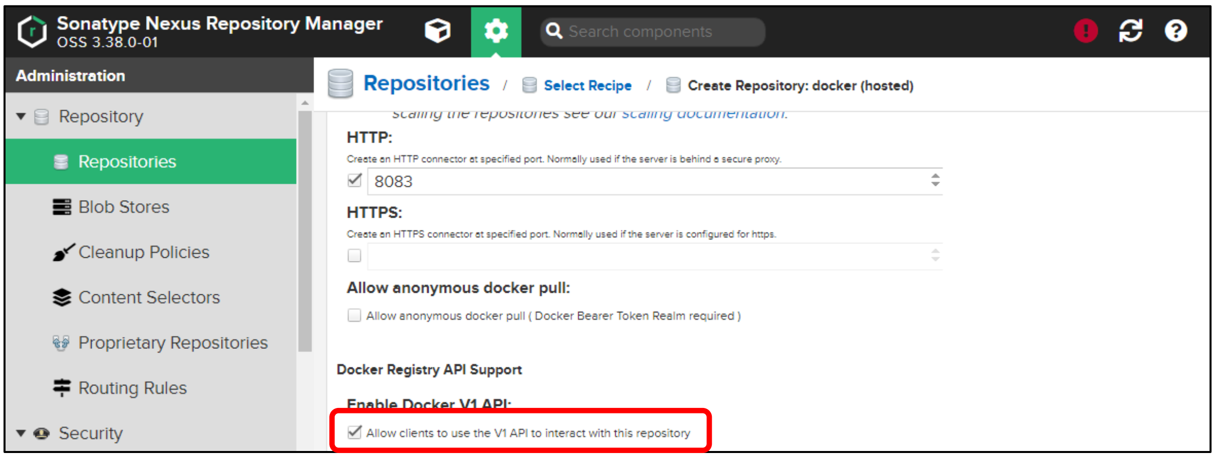
Repository Connectors

Connectors allow Docker clients to connect directly to hosted registries, but are not always required. Consult our documentation for which connector is appropriate for your use case. For information on scaling the repositories see our scaling documentation.

HTTP: Create an HTTP connector at specified port. Normally used if the server is behind a secure proxy.
✓ 8083

HTTPS: Create an HTTPS connector at specified port. Normally used if the server is configured for https.

Allow anonymous docker pull:


Repositories / Select Recipe / Create Repository: docker (hosted)

HTTP: Create an HTTP connector at specified port. Normally used if the server is behind a secure proxy.
✓ 8083

HTTPS: Create an HTTPS connector at specified port. Normally used if the server is configured for https.

Allow anonymous docker pull:
Allow anonymous docker pull (Docker Bearer Token Realm required)

Docker Registry API Support

Enable Docker V1 API:
✓ Allow clients to use the V1 API to interact with this repository

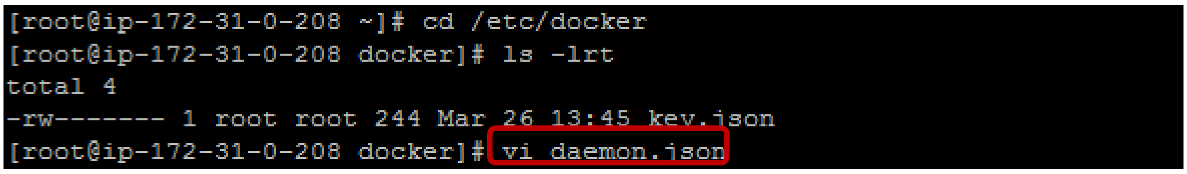
# Go to jenkins server and create a file called daemon.json.

# cd /etc/docker/daemon.json

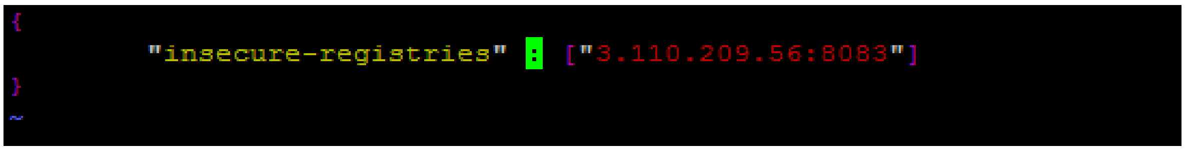
vi daemon.json
{
        “insecure-registries”:[“nexusserverIP:8083”]
}

# Restart the docker daemon by the following command.

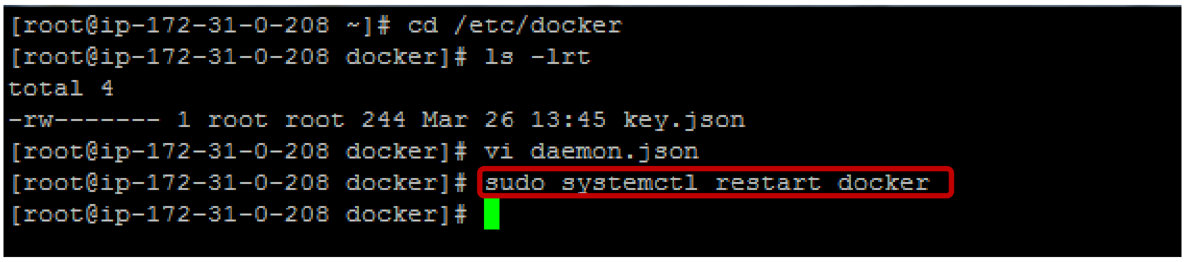
- ➢ sudo systemctl restart docker



```
[root@ip-172-31-0-208 ~]# cd /etc/docker
[root@ip-172-31-0-208 docker]# ls -lrt
total 4
-rw------- 1 root root 244 Mar 26 13:45 key.json
[root@ip-172-31-0-208 docker]# vi daemon.json
```



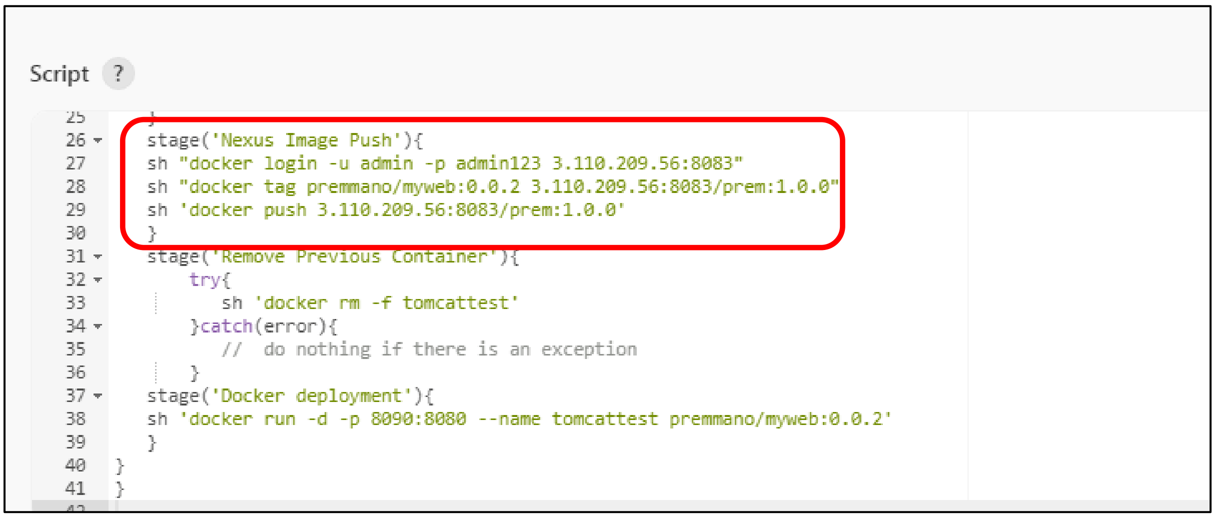
```
{
        "insecure-registries" : ["3.110.209.56:8083"]
}
~
```



```
[root@ip-172-31-0-208 ~]# cd /etc/docker
[root@ip-172-31-0-208 docker]# ls -lrt
total 4
-rw------- 1 root root 244 Mar 26 13:45 key.json
[root@ip-172-31-0-208 docker]# vi daemon.json
[root@ip-172-31-0-208 docker]# sudo systemctl restart docker
[root@ip-172-31-0-208 docker]#
```

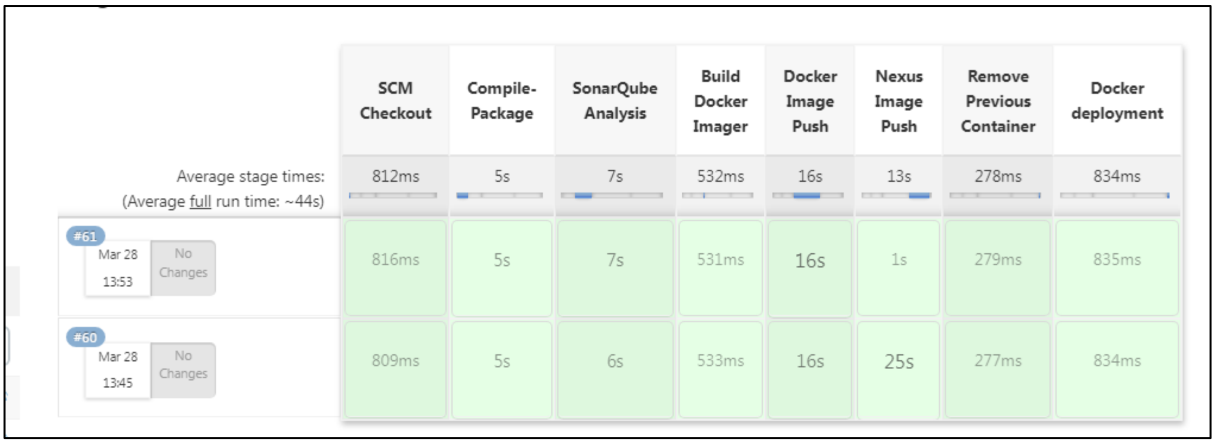
# Run the script.
# Push the image to nexus repository using this script.

```
stage('Nexus Image Push'){
sh "docker login -u admin -p admin123 3.110.209.56:8083"
sh "docker tag premmano/myweb:0.0.2 3.110.209.56:8083/prem:1.0.0"
sh 'docker push 3.110.209.56:8083/prem:1.0.0'
}
```



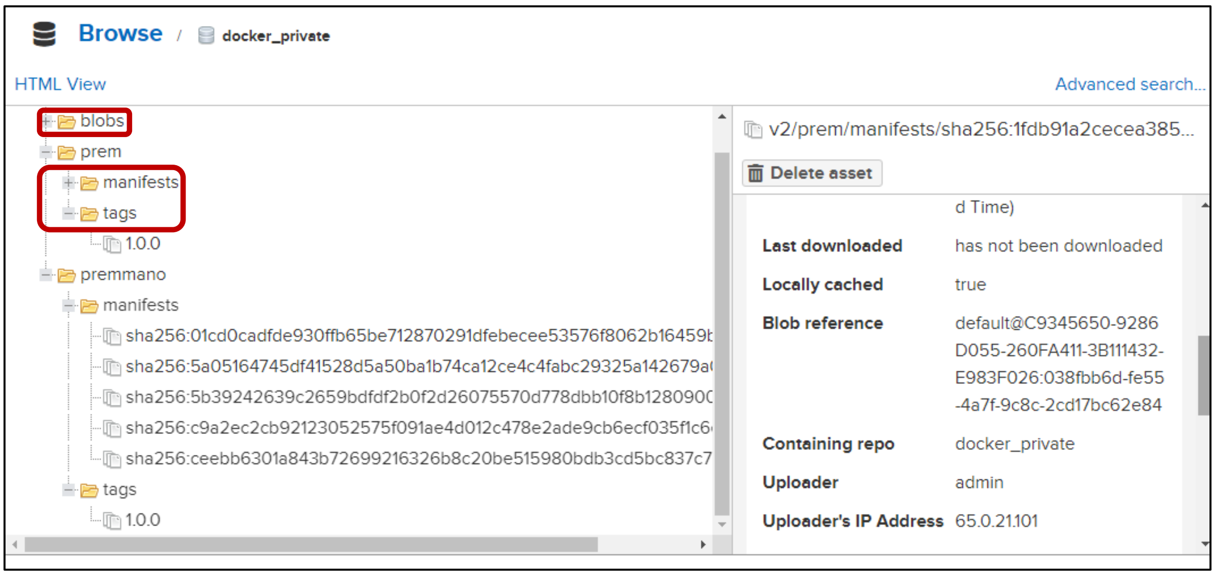
Script

```
    }
    stage('Nexus Image Push'){
    sh "docker login -u admin -p admin123 3.110.209.56:8083"
    sh "docker tag premmano/myweb:0.0.2 3.110.209.56:8083/prem:1.0.0"
    sh 'docker push 3.110.209.56:8083/prem:1.0.0'
    }
    stage('Remove Previous Container'){
        try{
            sh 'docker rm -f tomcattest'
        }catch(error){
            //  do nothing if there is an exception
        }
    stage('Docker deployment'){
    sh 'docker run -d -p 8090:8080 --name tomcattest premmano/myweb:0.0.2'
    }
}
}
```



	SCM Checkout	Compile-Package	SonarQube Analysis	Build Docker Imager	Docker Image Push	Nexus Image Push	Remove Previous Container	Docker deployment
Average stage times: (Average full run time: ~44s)	812ms	5s	7s	532ms	16s	13s	278ms	834ms
#61 Mar 28 13:53 No Changes	816ms	5s	7s	531ms	16s	1s	279ms	835ms
#60 Mar 28 13:45 No Changes	809ms	5s	6s	533ms	16s	25s	277ms	834ms

# Go to nexus console – browse – docker private – prem – 1.0.0
# Now docker images pushed to nexus console.



Browse / docker_private

HTML View Advanced search...

```
blobs
prem
  manifests
  tags
    1.0.0
premmano
  manifests
    sha256:01cd0cadfde930ffb65be712870291dfebecee53576f8062b16459b
    sha256:5a05164745df41528d5a50ba1b74ca12ce4c4fabc29325a142679a
    sha256:5b39242639c2659bdfdf2b0f2d26075570d778dbb10f8b1280900
    sha256:c9a2ec2cb92123052575f091ae4d012c478e2ade9cb6ecf035f1c6
    sha256:ceebb6301a843b72699216326b8c20be515980bdb3cd5bc837c7
  tags
    1.0.0
```

v2/prem/manifests/sha256:1fdb91a2cecea385...

Delete asset

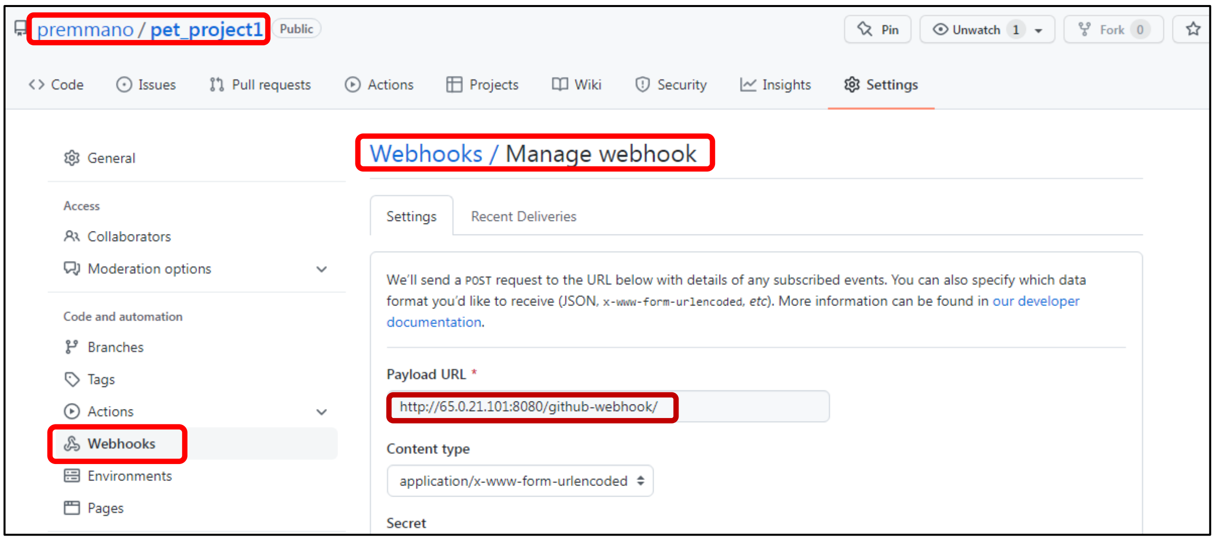
	d Time)
Last downloaded	has not been downloaded
Locally cached	true
Blob reference	default@C9345650-9286 D055-260FA411-3B111432-E983F026:038fbb6d-fe55 -4a7f-9c8c-2cd17bc62e84
Containing repo	docker_private
Uploader	admin
Uploader's IP Address	65.0.21.101

INTEGRATE GITHUB AND JENKINS:

# Go to github – setings – webhooks – add webhook

# Attach http://jenkins server ip:8080/github-webhooks/ on webhook place.

# context type: Application/json


premmano / pet_project1 Public

Webhooks / Manage webhook

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in our developer documentation.

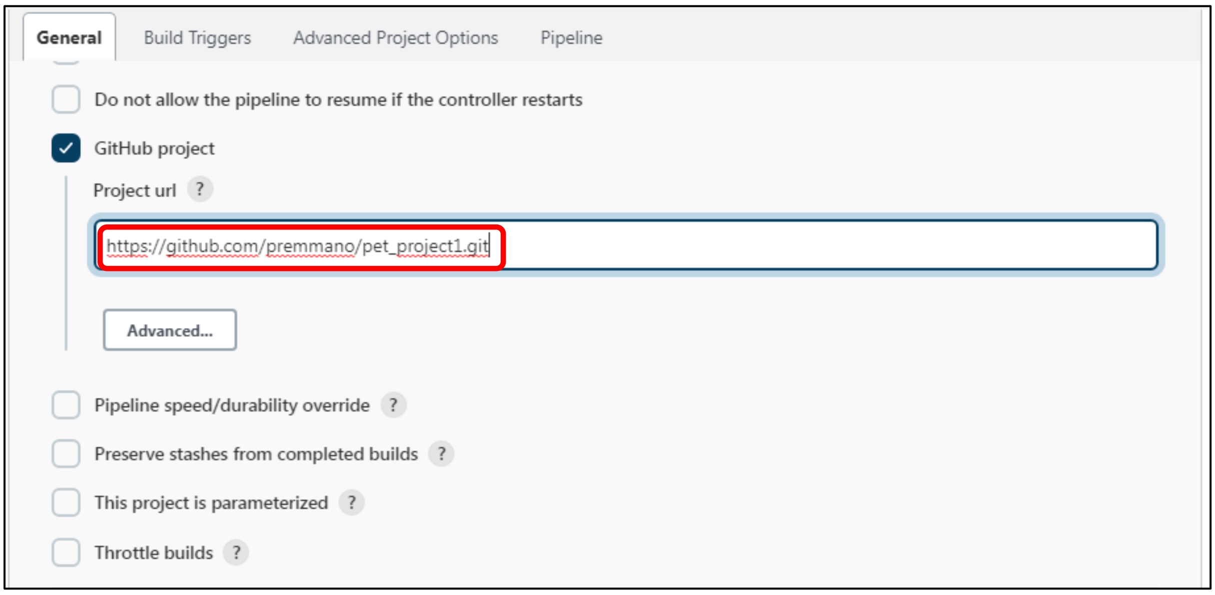
Payload URL *
http://65.0.21.101:8080/github-webhook/

Content type
application/x-www-form-urlencoded

Secret

# Go to jenkins console – configuration – general – github project

# Copy github pet_project1 url (https://github.com/premmano/pet_project1/my-app.git) and paste it in project url place.


Do not allow the pipeline to resume if the controller restarts

[✓] GitHub project

Project url
https://github.com/premmano/pet_project1.git

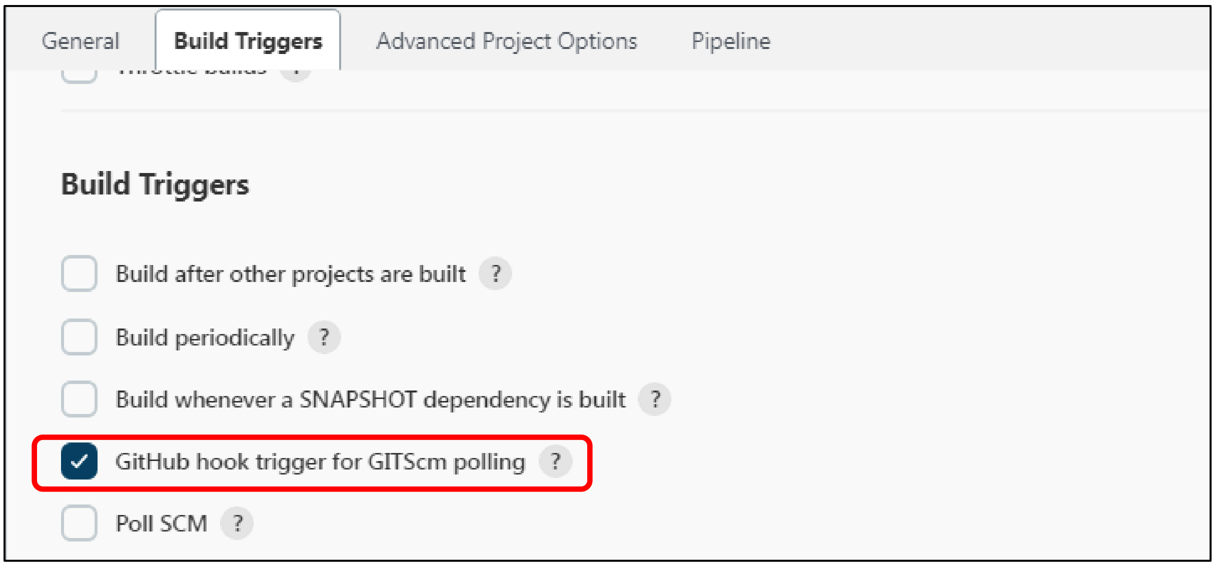
Pipeline speed/durability override

Preserve stashes from completed builds

This project is parameterized

Throttle builds

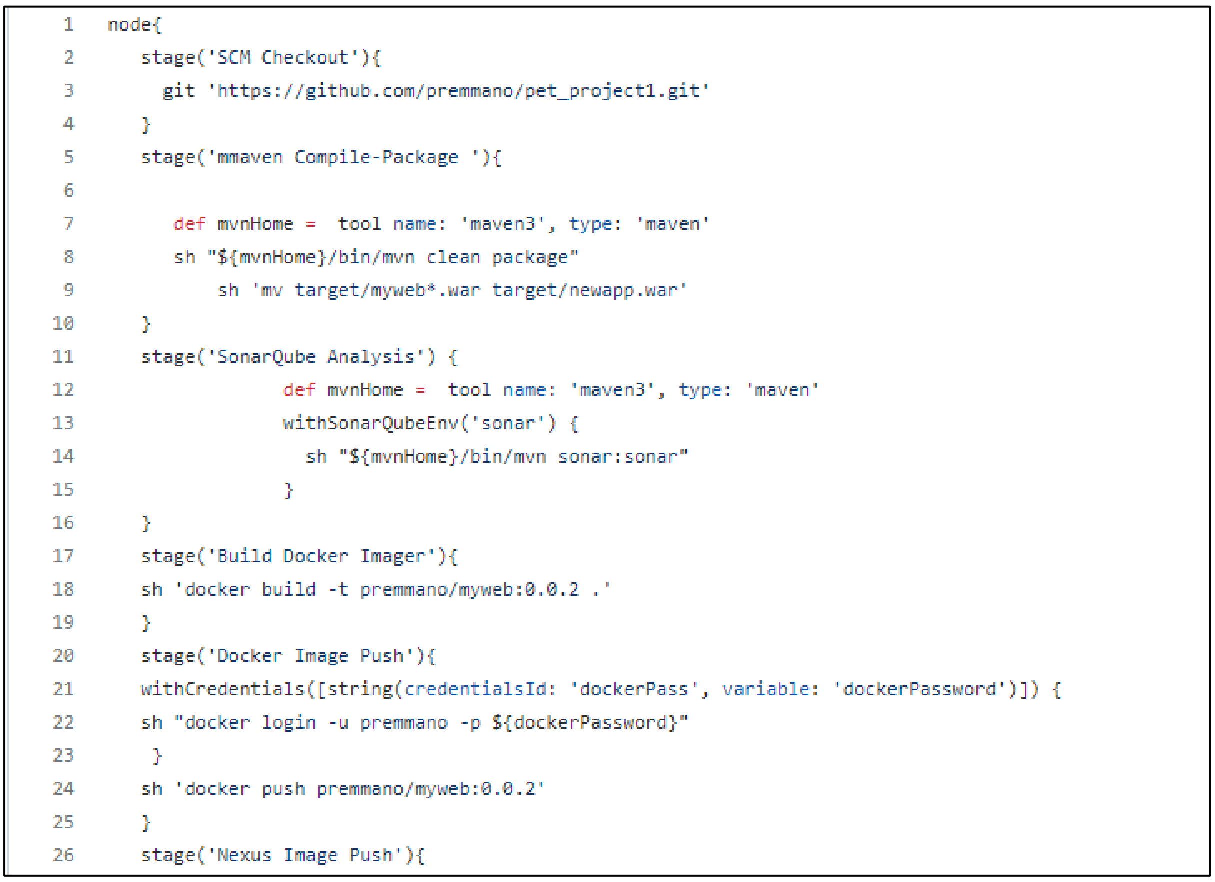
# Go to jenkins console – select "gith hub hook trigger for GIT SCM polling". whenever commit happens in github, the jenkins pipeline automatically trigger and run the code.



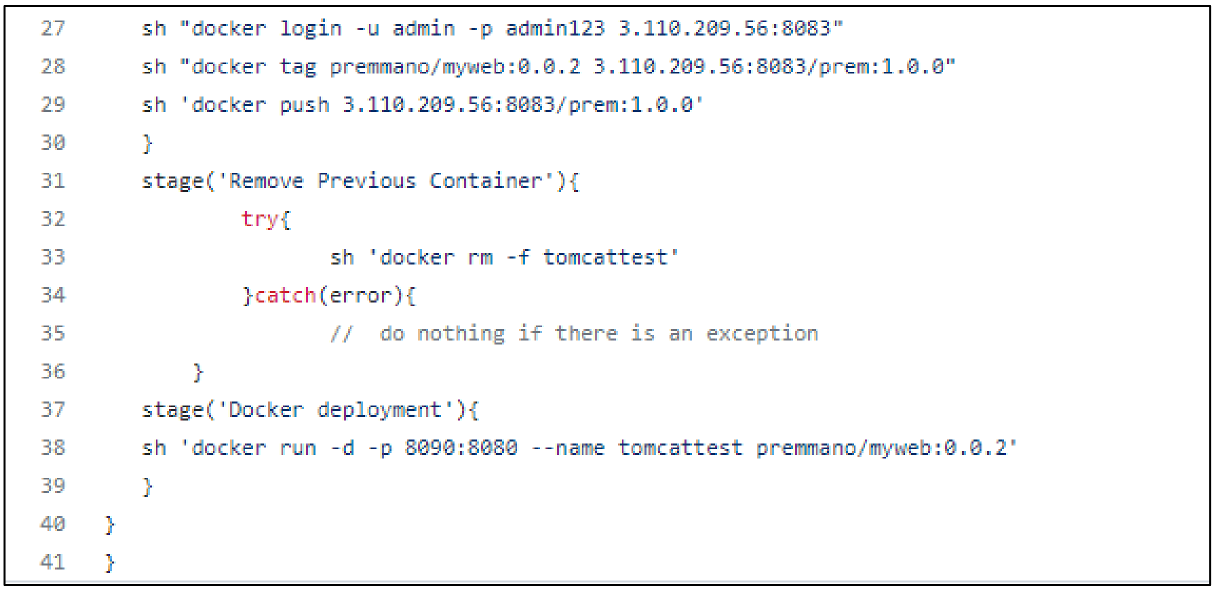
Build Triggers

- [ ] Build after other projects are built
- [ ] Build periodically
- [ ] Build whenever a SNAPSHOT dependency is built
- [✓] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

# Go to git hub – code – jenkins file
# Do commit or changes in jenkins file.

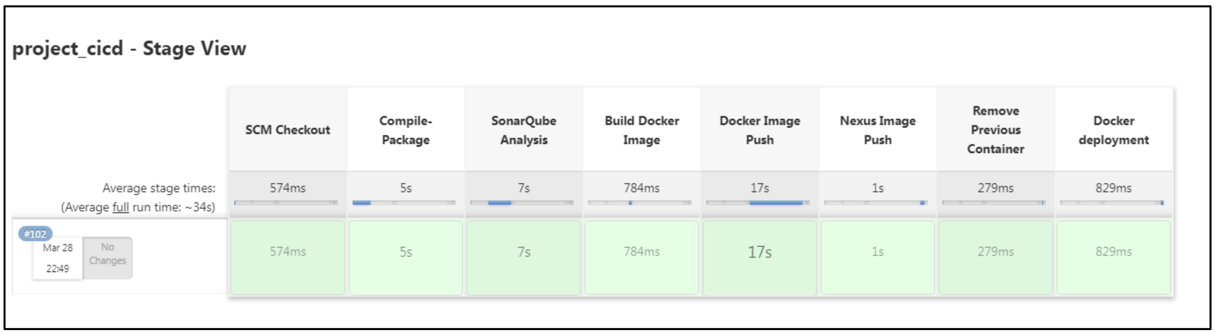


```
node{
   stage('SCM Checkout'){
     git 'https://github.com/premmano/pet_project1.git'
   }
   stage('mmaven Compile-Package '){

      def mvnHome =  tool name: 'maven3', type: 'maven'
      sh "${mvnHome}/bin/mvn clean package"
          sh 'mv target/myweb*.war target/newapp.war'
   }
   stage('SonarQube Analysis') {
                def mvnHome =  tool name: 'maven3', type: 'maven'
                withSonarQubeEnv('sonar') {
                  sh "${mvnHome}/bin/mvn sonar:sonar"
                }
   }
   stage('Build Docker Imager'){
   sh 'docker build -t premmano/myweb:0.0.2 .'
   }
   stage('Docker Image Push'){
   withCredentials([string(credentialsId: 'dockerPass', variable: 'dockerPassword')]) {
   sh "docker login -u premmano -p ${dockerPassword}"
    }
   sh 'docker push premmano/myweb:0.0.2'
   }
   stage('Nexus Image Push'){
```



```
        sh "docker login -u admin -p admin123 3.110.209.56:8083"
        sh "docker tag premmano/myweb:0.0.2 3.110.209.56:8083/prem:1.0.0"
        sh 'docker push 3.110.209.56:8083/prem:1.0.0'
        }
        stage('Remove Previous Container'){
                try{
                        sh 'docker rm -f tomcattest'
                }catch(error){
                        //  do nothing if there is an exception
            }
        stage('Docker deployment'){
        sh 'docker run -d -p 8090:8080 --name tomcattest premmano/myweb:0.0.2'
        }
    }
    }
```

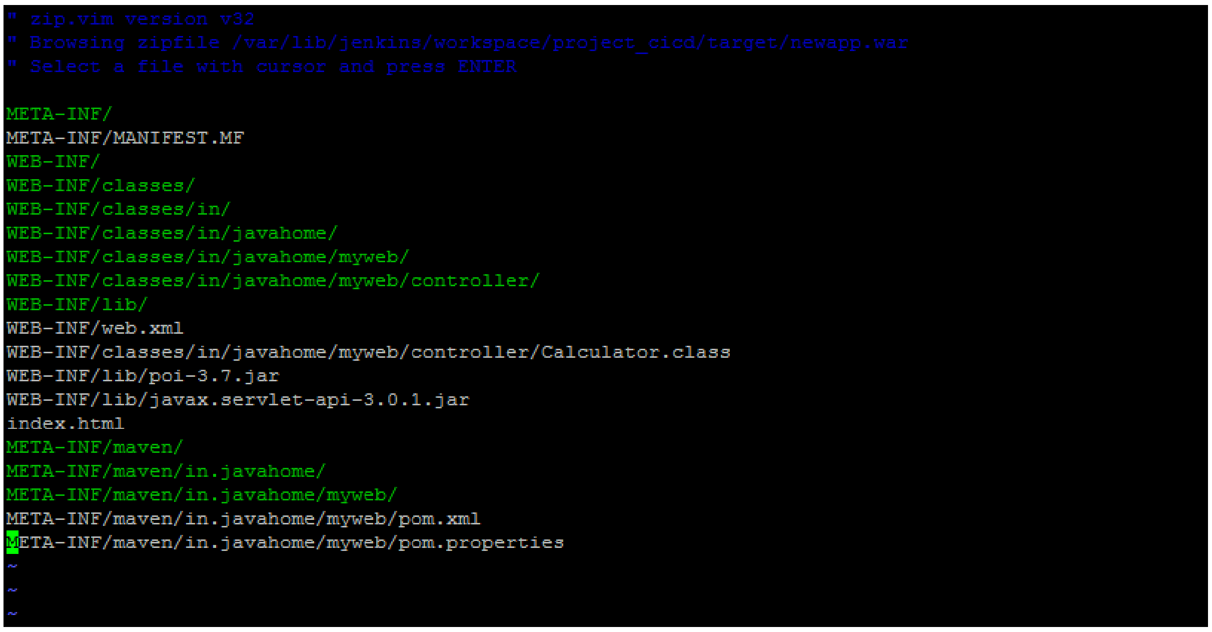
# Go to jenkins console
# It automatically run the code.



project_cicd - Stage View

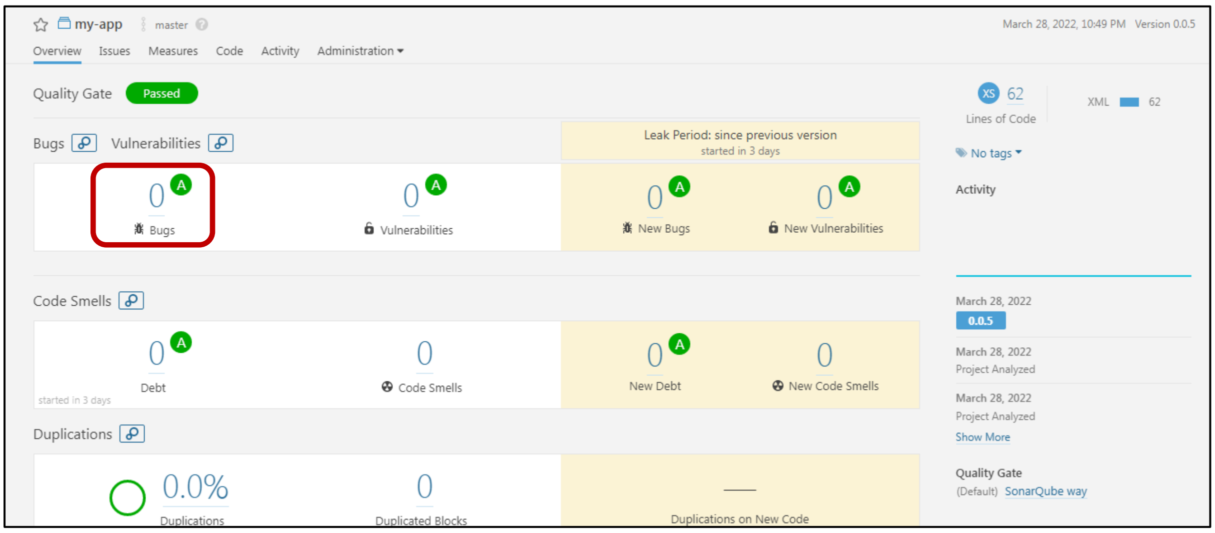
	SCM Checkout	Compile-Package	SonarQube Analysis	Build Docker Image	Docker Image Push	Nexus Image Push	Remove Previous Container	Docker deployment
Average stage times: (Average full run time: ~34s)	574ms	5s	7s	784ms	17s	1s	279ms	829ms
#102 Mar 28 22:49 No Changes	574ms	5s	7s	784ms	17s	1s	279ms	829ms

RESULT:
The below snapshot gives the result of each tool used.
Maven:



```
" zip.vim version v32
" Browsing zipfile /var/lib/jenkins/workspace/project_cicd/target/newapp.war
" Select a file with cursor and press ENTER

META-INF/
META-INF/MANIFEST.MF
WEB-INF/
WEB-INF/classes/
WEB-INF/classes/in/
WEB-INF/classes/in/javahome/
WEB-INF/classes/in/javahome/myweb/
WEB-INF/classes/in/javahome/myweb/controller/
WEB-INF/lib/
WEB-INF/web.xml
WEB-INF/classes/in/javahome/myweb/controller/Calculator.class
WEB-INF/lib/poi-3.7.jar
WEB-INF/lib/javax.servlet-api-3.0.1.jar
index.html
META-INF/maven/
META-INF/maven/in.javahome/
META-INF/maven/in.javahome/myweb/
META-INF/maven/in.javahome/myweb/pom.xml
META-INF/maven/in.javahome/myweb/pom.properties
~
~
~
```

SonarQube:



my-app master

March 28, 2022, 10:49 PM Version 0.0.5

Overview Issues Measures Code Activity Administration

Quality Gate Passed

62 Lines of Code XML 62

No tags

Activity

Bugs Vulnerabilities

Leak Period: since previous version
started in 3 days

0 A Bugs
0 A Vulnerabilities
0 A New Bugs
0 A New Vulnerabilities

Code Smells

0 A Debt
started in 3 days
0 Code Smells
0 A New Debt
0 New Code Smells

Duplications

0.0% Duplications
0 Duplicated Blocks
— Duplications on New Code

March 28, 2022
0.0.5

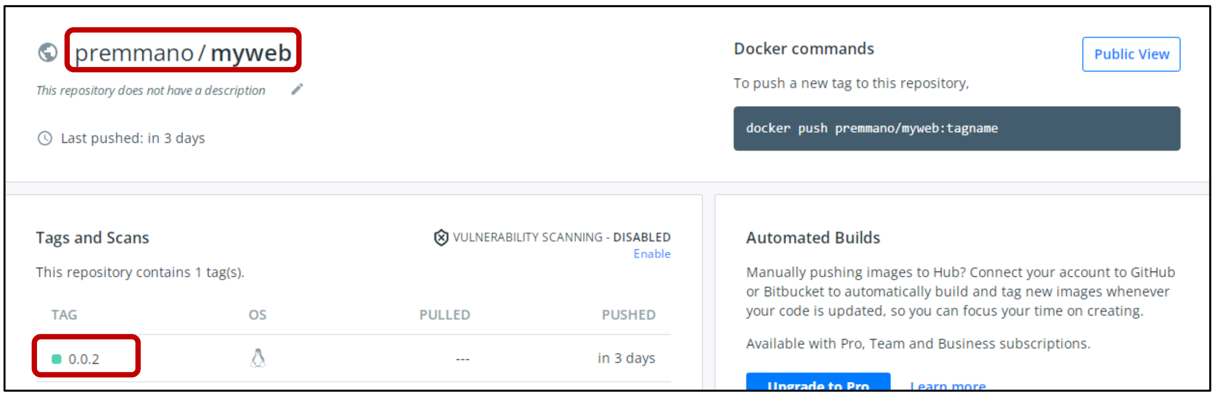
March 28, 2022
Project Analyzed

March 28, 2022
Project Analyzed

Show More

Quality Gate
(Default) SonarQube way

Docker



premmano / myweb

This repository does not have a description

Last pushed: in 3 days

Docker commands

Public View

To push a new tag to this repository,

```
docker push premmano/myweb:tagname
```

Tags and Scans

VULNERABILITY SCANNING - DISABLED
Enable

This repository contains 1 tag(s).

TAG	OS	PULLED	PUSHED
0.0.2		---	in 3 days

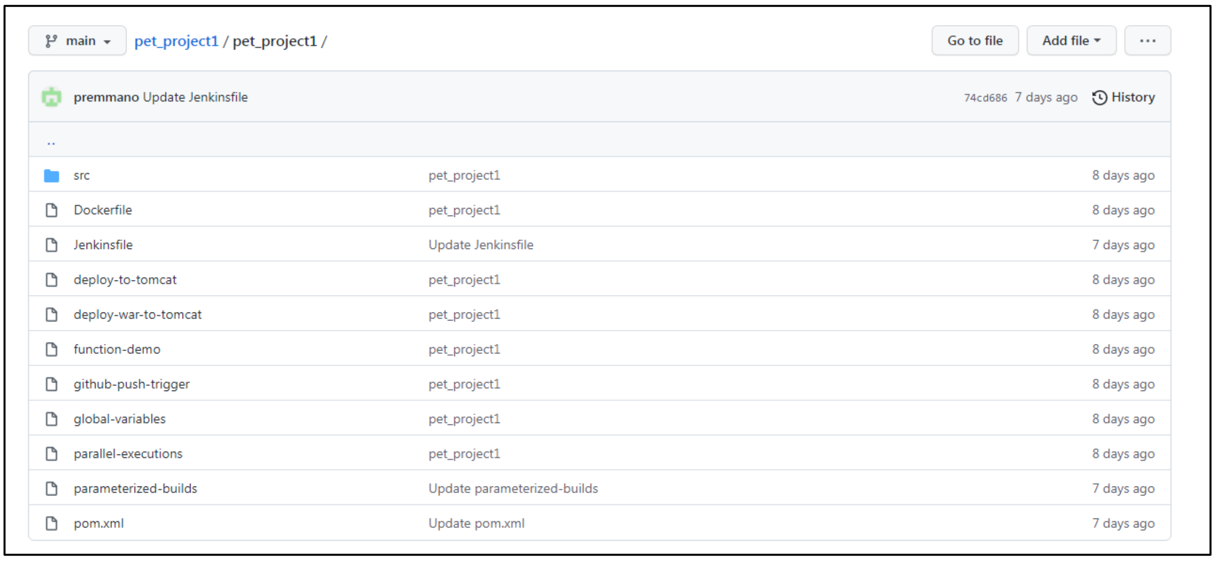
Automated Builds

Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions.

Upgrade to Pro Learn more

Github



main pet_project1 / pet_project1 /

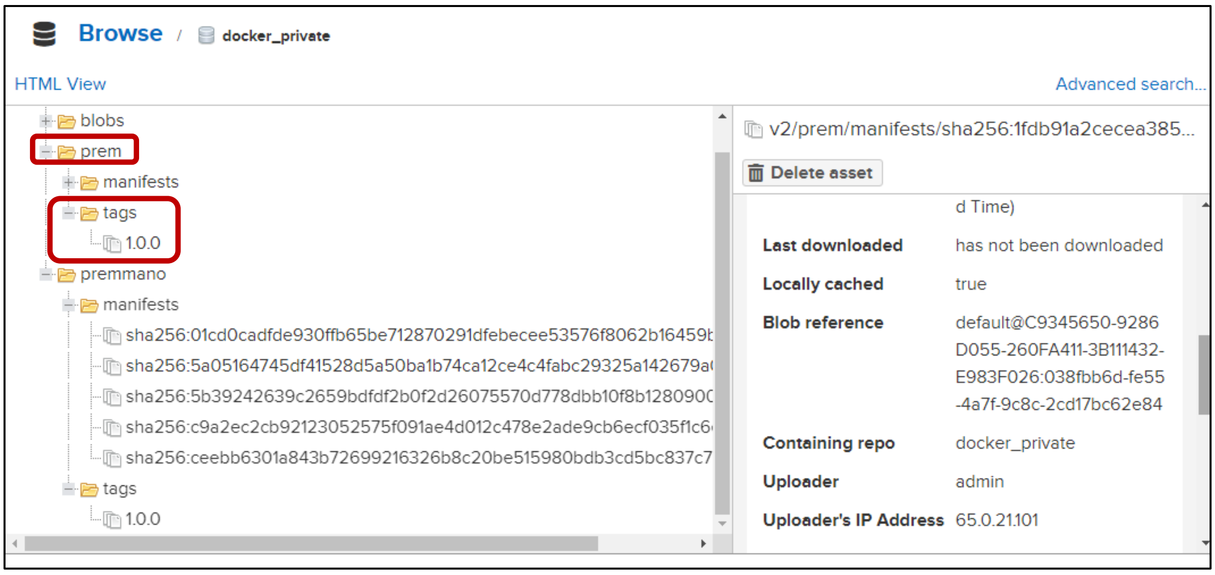
Go to file Add file ···

premmano Update Jenkinsfile 74cd686 7 days ago History

..

src	pet_project1	8 days ago
Dockerfile	pet_project1	8 days ago
Jenkinsfile	Update Jenkinsfile	7 days ago
deploy-to-tomcat	pet_project1	8 days ago
deploy-war-to-tomcat	pet_project1	8 days ago
function-demo	pet_project1	8 days ago
github-push-trigger	pet_project1	8 days ago
global-variables	pet_project1	8 days ago
parallel-executions	pet_project1	8 days ago
parameterized-builds	Update parameterized-builds	7 days ago
pom.xml	Update pom.xml	7 days ago

Nexus:


Browse / docker_private

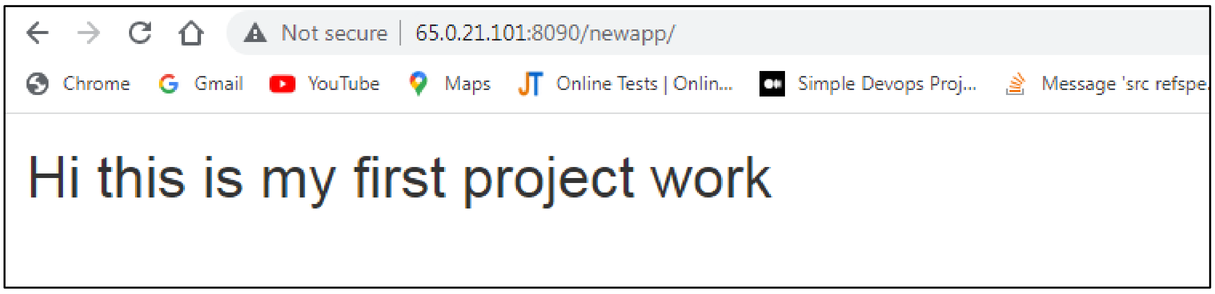
HTML View — Advanced search...

```
blobs
prem
  manifests
  tags
    1.0.0
premmano
  manifests
    sha256:01cd0cadfde930ffb65be712870291dfebecee53576f8062b16459b
    sha256:5a05164745df41528d5a50ba1b74ca12ce4c4fabc29325a142679a
    sha256:5b39242639c2659bdfdf2b0f2d26075570d778dbb10f8b128090C
    sha256:c9a2ec2cb92123052575f091ae4d012c478e2ade9cb6ecf035f1c6
    sha256:ceebb6301a843b72699216326b8c20be515980bdb3cd5bc837c7
  tags
    1.0.0
```

v2/prem/manifests/sha256:1fdb91a2cecea385...

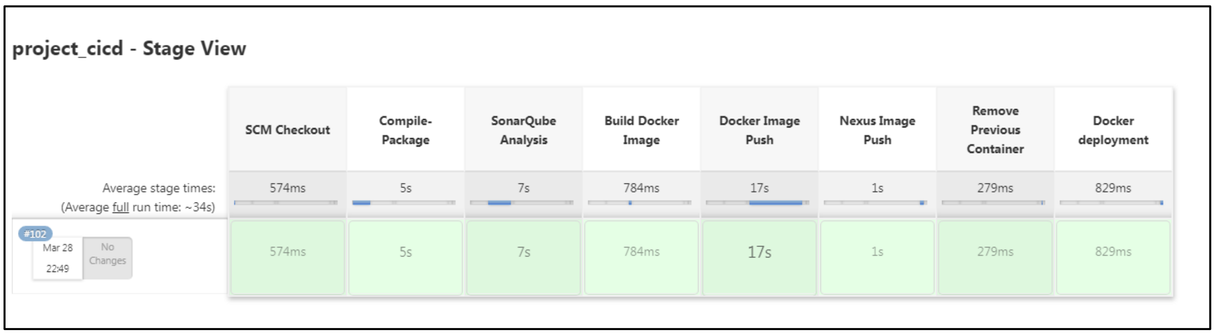
Delete asset

	d Time)
Last downloaded	has not been downloaded
Locally cached	true
Blob reference	default@C9345650-9286 D055-260FA411-3B111432-E983F026:038fbb6d-fe55 -4a7f-9c8c-2cd17bc62e84
Containing repo	docker_private
Uploader	admin
Uploader's IP Address	65.0.21.101


Not secure | 65.0.21.101:8090/newapp/

Hi this is my first project work

Jenkins:


project_cicd - Stage View

	SCM Checkout	Compile-Package	SonarQube Analysis	Build Docker Image	Docker Image Push	Nexus Image Push	Remove Previous Container	Docker deployment
Average stage times: (Average full run time: ~34s)	574ms	5s	7s	784ms	17s	1s	279ms	829ms
#102 Mar 28 22:49 No Changes	574ms	5s	7s	784ms	17s	1s	279ms	829ms

CONCLUSION:

This proposed project is successfully done using DeVops methodology. DevOps is a software development methodology that escalates to the amalgamation between software developers and information technology (IT) operation professionals. Its focuses mainly on delivering software product faster and reducing the failure rate of releases to make the product efficient. This system will be helpful for the developers or testers who need to fix the bugs rapidly and want to add extra features to the existing product according to the client requirement. At present DevOps is the most advanced approach in IT industry than waterfall model and agile model. This proposed project is successfully implemented using the DeVops methodology using aws, git, jenkins, maven, Sonarqube, docker and nexus.