****

Jenkins-Server Configuration

****

DevOps project setup with CI CD Pipeline:

~~~~~~

Pre-requisites:

----

1) GitHub

(Repo url:https://github.com/Pankaj1006/Devops-project-in-AWS.git**)**

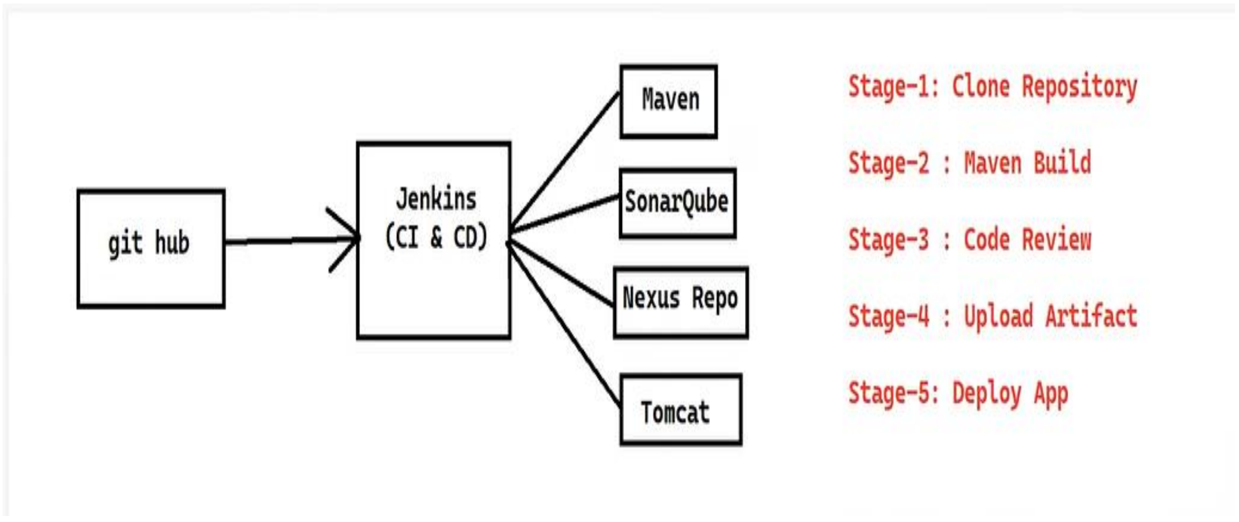
2) Tomcat (url:http://public-ip:8080 **)**

3) Nexus Repo (url:http://public-ip:8081 **)**

4) SonarQube (url:http://public-ip:9000 **)**

5) Jenkins Server (url:http://public-ip:8080 **)**

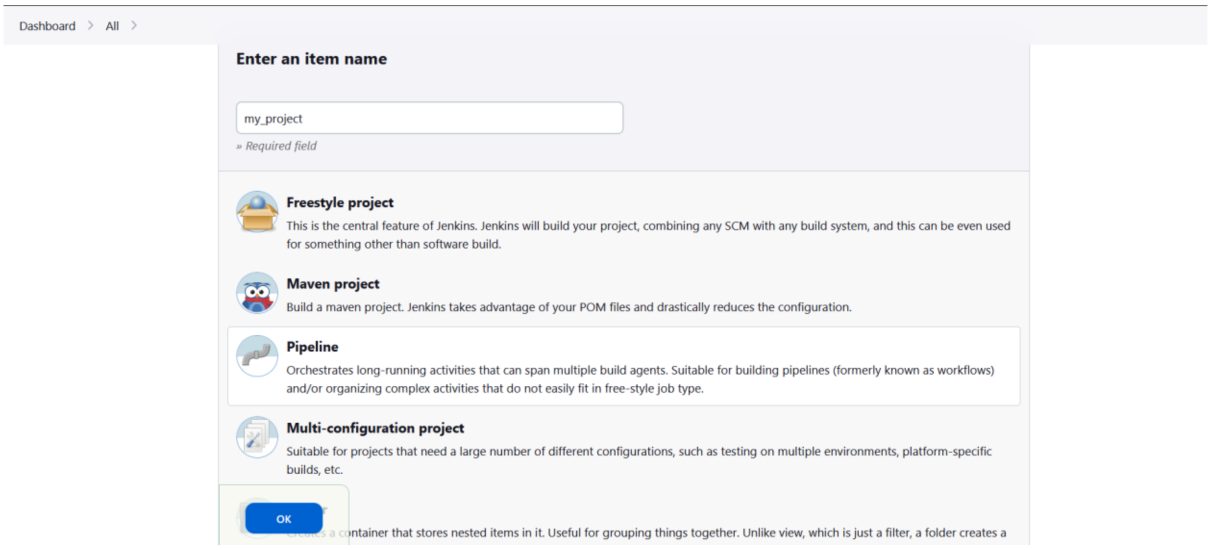
6) Maven (as a global tool in Jenkins**)**



**Note: ***The installation process of Pre-requisites are present in a github repository link.
~~~~~~

Start from the Jenkins server:

From the Dashboard + New item


Dashboard > All >
Enter an item name
my_project
» Required field
Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
OK
container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a

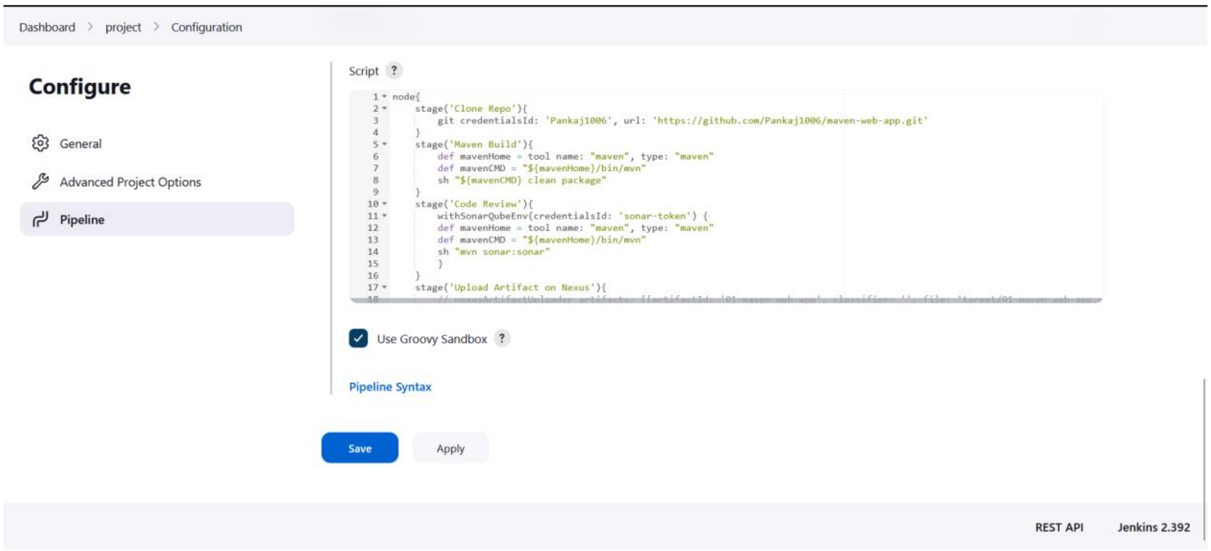
Enter an item name<project_name>

Select Pipeline

OK

Now from Jenkins dashboard select the <project_name> after that select configure, click on pipeline

Here you can write your pipeline script.



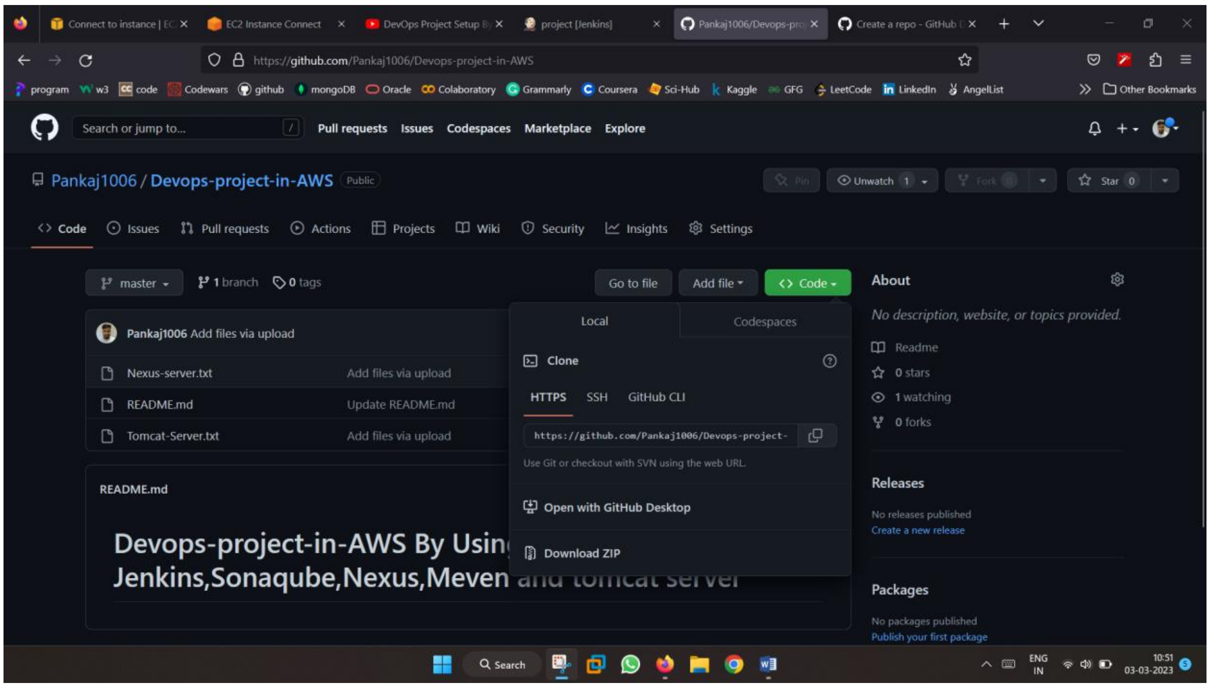
```
node{
    stage('Clone Repo'){
        git credentialsId: 'Pankaj1006', url: 'https://github.com/Pankaj1006/maven-web-app.git'
    }
    stage('Maven Build'){
        def mavenHome = tool name: "maven", type: "maven"
        def mavenCMD = "${mavenHome}/bin/mvn"
        sh "${mavenCMD} clean package"
    }
    stage('Code Review'){
        withSonarQubeEnv(credentialsId: 'sonar-token') {
        def mavenHome = tool name: "maven", type: "maven"
        def mavenCMD = "${mavenHome}/bin/mvn"
        sh "mvn sonar:sonar"
        }
    }
    stage('Upload Artifact on Nexus'){
        // nexusArtifactUploader artifacts: [[artifactId: '01-maven-web-app', classifier: '', file: 'target/01-maven-web-app
```

*Script start from node or pipeline.

Stage-1 : Clone Repository

First log-in with your Github account, then create new repository for the project.

Now, copy the url of the repository.



Pankaj1006 Add files via upload

Nexus-server.txt	Add files via upload
README.md	Update README.md
Tomcat-Server.txt	Add files via upload

https://github.com/Pankaj1006/Devops-project-

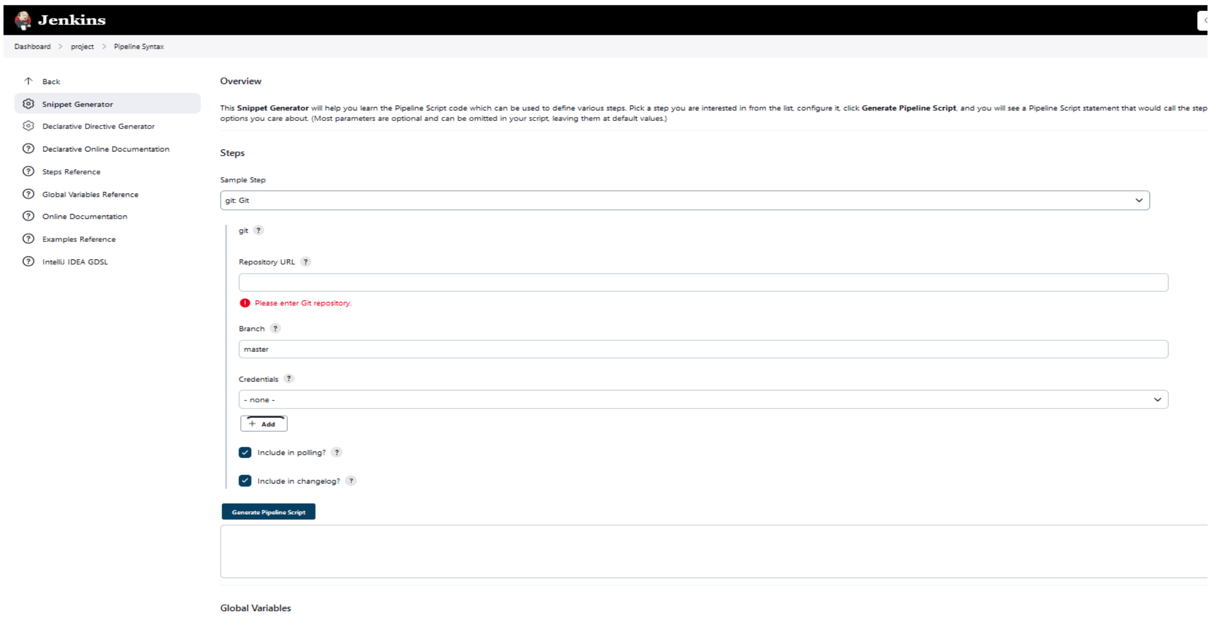
Use Git or checkout with SVN using the web URL.

README.md

Devops-project-in-AWS By Using Jenkins,Sonaqube,Nexus,Meven and tomcat server

No description, website, or topics provided.

To configure(clone) github repository you need to click **pipeline syntax** option given below on Jenkins pipeline script.


Jenkins
Dashboard > project > Pipeline Syntax
Back
Snippet Generator
Declarative Directive Generator
Declarative Online Documentation
Steps Reference
Global Variables Reference
Online Documentation
Examples Reference
IntelliJ IDEA GDSL
Add
Generate Pipeline Script

Overview

This **Snippet Generator** will help you learn the Pipeline Script code which can be used to define various steps. Pick a step you are interested in from the list, configure it, click **Generate Pipeline Script**, and you will see a Pipeline Script statement that would call the step options you care about. (Most parameters are optional and can be omitted in your script, leaving them at default values.)

Steps

Sample Step: git: Git

git

Repository URL:

Please enter Git repository.

Branch: master

Credentials: - none -

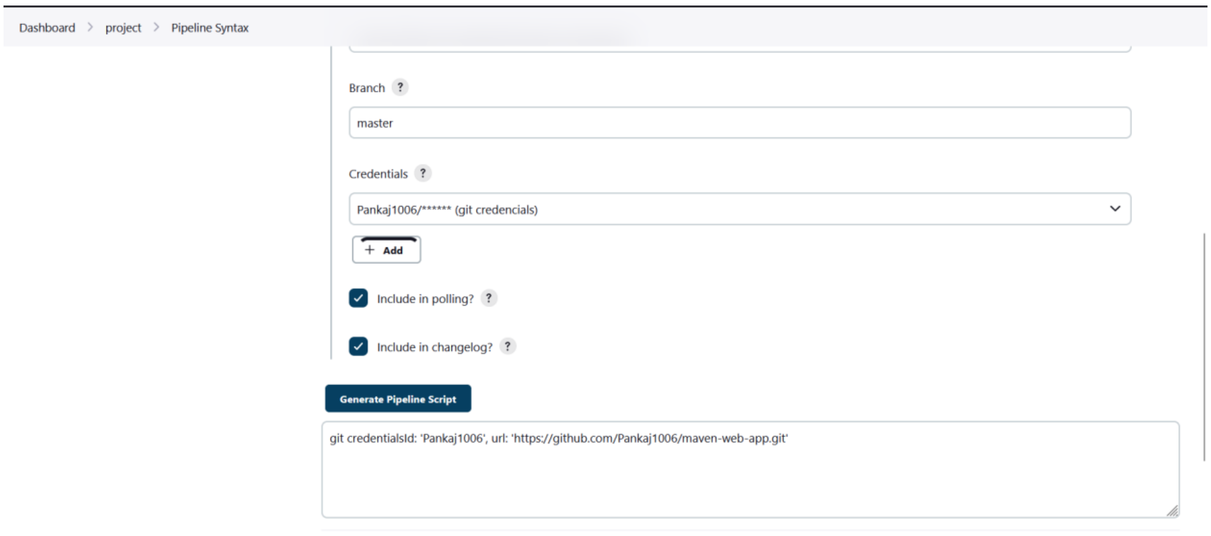
✓ Include in polling?

✓ Include in changelog?

Global Variables

Select Steps as Git, paste github repositotry url Then click on the add credentials give username and password of your github account.

Now Generate Pipeline Script



Branch: master

Credentials: Pankaj1006/****** (git credencials)

✓ Include in polling?

✓ Include in changelog?

```
git credentialsId: 'Pankaj1006', url: 'https://github.com/Pankaj1006/maven-web-app.git'
```

Copy the generated script and paste in pipeline script.

As shown below:

```
node{
    stage('Clone Repo'){
        git credentialsId: 'Pankaj1006', url:
'https://github.com/Pankaj1006/maven-web-app.git'
    }
```

Apply & SAVE

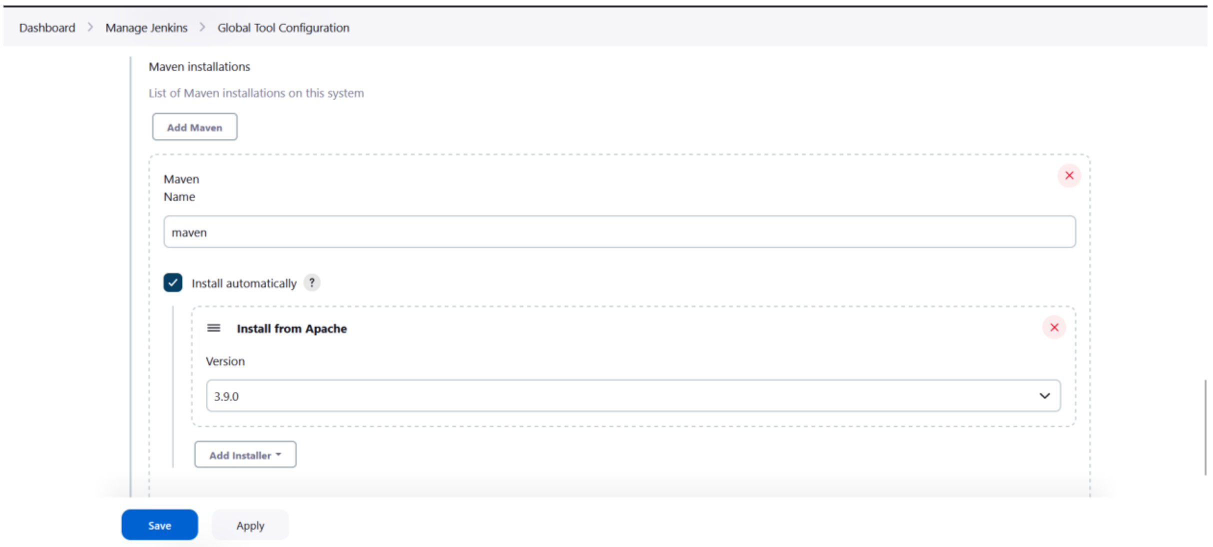
Stage-2 : Maven Build

To setup a Maven as a global tool Go to the Jenkins Dashboard

Click on manage Jenkins here you have option called Global tool configuration.

Scroll down then you see maven installations

Enter name of the maven after that select required version of maven.



Maven installations

List of Maven installations on this system

Maven Name: maven

Install automatically: ✓

Install from Apache

Version: 3.9.0

Apply & SAVE

Now from the pipeline script at second stage we need to tell to the Jenkins that maven has configure as a global tool.

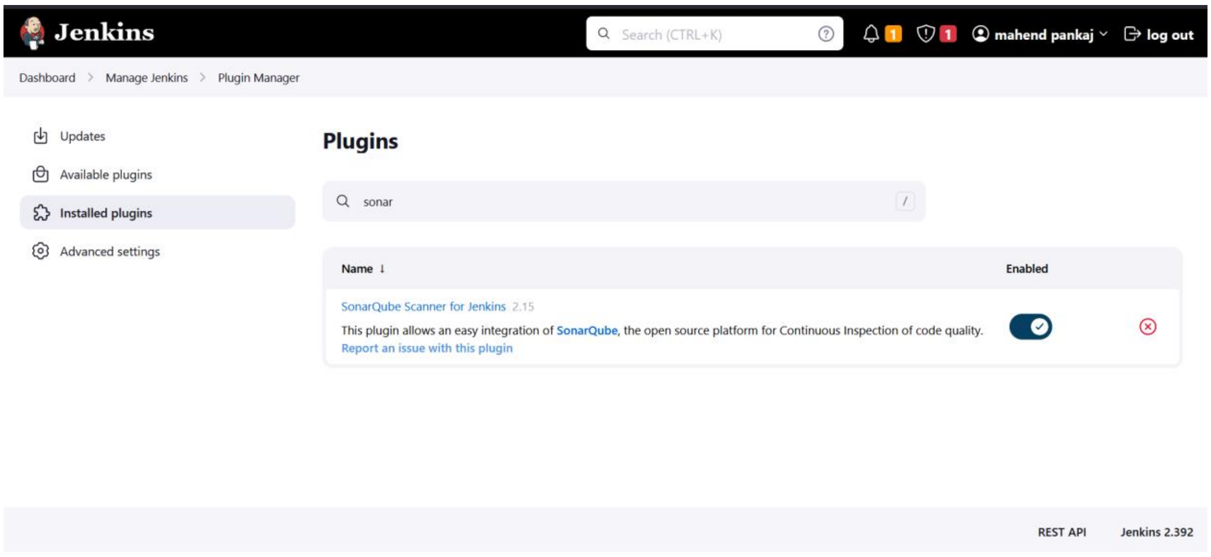
```
stage('Maven Build'){
   def mavenHome = tool name: "maven", type: "maven"
   def mavenCMD = "${mavenHome}/bin/mvn"
   sh "${mavenCMD} clean package"
}
```

Apply & SAVE

Stage-3 : Code Review (By using SonarQube)

In order to integrate Jenkins with sonar you have to configure plugin called “**Sonar Qube Scanner Plugin**”.

Go to mange Jenkins →manage plugin → Available plugin→Install.



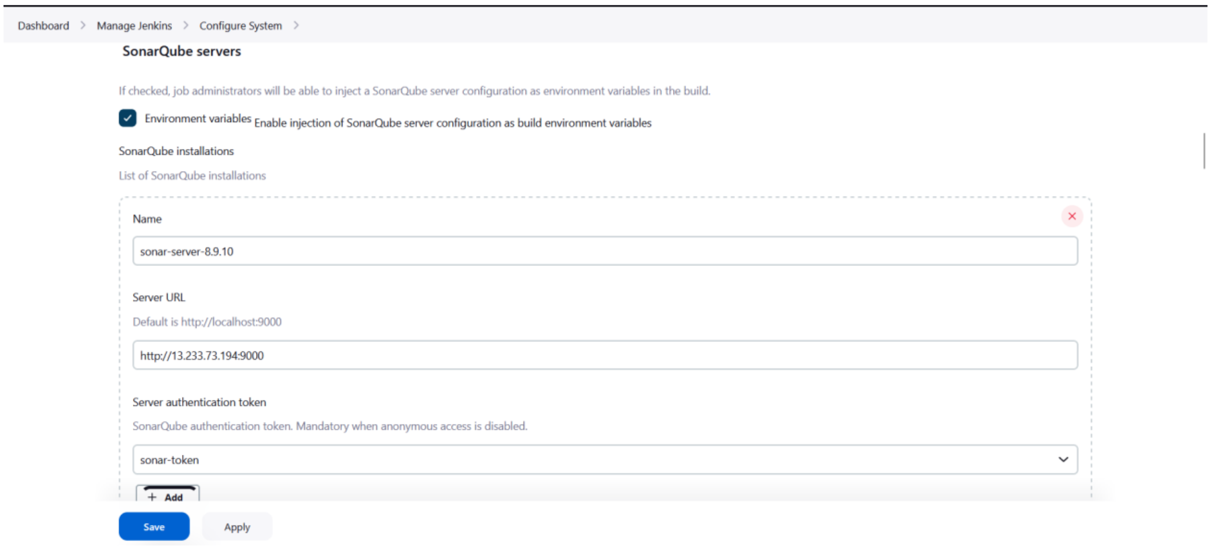
Plugins

sonar

Name ↓	Enabled
SonarQube Scanner for Jenkins 2.15 This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality. Report an issue with this plugin	✓

Now we need to configure sonar with our Jenkins. for that

Manage Jenkins→configure systems→Sonarqube Server(Add sonarqube)



SonarQube servers

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

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

SonarQube installations

List of SonarQube installations

Name: sonar-server-8.9.10

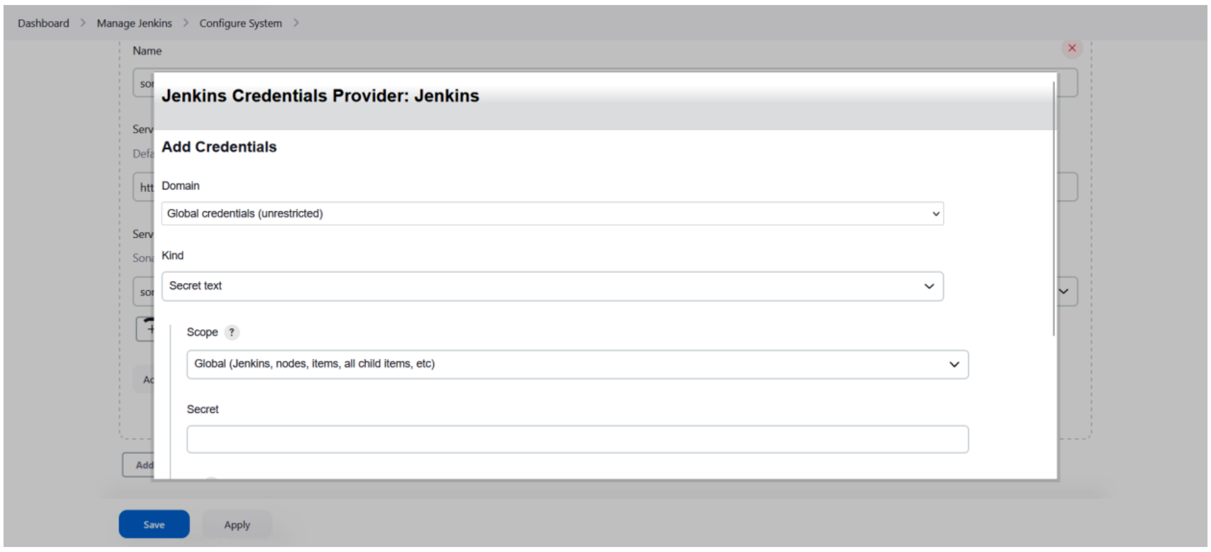
Server URL
Default is http://localhost:9000
http://13.233.73.194:9000

Server authentication token
SonarQube authentication token. Mandatory when anonymous access is disabled.
sonar-token

You can give any name <sonar_server>

Copy url of the sonar server from the browser and paste in server url.

Now for Server authentication token click add.



Jenkins Credentials Provider: Jenkins

Add Credentials

Domain: Global credentials (unrestricted)

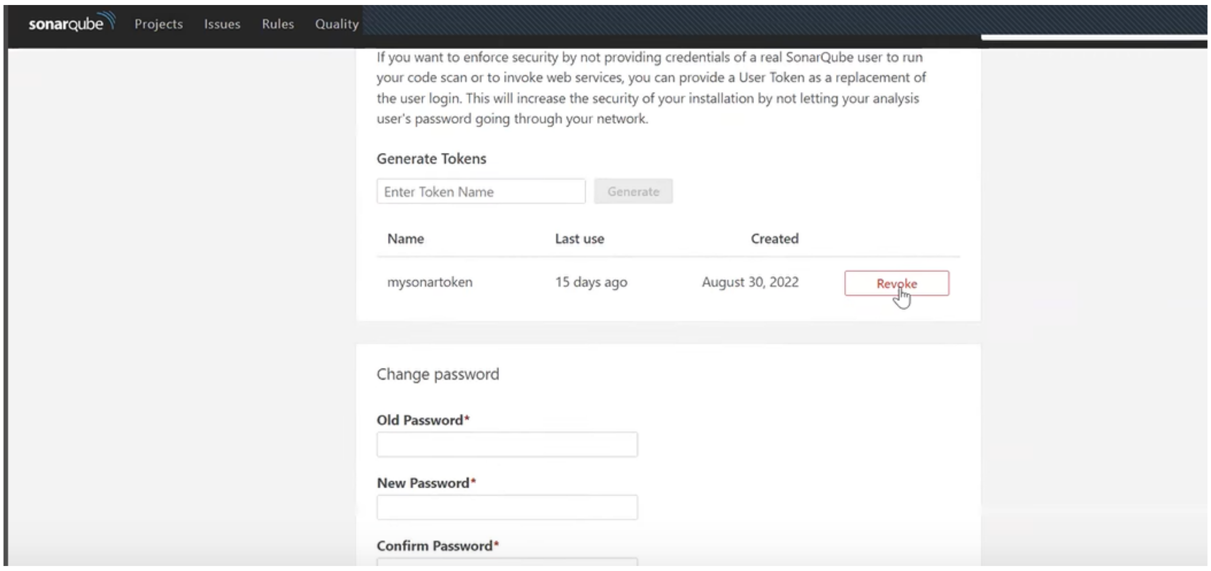
Kind: Secret text

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

Secret

Kind : secret text you have to go to the sonar server and configure token.

Login to sonar server → account → security →Generate token(copy)



If you want to enforce security by not providing credentials of a real SonarQube user to run your code scan or to invoke web services, you can provide a User Token as a replacement of the user login. This will increase the security of your installation by not letting your analysis user's password going through your network.

Generate Tokens

Enter Token Name | Generate

Name	Last use	Created	
mysonartoken	15 days ago	August 30, 2022	Revoke

Change password

Old Password*

New Password*

Confirm Password*

Paste as a secret text of credentials.

Note:* if you note able to add credentials hare then you need to configure as a global credential.

(manage Jenkins→credentials →Global→add credentials)

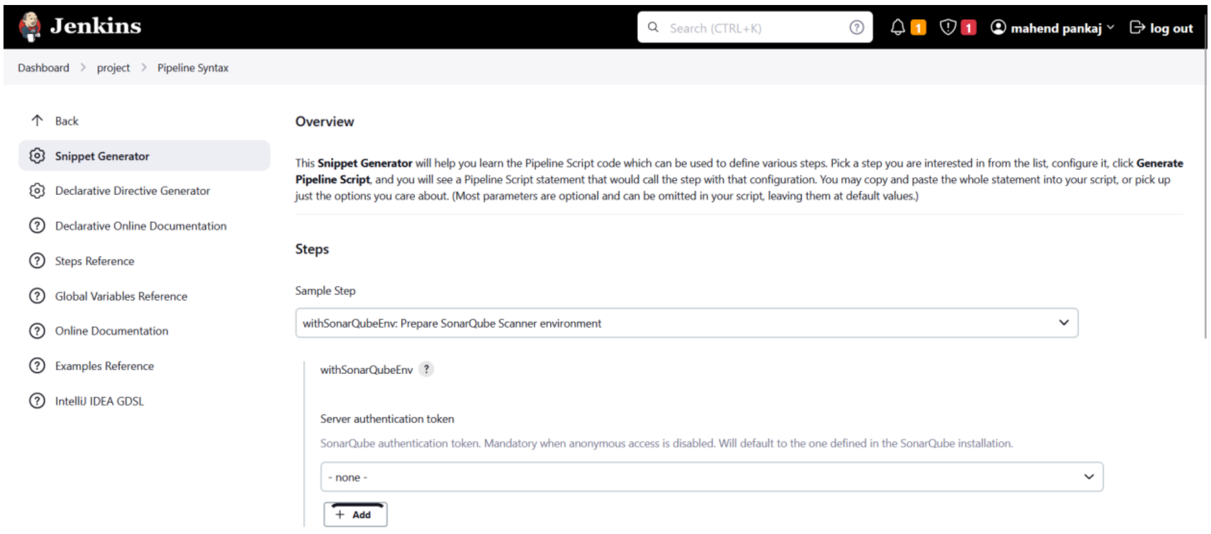
Now from configure system select the credential at drop-down.

Apply & SAVE

After that we need to add next stage as a 'Code Review' in a pipeline script.

Click pipeline syntax → select the option called

withSonarQubeEnv: Prepare Sonarqube scanner environment →select authentication token → Generate Pipeline Script



Overview

This **Snippet Generator** will help you learn the Pipeline Script code which can be used to define various steps. Pick a step you are interested in from the list, configure it, click **Generate Pipeline Script**, and you will see a Pipeline Script statement that would call the step with that configuration. You may copy and paste the whole statement into your script, or pick up just the options you care about. (Most parameters are optional and can be omitted in your script, leaving them at default values.)

Steps

Sample Step: withSonarQubeEnv: Prepare SonarQube Scanner environment

withSonarQubeEnv

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled. Will default to the one defined in the SonarQube installation.

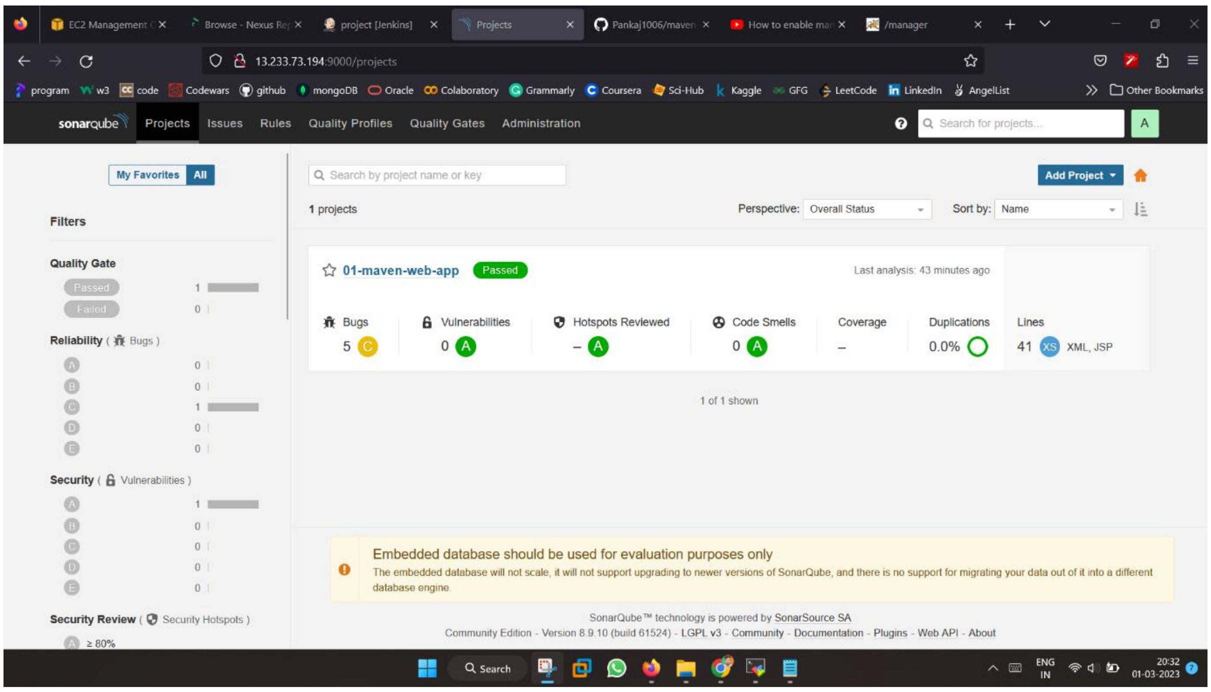
- none -

Copy & paste in pipeline script stage:

```
stage('Code Review'){
        withSonarQubeEnv(credentialsId: 'sonar-token') {
        def mavenHome = tool name: "maven", type: "maven"
        def mavenCMD = "${mavenHome}/bin/mvn"
        sh "mvn sonar:sonar"
        }
```

Apply & SAVE

*If you build the stages you get new project in sonar server with code review.


13.233.73.194:9000/projects

1 projects

Perspective: Overall Status   Sort by: Name

01-maven-web-app Passed   Last analysis: 43 minutes ago

Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications	Lines
5 C	0 A	– A	0 A	–	0.0%	41 XS XML, JSP

1 of 1 shown

Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

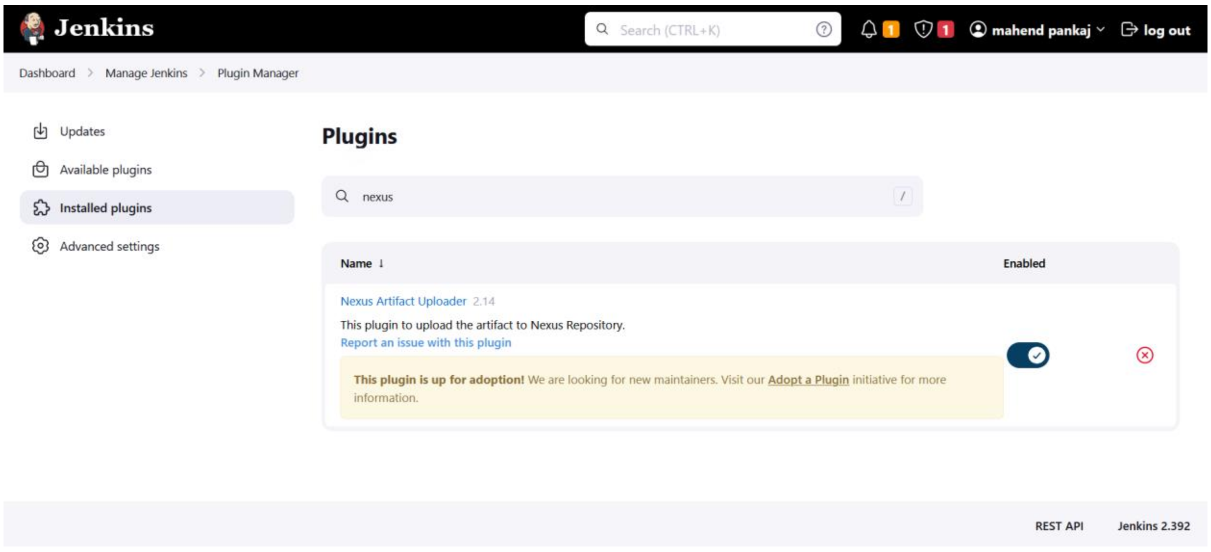
SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 8.9.10 (build 61524) - LGPL v3 - Community - Documentation - Plugins - Web API - About

Stage-4 : Upload Artifacts (By using Nexus-server)

We need to integrate nexus in Jenkins to upload the artifacts.

Go to dashboard→manage Jenkins→manage plugin→install

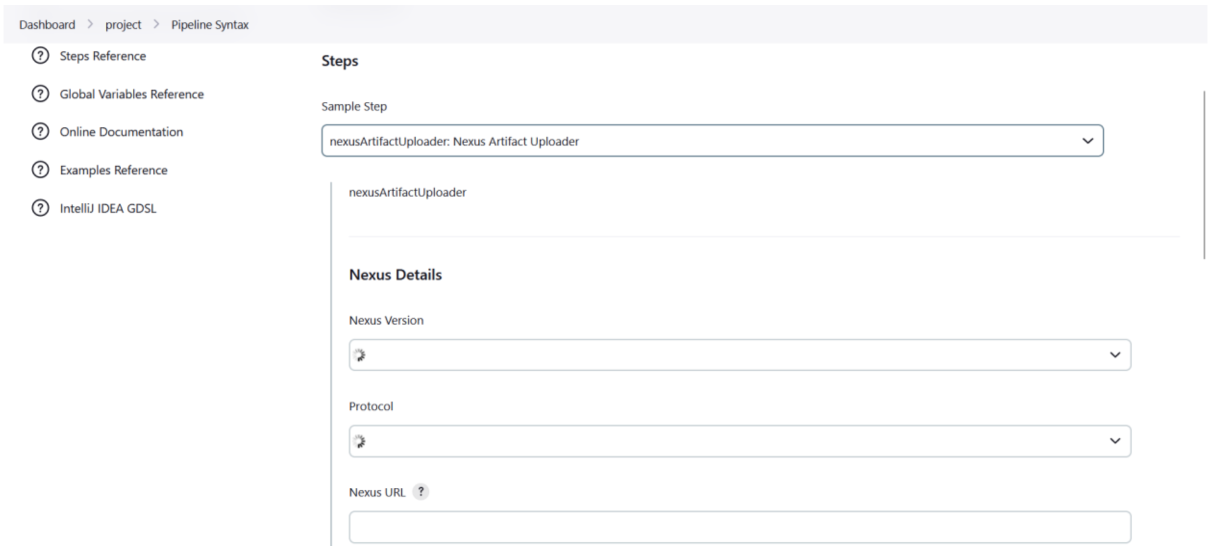
Plugin called 'Nexus Artifact Uploader'.



Nexus Artifact Uploader 2.14
This plugin to upload the artifact to Nexus Repository.
Report an issue with this plugin
This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.

Ones the plugin install successfully then go to configure pipeline script for next stage.

Click the pipeline syntax to configure pipeline script for nexus stage.



Sample Step: nexusArtifactUploader: Nexus Artifact Uploader

nexusArtifactUploader

Nexus Details

Nexus Version

Protocol

Nexus URL

Select step called “nexusArifactUploader: Nexus Artifact Uploader”

Nexus version: see on nexus browser which version you have configure (NEXUS2/NUXUS3)

Protocal : http

Nexus url : <public-ip of nexus server>:8081 (http:// are not required)

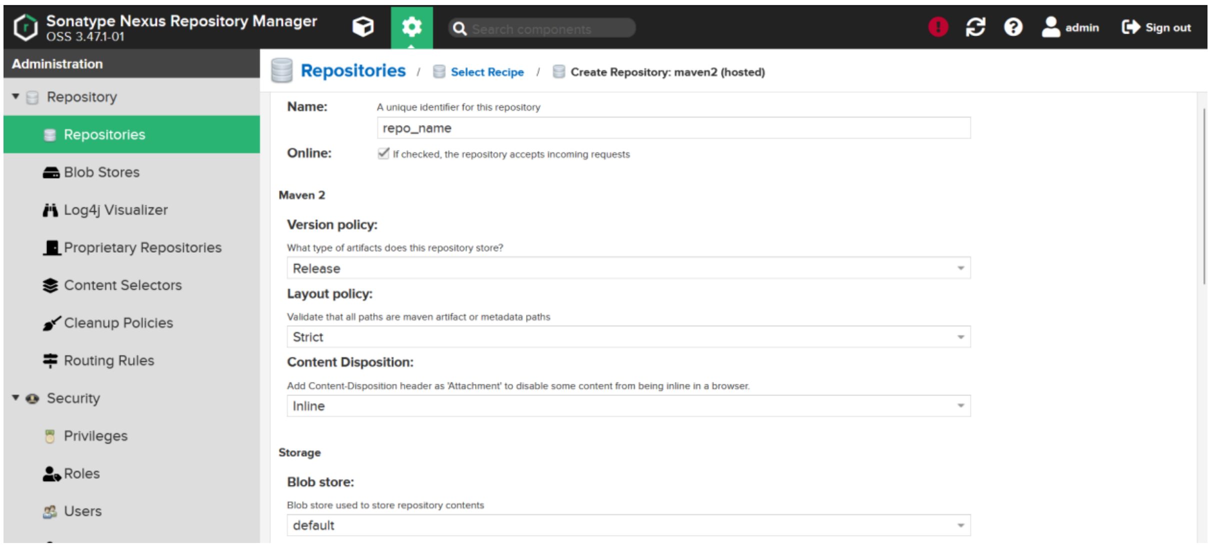
Credentials: configure credentials username & password as admin.

Group id: (in.project) you can give anything.

Version: 01-SNAPSHOT(version of your snapshot)

Repository: make new repository on nexus server:

Login in nexus server then create new repository.



Repositories / Select Recipe / Create Repository: maven2 (hosted)

Name: A unique identifier for this repository
repo_name

Online: ✓ If checked, the repository accepts incoming requests

Maven 2

Version policy:
What type of artifacts does this repository store?
Release

Layout policy:
Validate that all paths are maven artifact or metadata paths
Strict

Content Disposition:
Add Content-Disposition header as 'Attachment' to disable some content from being inline in a browser.
Inline

Storage

Blob store:
Blob store used to store repository contents
default

Select maven2(hosted) and Development policy: Allow redeploy

Create repository

Now in repository fill the repo_name as per created in nexus server.

Artifacts:

Artifactid:<01-maven-web-app> (is the project name that you want to use)

Type:war

Classifier:………………

File: target/<01-maven-web-app.war>(where the war file will be created)

Generate pipeline script

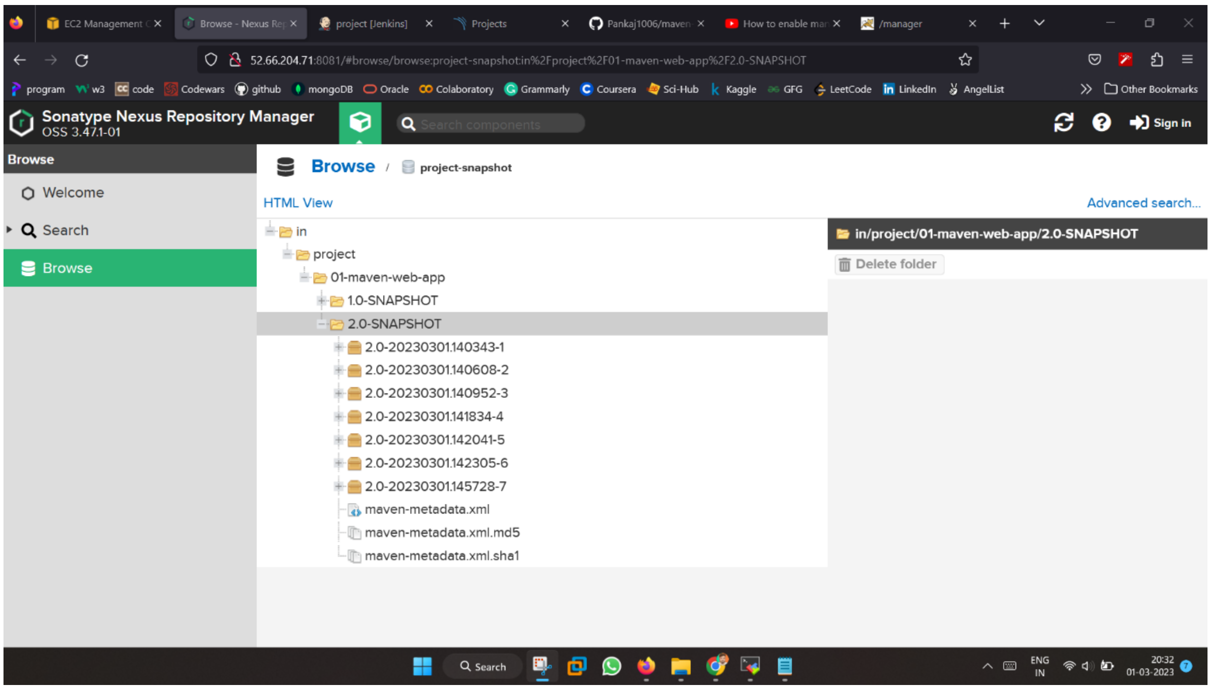
You can just copy the generated syntax and paste in a pipeline script next stage.

As shown below:

```
stage('Upload Artifact on Nexus'){
        nexusArtifactUploader artifacts: [[artifactId: '01-maven-web-
app', classifier: '', file: 'target/01-maven-web-app.war', type: 'war']],
credentialsId: 'nexus-credentials', groupId: 'in.project', nexusUrl:
'52.66.204.71:8081', nexusVersion: 'nexus3', protocol: 'http',
repository: 'project-snapshot', version: '2.0-SNAPSHOT'
    }
```

Apply & SAVE

*if you build the stage you can get a snapshot created in Nexus server. As shown below:



52.66.204.71:8081/#browse/browse:project-snapshot:in%2Fproject%2F01-maven-web-app%2F2.0-SNAPSHOT

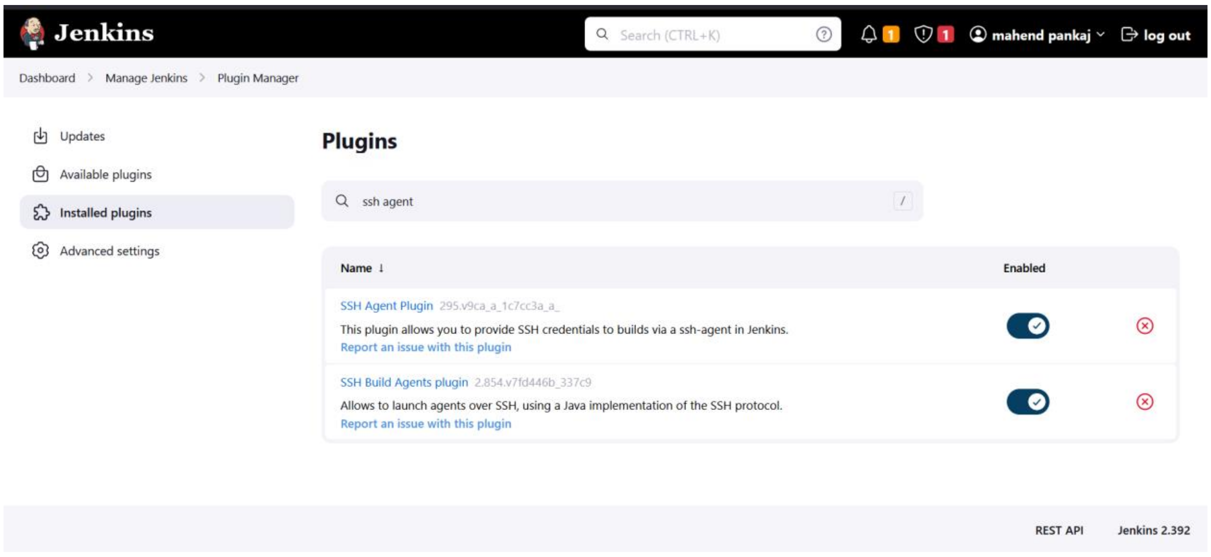
in/project/01-maven-web-app/2.0-SNAPSHOT

```
in
  project
    01-maven-web-app
      1.0-SNAPSHOT
      2.0-SNAPSHOT
        2.0-20230301.140343-1
        2.0-20230301.140608-2
        2.0-20230301.140952-3
        2.0-20230301.141834-4
        2.0-20230301.142041-5
        2.0-20230301.142305-6
        2.0-20230301.145728-7
        maven-metadata.xml
        maven-metadata.xml.md5
        maven-metadata.xml.sha1
```

That means your nexus server being configured by Jenkins server Successfully.

Stage-5 : Deploy App (By using Tomcat-Server)

To copy the war file generated in nexus server in Tomcat server we need to install plugin in Jenkins called 'SSH Agent'



Name ↓	Enabled
SSH Agent Plugin 295.v9ca_a_1c7cc3a_a_ This plugin allows you to provide SSH credentials to builds via a ssh-agent in Jenkins. Report an issue with this plugin	✓
SSH Build Agents plugin 2.854.v7fd446b_337c9 Allows to launch agents over SSH, using a Java implementation of the SSH protocol. Report an issue with this plugin	✓

Now go to our pipeline script create a new stage and for that you need to configure pipeline syntax.

Select step called "sshagent: SSH Agent" then configure credentials of tomcat server:

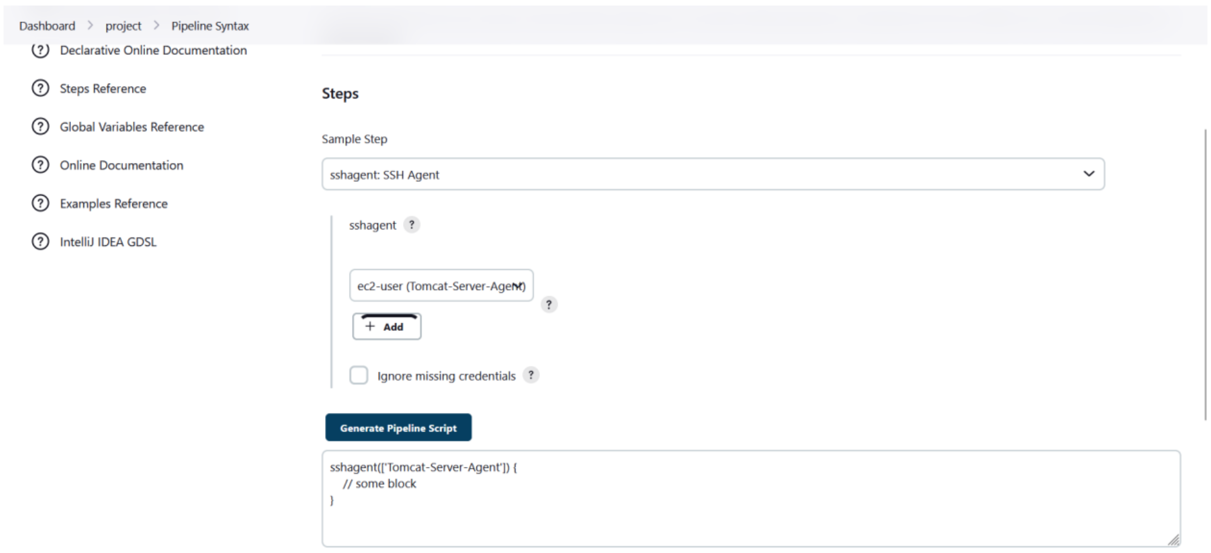
Kind: SSH username and private key

Username : ec2-user

Private key : Here you need to copy your private key .pem file content.

Then select credentials.

Generate pipeline script



Sample Step: sshagent: SSH Agent

sshagent: ec2-user (Tomcat-Server-Agent)

Ignore missing credentials: ☐

```
sshagent(['Tomcat-Server-Agent']) {
    // some block
}
```

SSH Agent get successfully configure now you need to deploy war file in tomcat server by using ssh agent configured in Jenkins server.

war file locaton: target/01-maven-web-app.war

(on Jenkins server)

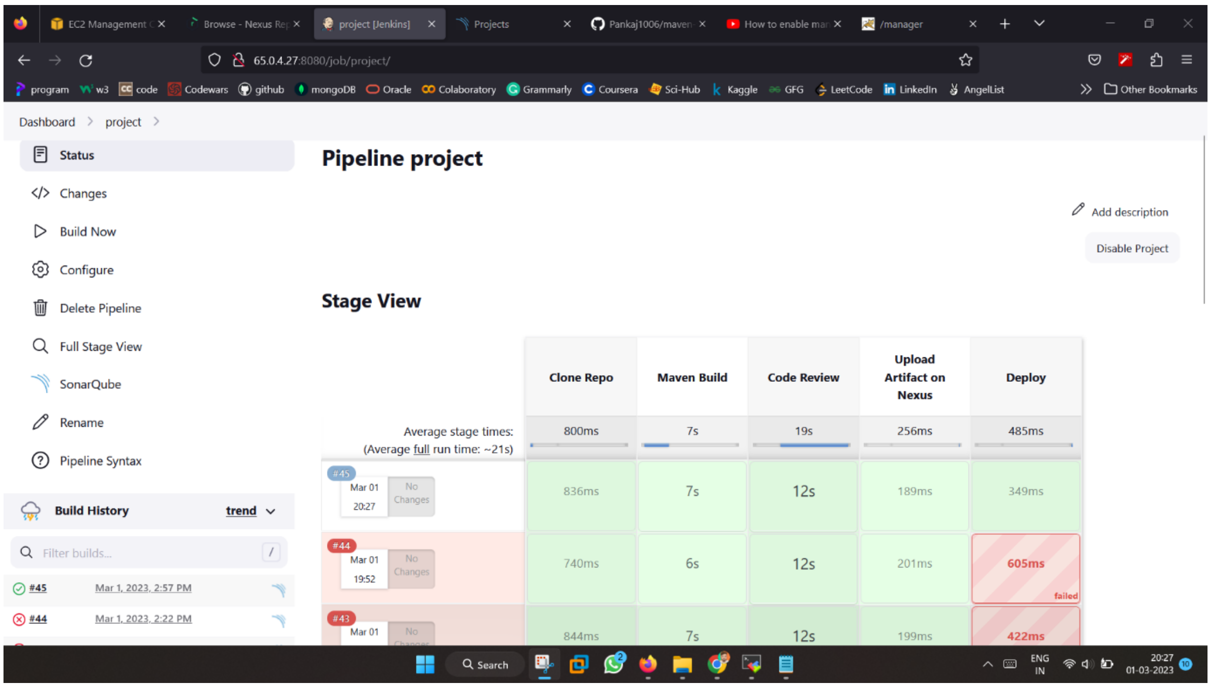
destination location : home/ec2-user/apache-tomcat-9.0.72/webapps

(on tomcat server)

```
   stage('Deploy'){
       sshagent(['Tomcat-Server-Agent']) {
           sh 'scp -o StrictHostKeyChecking=no target/01-maven-web-app.war ec2-user@65.0.32.119:/home/ec2-user/apache-tomcat-9.0.72/webapps'
          }
   }
```

Apply & SAVE

At this point your pipeline script is configure successfully now you click Build Now.



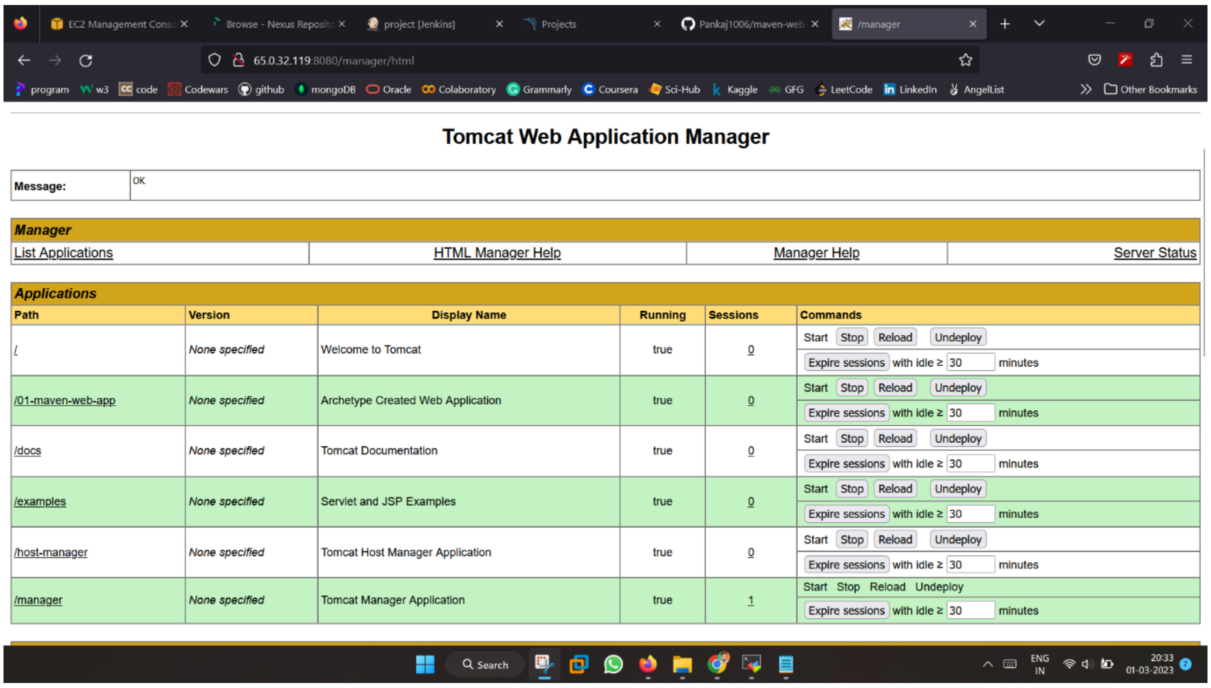
Pipeline project

Stage View

	Clone Repo	Maven Build	Code Review	Upload Artifact on Nexus	Deploy
Average stage times: (Average full run time: ~21s)	800ms	7s	19s	256ms	485ms
#45 Mar 01 20:27 No Changes	836ms	7s	12s	189ms	349ms
#44 Mar 01 19:52 No Changes	740ms	6s	12s	201ms	605ms failed
#43 Mar 01 No Changes	844ms	7s	12s	199ms	422ms

#45	Mar 1, 2023, 2:57 PM
#44	Mar 1, 2023, 2:22 PM

Then you can see on Tomcat server the web page is deployed.



Tomcat Web Application Manager

Message:	OK

Manager

List Applications	HTML Manager Help	Manager Help	Server Status

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/01-maven-web-app	None specified	Archetype Created Web Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

At this point your CI CD Pipeline project with security is being configured successfully.

****