Install JAVA ON MASTER

sudo amazon-linux-extras install java-openjdk11 -y
sudo update-alternatives --config java
vim ~/.bash_profile
JAVA_HOME="/usr/lib/jvm/java-11-openjdk-11.0.11.0.9-1.amzn2.0.1.x86_64/bin/java"
source ~/.bash_profile
echo $JAVA_HOME

Install Jenkins on MASTER

sudo wget -O /etc/yum.repos.d/jenkins.repo \
https://pkg.jenkins.io/redhat-stable/jenkins.repo

sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
sudo yum upgrade -y
sudo yum install jenkins java-11-openjdk-devel -y
sudo systemctl daemon-reload

sudo service jenkins start
sudo service jenkins status

---

sudo systemctl status Jenkins
sudo systemctl start jenkins.service

OPTIONAL
sudo yum install httpd -y **(global config error path error maven http error 403)** **(ERROR1)**

Enter the public ip of the master Jenkins server

Public_ip_of_master:8080
Get the initial admin password of Jenkins
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
Install suggested plugins

Create a jenkins user named master from jenkins dashboard

Install git and maven on master & slave

sudo yum install git -y
sudo yum install maven -y

Repo clone and pushed to github

https://github.com/laxapatiakshaylearning/cicdpipelinejenkins

Setup master slave

Slave setup on master
Manage jenkins → Manage node & clouds → New Node → give node name linux_node1 → add to permanent

Remote root directory
/home/ec2-user/

Tick on (use websocket) & save

Click on the slave node & right click on agent.jar then copy link address

On slave machine
Download the agent.jar file at /home/ec2-user using wget
wget http://54.198.69.61:8080/jnlpJars/agent.jar

Join from slave using below command
java -jar agent.jar -jnlpUrl http://54.198.69.61:8080/computer/linux_slave1/jenkins-agent.jnlp -secret aa224035b5f55c28c926847bcf9bf60d4cf2dcf75b53be7fb350581a0413b874 -workDir "/home/ec2-user/"

java -jar agent.jar -jnlpUrl http://54.147.245.0:8080/computer/linux_node1/jenkins-agent.jnlp -secret a4d1e2ae0a49dc202811fe9b1a34accf06247077e31b64278306ba562ba1624e -workDir "/home/ec2-user/"

Check if the node/slave is connected

Configure global config
Manage jenkins → Global tool configuration

Jdk
Add jdk
Name: java
Untick install automatically
Java_home =/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.302.b08-0.amzn2.0.1.x86_64

/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.302.b08-0.amzn2.0.1.x86_64

Git

To find the path of git below is the command
Whereis git
/usr/bin/git **(ERROR1 optional)**

Maven add maven
Name : maven
Untick install automatically

To find the path of maven below is the command
Mvn -v
/usr/share/maven

Name : maven
Path : /usr/share/maven

Click Save

SETUP SONAR ON NEW SERVER

Launch at least instance type t2.medium because the minimum ram required for sonar is 4 gb

Install java

sudo amazon-linux-extras install java-openjdk11 -y

Check java path

sudo update-alternatives --config java

/usr/lib/jvm/java-11-openjdk-11.0.11.0.9-1.amzn2.0.1.x86_64/bin/java

Set environment variable

vim ~/.bash_profile
JAVA_HOME="/usr/lib/jvm/java-11-openjdk-11.0.11.0.9-1.amzn2.0.1.x86_64/bin/java"
source ~/.bash_profile

Install sonarqube

download binary of sonar

wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-8.1.0.31237.zip

unzip sonarqube-8.1.0.31237.zip

cd sonarqube-8.1.0.31237/bin/linux-x86-64/
./sonar.sh console

Login using admin as username & password
http://50.17.69.175:9000/ (public ip of sonar server : 9000)

Managejenkins → manage plugins → available plugins → search plugin

SonarQube Scanner

& install without restart

Manage Jenkins → Configure system

SonarQube servers

Tick

Environment variables Enable injection of SonarQube server configuration as build environment variables

Click Add sonarqube
Name : sonarqube
http://publicip_of_sonarqube:9000
http://34.203.248.145:9000

Add credentials for sonarqube
Server authentication token
Add → jenkins → secret text in kind → Secret -- paste the token that was copied from sonar server → Description : sonar token →
click on none & select sonar token from drop down menu

Tools Path configuration
Manage jenkins → manage nodes & clouds → click on settings of node → tools location add →

GIT
To find the path of git below is the command
Whereis git (Execute this command on slave node)
/usr/bin/git

JAVA
To find the path of git below is the command
update alternatives command (Execute this command on slave node)
/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.302.b08-0.amzn2.0.1.x86_64

MAVEN
To find the path of maven below is the command
mvn -v (Execute this command on slave node)
/usr/share/maven

CREATE A PIPELINE PROJECT
Name for the project

Pipeline script

```
pipeline{
    agent  any

    tools{
        git 'git'
        maven 'maven'
        jdk 'java'
    }

    stages{
        stage('checkout'){
            steps{
            git 'https://github.com/laxapatiakshaylearning/cicdpipelinejenkins.git'
            echo 'inside checkout'
            }
        }

        stage('build'){
            steps{
                sh 'mvn clean install -f pom.xml'
                echo 'inside build'
            }
        }
        stage('Code Quality'){
            steps{
                withSonarQubeEnv('sonarqube'){
                    sh 'mvn -f pom.xml sonar:sonar'
                }

            }

        }
        stage('Sonarqube') {


  steps {
    withSonarQubeEnv('sonarqube') {
        sh "/home/ec2-user/sonar-scanner-4.6.2.2472-linux/bin/sonar-scanner"
        echo 'inside sonar scanner'
```

```
        }
        timeout(time: 10, unit: 'MINUTES') {
            waitForQualityGate abortPipeline: true
            echo 'inside sonar environment'
                }
            }
        }
        stage('docker stage'){
            steps{
            echo 'inside docker stage'
            }
        }

        }
}



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
pipeline{
        agent  any

        tools{
            git 'git'
            maven 'maven'
            jdk 'java'
        }

        stages{
            stage('checkout'){
                steps{
                git 'https://github.com/laxapatiakshaylearning/cicdpipelinejenkins.git'
                echo 'inside checkout'
                }
            }

            stage('build'){
                steps{
                    sh 'mvn clean install -f pom.xml'
                    echo 'inside build'
                }
            }
            stage('Code Quality'){
```

```
            steps{
                withSonarQubeEnv('sonarqube'){
                    sh 'mvn -f pom.xml sonar:sonar'
                }

            }

        }
        stage('Sonarqube') {
    environment {
        scannerHome = tool 'sonarscanner'

    }

    steps {
            withSonarQubeEnv(installationName: 'sonarqube',credentialsId:
'519a31de-51c8-4cb6-9803-b0684513d2d2') {
            sh '''  $scannerHome/bin/sonar-scanner \
                        -Dsonar.passsword=admin \
                        -Dsonar.projectKey=pipeline-1 \
                -Dsonar.java.binaries=/var/lib/jenkins/workspace/pipeline1/target \
                -Dsonar.host.url=http://54.227.123.185:9000/ \
                -Dsonar.sources=/var/lib/jenkins/workspace/pipeline1/src '''
                echo 'inside sonar scanner properties'
            }
            timeout(time: 10, unit: 'MINUTES') {
            waitForQualityGate abortPipeline: true
            echo 'inside sonar environment'
            }
        }
    }
        stage('docker stage'){
            steps{
            echo 'inside docker stage'
            }
        }

        }
}
```

%%

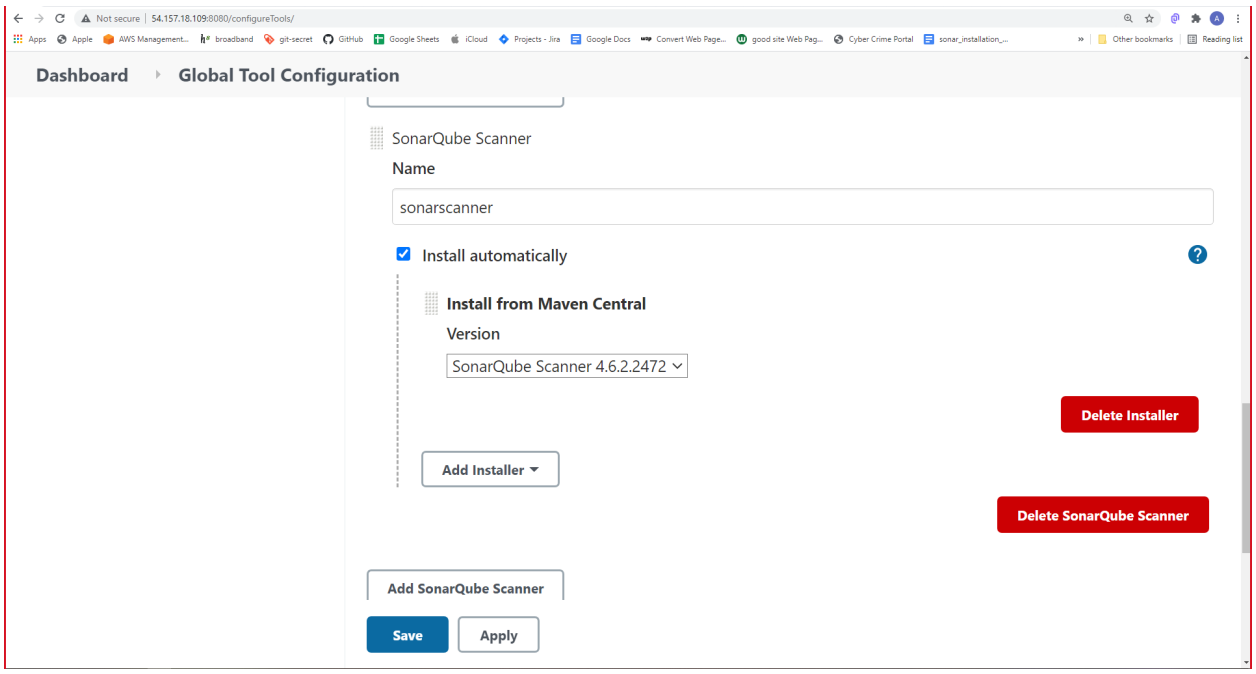
Get the syntax by using the option pipeline syntax & use the same in the above code

Apply save

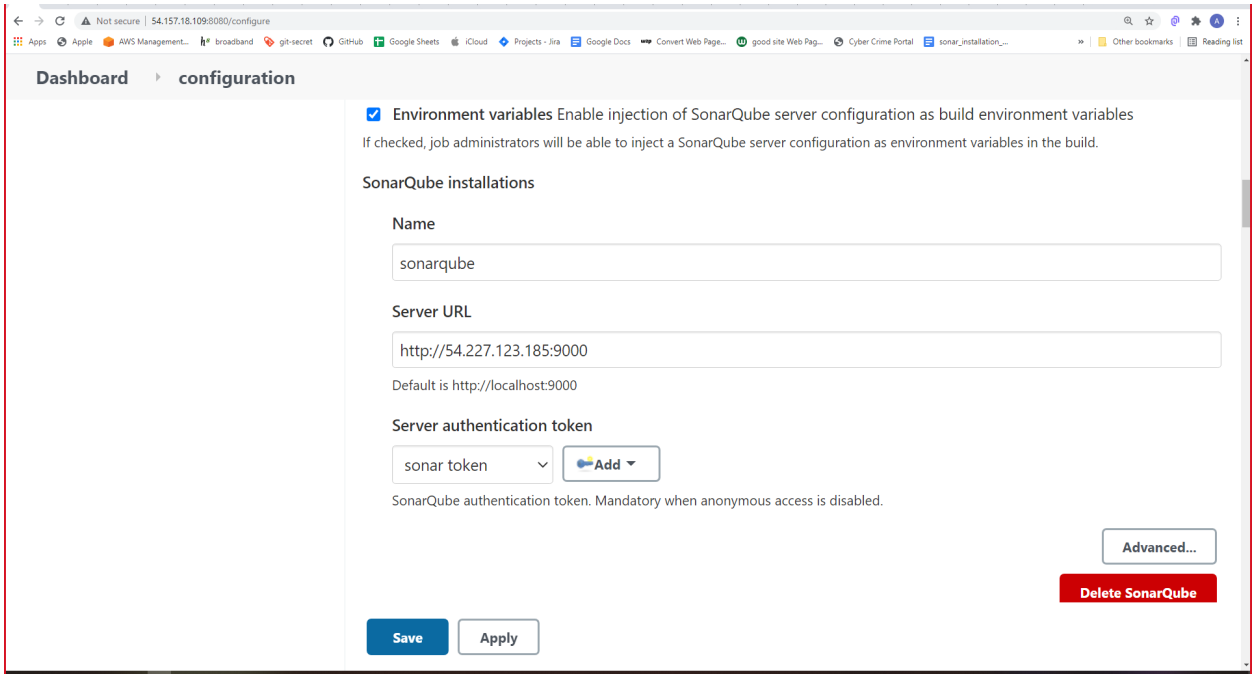
OPTIONAL : note if we want to use manual sonar scanner
We have to add sonar-project.properties in git repository

```
# this file is not used now
# must be unique in a given SonarQube instance
#sonar.projectKey=Project1
#sonar.projectVersion=1.0
#sonar.projectName=CustomHelloApp
#sonar.sources=.
#sonar.java.binaries=./target
#sonar.host.url=http://54.227.123.185:9000
#sonar.host.url=http://172.27.12.78/
```

Sonar scanner binary
https://binaries.sonarsource.com/Distribution/sonar-scanner-cli/sonar-scanner-cli-4.6.2.2472-linux.zip



sonarscanner


Not secure | 54.157.18.109:8080/configure
Dashboard
configuration
Environment variables Enable injection of SonarQube server configuration as build environment variables
If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.
SonarQube installations
Name
sonarqube
Server URL
http://54.227.123.185:9000
Default is http://localhost:9000
Server authentication token
sonar token
Add
SonarQube authentication token. Mandatory when anonymous access is disabled.
Advanced...
Delete SonarQube
Save
Apply

sonarqube