### Java CICD Pipeline

**Flow Diagram:**

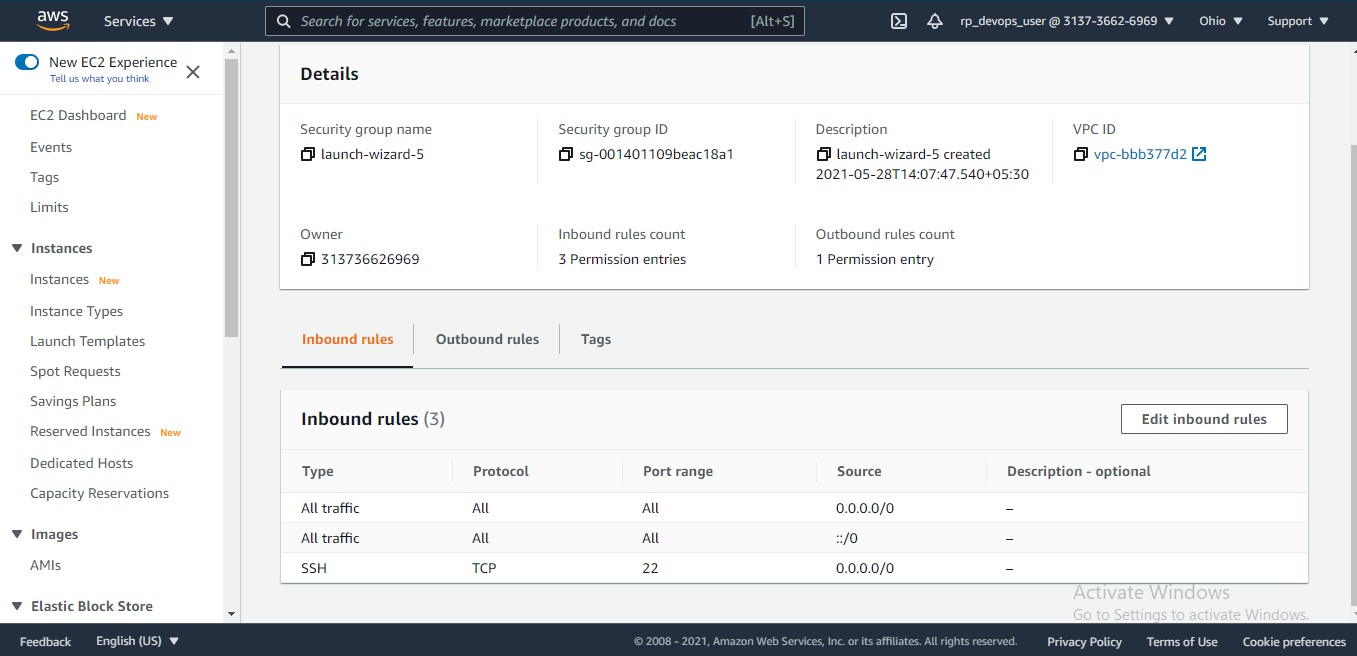
### Create Instances:

Create an EC2 Instance in AWS with Ubuntu-18.0 as the operating system with instance type t2.large and Volume size 30Gb.

**EC2 Security Groups:**

Open all traffic for the security group to allow all traffic the asses the Instance.

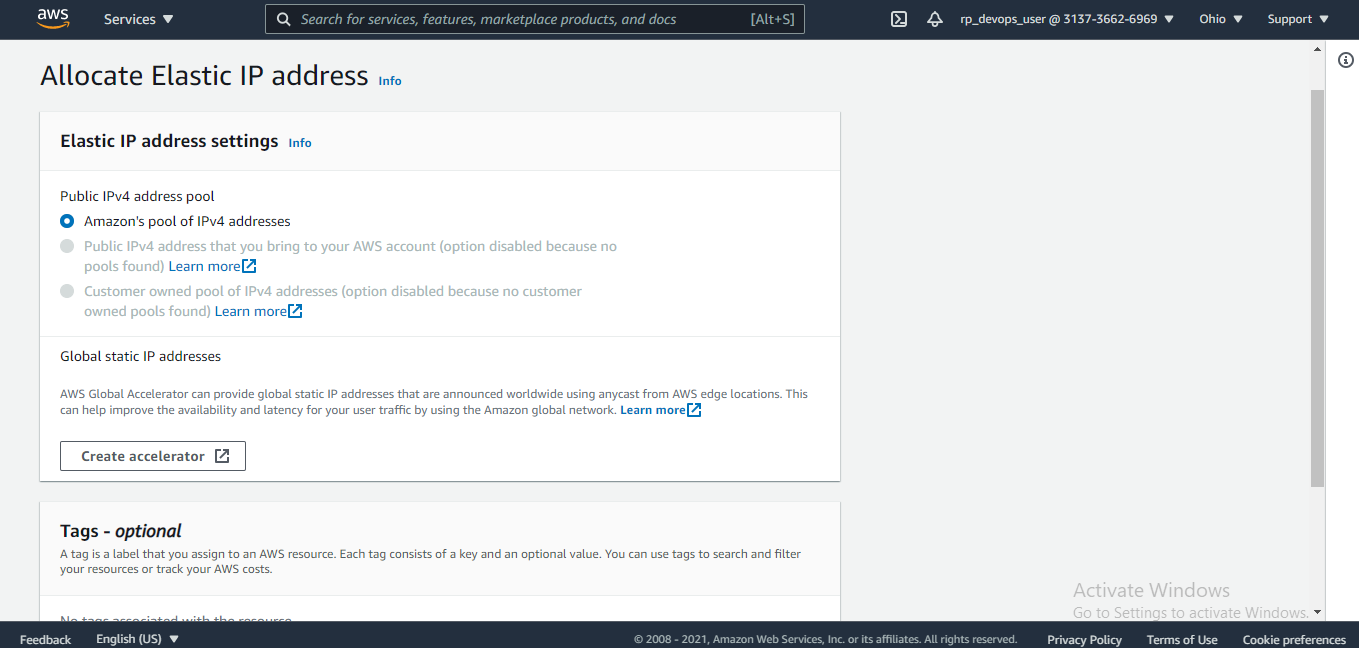
**Note**: Opening all traffic ports can cause security threats, so only open port we need to assess and to be assessed by when using for Production purposes.



**Allocate Elastic IP**

Create an Elastic IP address for the Instance by going to Elastic IP page in Ec2-Dashboard Section.

And Click on Allocate IP address Button.



After the Elastic IP address is allocated, the IP address should be associated to the Ec2 Instance Created by us. We would get a new Public IP address assigned to us which would remain **Static** till the lifetime of the Ec2 instance.

**SSH into the EC2 Instance :**

**Use the pem key assigned to us while creating the instance and DNS address to SSh into the EC2 instance.**

Use:

1.sudo apt-get update

**Install Jenkins:**

**Initially, install Java-jdk.**

(version 8)

1.sudo apt install openjdk-8-jdk.

**Jenkins:**

1. First, add the repository key to the system:

wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add –

2. When the key is added, the system will return OK. Next, append the Debian package repository address to the server’s sources.list.

sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'

3. When both of these are in place, run update so that apt will use the new repository:

sudo apt update

4. Finally, install Jenkins and its dependencies:

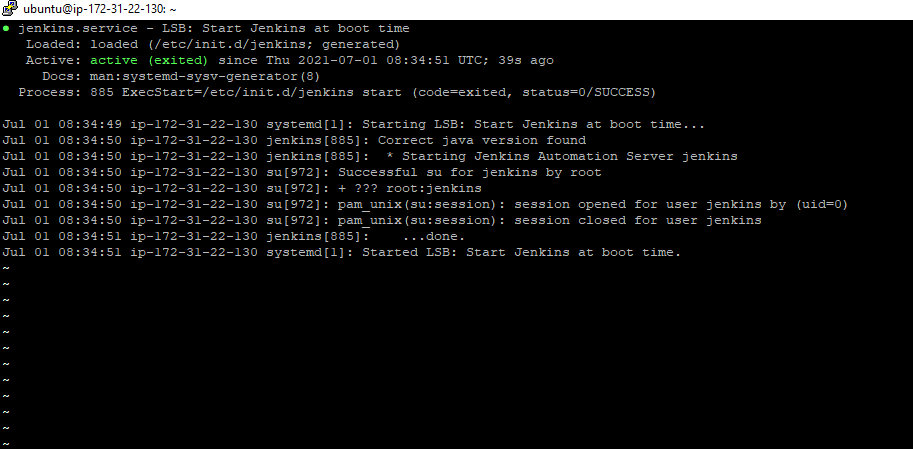
sudo apt install Jenkins

5. After Jenkins and its dependencies are installed, we’ll start the Jenkins server

sudo systemctl start Jenkins

6. Check the status of the Jenkins application installed.

sudo systemctl status Jenkins.

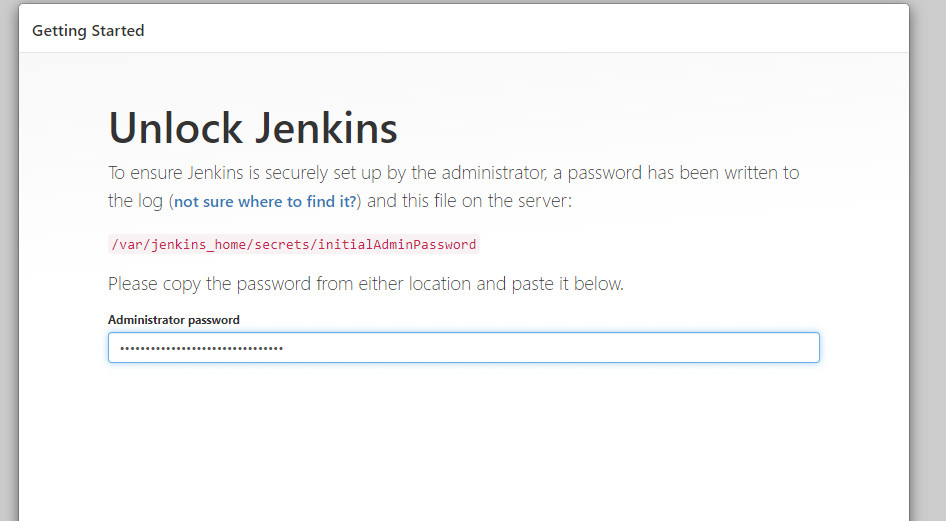


If the Status is Active then we are can access the Jenkins service in our Ec2 by using Public IP on port 8080.

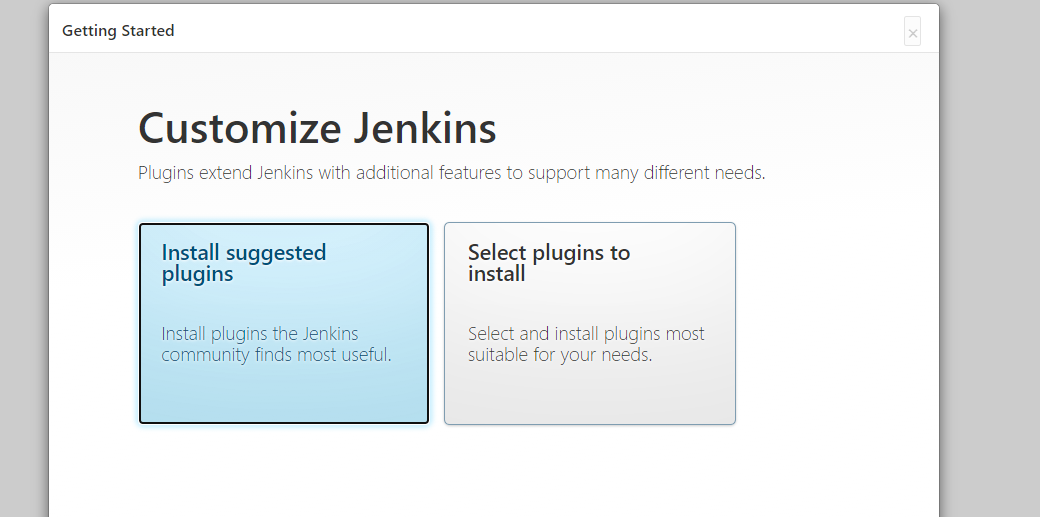
3.129.234.9:8080

Unlock Jenkins by using command:

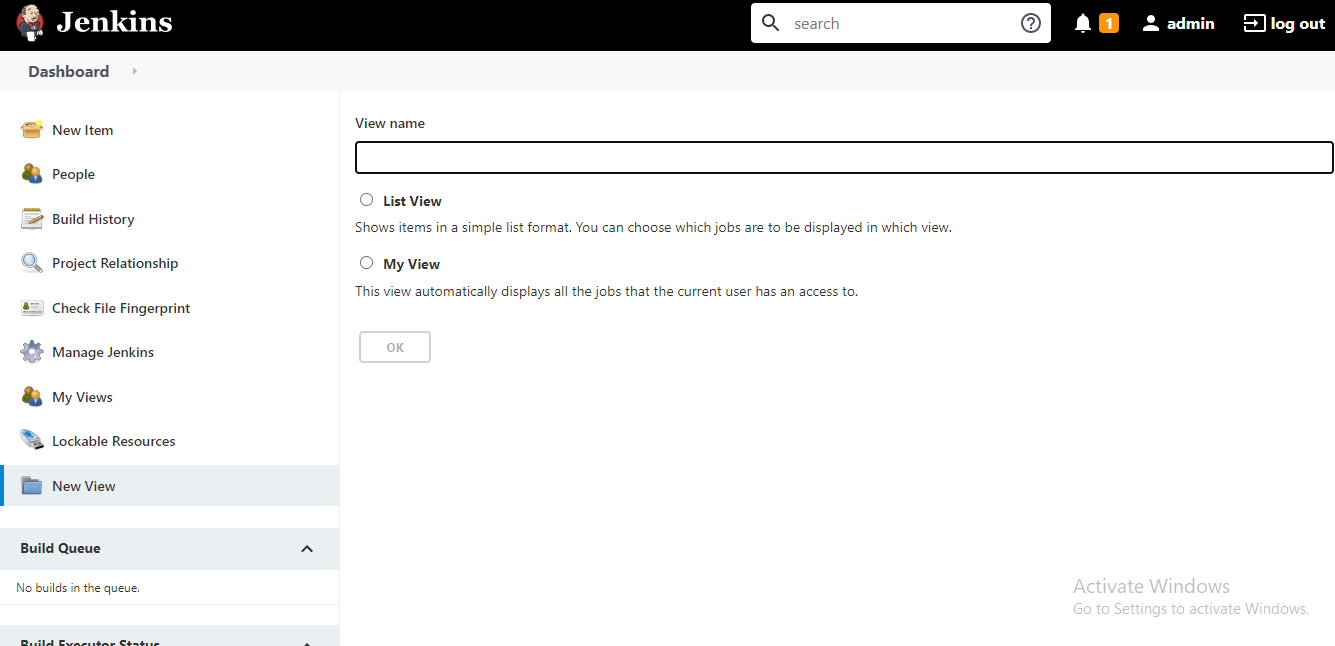
cat var/Jenkins\_home/secrets/initialAdminPassword



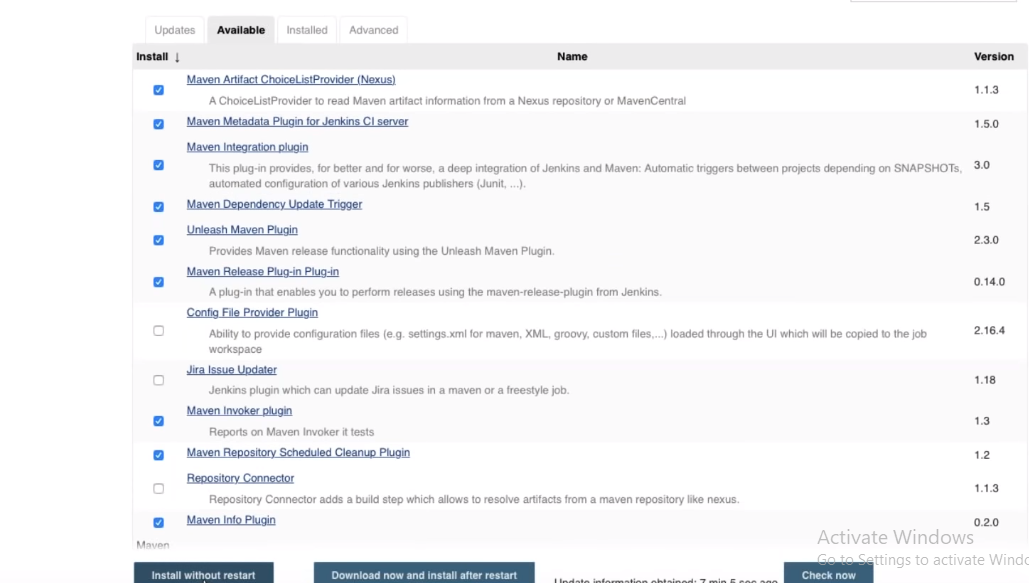
Install suggested plugins :



Install all the plugins for the process by going to Manage Pluggins ->Available and search for these and install them.

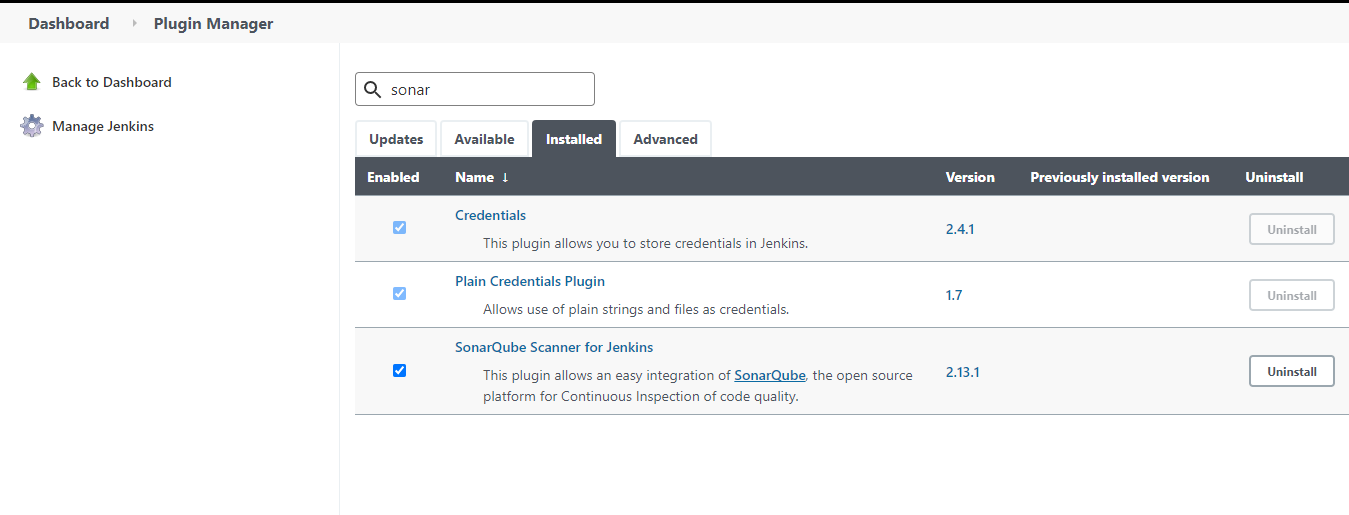


Installing Maven plugin in Jenkins, Search for maven in search panel and select the following plugins click on install without restart.

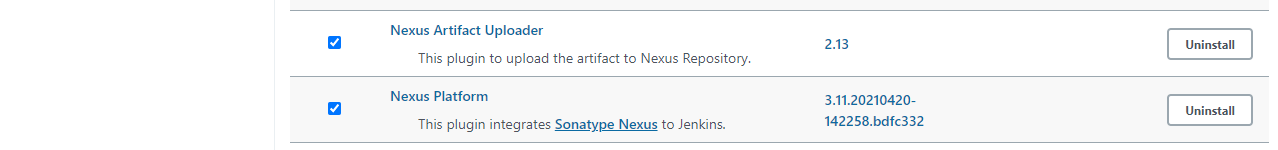


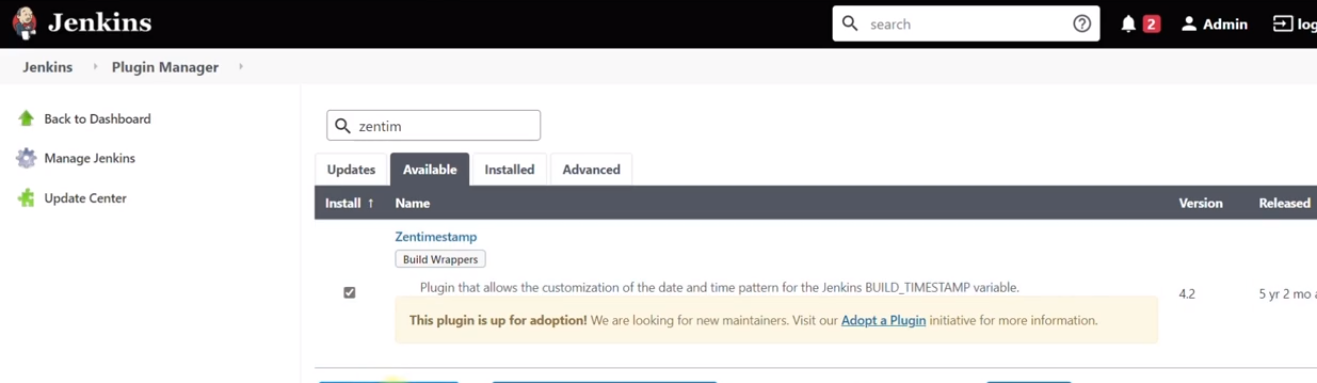
Install Sonarqube and Nexus plugins in Jenkins:

Sonarqube plugins:



Nexus plugins:

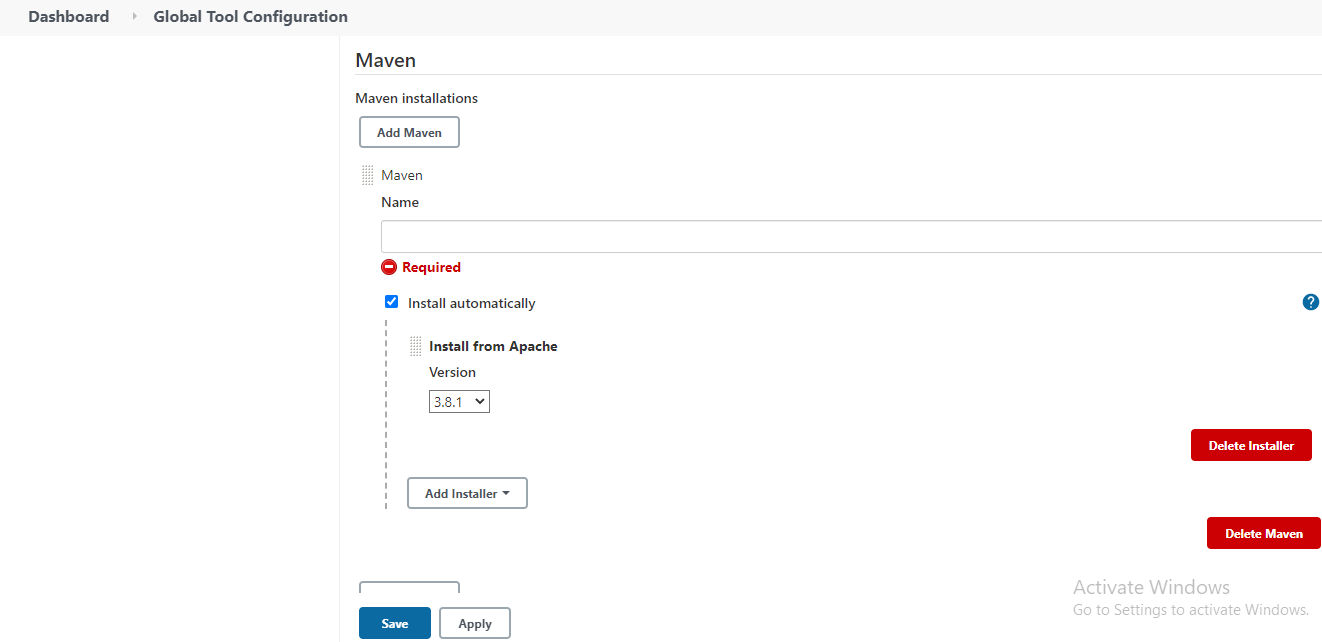


****

Goto Manage Jenkins->Global Tool Configuration. Scroll down to SonarQube Scanner and Enter the Name and Check the Install automatically box. Leave rest of the field default. Click apply.

****

After completing the above step scroll down and Click Add Maven. Enter the Name and Check the Install automatically box. Leave rest of the field default. Click apply.

****

**Install Maven in EC2:**

**Method 1:**

**Installing Maven on Ubuntu using apt:**

1.Start by updating the package index:

sudo apt update

2. Next, install Maven by typing the following command:

sudo apt install maven

3. Verify the installation by running the mvn -version command:

mvn –version

**Alternative Method 2:**

**Install the Latest Release of Apache Maven:**

1. Assuming you have already installed Java Jdk, if you have go to /usr/lib/jvm location and check out the java jdk folder installed.
2. Download the required Apache Maven version, the version used here is 3.6.0. Start by downloading the Apache Maven in the /tmp directory using the following wget command:

$ wget <https://www-us.apache.org/dist/maven/maven-3/3.6.0/binaries/apache-maven-3.6.0-> bin.tar.gz -P /tmp

1. Once the download is completed, extract the archive in the /opt directory:

$ sudo tar xf /tmp/apache-maven-\*.tar.gz -C /opt

4. To have more control over Maven versions and updates, we will Create a symbolic link maven that will point to the Maven installation directory:

$ sudo ln -s /opt/apache-maven-3.6.0 /opt/maven

5.set up the environment variables. To do so, open your text editor and create a new file name maven,sh inside etc/profile.

$sudo nano /etc/profile.d/maven.sh

6. Paste the following into the nano editor

export JAVA\_HOME=/usr/lib/jvm/ java-8-openjdk-amd64

export M2\_HOME=/opt/maven

export MAVEN\_HOME=/opt/maven

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

**Note**: Assuming the Java-jdk we downloaded is java-8-openjdk-amd64

7. Make the Script executable using chmod:

sudo chmod +x /etc/profile.d/maven.sh

8. Finally load the environment variables using the Source command:

$ source /etc/profile.d/maven.sh

9. check the installed maven version:

mvn –version

**Install Sonarqube:**

sudo su

apt-get update

wget [https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-7.3.zip](https://www.youtube.com/redirect?event=comments&redir_token=QUFFLUhqbTFyOFQyam1nTEZKajgzdDJMQjgyN0c3RzRWd3xBQ3Jtc0trWDQ3blEtX1pRVkwyZVVYa3ptd3VXZHh3U1pmQzkzZXRSN2ZJM2c1MU1FbEhJODZYbFhCU0tqcVJoVmRNS1h6bVF0RFI2Y2dqdlJadHRpM1ZmM3NJTlpBQ1BHeEUtcjdDb1Nqc0hzeGItcFB4dmNzYw&q=https%3A%2F%2Fbinaries.sonarsource.com%2FDistribution%2Fsonarqube%2Fsonarqube-7.3.zip&stzid=UgxCI2I5MG5K-oqxHLZ4AaABAg.9KPv0LrJbAs9KzediPgoAo)

apt-get install unzip –y

unzip [sonarqube-7.3.zip](https://www.youtube.com/redirect?event=comments&redir_token=QUFFLUhqbDFjenFLTjhEdkMzX2dIVWVxQmpwODVycUhXQXxBQ3Jtc0tubmJFSnliYXZocGFrWGU0TG0xc0VNODhmTU04VkpMSF9IMmR0bXI2UTg0WHVqM0IxX3M5RHlyRTkwcDRRQzRKTVdOVkNrRHJJa3FSa1NMdXl1dHNDZjFwSHVOUzItbDFPbV9Ob3NBeVlVR2p4UzhqZw&q=http%3A%2F%2Fsonarqube-7.3.zip%2F&stzid=UgxCI2I5MG5K-oqxHLZ4AaABAg.9KPv0LrJbAs9KzediPgoAo)

sudo chown -R ubuntu:ubuntu sonarqube-7.3

su ubuntu

cd /home/ubuntu/sonarqube-7.3/bin/linux-x86-64

./[sonar.sh](https://www.youtube.com/redirect?event=comments&redir_token=QUFFLUhqbmxrcC1qLWphSTI2X3VMeWdjTTlxWnZVNnJSd3xBQ3Jtc0tuZHhYNktpdndfQXJwS0VIcGF6OHdCaTdQRlVMN2hzT2FyX2J4RXhncGFPQkVuMkhrX1BEZ1Z0RlZ1NVFNUDA2VWJYdFAtRTJJZlRRdTZPUldxeDBGZXNlQkFmOExRRGVKMWJ4NmUyQW1LdGlsVkVaRQ&q=http%3A%2F%2Fsonar.sh%2F&stzid=UgxCI2I5MG5K-oqxHLZ4AaABAg.9KPv0LrJbAs9KzediPgoAo" \t "_blank) console &

To see different sonarqube process commands use:

./sonar.sh asdf

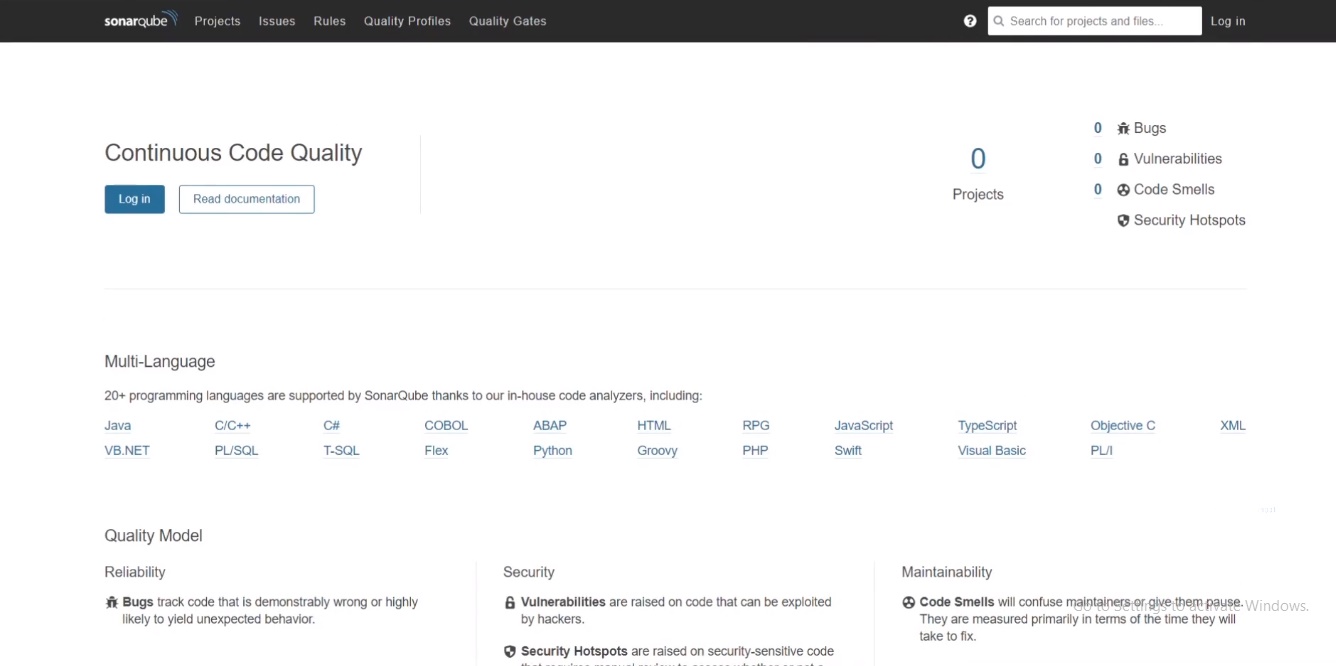
**Note**: If you run sonarqube command with ROOT access, Elasticsearch might not run with Root so sonarqube will not start. Sonarqube-7.3 version is used at the time of creating this document.

Access Sonarqube using public ip on port 9000:

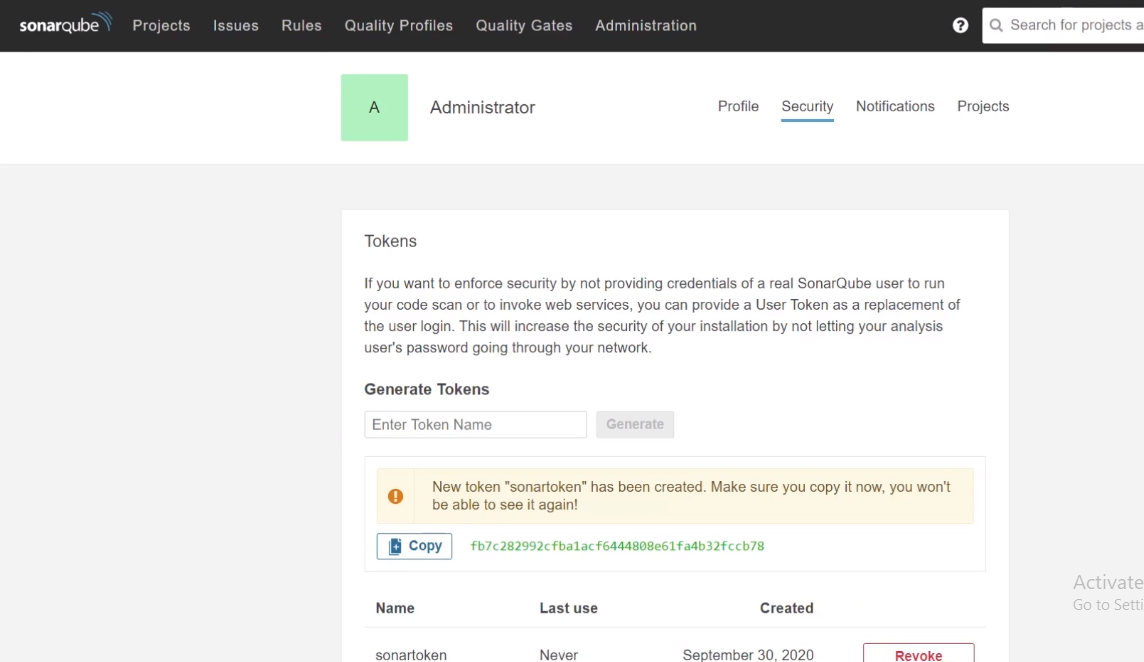
3.129.234.9:9000

Default creds: admin/admin123

SonarQube Dashboard:



Goto Administator->My Account->Security->Enter token name(Give any name). Copy the token Generated and save it somewhere we would use it later.



Install Nexus:

1. Download the tar file and unzip and install nexus

sudo su

cd /opt

wget https://sonatype-download.global.ssl.fastly.net/nexus/3/nexus-3.0.2-02-unix.tar.gz

tar -zxvf nexus-3.0.2-02-unix.tar.gz

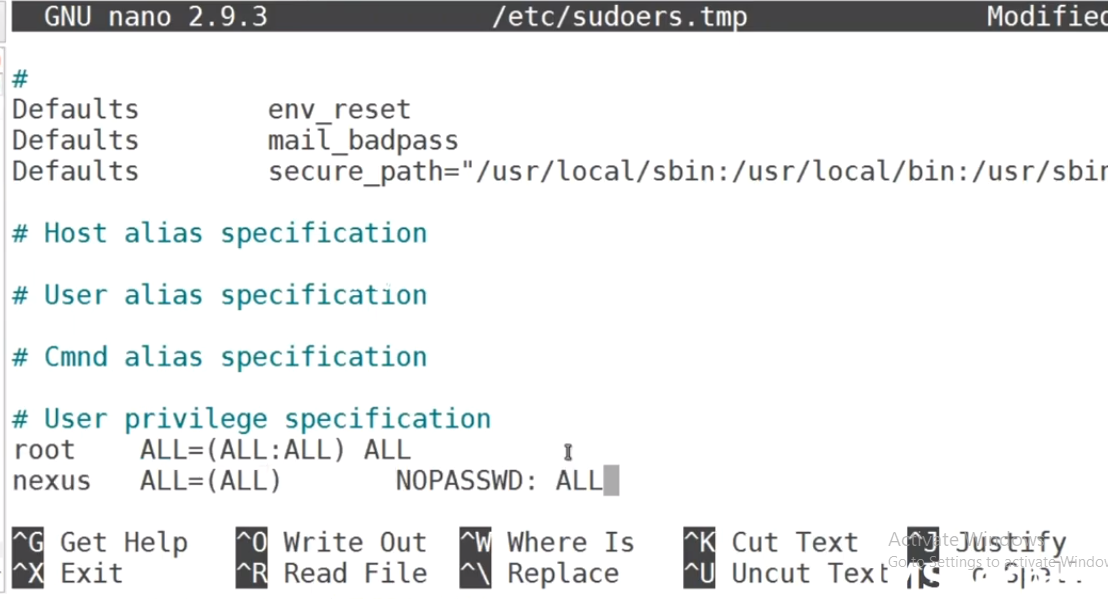
mv /opt/nexus-3.0.2-02/opt/nexus

2. As a good security practice, it is not advised to run nexus service as root. so create new user called nexus and grant sudo access to manage nexus services.

sudo adduser nexus

//(Enter New Nexus Password):

sudo visudo(to edit the sudo file and add the nexus details specified below)



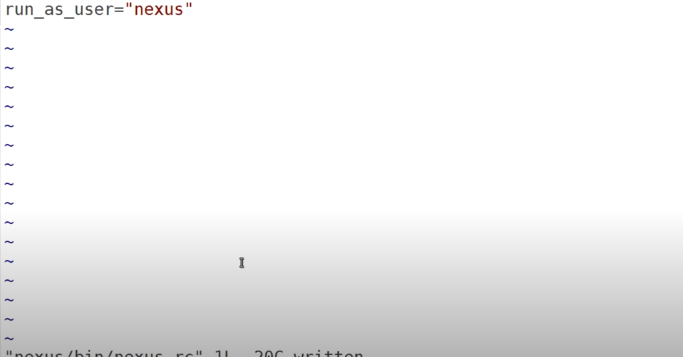
sudo chown -R nexus:nexus /opt/nexus

sudo chown -R nexus:nexus /opt/sonatype-work

3. Open /opt/nexus/bin/nexus.rc file, uncomment run\_as\_user parameter and set it as following.

sudo vim /opt/nexus/bin/nexus.rc

(edit using vim editor and add this below detail in the page and save) &&



4. Add nexus as a service at boot time

sudo ln -s /opt/nexus/bin/nexus /etc/init.d/nexus

5. Login as a nexus user and start service.

$su - nexus

$/etc/init.d/nexus start

(to check the active nexus servers running) sudo netstat -plnt

Use public ip address with port number 8081 to open Nexus Dashboard.

http:// 3.129.234.9:8081

**Note:** (If the port is not opening use) - sudo ufw allow 8081

Use default credentials to login

username : admin  
password : admin123

# Note: If you encounter this error “[Installing nexus error: The version of the JVM must be at least 1.8 and at most 1.8](https://stackoverflow.com/questions/55111719/installing-nexus-error-the-version-of-the-jvm-must-be-at-least-1-8-and-at-most)”

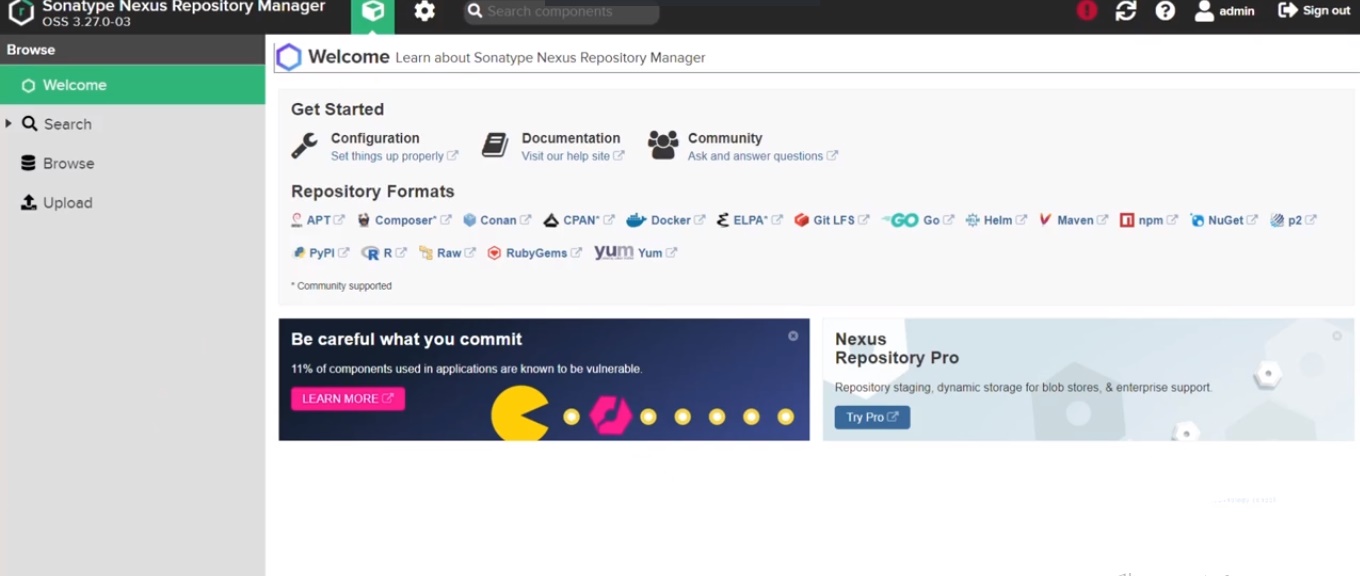
1.Edit the file /opt/nexus/bin/nexus using a text editor:

sudo vi /opt/nexus/bin/nexus

2.Add the Below value and save the changes INSTALL4J\_JAVA\_HOME\_OVERRIDE=

INSTALL4J\_JAVA\_HOME\_OVERRIDE=/usr/lib/jvm/java-8-openjdk-amd64

Nexus Dashboard:



After accessing the Nexus Dashboard:

1.Goto Settings->Repository->Create Repository-> maven2(Hosted).

a)Enter the Repo Name: (My Repo Name: maven-releases)

b)Leave rest as default and Click on Create Repository

2. Click on Create Repository-> maven2(proxy).

a)Enter the Repo Name: (My Repo name:vpro-maven-central)

b) In the Enter URL field (specify the Maven Public repository URL):<https://repo1.maven.org/maven2/>

c) Click on Create Repository

3. Create a snapshot Repo, Click on Create Repository-> maven2(hosted).

a)Enter a Name : (My Repo Name: vprofile-snapshot)

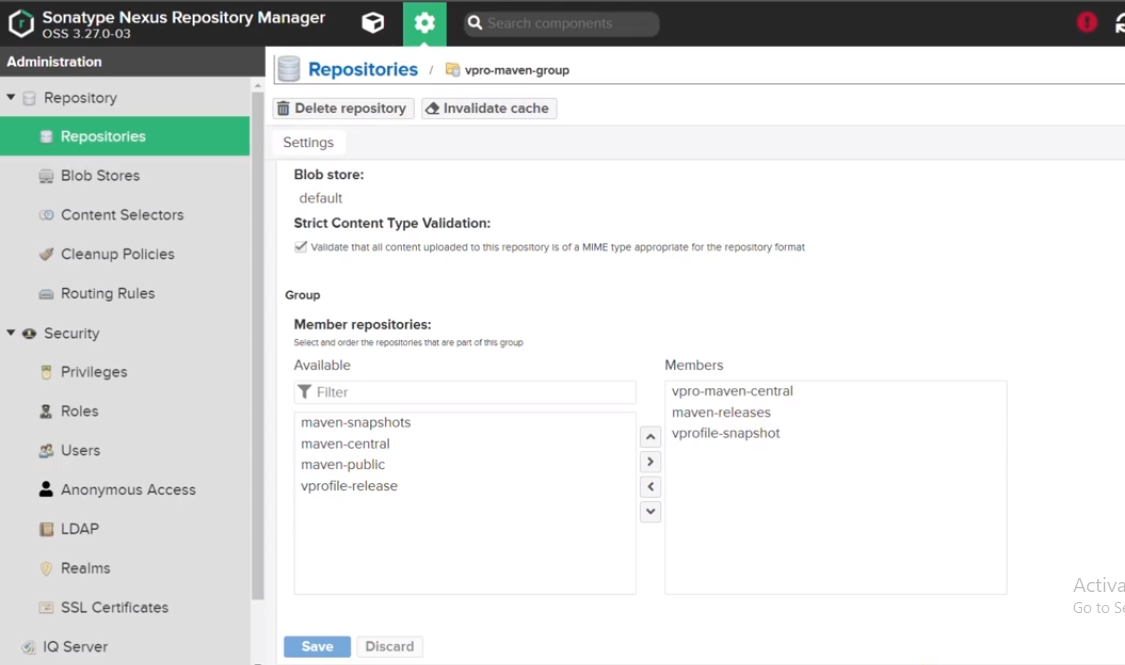
b)under version choose snapshot. And click on Create Repository

4. Click on Create Repository-> maven2(group).

a)Enter the Repo name: (My Repo Name: vpro-maven-group)

b)Under Members Repository add all the above Created repository and Save

1. Dashboard view of the group-Repository created in step-4.



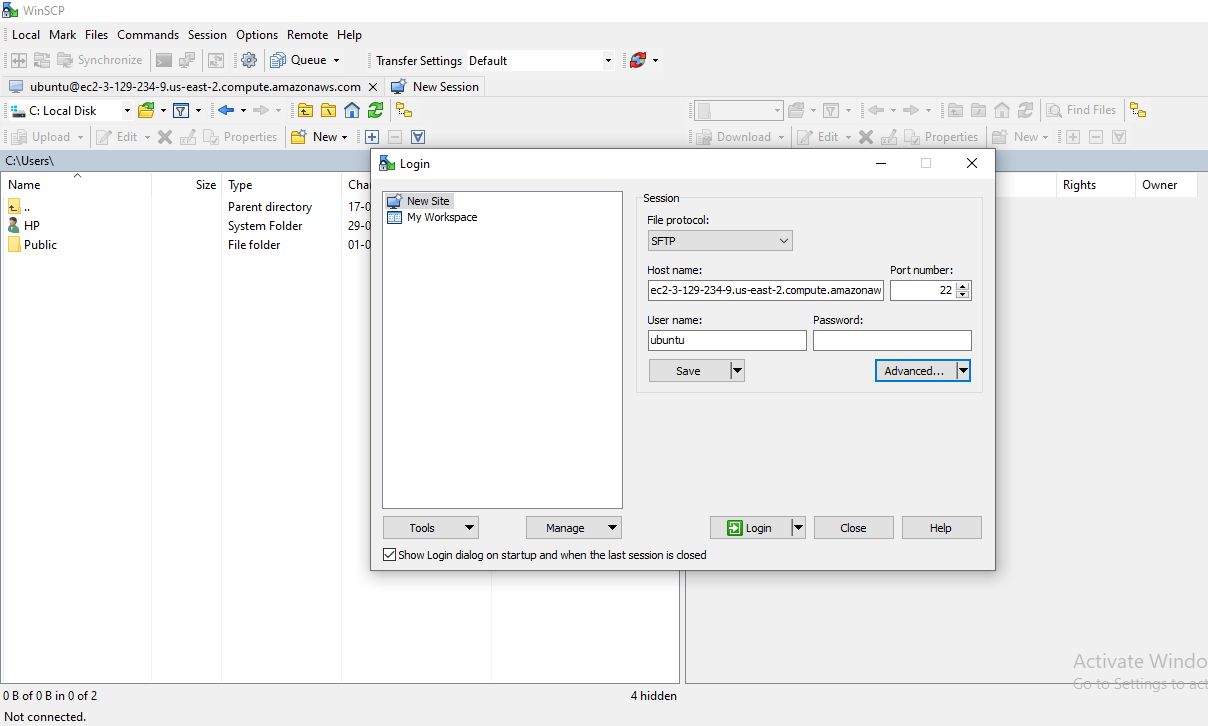
**Install Jira:**

1. Go to the below url to Download the required Jira version zip file in your Local, to run it for Linux os.

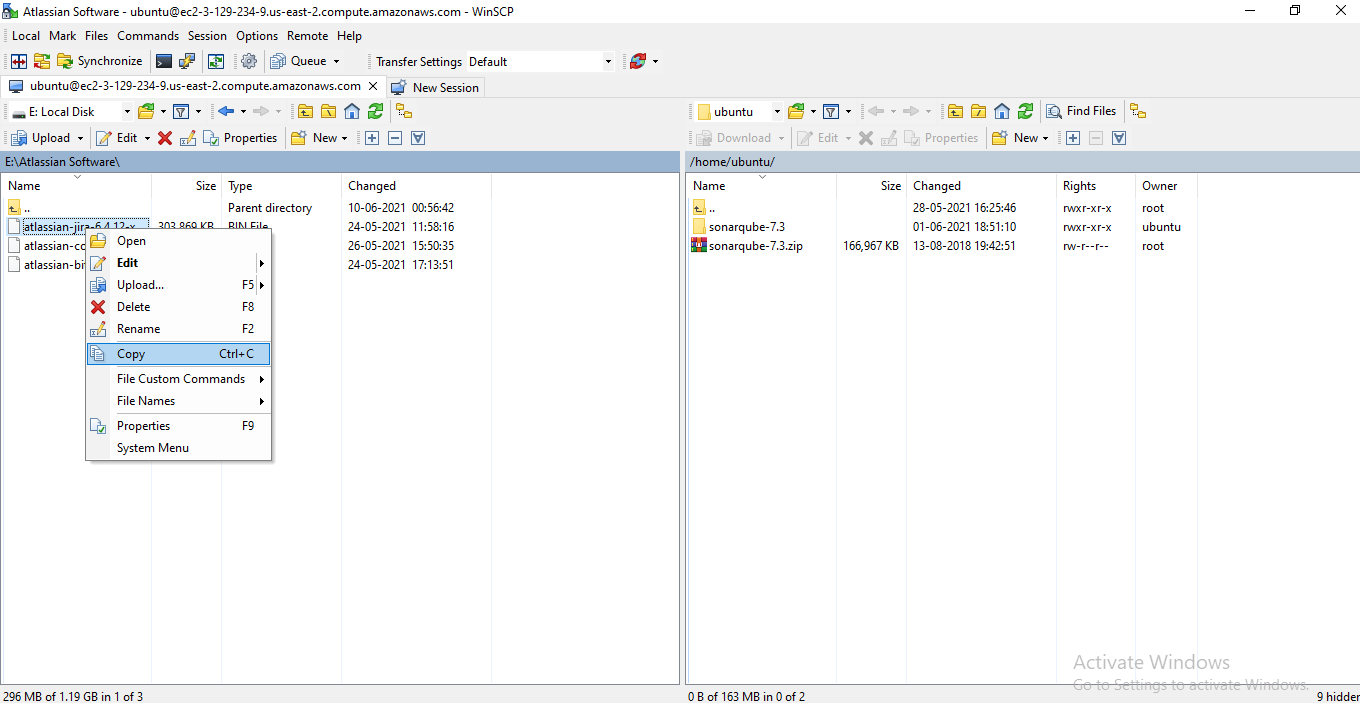
https://www.atlassian.com/software/jira/update

2.Download Winscp Software in your local and access the Ubuntu Instance you want to install the Jira application to:

<https://winscp.net/eng/download.php> (winscp download link)



3.Transfer the Jira zip file from your local by copy pasting the software to the cd home/ubuntu location



4.unzip the file using

chmod a+x atlassian-jira-7.12.3-x64.bin (Jira 7.12.3-64x was the version used during creation of this doc)

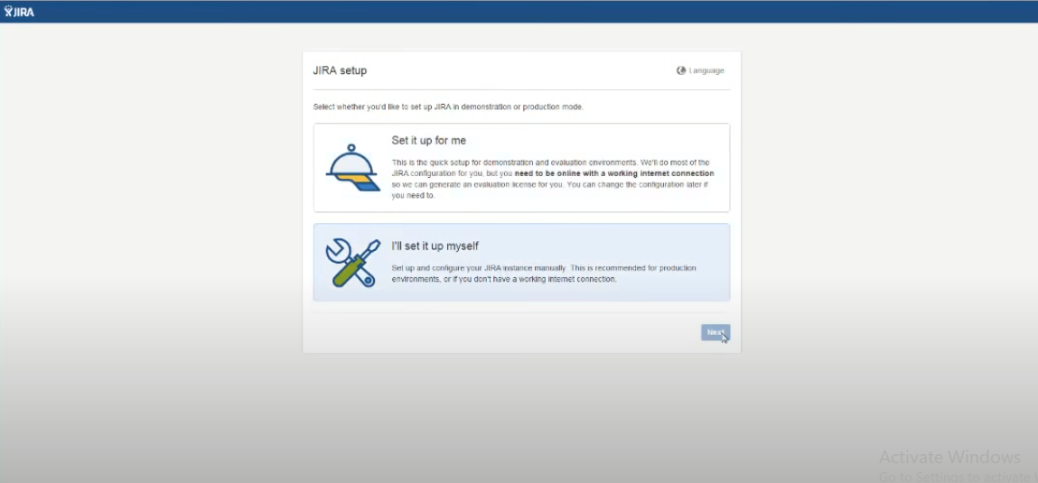
sudo ./ atlassian-jira-7.12.3-x64.bin

Press 1 for Default version

Enter the port number of your choice and press i

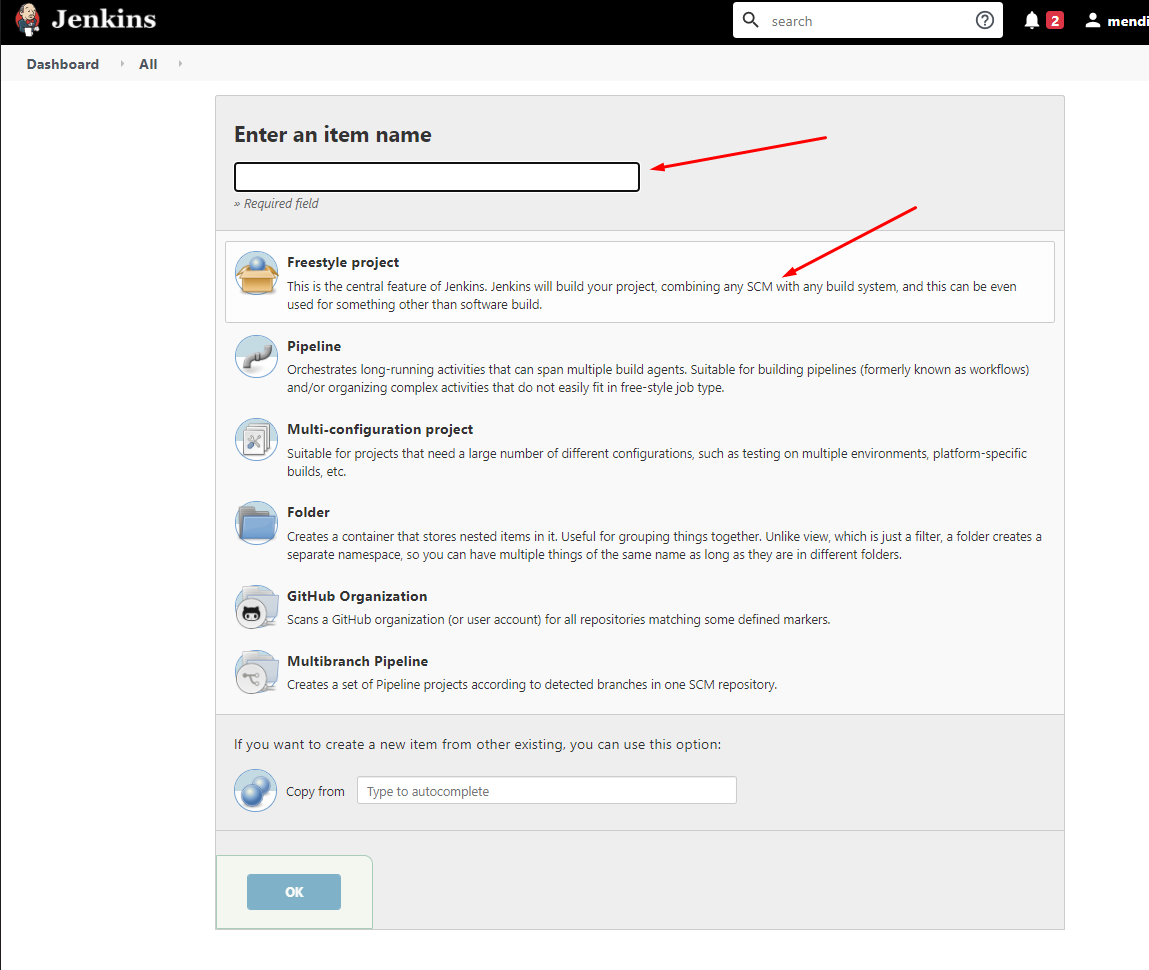
jira would be available in the public ip on the user specified port: 3.129.234.9:<specified\_port>

Jira Dashboard:



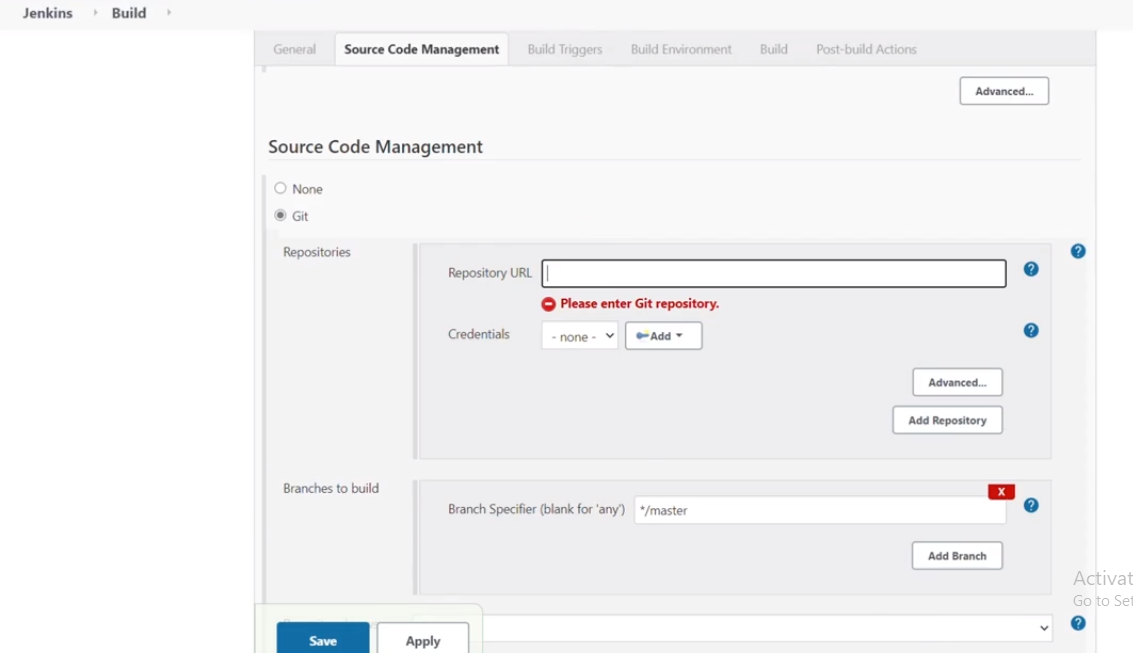
Access Jenkins:

1. On the Jenkins Dashboard Goto New Item->FreeStyle Project. Enter Name and click ok.



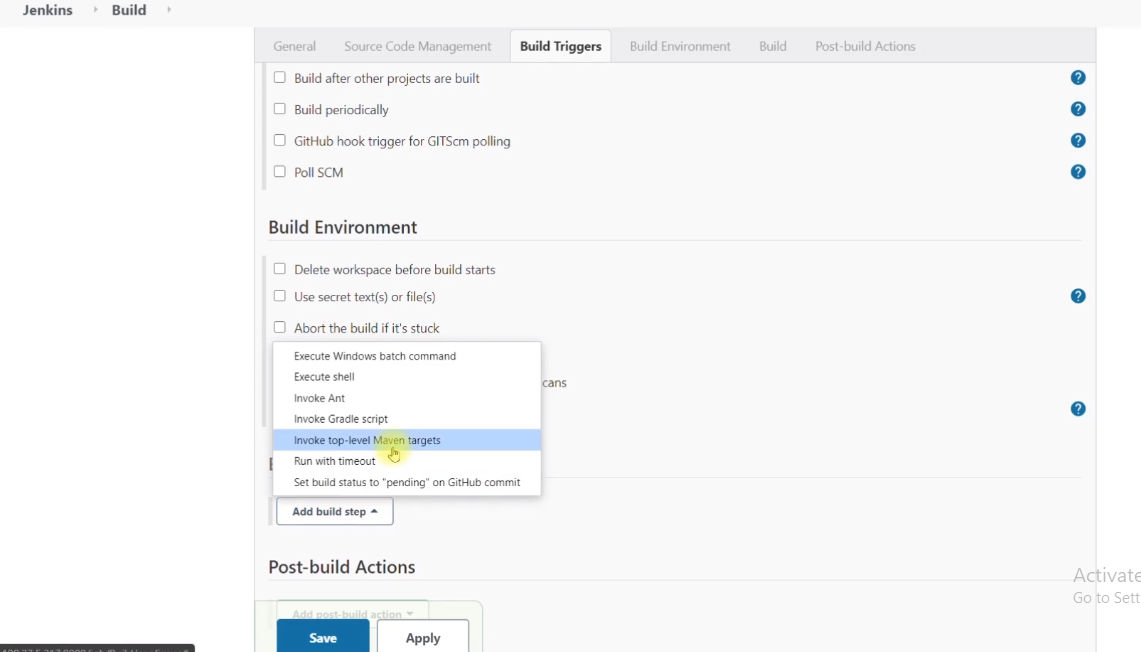
1. Paste the Git Repository name you want to use.

**Use**: https://github.com/Roshan05mura/Java-CICD.git

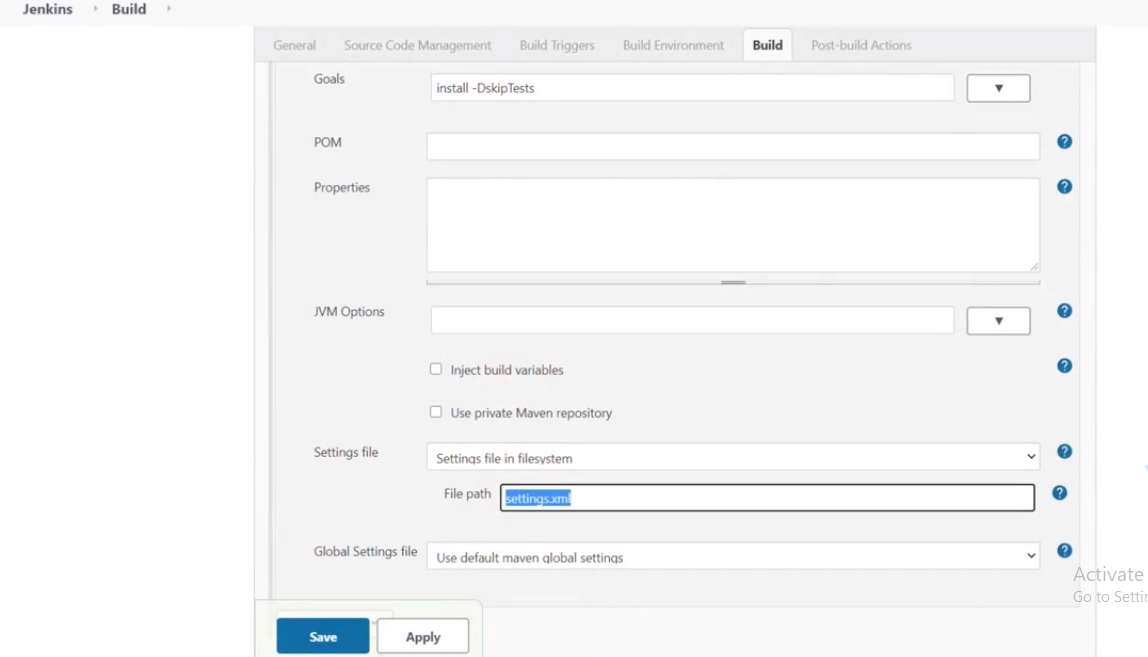


cccc

3.Goto Build Environment->Add Build-step->Invoke top-level maven target.



1. Add in the below details.



1. Add the details of all the Created Nexus Repository names the Ip, port number and login creds of nexus in the properties section.

**Note**: My Properties Details. Access the IP in public Ipv4 from Ec2 dashboard.

SNAP-REPO=vprofile-snapshot

NEXUS-USER=admin

NEXUS-PASS=admin123

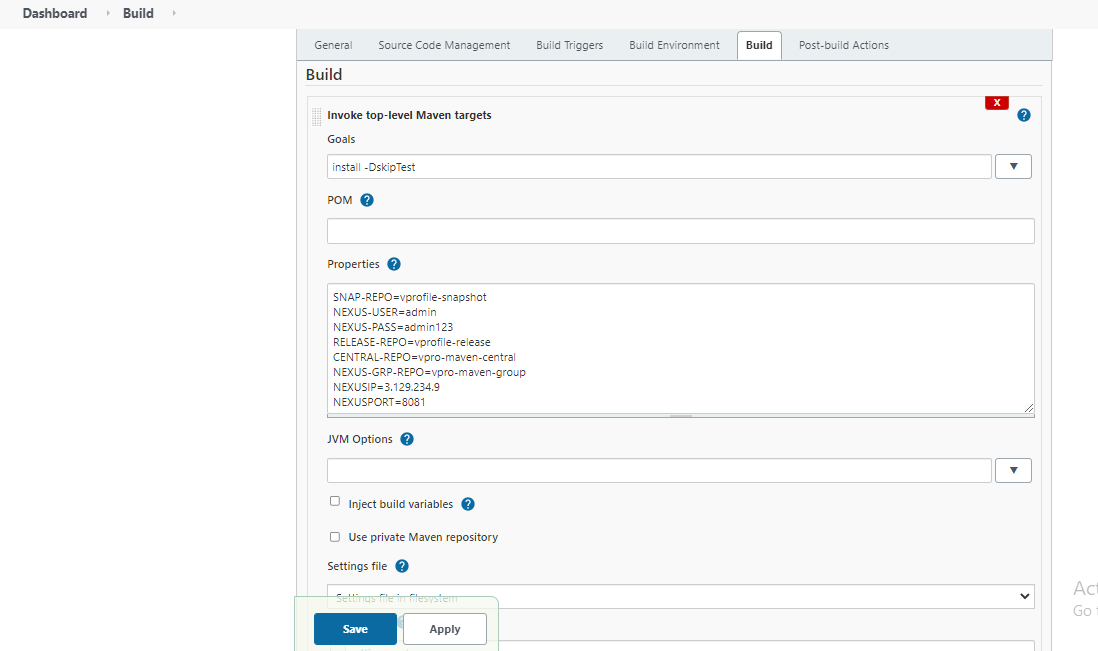
RELEASE-REPO=vprofile-release

CENTRAL-REPO=vpro-maven-central

NEXUS-GRP-REPO=vpro-maven-group

NEXUSIP=3.129.234.9

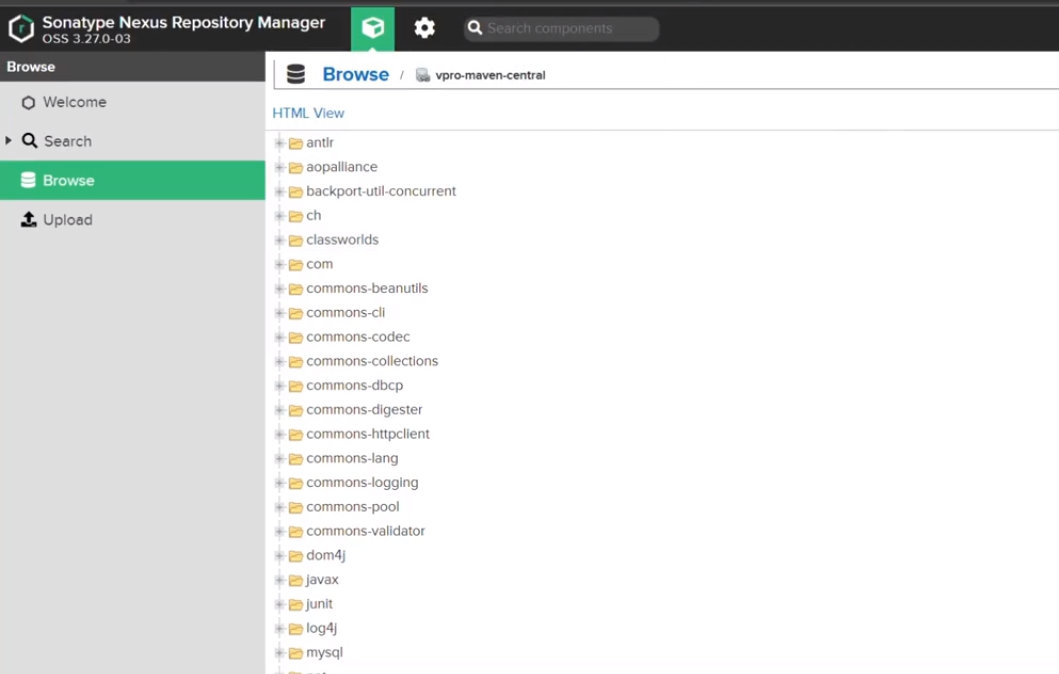
NEXUSPORT=8081



1. Click save and build.

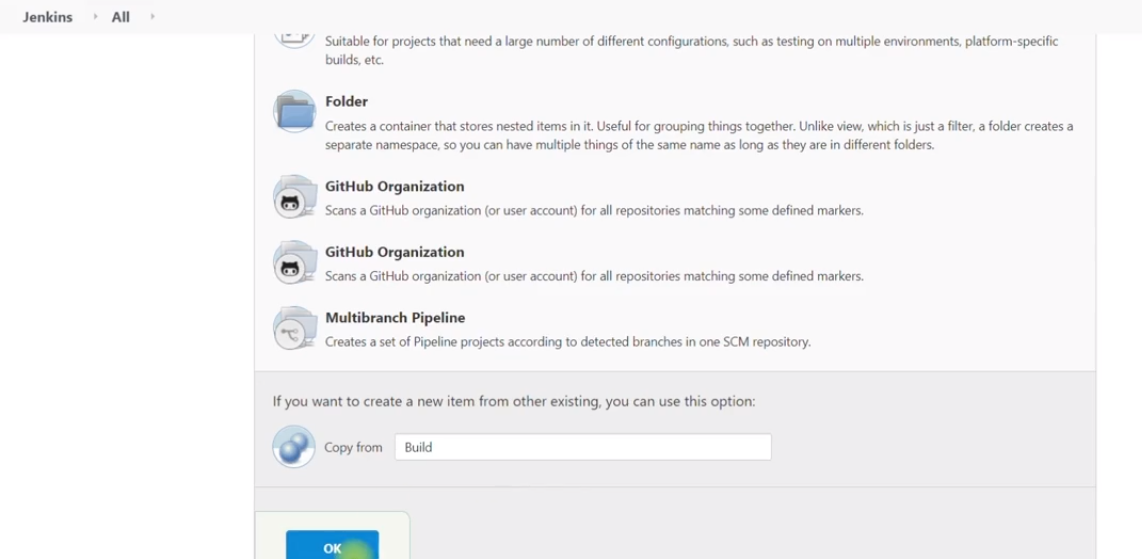
8.If the Build is successful you can see the Build dependencies in the Nexus repository dashboard under maven-central-repo.

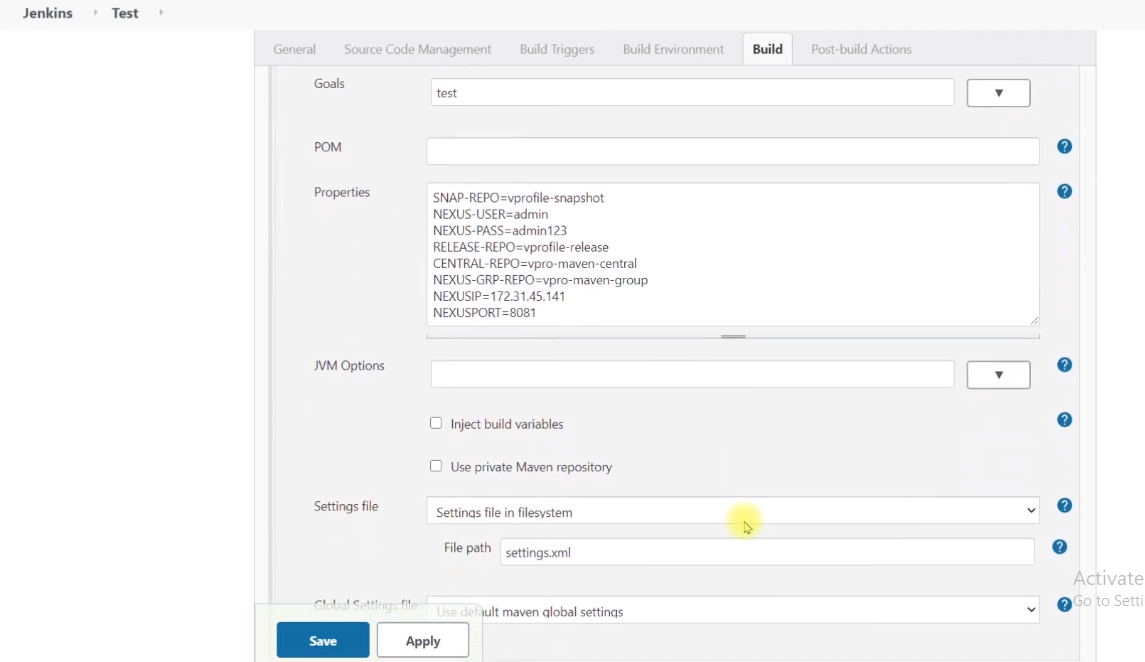
Note: These are installed from the Maven Public repository which we specified the URL for- https://repo1.maven.org/maven2/

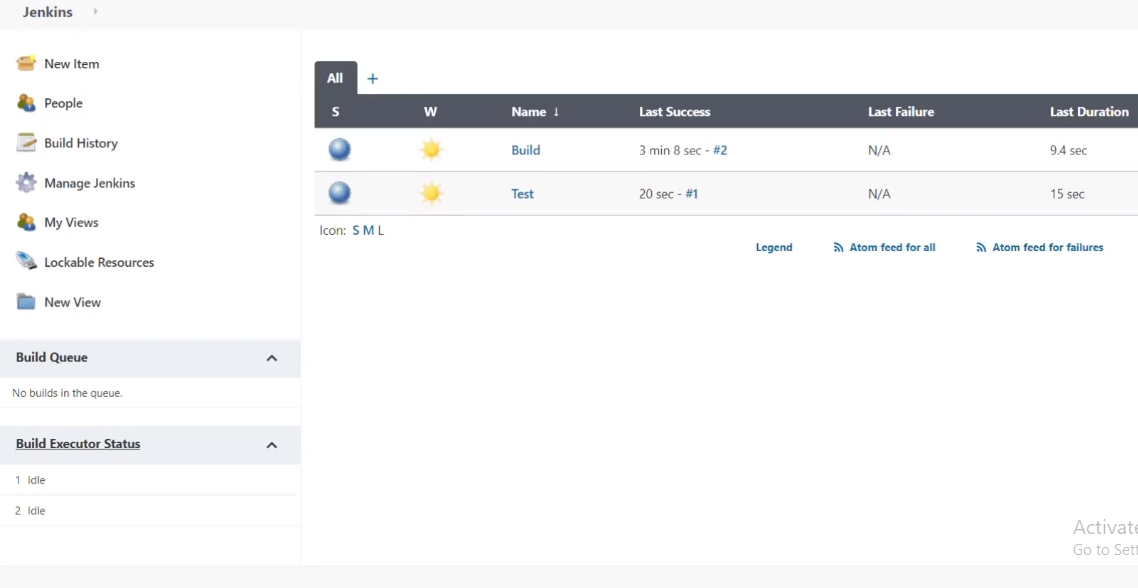


**Create Unit Test Case in Jenkins:**

1. Create a new Free style Project and under copy from section type the pevous build project name as shown below.

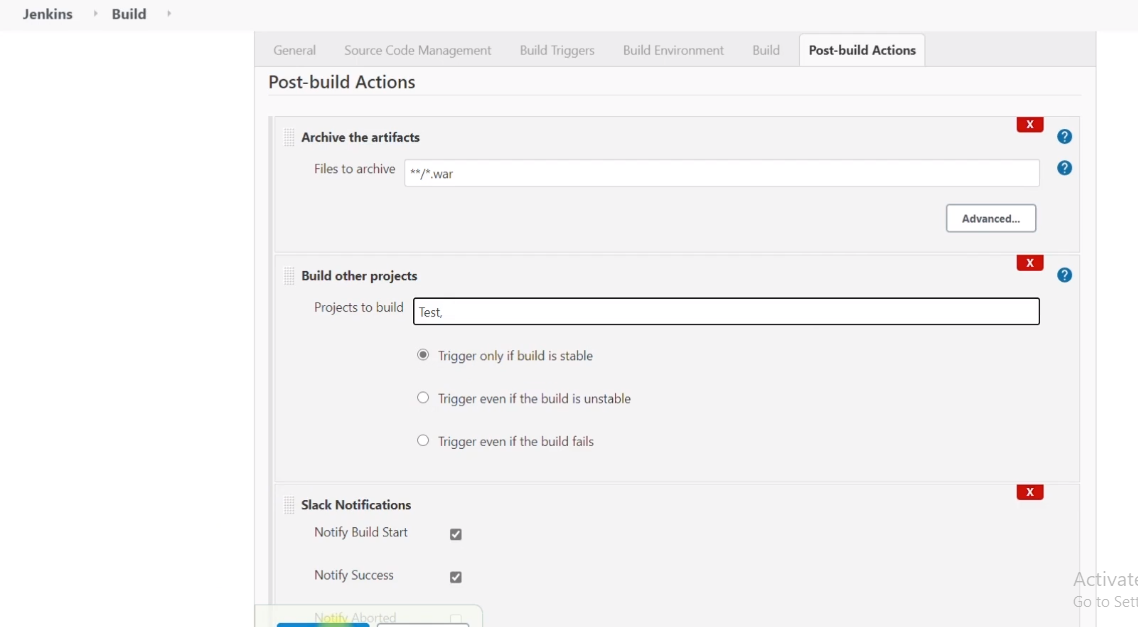


1. Select Invoke top-level maven target, under Goals type: test. Leave the rest of the details as it is.
2. Click save and Build now. The successful build will look like this.



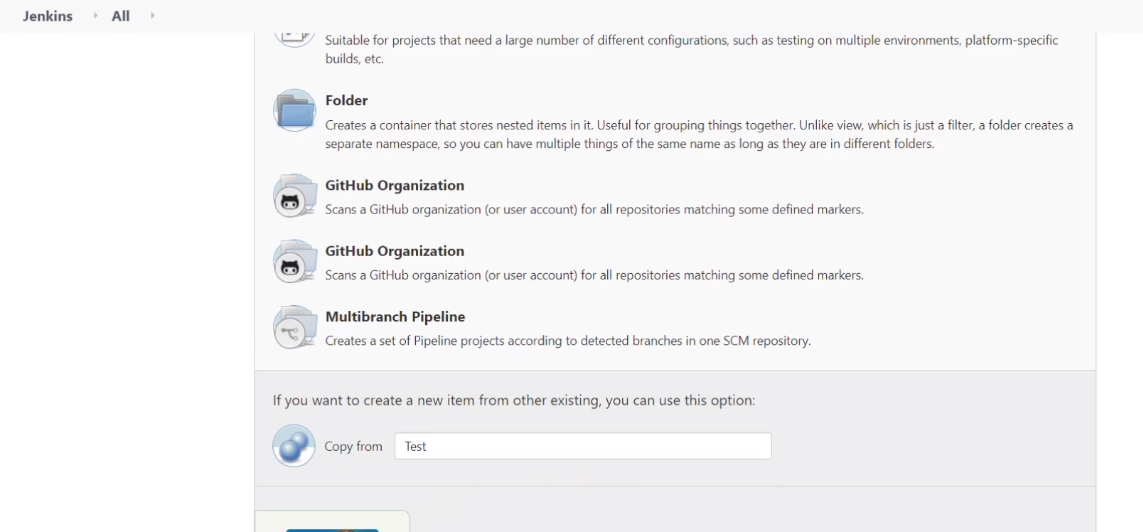
1. To bring a flow to the Jenkins build process from One project to another (i.e: From Build -> Test).

Goto project named “Build” as see in the above fig and click configure. Once Inside scroll down to Add Post-Build action and click Build-other-projects. Add the next project you want to trigger build for post action (i.e “Test” in the above fig). Finally Save the Changes.

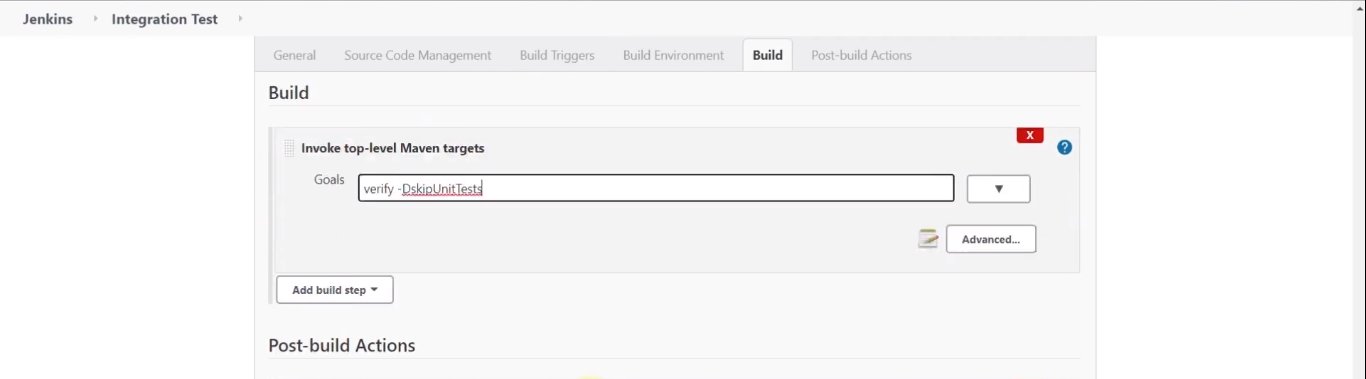


**Create Integration Test case in Jenkins**

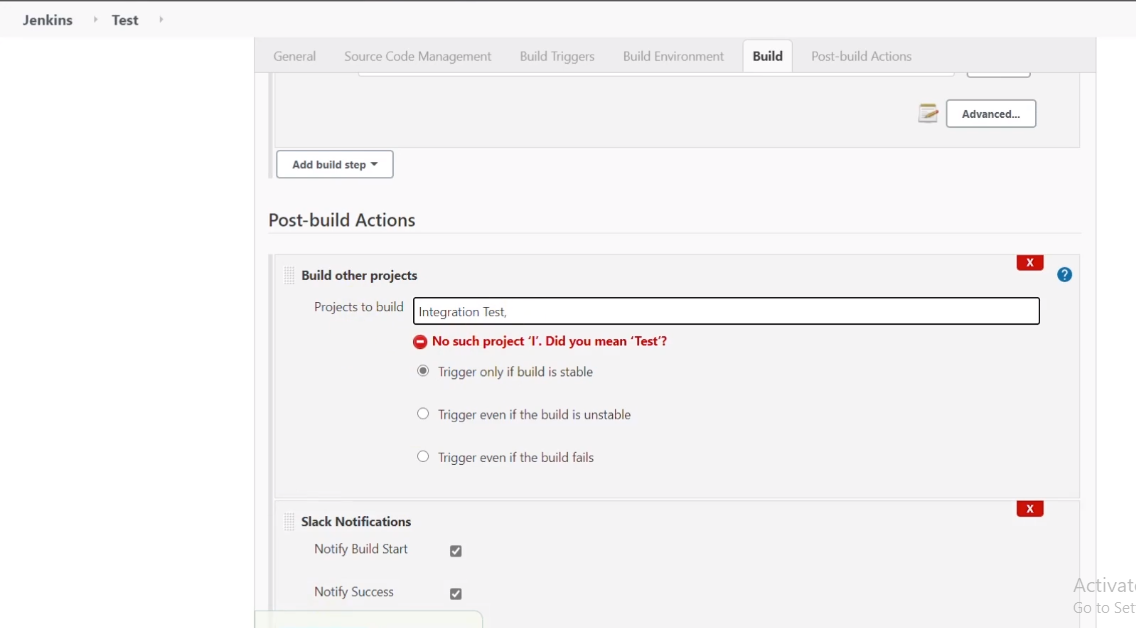
1. Create a New FreeStyle Project and Enter a name of your choice. Under Copy from give the name of the project used for unit testing.



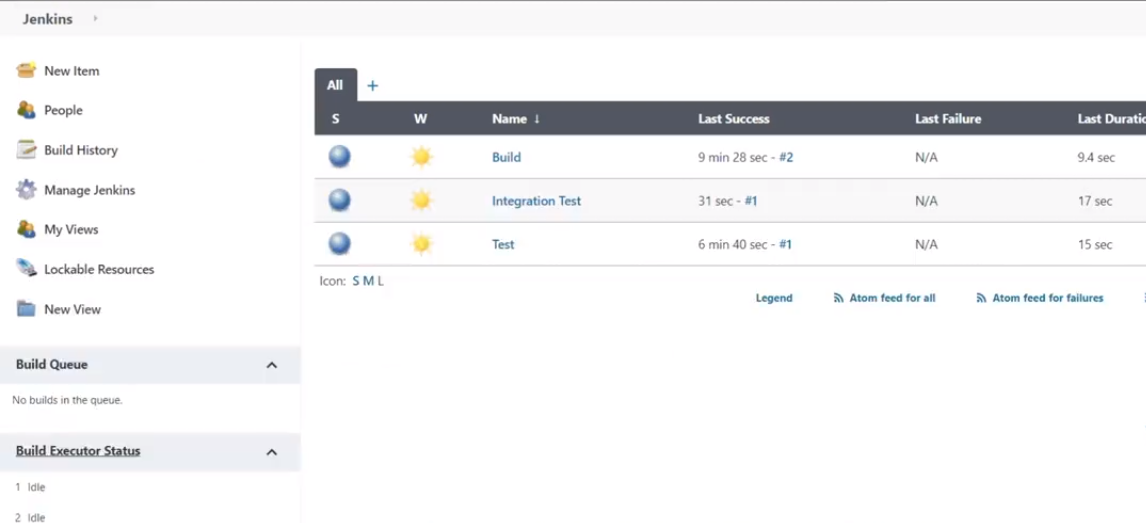
1. Enter the Goals for the Integration test as below. And Click on Save button.



1. Goto configure of the Unit testing project. And Add the Post build action to Integration testing project name as shown in the fig below. Finally Click on Save



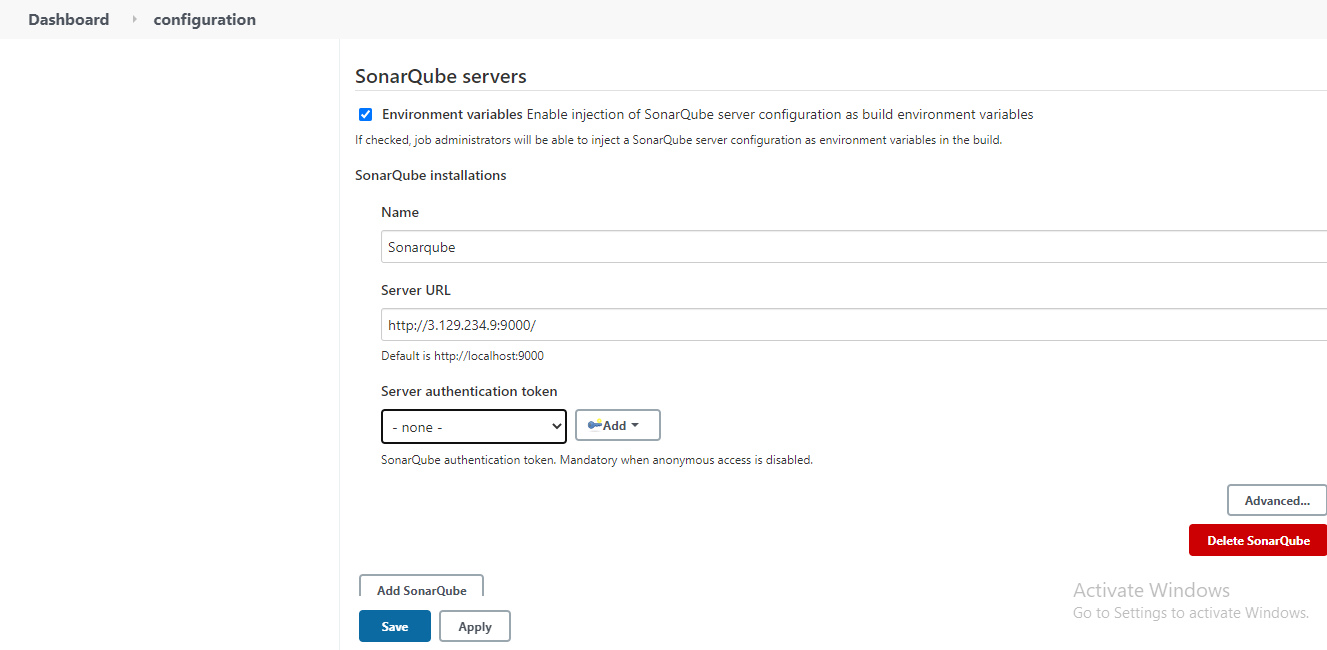
1. Click Build now from 1st Free style project(i.e Build as shown in fig above). You can see the build trigger flow from 1st->2nd ->3rd project. The successful build is displayed as shown below in the fig.



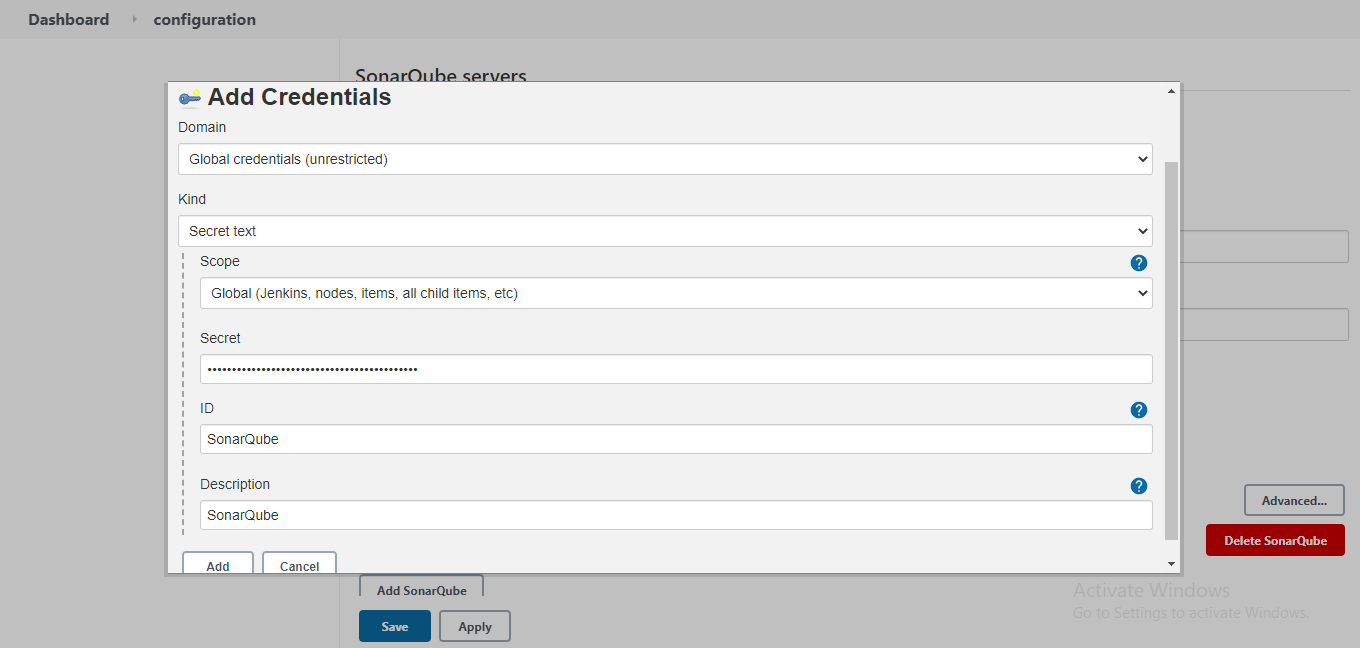
**Create SonarQube code quality check from Jenkins**

1.Goto Manage Jenkins->Configure System->SonarQube Server. Check-in the Environment variables box and click on Add SonarQube. Enter the Name and

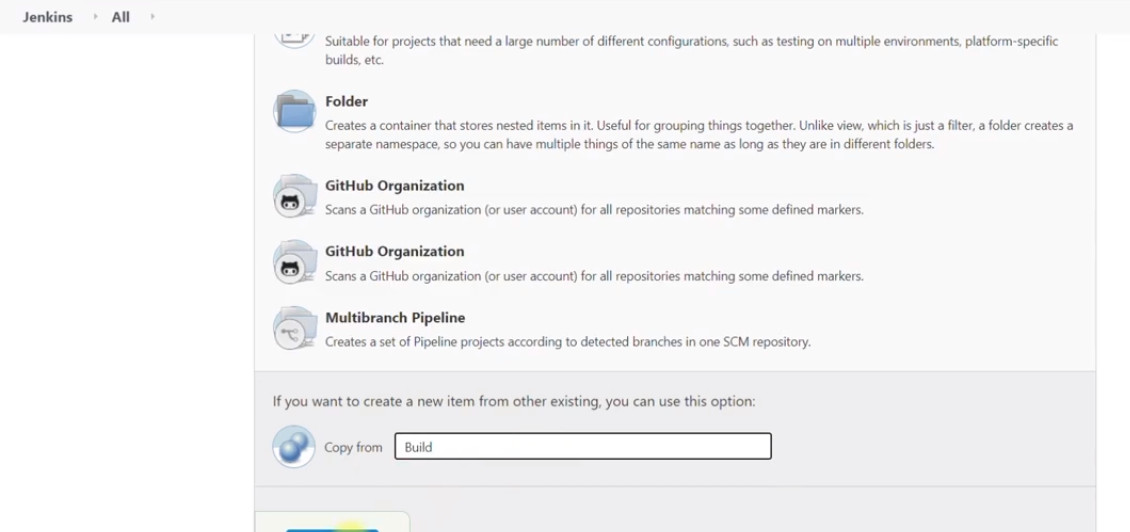
Server URL( Public ipv4 address of Ec2 and port=9000).



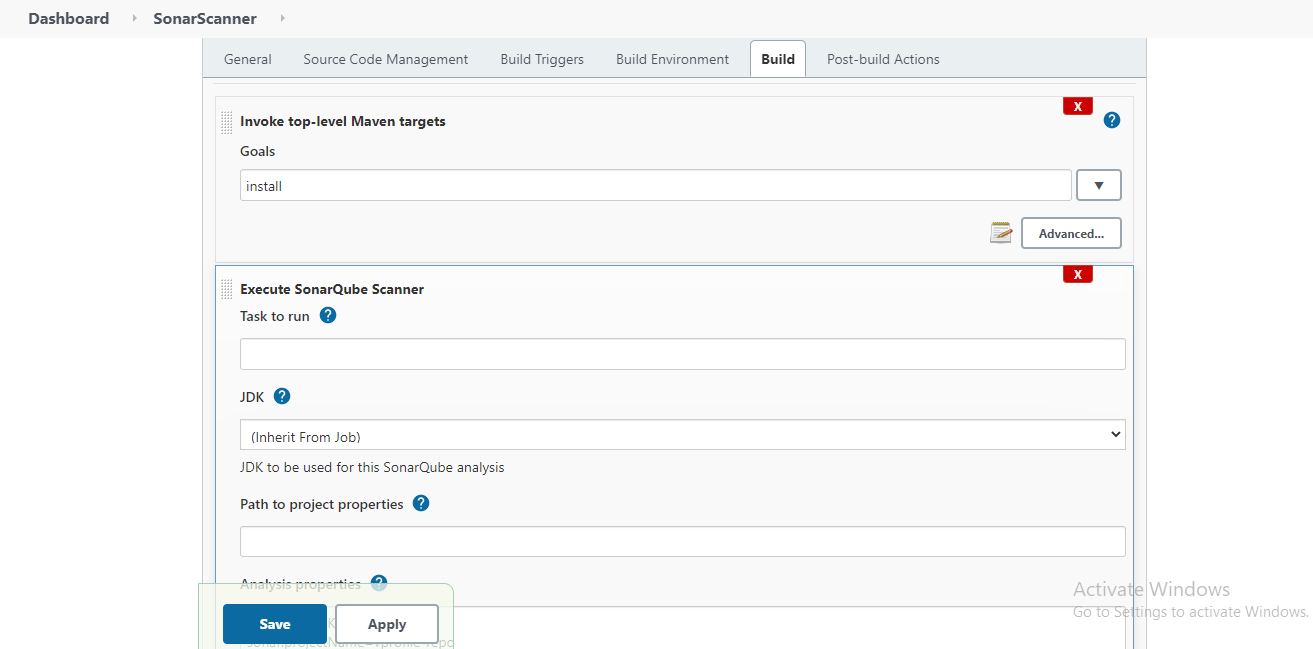
1. Click on the add button below Server authentication token. Select Secret Text from Kind-drop-box. Paste the Previously copied Secret key from SonarQube token-generator. Enter the Id and Description. Finally Click Add and Save buttons respectively.



1. Create a 4th FreeStyle Project and Enter a name for your project. Scroll Down under copy from Enter the 1st Freestyle Project Name (i.e “Build”).



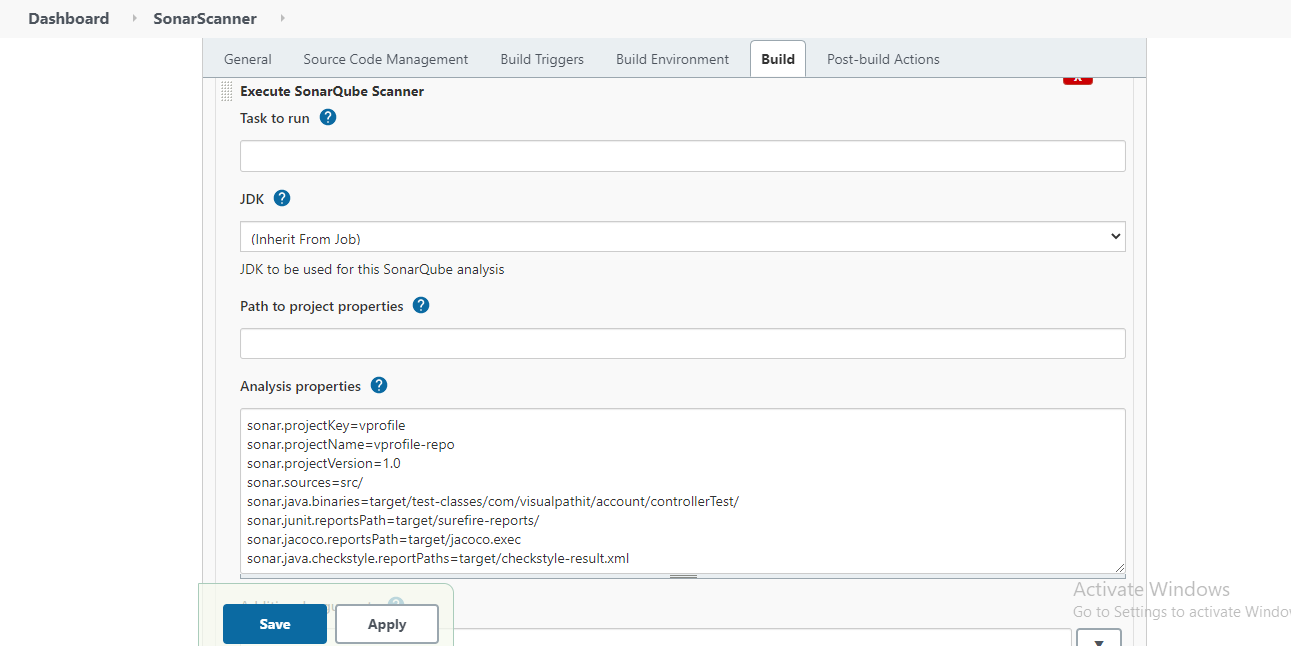
1. Enter the Goal as install under Invoke-toplevel-maven. Click add-build-Step->Execute-SonarQube-Scanner and provide the below details shown in the Fig.



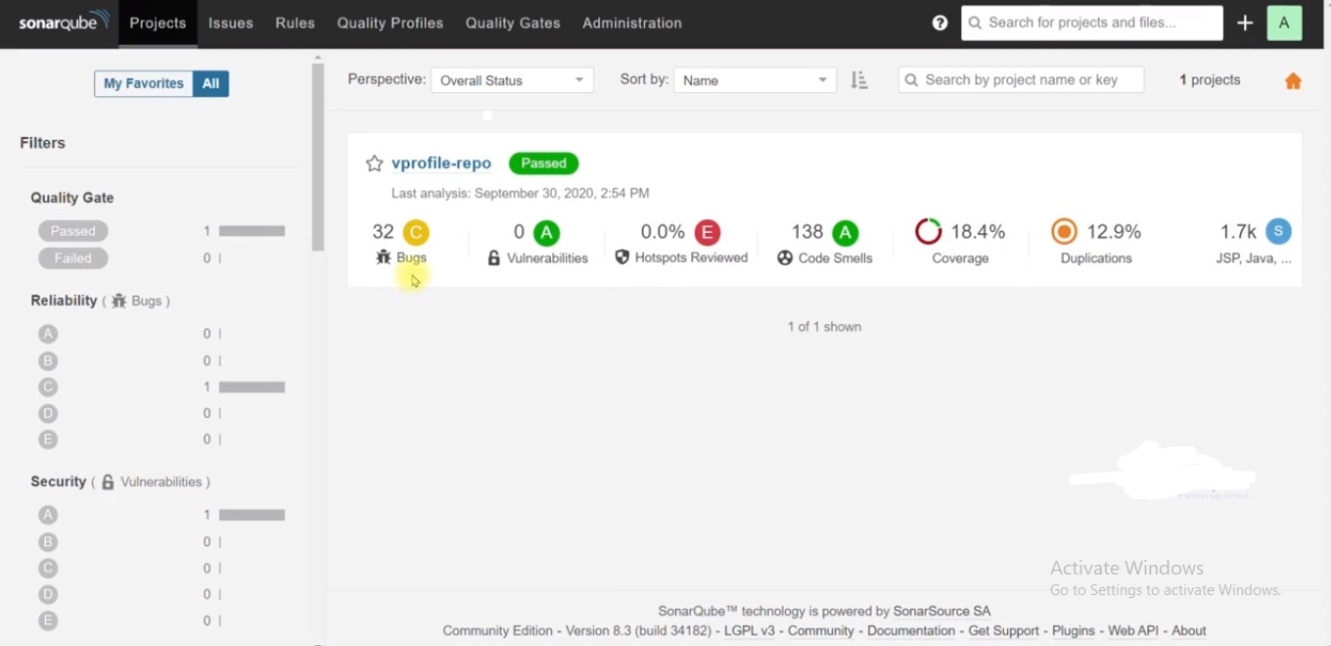
## Paste the commands given in git folder:-[Java-CICD](https://github.com/Roshan05mura/Java-CICD)/[userdata](https://github.com/Roshan05mura/Java-CICD/tree/main/userdata)/****sonar-analysis-properties.****

Click save and Build.

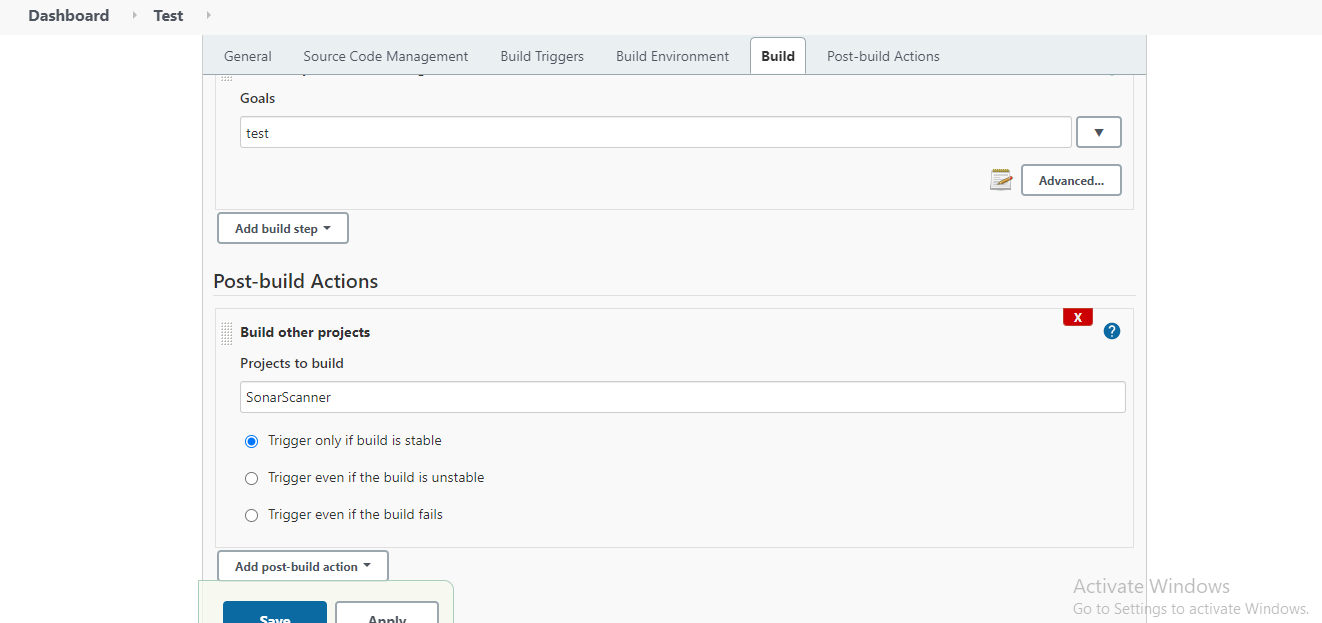
Note: Github link-( https://github.com/Roshan05mura/Java-CICD/blob/main/userdata/sonar-analysis-properties).



1. Sonarqube Dashboard Result would look similar to this.

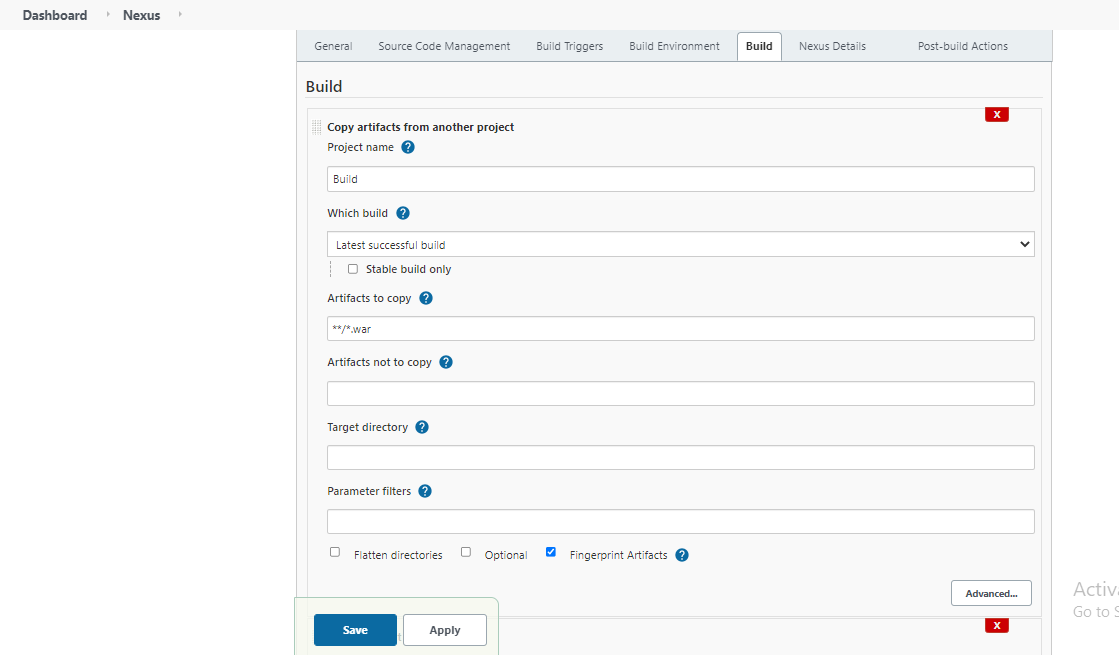


1. Add the Post build Step for the 3rd Freestyle project and point it to the 4th Project.



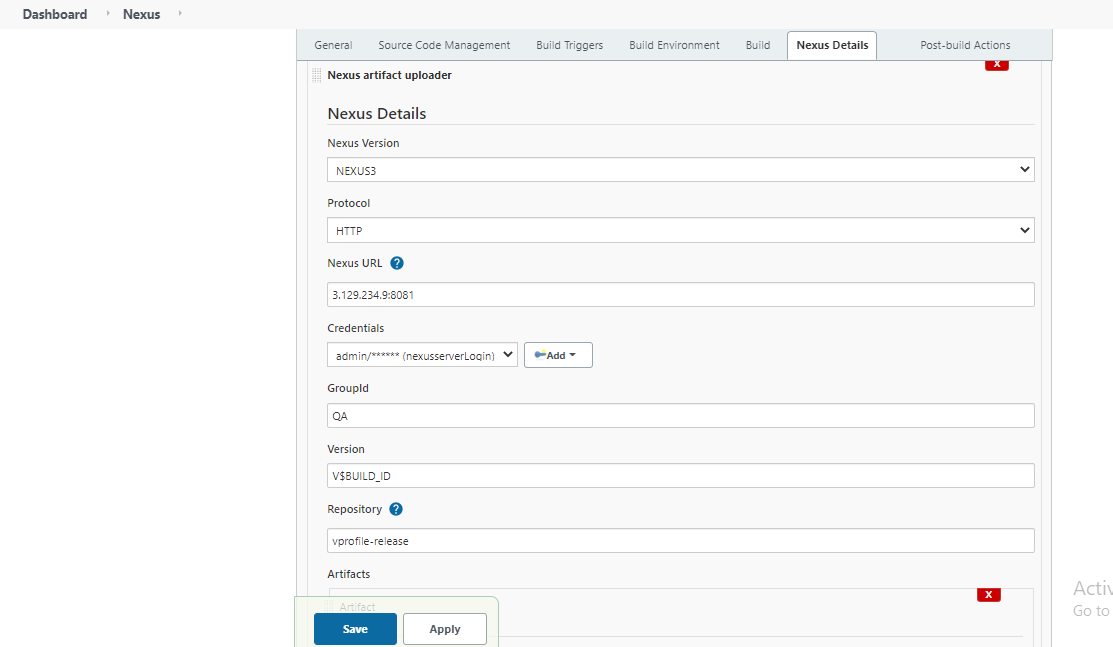
**Project Build-push war file to Nexus Repository from Jenkins.**

1. Create a New Freestyle Project and, Enter the name of your choice.
2. Scroll Down to Build and Click add-build-step. From the Drop down Click copy-artifact-from-another-project and Nexus-artifact-uploader. Enter the details shown in the below Fig.

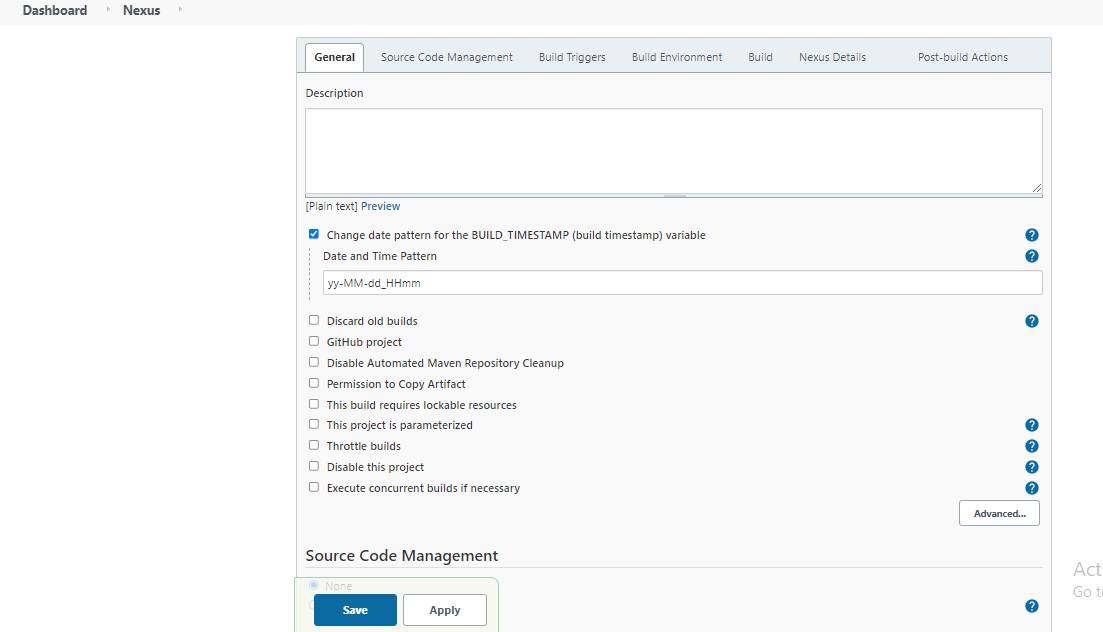
****

1. Under Nexus-artifact-uploader, the URL is the Public ipv4 of Ec2. Click add credentials to input the login credentials for Nexus.

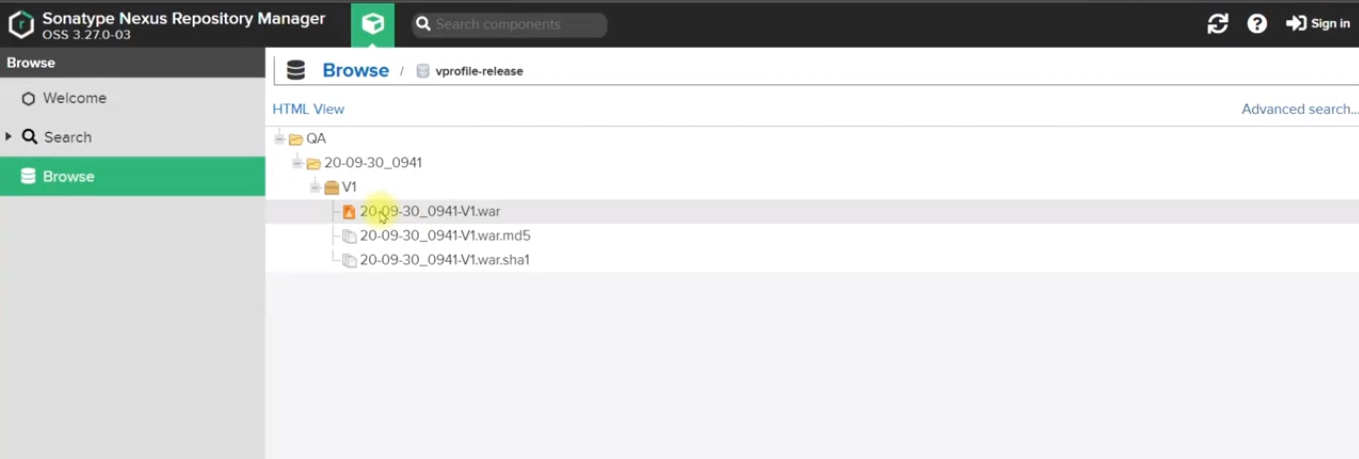
Note:[Repository should be the name of the Nexus Repository where you want to upload the war file to]



1. Checkin the Change-date-pattern-for-the-Build\_time\_stamp. Enter the below shown details in the field. Finally click save. And Press Build- Now for the 5th Freestyle project.

****

1. Go to the Nexus Repository where you wanted to upload the artifact to. The war file we be stored and would look as shown in the Fig below.

****

1. Add the Post build Step for the 4th Freestyle project and point it to the 5th Project. Click save. When you Press Build-Now from project 1, The Jenkins build trigger flow would be from 1st->2nd->3rd->4th->5th project.

