**Jenkins Set-up**

1. Install Java 8 and Java Environment

2. Install Jenkins

3. Install Maven and Maven Environment

4. Configure Jenkins

5. Install and Configure Apache Tomcat

6. Create & Clone Git Repository (In Git-Hub document)

7. Jenkins Standalone Mode

Project 1: Build MVN Artifact & Deploy MVN Artifact on the Tomcat Container

8. Jenkins Master/Slave Setup

Project 1: Build Project in the Salve Mode

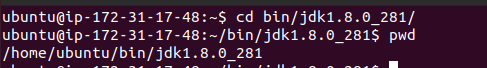
Project 2: Build MVN Artifact in the Salve Node & Deploy MVN Artifact in the Master Node

**Install Java 8**

1. Launch an EC2 Instance and open ports 22/8080/8081
2. Login to Oracle site and either download or wget link e.g. jdk-8u281-linux-x64.tar.gz for the Linux Ubuntu, tar it tar –zxvf jdk-8u281-linux-x64.tar.gz
3. Java Home: java –version will through error because we need to set Java Environment variable to set the PATH as Global in the Bash Profile.

The **Bash profile** is a file that runs every time a new **Bash** session is created. This is useful because we may need to run certain code every time system starts. OS doesn't include a **Bash profile** by default.

Bash **profile** file is located at /home/<user>/. bash\_profile

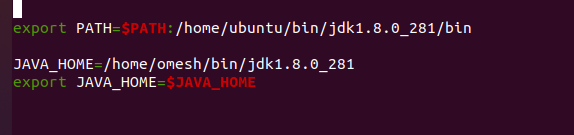


ubuntu@ip-172-31-17-48:~$ sudo nano ~/.bash\_profile

export PATH=$PATH: /home/ubuntu/bin/jdk1.8.0\_281/bin

JAVA\_HOME=/home/omesh/bin/jdk1.8.0\_281

export JAVA\_HOME=$JAVA\_HOME



Note: /bin is added as executables are here and also JAVA Home environment set.

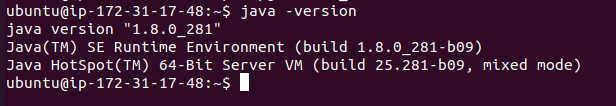
ubuntu@ip-172-31-17-48:~$ source ~/.bash\_profile

source ~/.bash\_profile

ubuntu@ip-172-31-17-48:~/bin/jdk1.8.0\_281$ echo $PATH

/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/home/ubuntu/bin/jdk1.8.0\_281/bin

Java version running successfully suggests that Java is installed.



**Install Jenkins**

Use official Jenkins site Jenkins.io and select debian platform, follow instructions listed based on the OS type.



1.

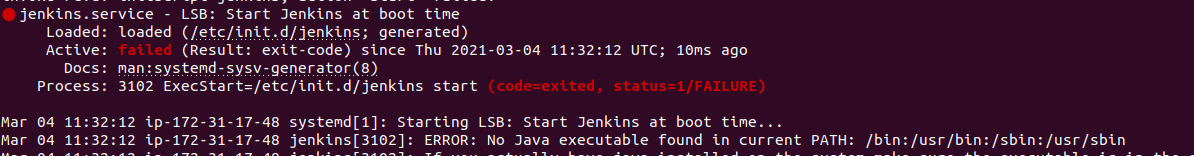


2. ubuntu@ip-172-31-17-48:~$ sudo nano /etc/apt/sources.list

3. ubuntu@ip-172-31-17-48:~$ sudo apt update

ubuntu@ip-172-31-17-48:~$ sudo apt install jenkins –y

Jenkins will fail to start

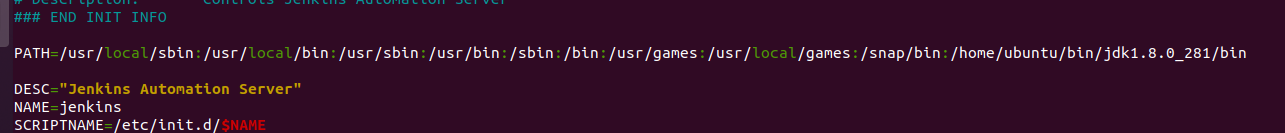


Jenkins will give java path error so we need to provide the correct path by editing the file /etc/init.d/Jenkins and the correct is provide by the echo $PATH.

ubuntu@ip-172-31-17-48:~$ echo $PATH

/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/home/ubuntu/bin/jdk1.8.0\_281/bin

ubuntu@ip-172-31-17-48:~$ sudo nano /etc/init.d/jenkins

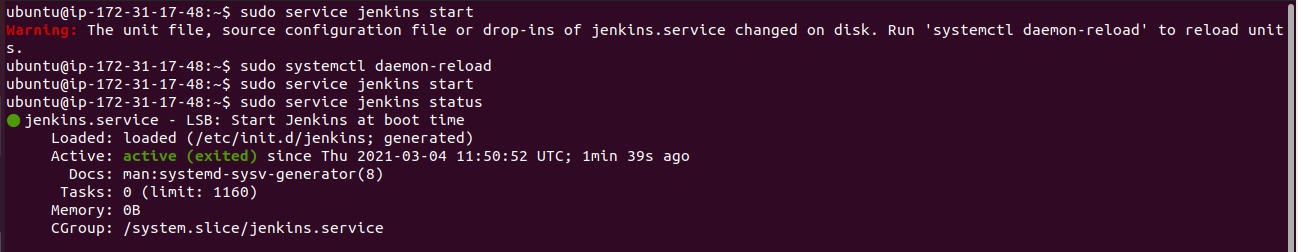


ubuntu@ip-172-31-17-48:~$ sudo service jenkins start

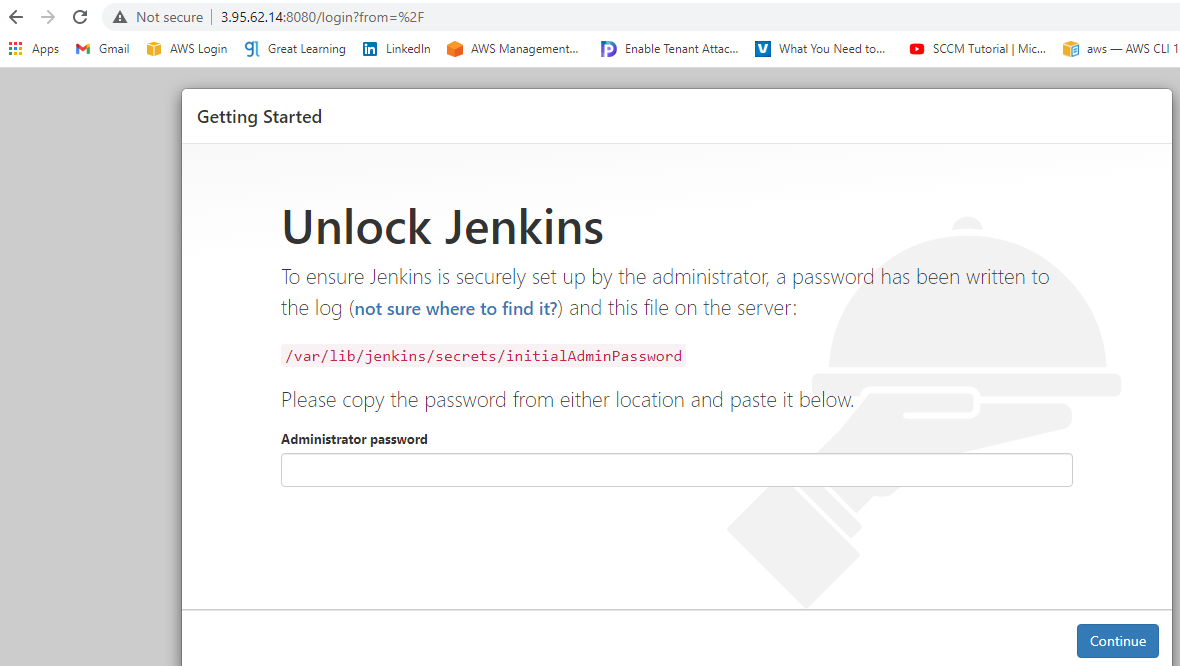
ubuntu@ip-172-31-17-48:~$ sudo systemctl daemon-reload

ubuntu@ip-172-31-17-48:~$ sudo service jenkins start

ubuntu@ip-172-31-17-48:~$ sudo service jenkins status



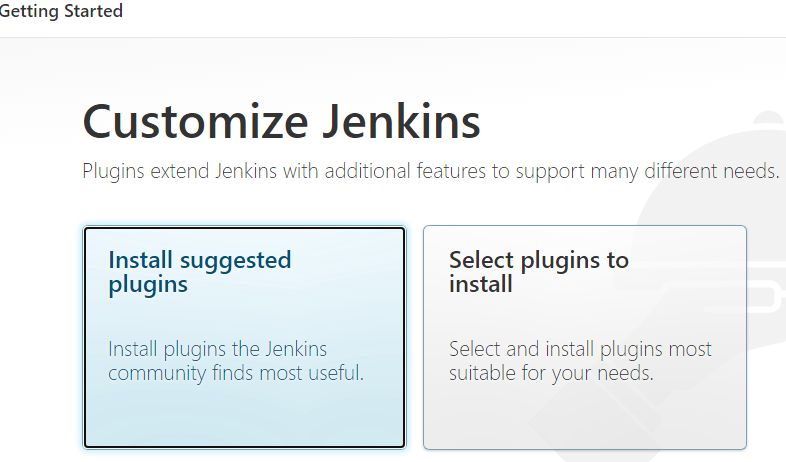
Login to Jenkins using Publicip:8080

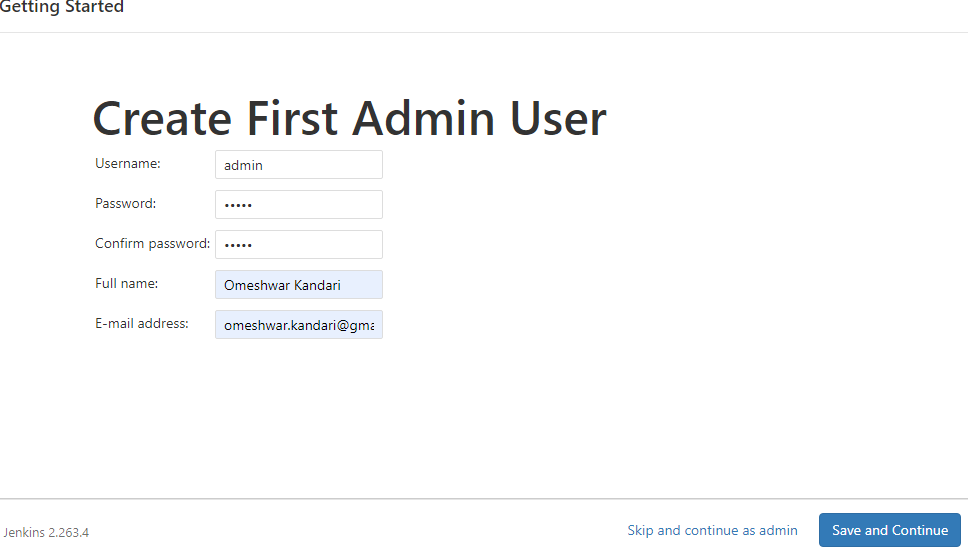


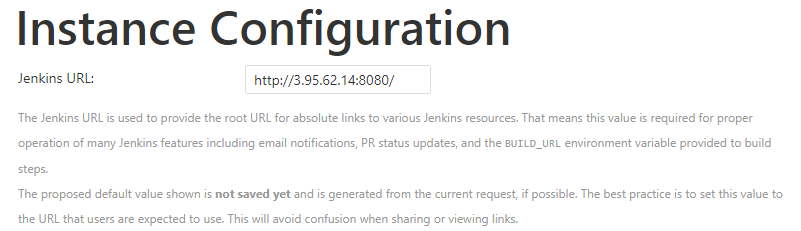
Copy the initial password from the file, paste and continue

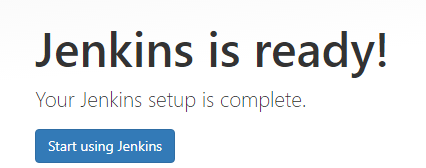
ubuntu@ip-172-31-17-48:~$ sudo nano /var/lib/jenkins/secrets/initialAdminPassword

Install the plugin and create the First Admin user









**Install Maven**

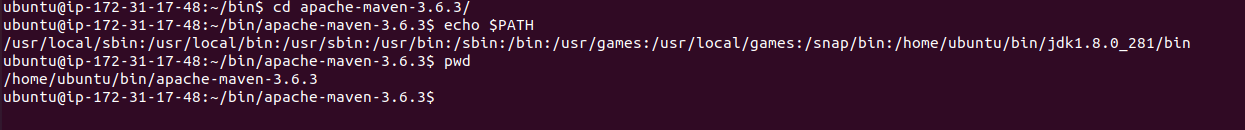
1. Copy the download the link for binary tar.gz from Apache Maven site

e.g. <https://downloads.apache.org/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.tar.gz>

2. Install Maven: ubuntu@ip-172-31-17-48:~/bin$ wget <https://downloads.apache.org/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.tar.gz>

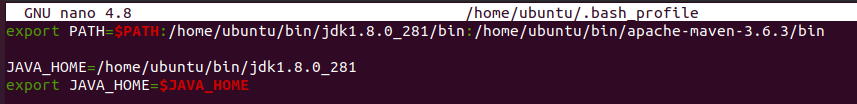
ubuntu@ip-172-31-17-48:~/bin$ tat -zxvf apache-maven-3.6.3-bin.tar.gz

(tar will extract the executables)

3. Set the MVN Path 

Add maven path /home/ubuntu/bin/apache-maven-3.6.3/bin in Java path with : and this way we can add anyother environment as needed.

ubuntu@ip-172-31-17-48:~/bin/apache-maven-3.6.3$ sudo nano ~/.bash\_profile



ubuntu@ip-172-31-17-48:~/bin/apache-maven-3.6.3$ source ~/.bash\_profile

ubuntu@ip-172-31-17-48:~/bin/apache-maven-3.6.3$ mvn -v

Apache Maven 3.6.3 (cecedd343002696d0abb50b32b541b8a6ba2883f)

Maven home: /home/ubuntu/bin/apache-maven-3.6.3

Java version: 1.8.0\_281, vendor: Oracle Corporation, runtime: /home/ubuntu/bin/jdk1.8.0\_281/jre

Default locale: en, platform encoding: UTF-8

OS name: "linux", version: "5.4.0-1038-aws", arch: "amd64", family: "unix"

**Configure Jenkins**

Global Tool Configuration:

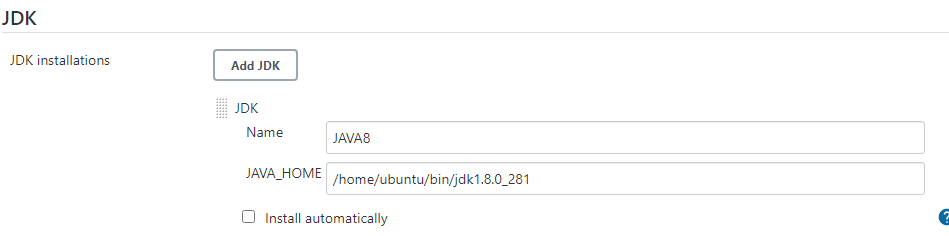
Installation packages needed to build the artifacts are set in here so that Build runs successfully and this configuration is also followed by the Slave Nodes which means the path for packages installed in Slave Nodes should be in line with this configuration. Since Git Repository is already configured so we will ser only Java and MVN for our MVN Program

JDK :

We have already installed Java so installation check box is unchecked and other variables like Name and Java\_Home path are entered based on the Java path in the bash\_profile.

ubuntu@ip-172-31-17-48:~/bin/apache-maven-3.6.3$ echo $JAVA\_HOME

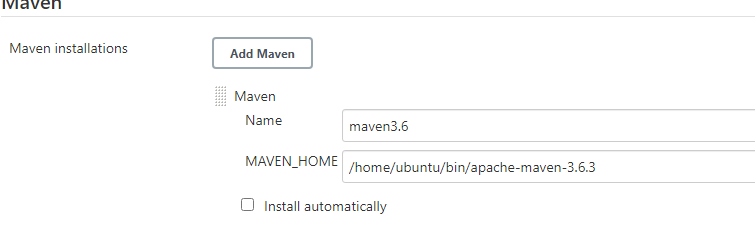
/home/ubuntu/bin/jdk1.8.0\_281



MAVEN:

MVN also installed so we need to update Name and MVN Path.

ubuntu@ip-172-31-17-48:~/bin/apache-maven-3.6.3$ mvn –version



**Install and configure Tomcat**

1. Install Tomcat using download link from Tomcat Apache tar gz

2. Port Change: By default Tomcat runs on 8080 and Jenkins already using it so we need to assign another port e.g. 8081 to Tomcat by editing Server.xml file

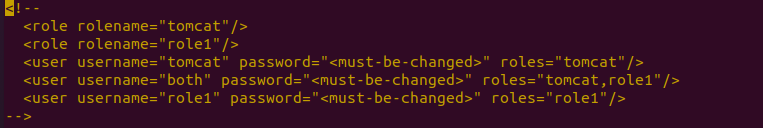
ubuntu@ip-172-31-17-48:~/bin/apache-tomcat-9.0.43/conf$ sudo nano server.xml



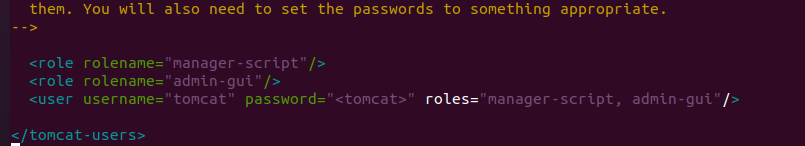
3. Permissions: Jenkins Server needs access to deploy artifacts into Tomcat container by editing the file tomcat-users.xml

ubuntu@ip-172-31-17-48:~/bin/apache-tomcat-9.0.43/conf$ sudo nano tomcat-users.xml

Uncomment and add user with role:



Add two roles and set the user with password:



1. Container Access Permission Error:

In case Deployment issue inside the container then a security permission issue might be the reason and we need to edit context.xml file.

ubuntu@ip-172-31-17-48:~/bin/apache-tomcat-9.0.43/webapps/manager/META-INF$ sudo nano context.xml

Below line needed to be commented because it means “only allow deployment from within the container using local-host but we want to deploy using Public IP”.

Default Settings:



Updated Settings:



1. Stop and start Tomcat to load the changes:

Use bin directory which has executable

ubuntu@ip-172-31-17-48:~/bin/apache-tomcat-9.0.43/bin$ ./shutdown.sh

ubuntu@ip-172-31-17-48:~/bin/apache-tomcat-9.0.43/bin$ ./startup.sh

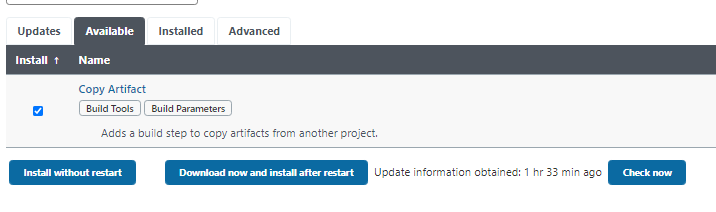
1. Check tomcat @ public ip:8081

**Project 1: Build MVN Artifact & Deploy MVN Artifact on the Tomcat Container**

1. Install Plug-in on the Jenkins:

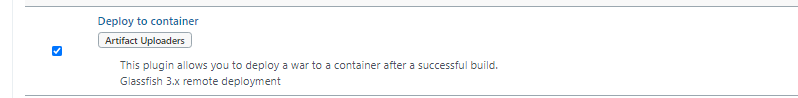
Copy Artifact:

It enables copy artifacts from the other projects during the Build Process.



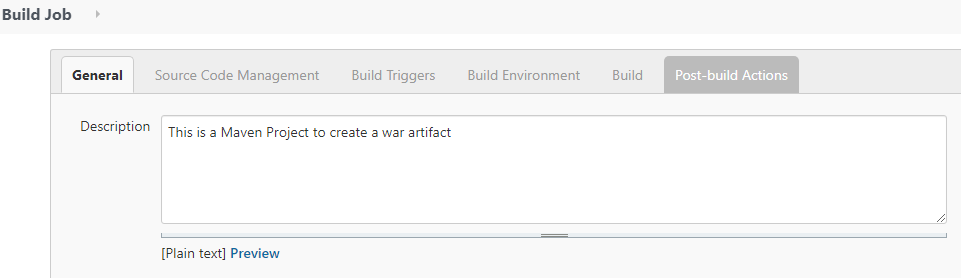
Deploy To Container:

It enables to deploy artifacts to another project during the Deploy Process.

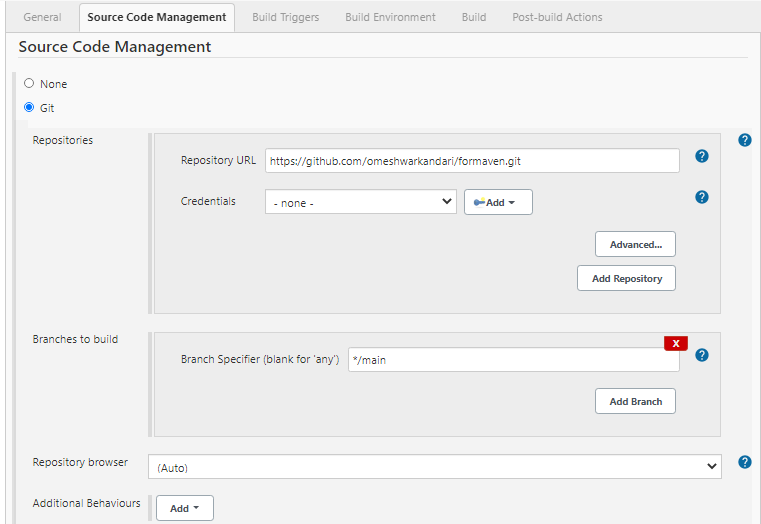


2. Create the Build Project:

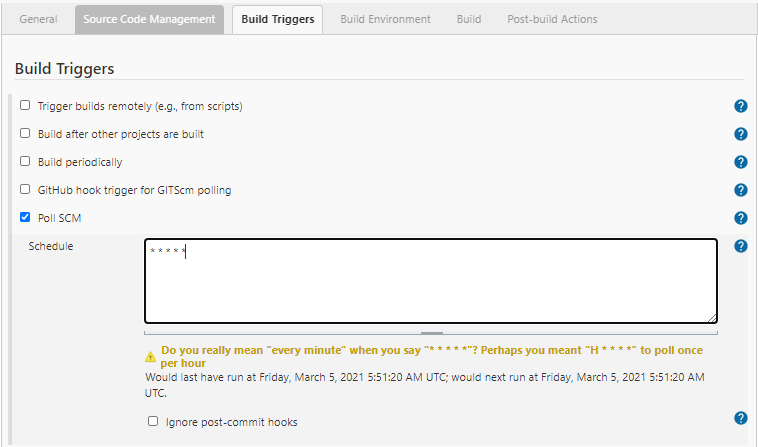
Create a new job Build MVN Artifact and describe the purpose.



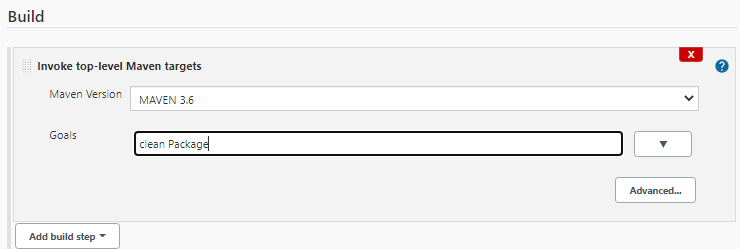
Source Code – Enter the path from Git Repo which is cloned on the local system



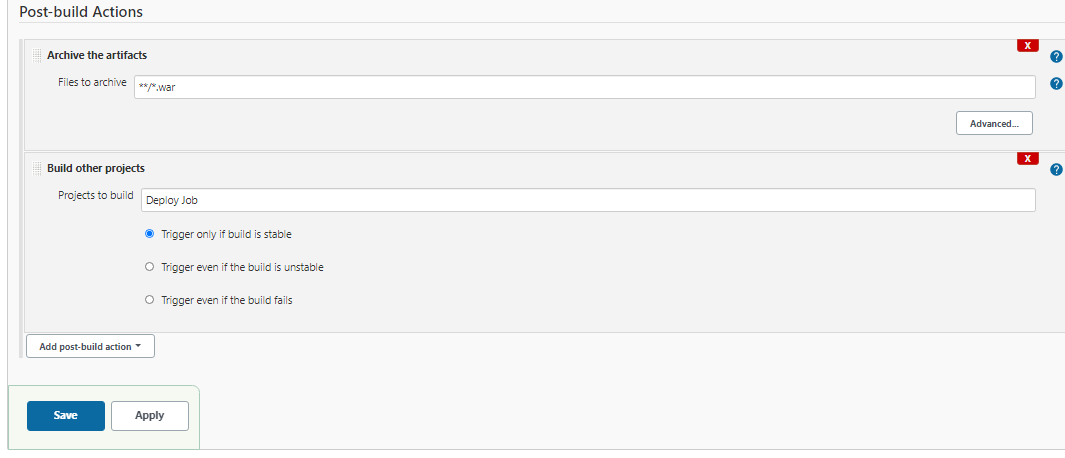
Build Trigger is set to Poll SCM (Source Code Manager i.e. Repo) with a crone job expression where 5 stars meaning polling will be every minute for new commit. Build Environment is not set for this test case.



Build: Invoke top-level Maven target which is Maven version 3.6 and it will clean the package before every new build.

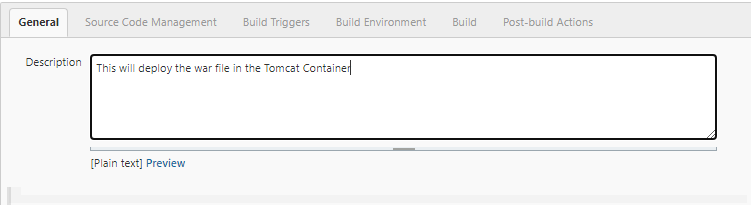


Post Build Actions: It will search every build with .war and archive it as a Build for other project which we need to mention e.g. Deploy Project

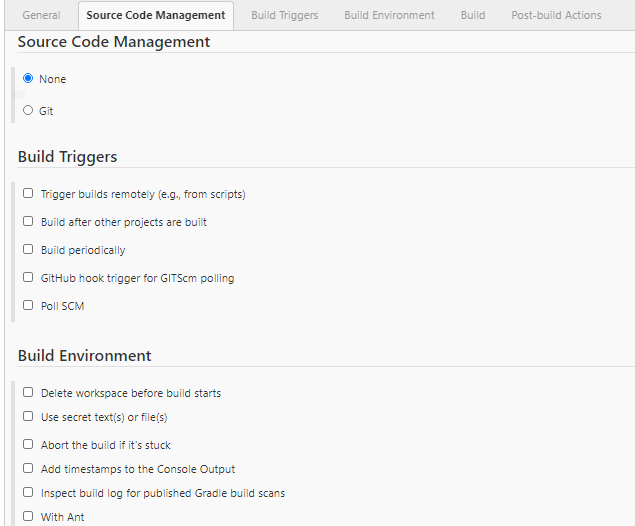


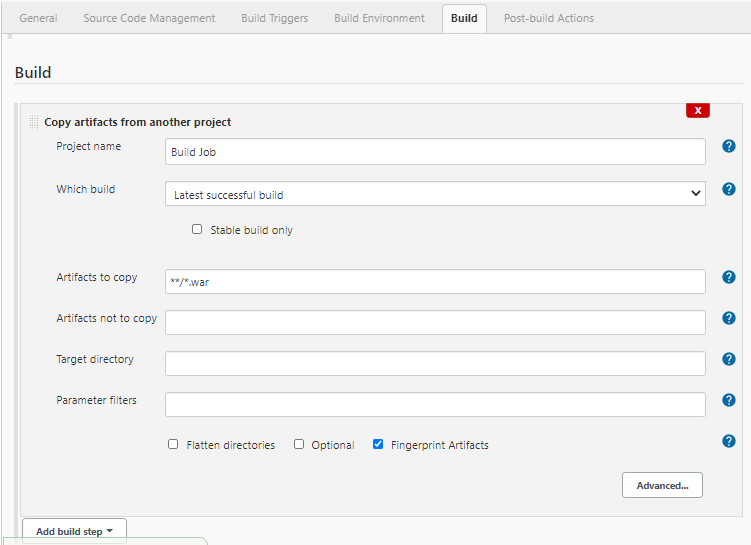
3. Create A Deploy Job: Deploy MVN Artifact on the Tomcat Container

New Item and create a Deploy Job in freestyle.

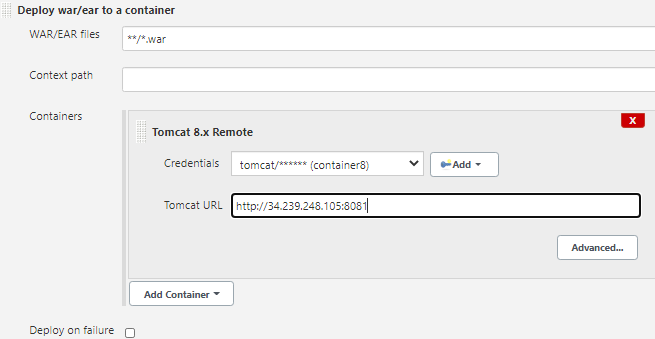


Source Code is None as Build will provide war file and no Build Trigger/ Build Environment selected/needed.

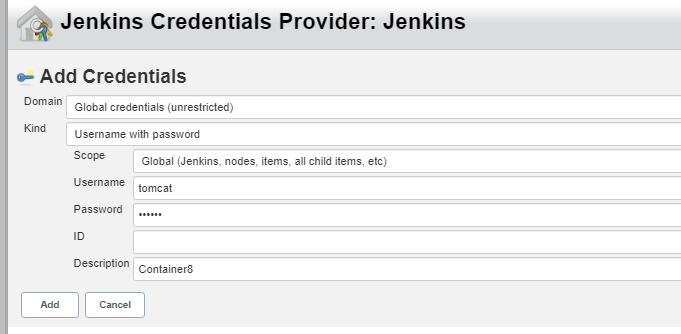


Build: Copy artifacts from another project 

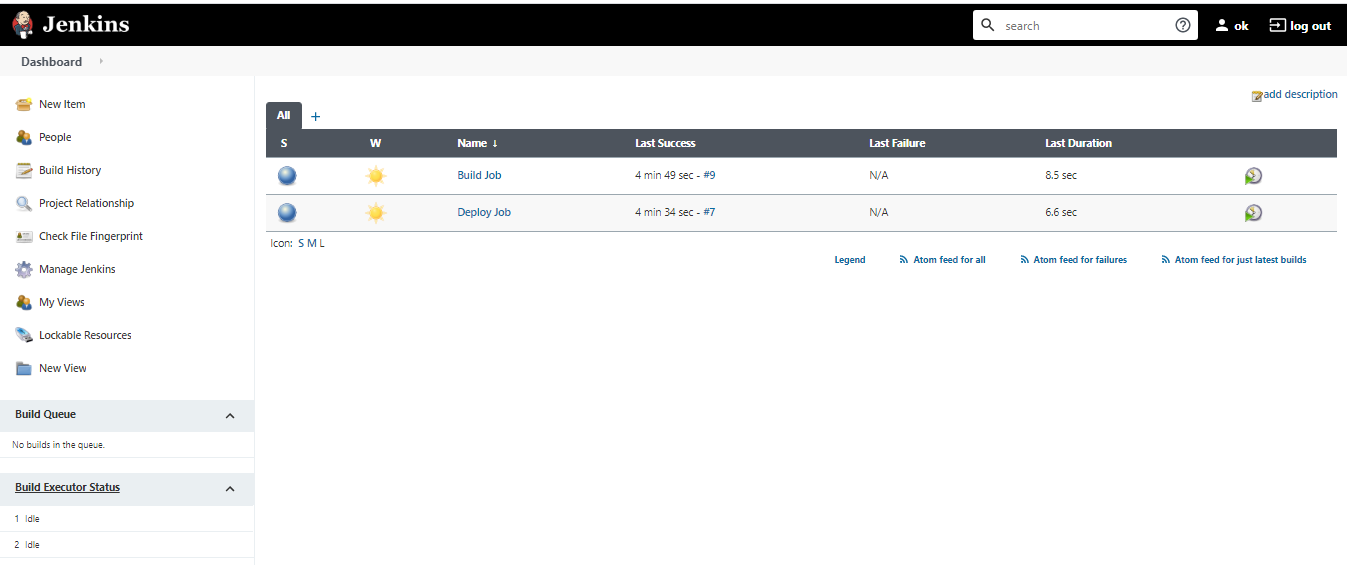
Post Build Actions: Deploy war to the tomcat container.



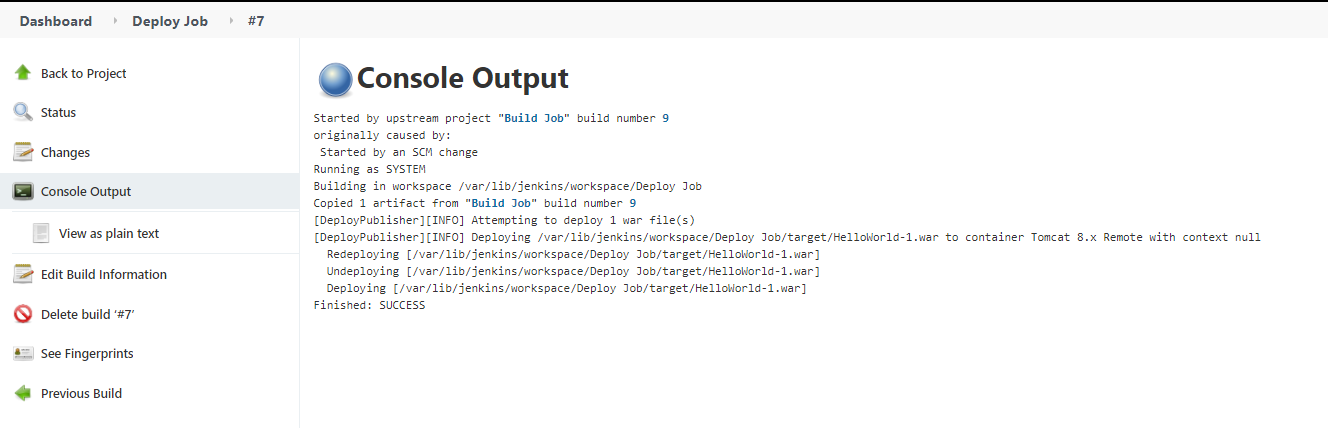
Add the tomcat user settings in the add section and give name to the container

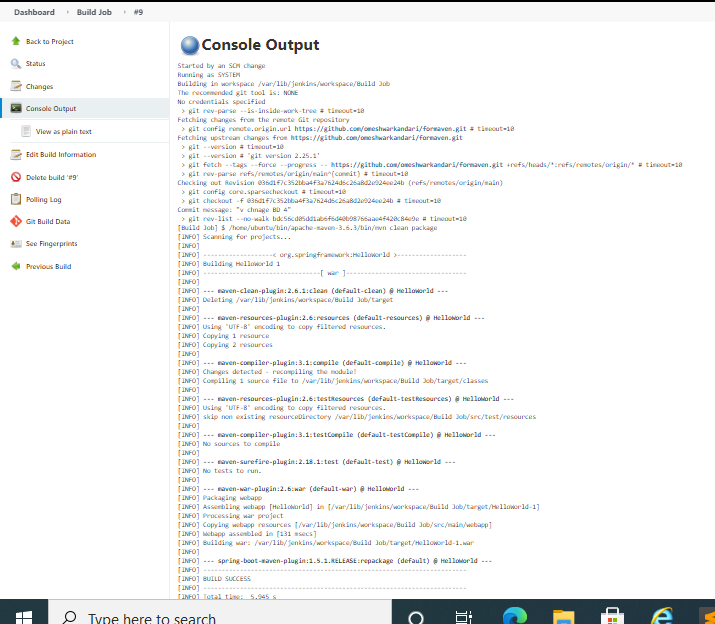


**Sample Output:** Git commit/add (-am) will commit the code change and Git Push will create a new Build within 1 minute based on the SCM Poll settings which is 5-star(every minute) and new build will deploy it automatically into the Tomcat Container. Sample Output also attached.



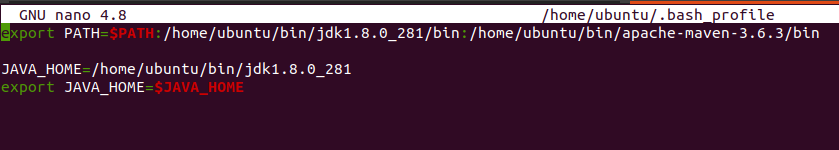
**Deploy Output:**



**Build Output:**

**Jenkins Master Slave Configuration:**

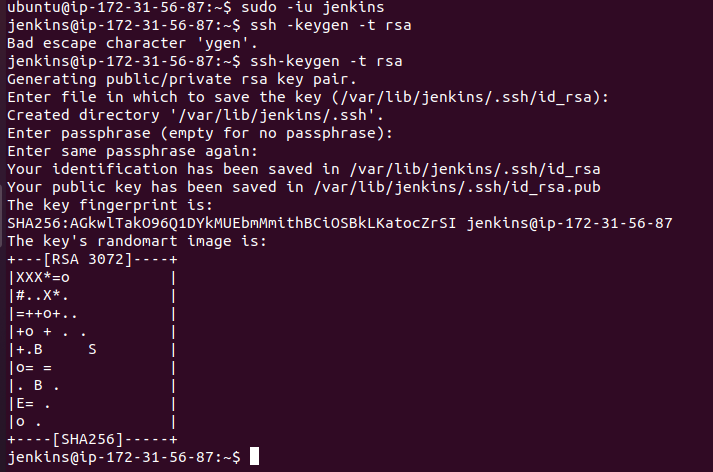
1. Create a Slave machine and install Java/MVN with environment set as per the path in the Jenkins Global Configuration Settings.



1. Create a key pair (Pub/Private): This is needed for the user Jenkins to ssh into Slave Node from the Master Node. Jenkins user Public key from the Master Node will be saved in the root .ssh/authorized\_key of the Slave Node.
2. Public Key Access: Login as Jenkins user in the Master Node use Keygen to access public/private keys.

ubuntu@ip-172-31-56-87:~$ sudo -iu Jenkins

jenkins@ip-172-31-56-87:~$ ssh-keygen -t rsa (-t is type)



jenkins@ip-172-31-56-87:~$ ls –al (it will show .ssh folder)

jenkins@ip-172-31-56-87:~$ ls .ssh

id\_rsa id\_rsa.pub

1. Copy Public Key from the Master Node:

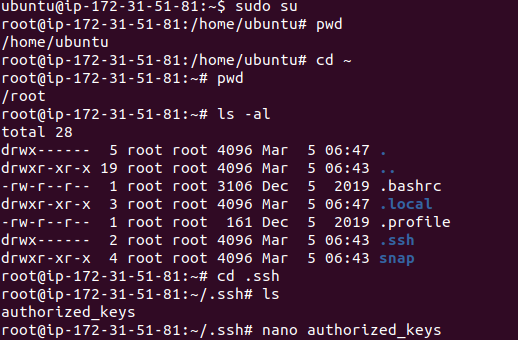
jenkins@ip-172-31-56-87:~$ nano .ssh/id\_rsa.pub

or

jenkins@ip-172-31-56-87:~$ cat .ssh/id\_rsa.pub

1. Paste in the Slave Node:

root@ip-172-31-51-81:~/.ssh# nano authorized\_keys



1. SSH into Slave from Master with Jenkins login:

jenkins@ip-172-31-56-87:~$ ssh [root@100.27.11.225](mailto:root@100.27.11.225)

Confirm with below command to match the Private IP of the Slave Node:

root@ip-172-31-51-81:~# hostname

ip-172-31-51-81

1. Download a jar file slave.jar which is required to communicate between Jenkins Master Node and Slave Nodes.

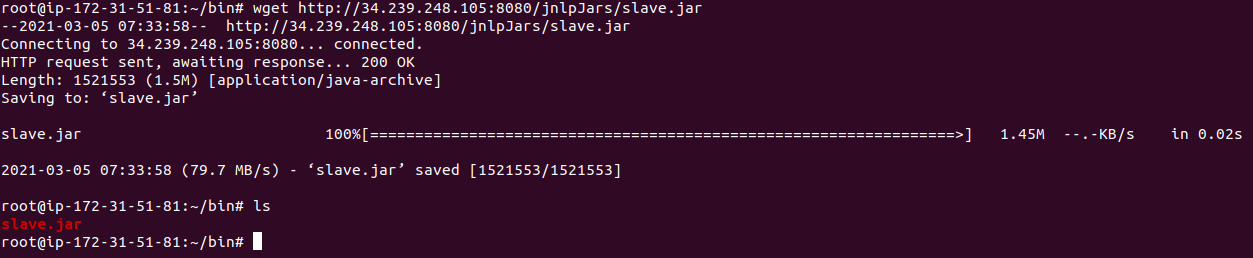
root@ip-172-31-51-81:~# mkdir bin

root@ip-172-31-51-81:~# cd bin/

root@ip-172-31-51-81:~/bin# ls

root@ip-172-31-51-81:~/bin# wget <http://34.239.248.105:8080/jnlpJars/slave.jar>

( [http://](http:// ) Pub-ip Master Node : Jenkin Default Port / jnlpJars / slave.jar)



1. Install a Jre file which will create JVM environment for Master/Slave connection

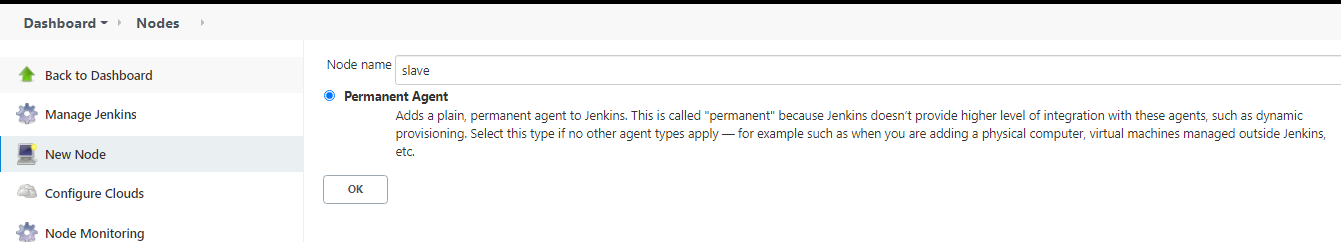
root@ip-172-31-51-81:~/bin# sudo apt update

root@ip-172-31-51-81:~/bin# apt install default-jre

1. Login to Jenkins Console: Add/configure Slave Nodes.

Go to Manage Jenkins -🡪 Manage Nodes & Clouds

1. Create a New Node slave as Permanent Agent



1. Node Configuration:

Remote Root Directory: /var/Jenkins

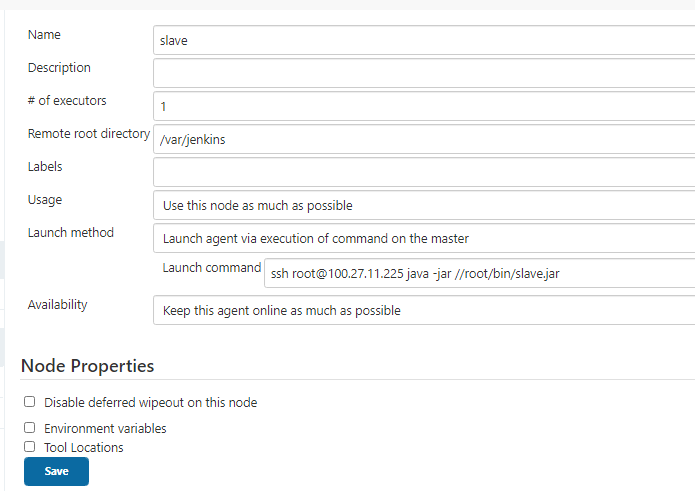
This directory is kind of a storage/volume which will be used by Master to create Workspace where built artifacts will be stored.

Launch Method:

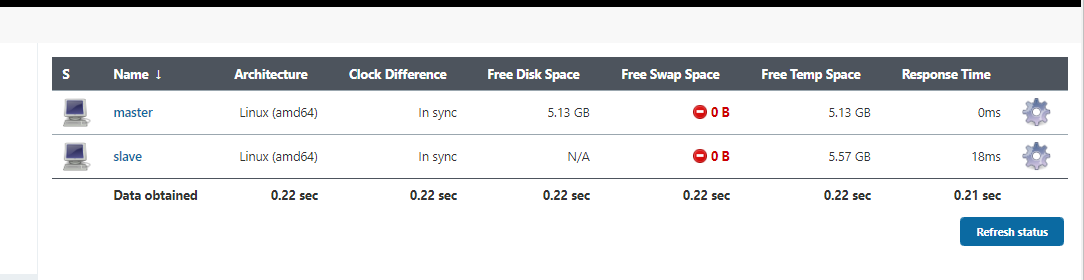
Execution of the command on the master

ssh root@100.27.11.225 java -jar /root/bin/slave.jar

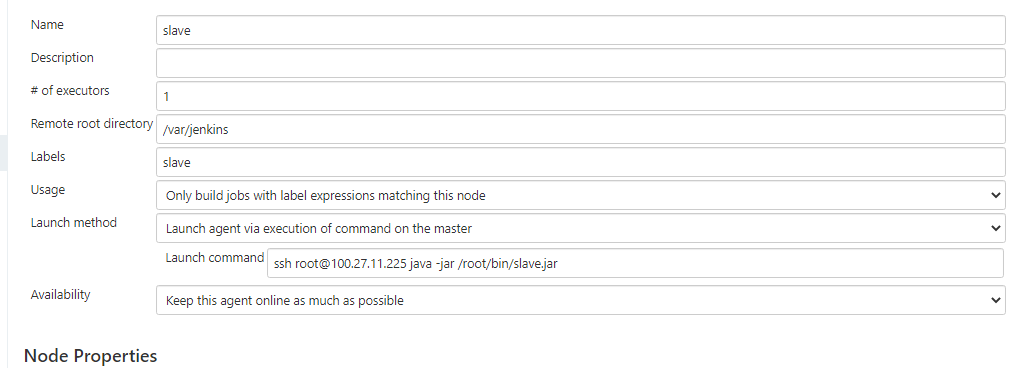
(IP – Public IP of Slave Node and /root/bin/slave.jar is based on slave.jar location in the slave node)

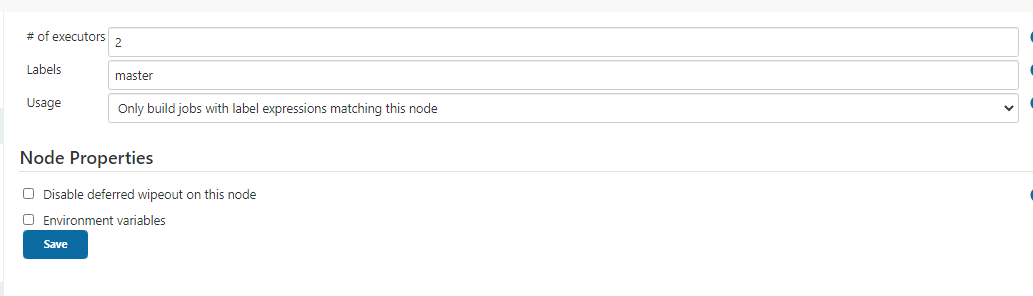


1. Check Node status



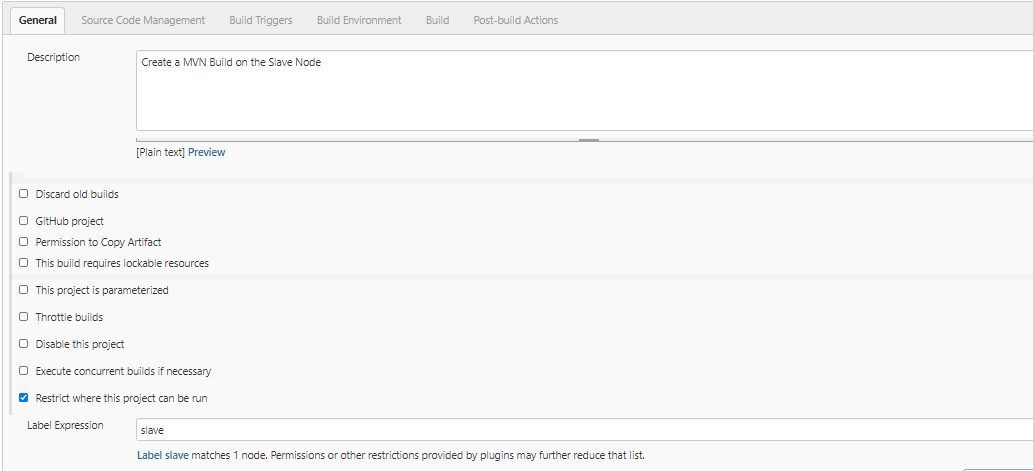
1. Labels: Setting Label and related information is critical to ensure Build/Deploy roles of the Master and Slave.

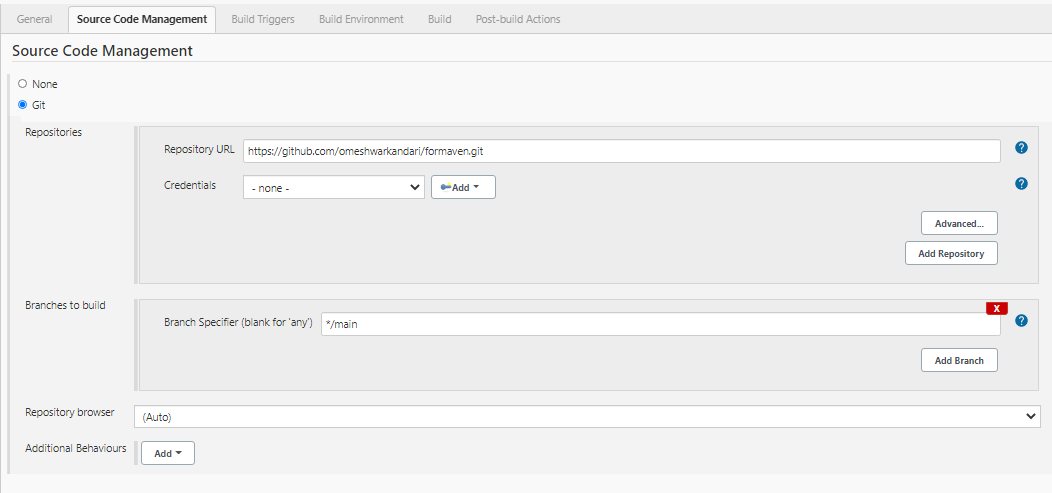
Slave Config:

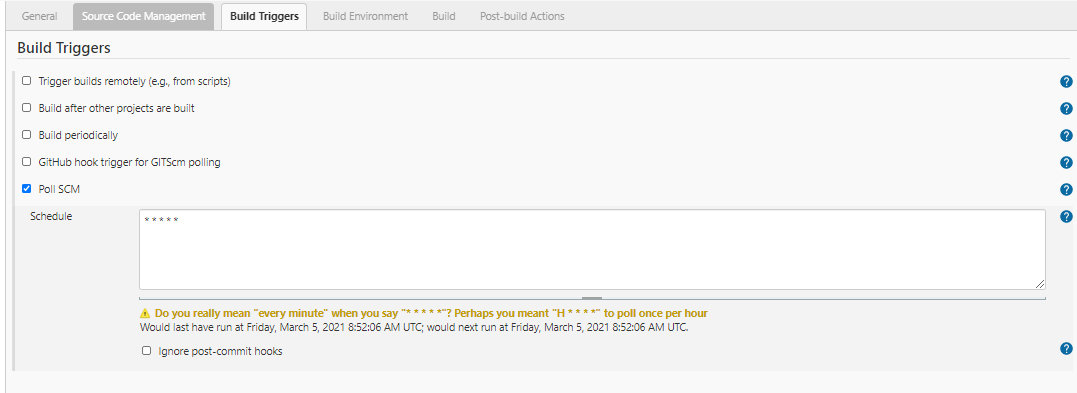
Master Config:

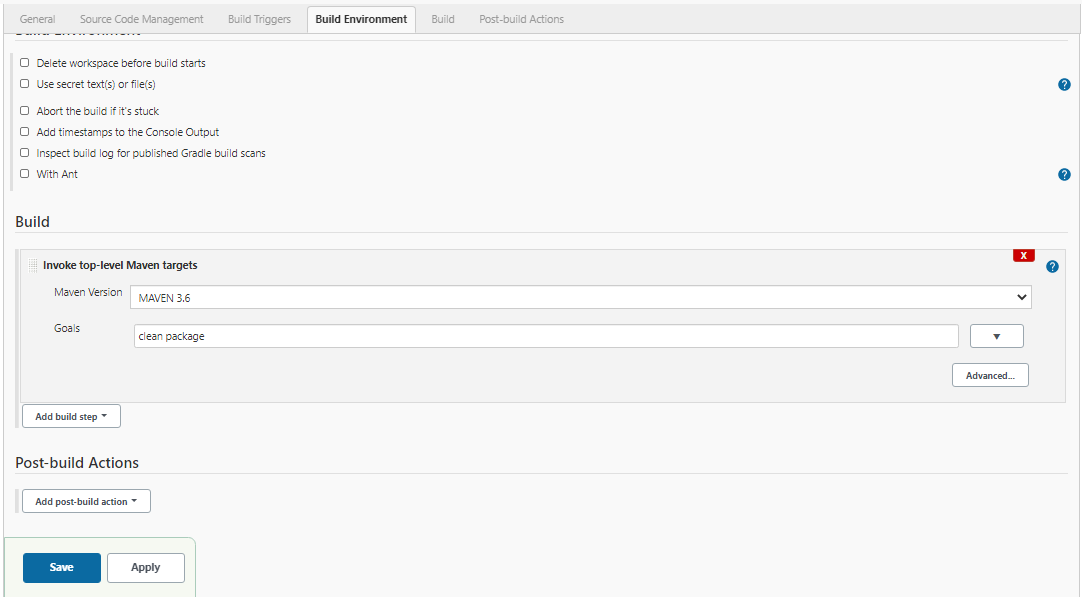
**Build a Maven Project “New Project” in Slave Node**

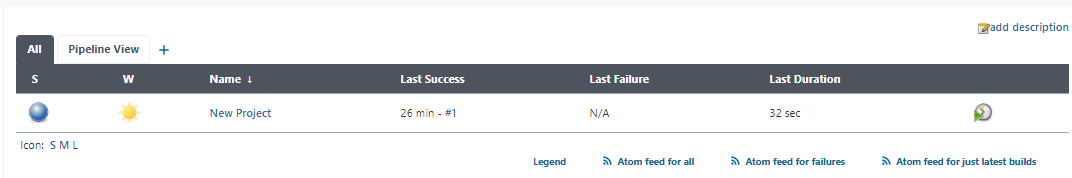
1. Create a new build job “New Project” to create a MVN Build in the Slave Node.











Build was successful and artifact should be in the Workspace under /var/Jenkins as visible below as HellowWorld-1 artifact is created in the directory accessed in slave node.



ubuntu@ip-172-31-56-87:~$ sudo -iu jenkins

jenkins@ip-172-31-56-87:~$ ssh [root@100.27.11.225](mailto:root@100.27.11.225)

root@ip-172-31-51-81:~# cd /var/jenkins/

root@ip-172-31-51-81:/var/jenkins# ls

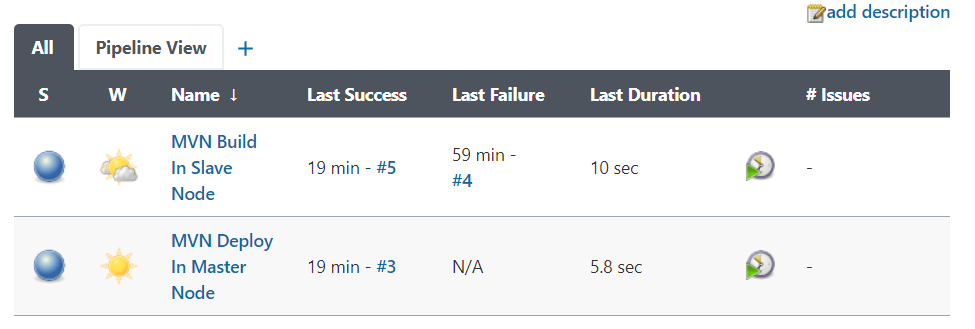
workspace

root@ip-172-31-51-81:/var/jenkins/workspace/New Project/target# ls

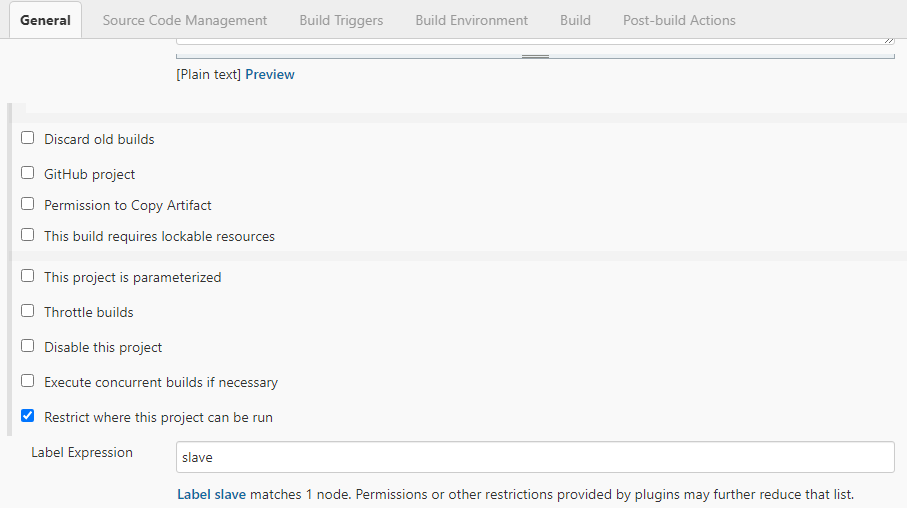
HelloWorld-1 HelloWorld-1.war HelloWorld-1.war.original classes generated-sources maven-archiver maven-status

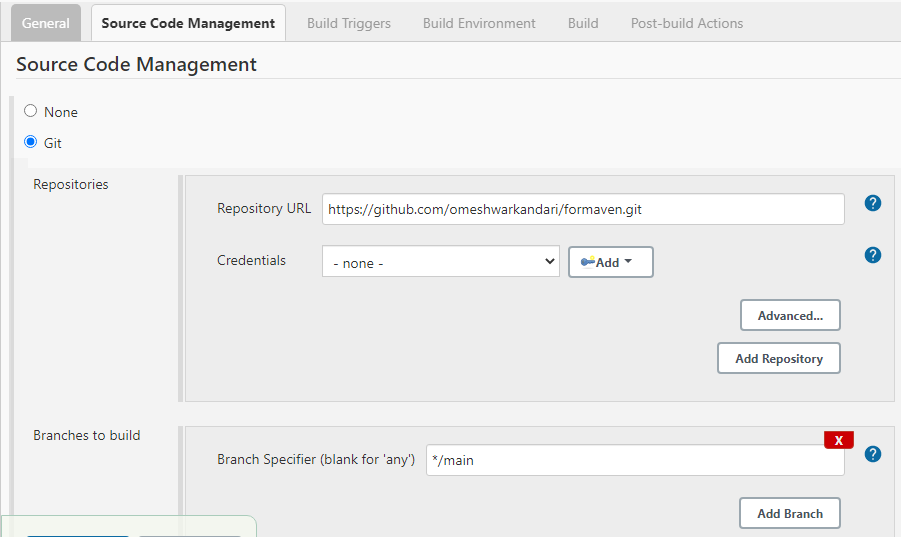
**Project 2: Build MVN Artifact in the Salve Node & Deploy MVN Artifact in the Master Node**

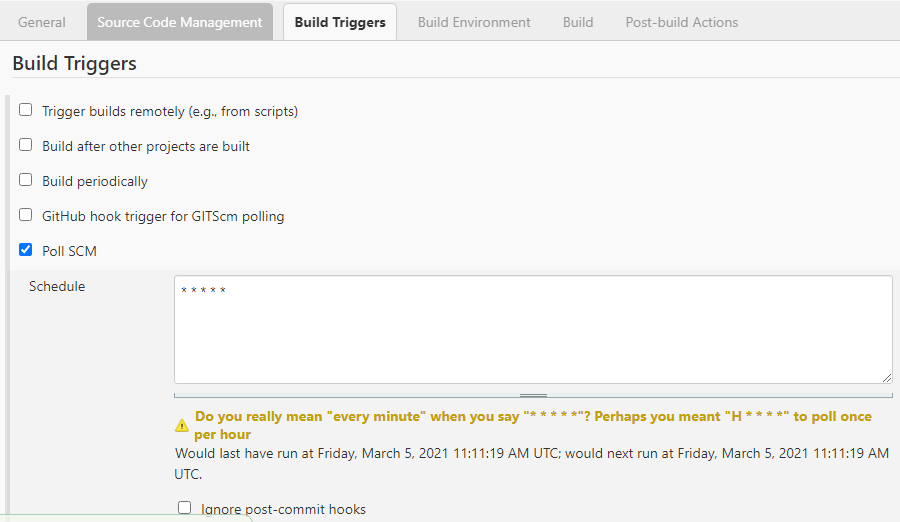
Artifact the war file is built In Slave Node and deployed in the Tomcat Container in the Master Node

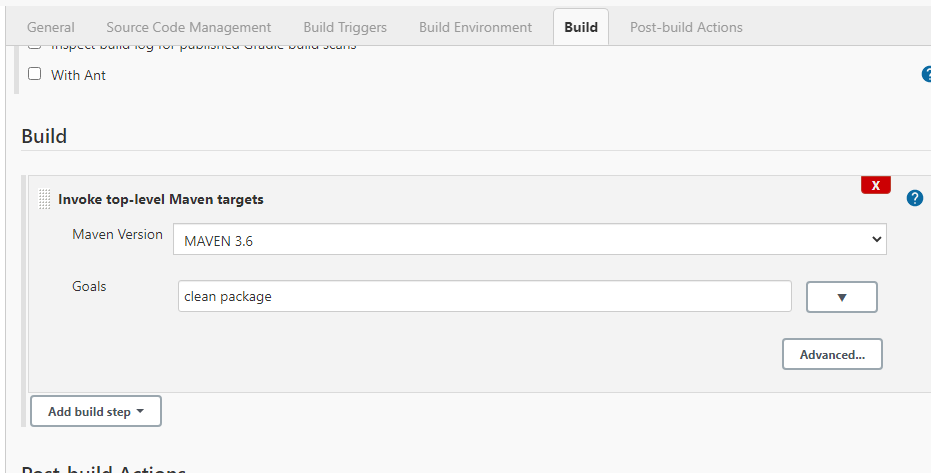


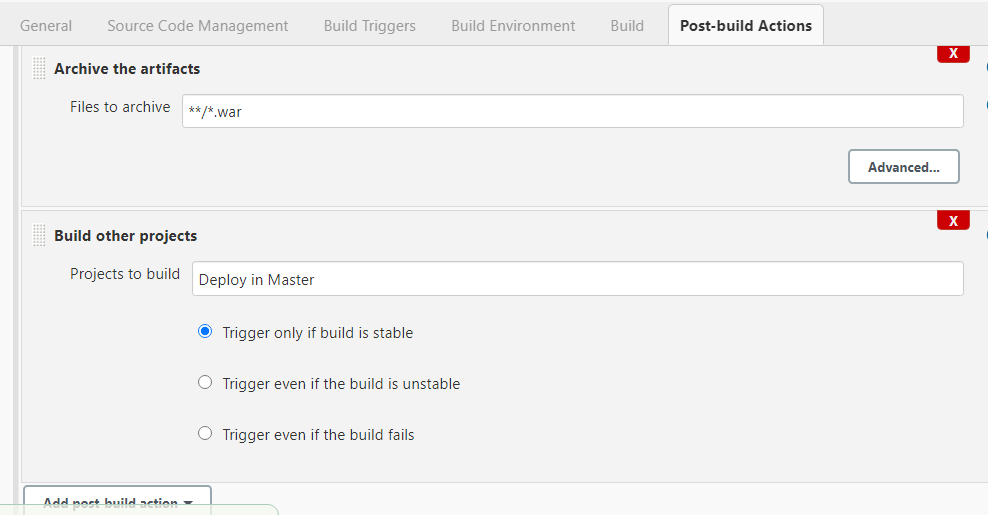
Build Project: MVN Build In Slave Node



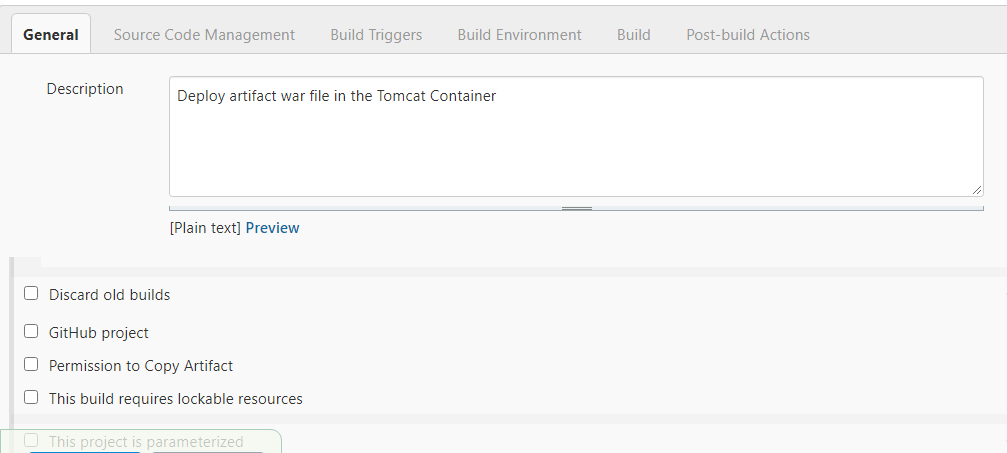








Deploy Project: MVN Deploy In Master Node



Nothing in SCM/BT/BE:

