\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**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**)**

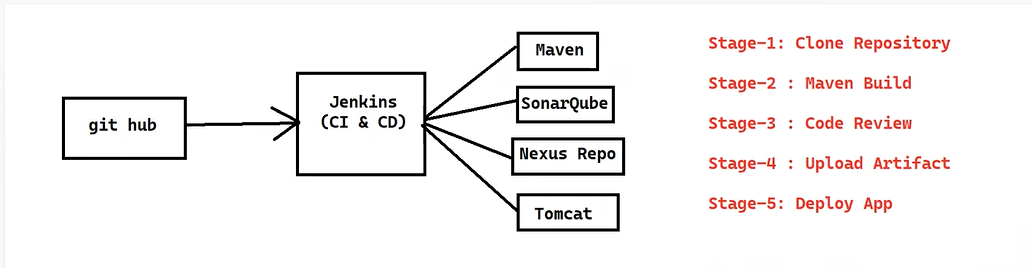
**2) Tomcat (url:**http://public-ip:8080 **)**

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

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

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

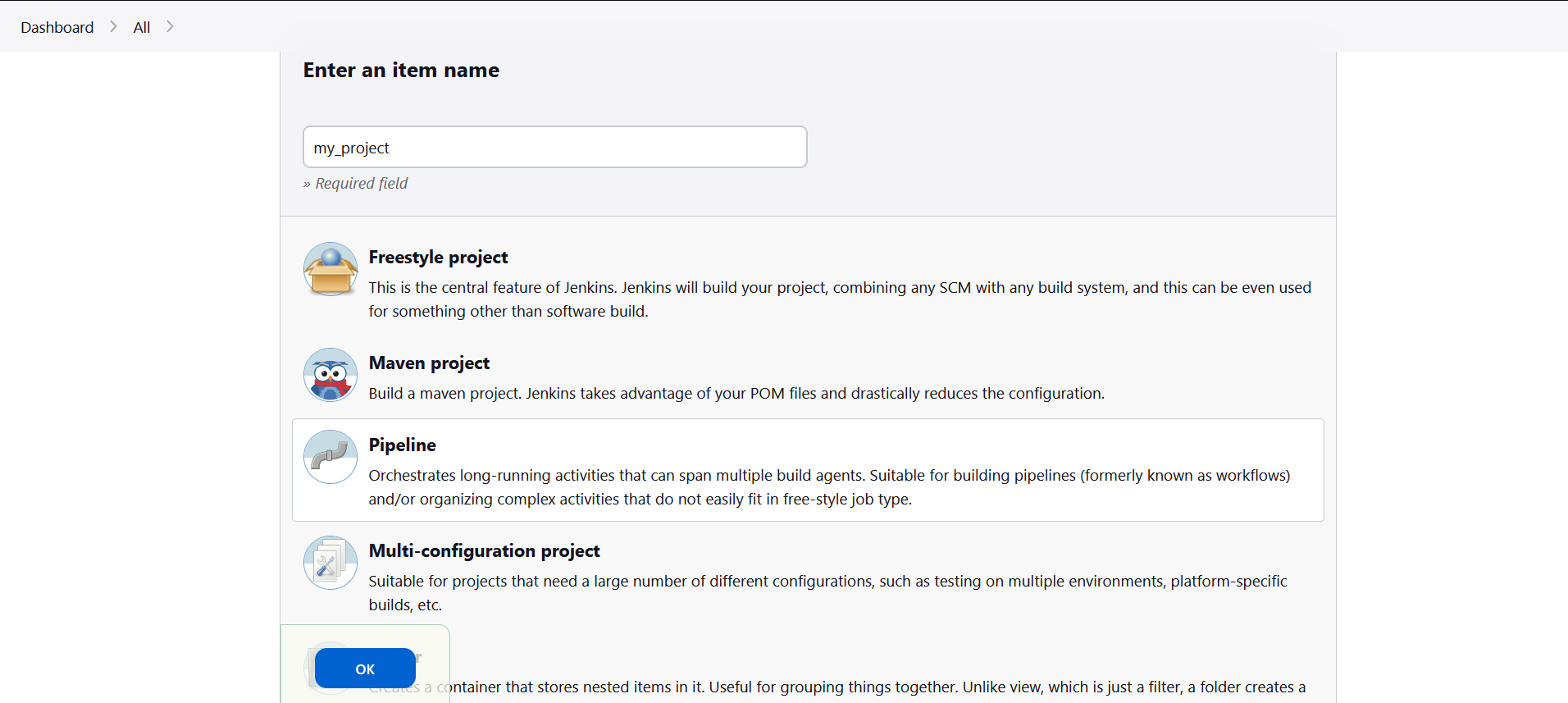
**6) Maven (**as a global tool in Jenkins**)**

****

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

**Start from the Jenkins server:**

From the Dashboard + New item



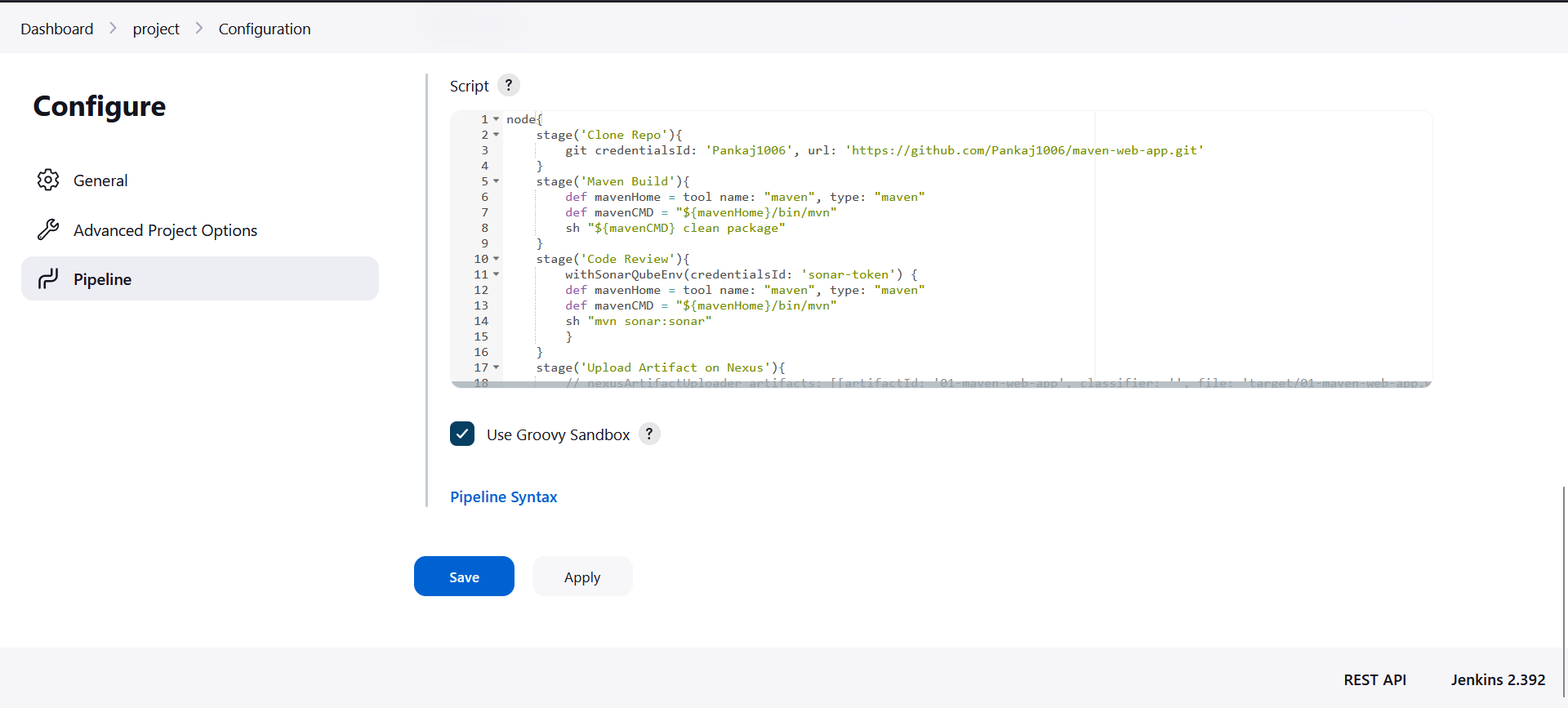
Enter an item name<project\_name>

Select Pipeline

OK

Now from Jenkins dashboard select the <project\_name> after that select configure, click on pipeline

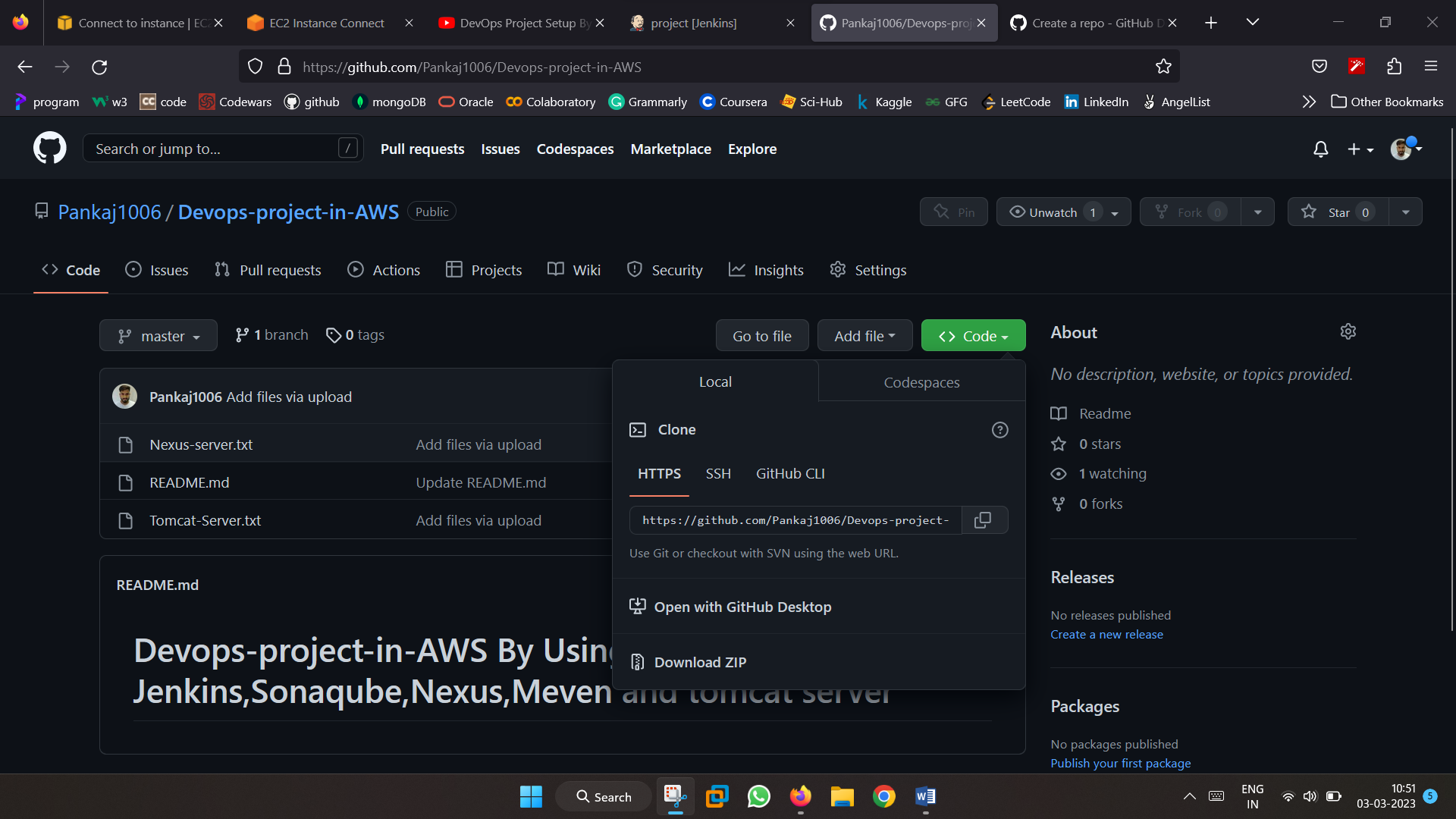
Here you can write your pipeline script.

****\*Script start from node or pipeline.

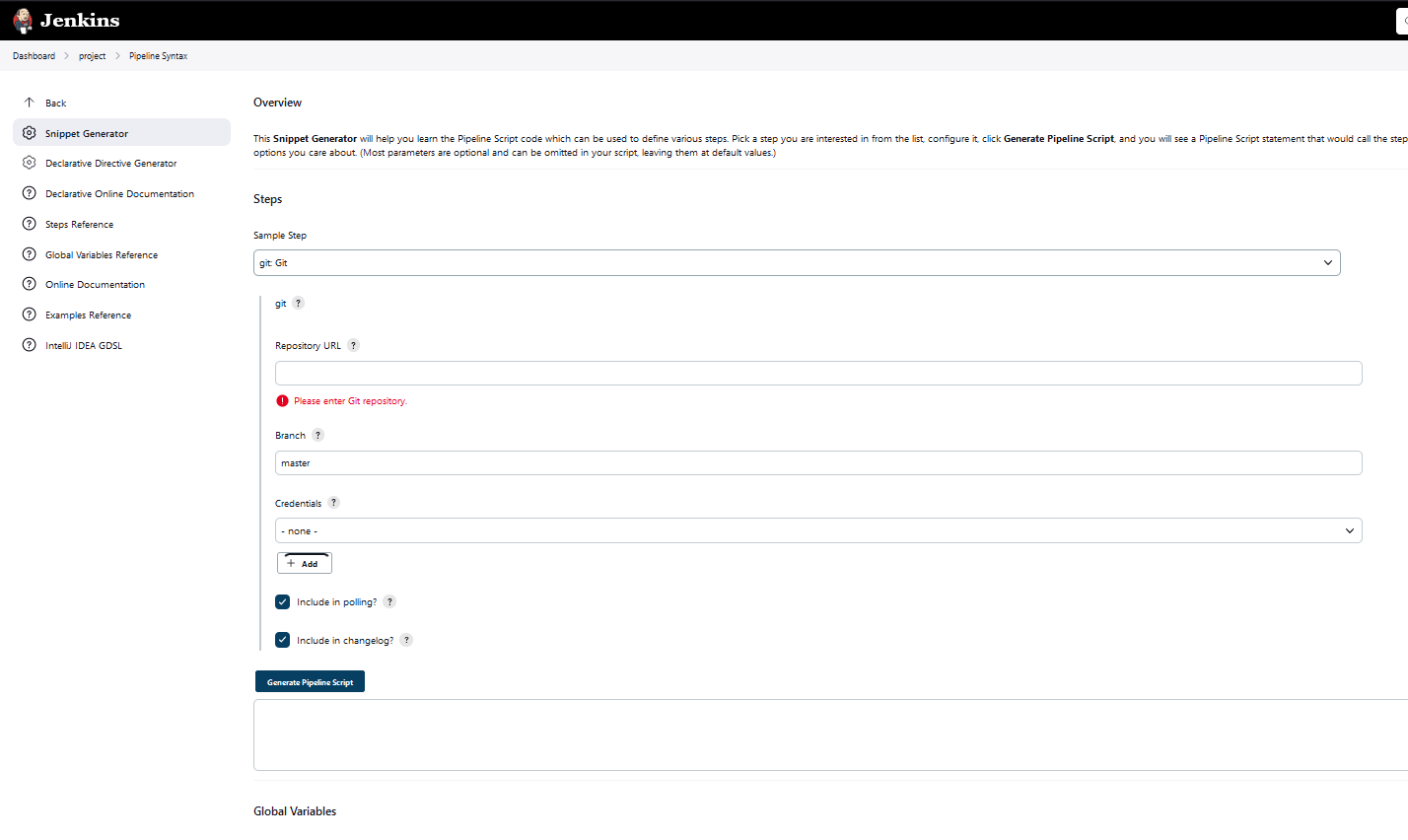
**Stage-1 : Clone Repository**

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

Now, copy the url of the repository.

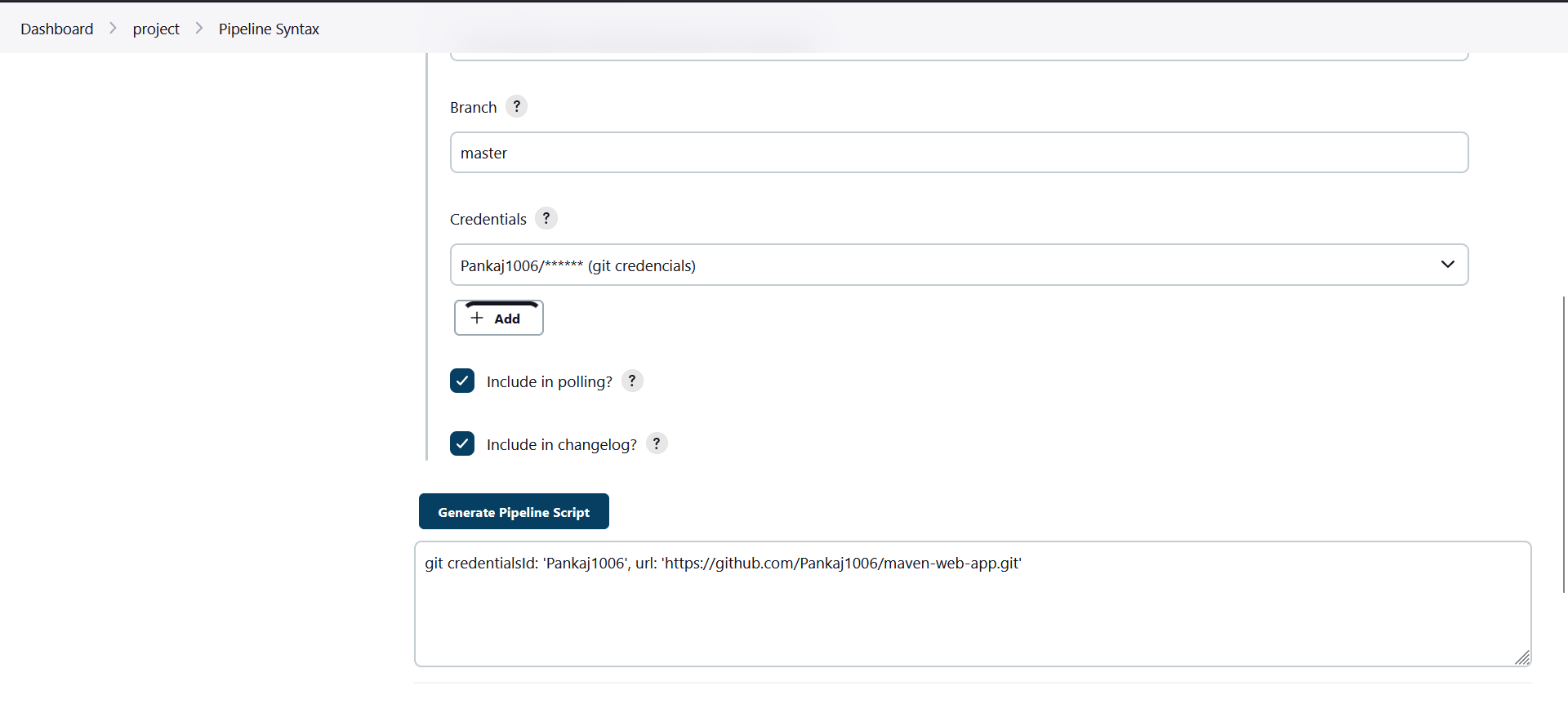


To configure(clone) github repository you need to click **pipeline syntax** option given below on Jenkins pipeline script.



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



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

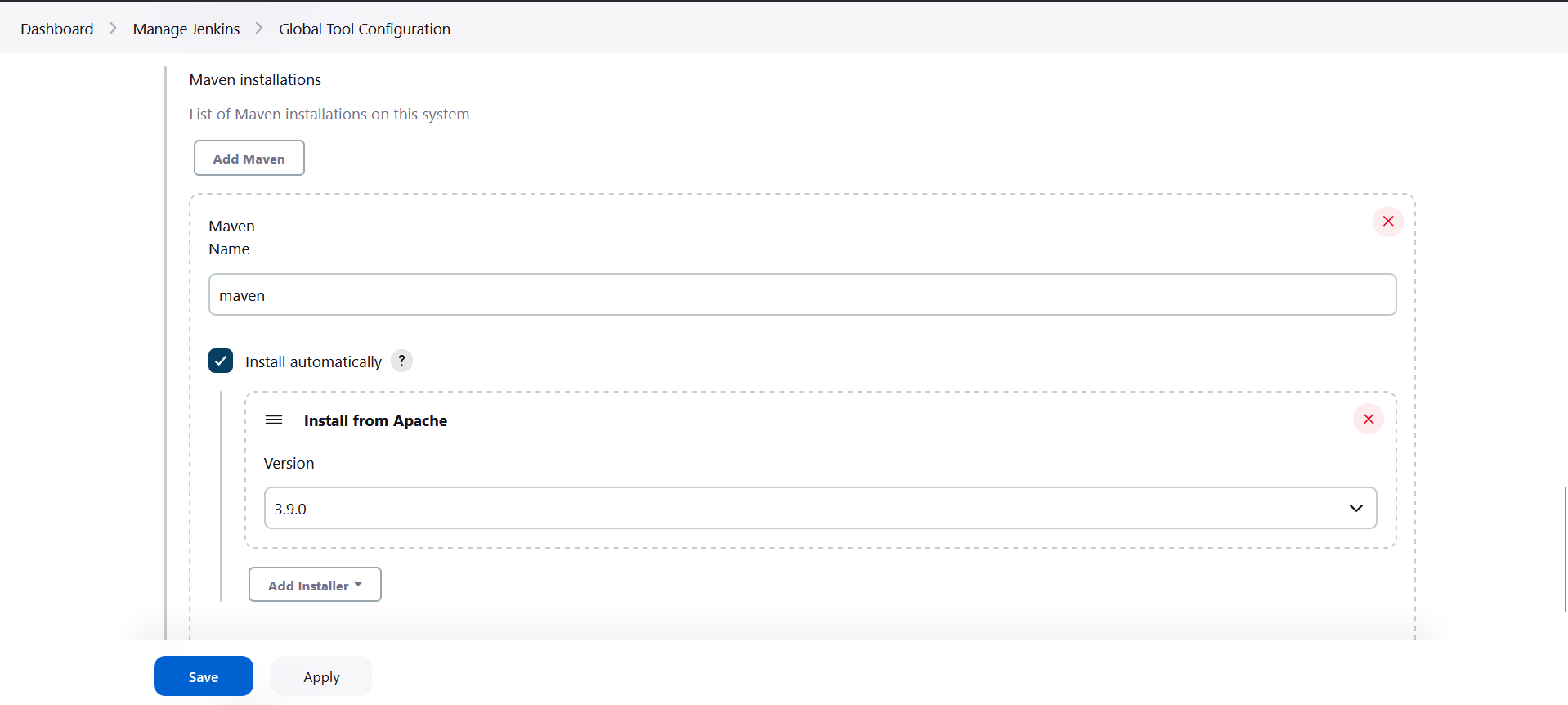
**Stage-2** **: Maven Build**

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.



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"

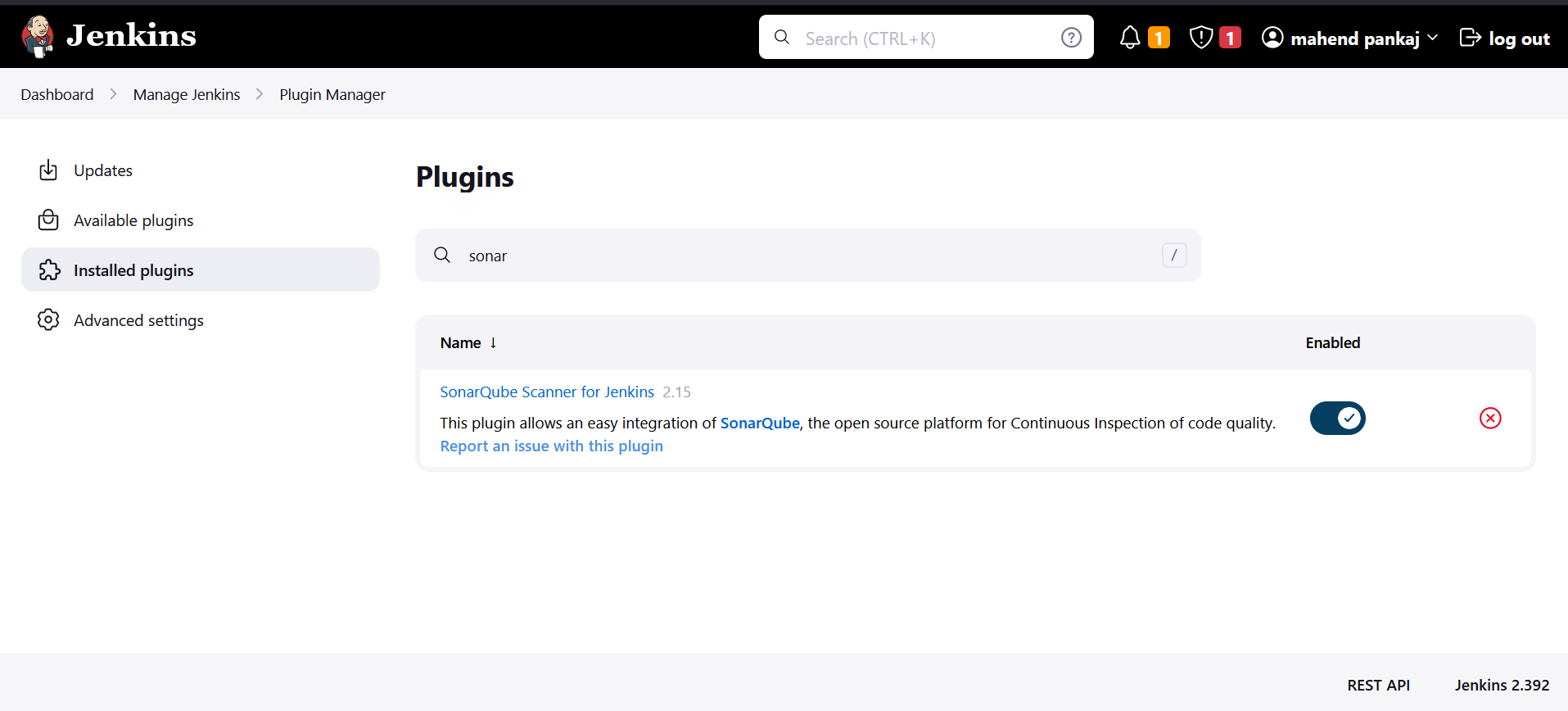
}

Apply & SAVE

**Stage-3 : Code Review (By using SonarQube)**

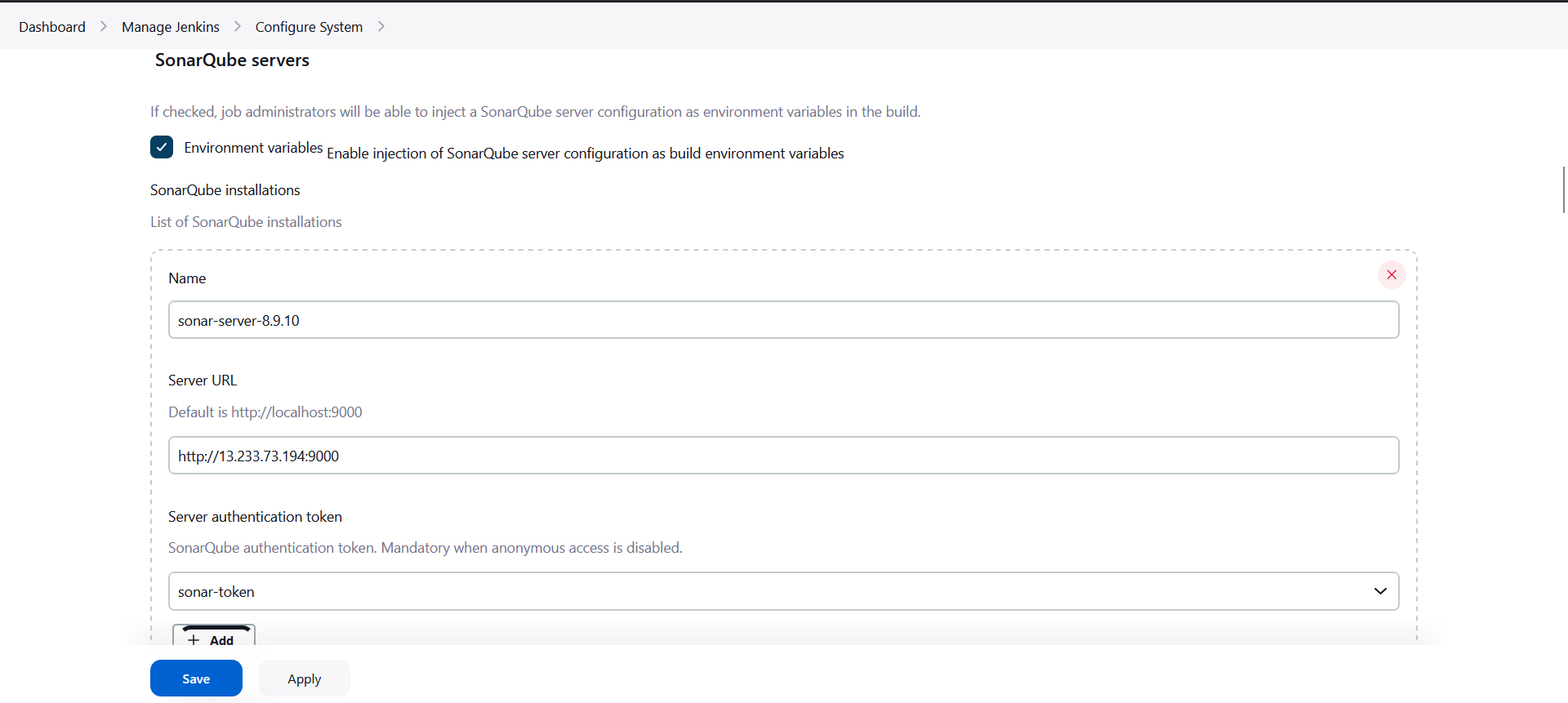
**I**n order to integrate Jenkins with sonar you have to configure plugin called “**Sonar Qube Scanner Plugin**”.

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



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

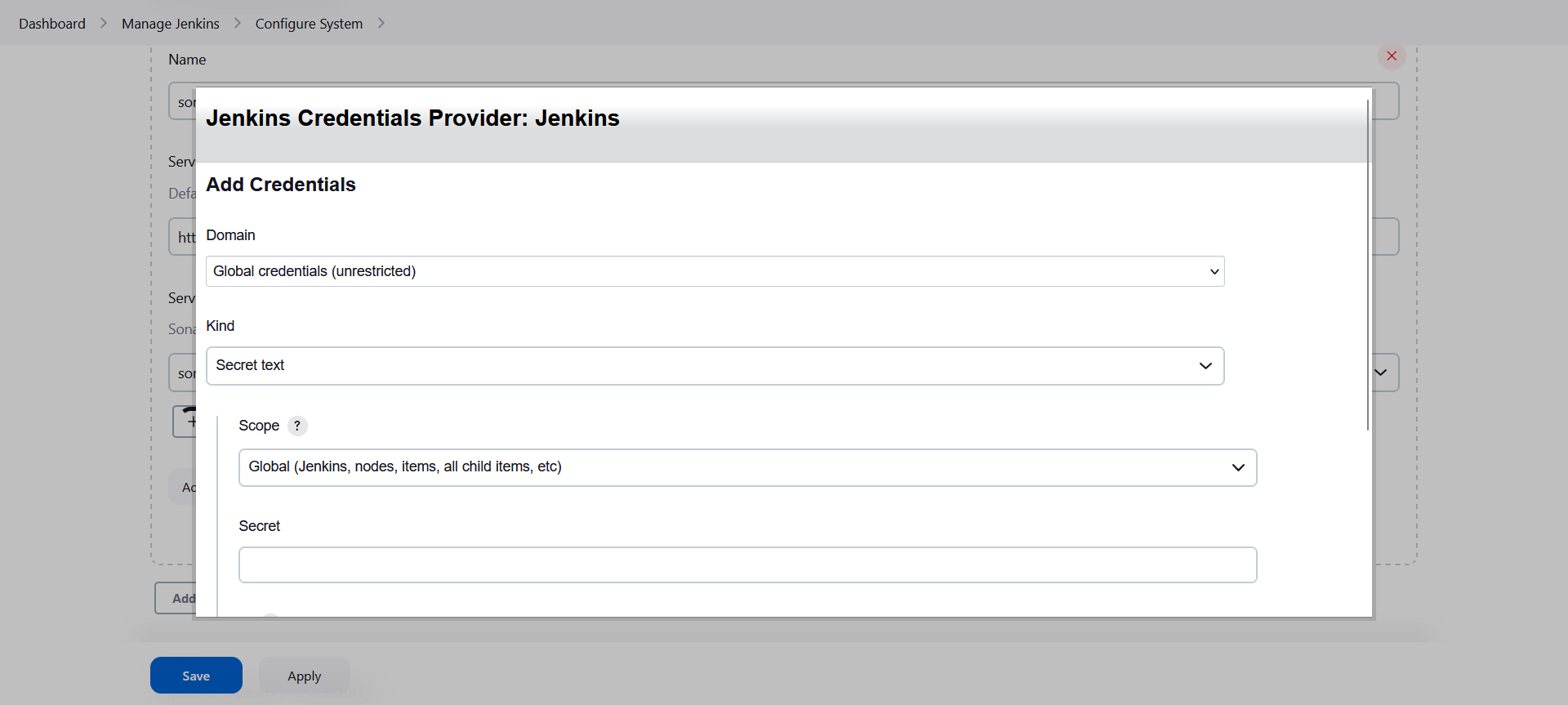
Manage Jenkins🡪configure systems🡪Sonarqube Server(Add sonarqube)



You can give any name <sonar\_server>

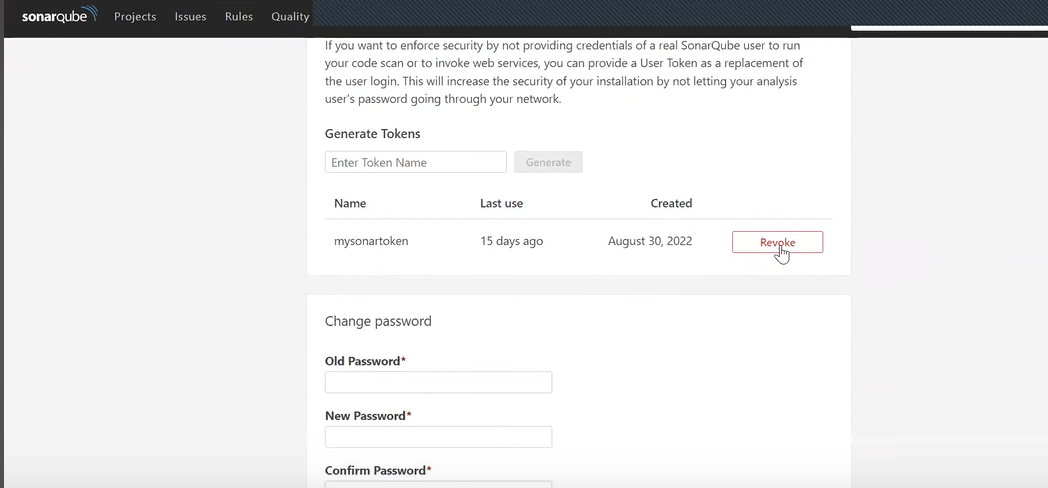
Copy url of the sonar server from the browser and paste in server url.

Now for Server authentication token click add.



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

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



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)

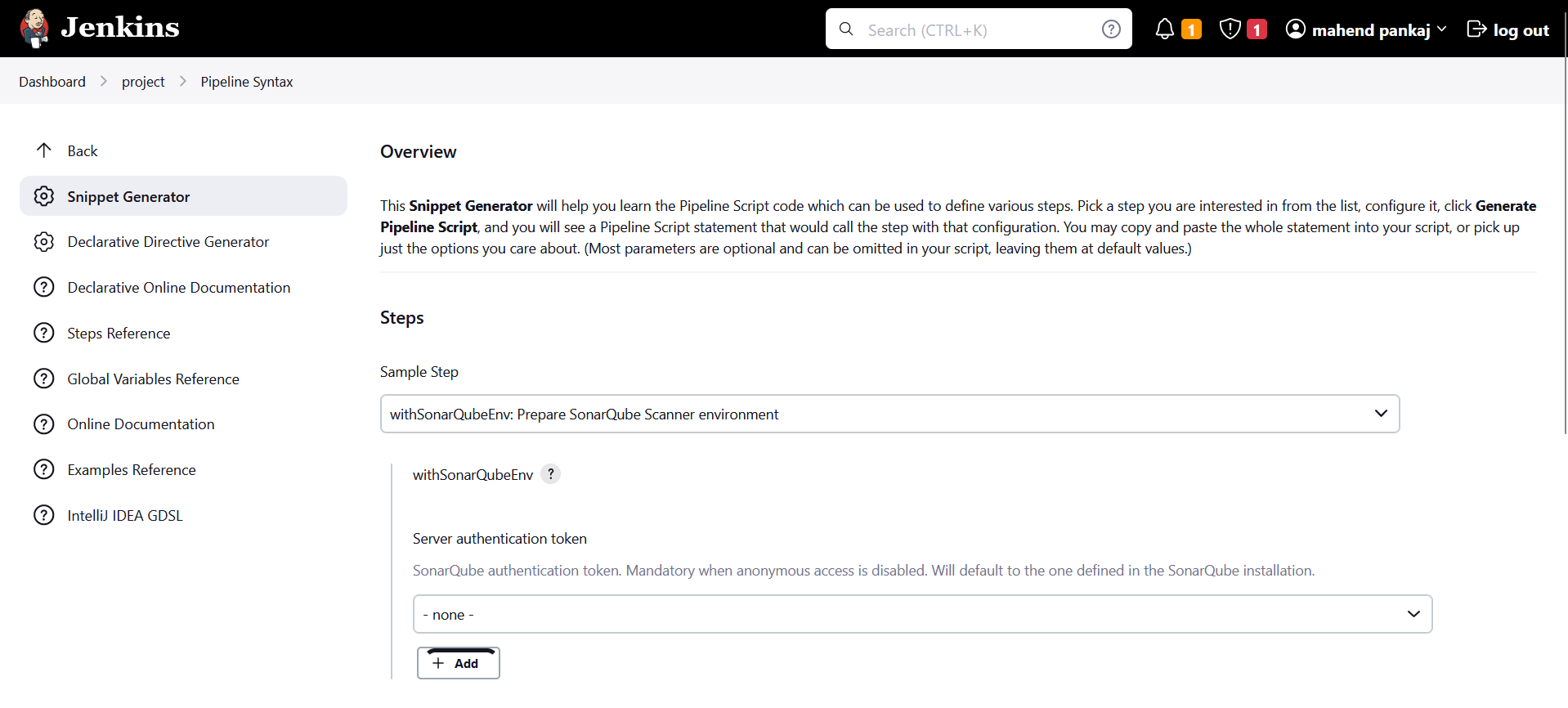
Now from configure system select the credential at drop-down.

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



Copy & paste in pipeline script stage:

stage('Code Review'){

withSonarQubeEnv(credentialsId: 'sonar-token') {

def mavenHome = tool name: "maven", type: "maven"

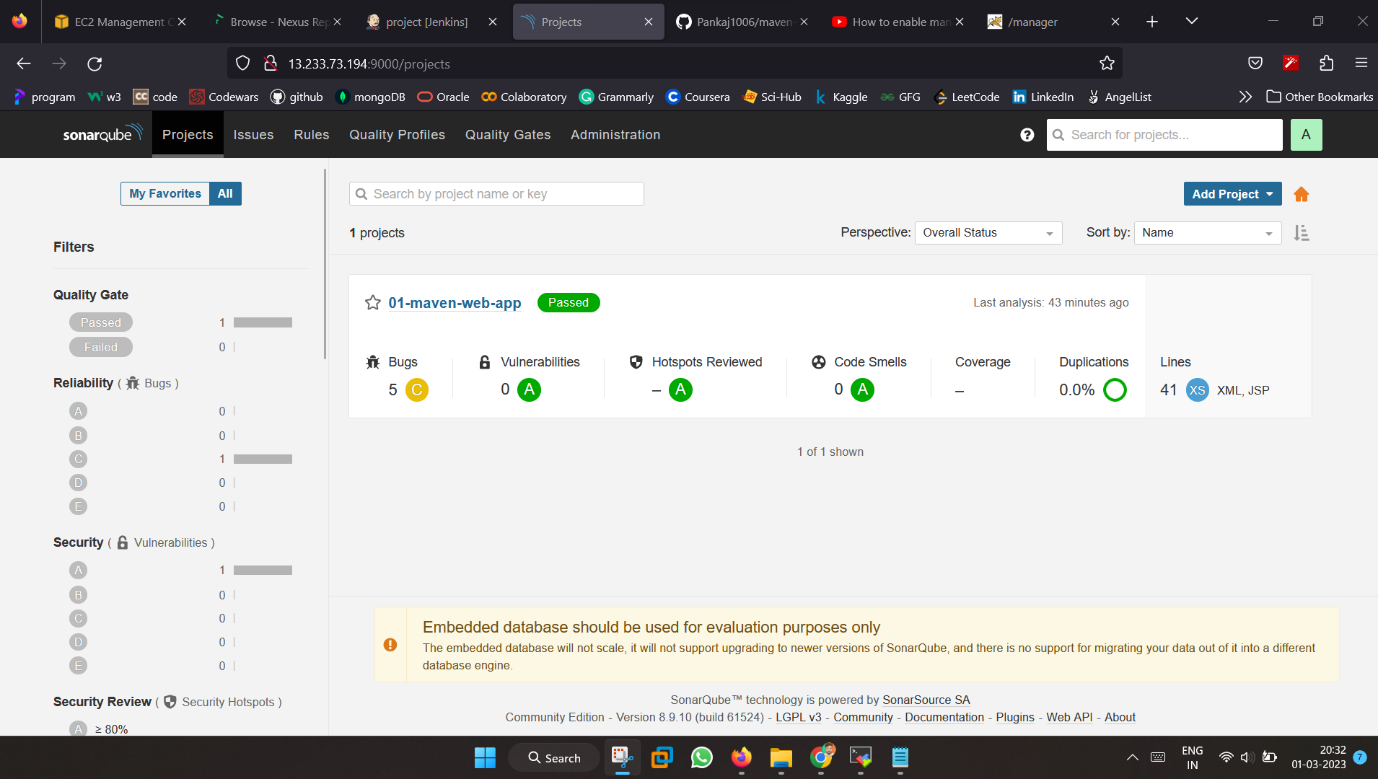
def mavenCMD = "${mavenHome}/bin/mvn"

sh "mvn sonar:sonar"

}

Apply & SAVE

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

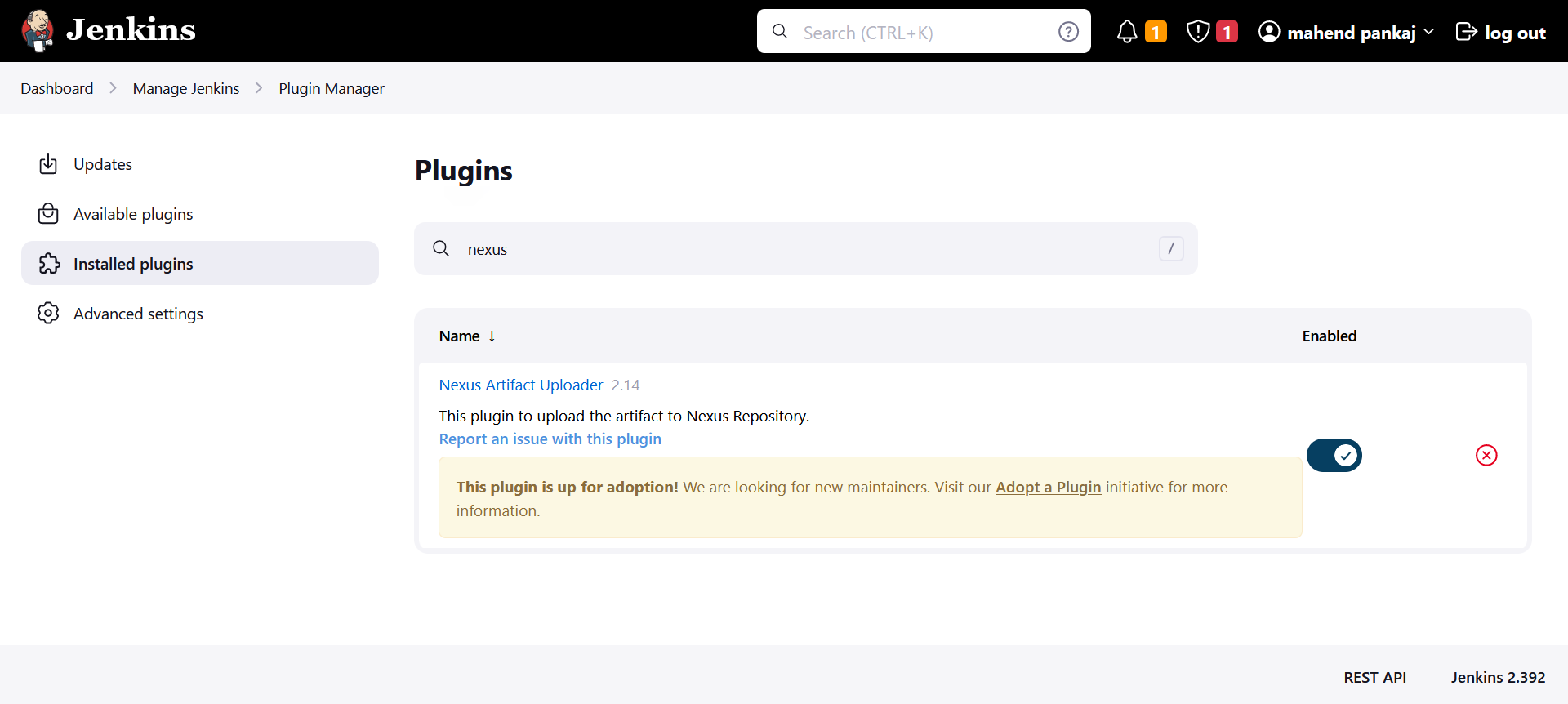


**Stage-4 : Upload Artifacts (By using Nexus-server)**

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

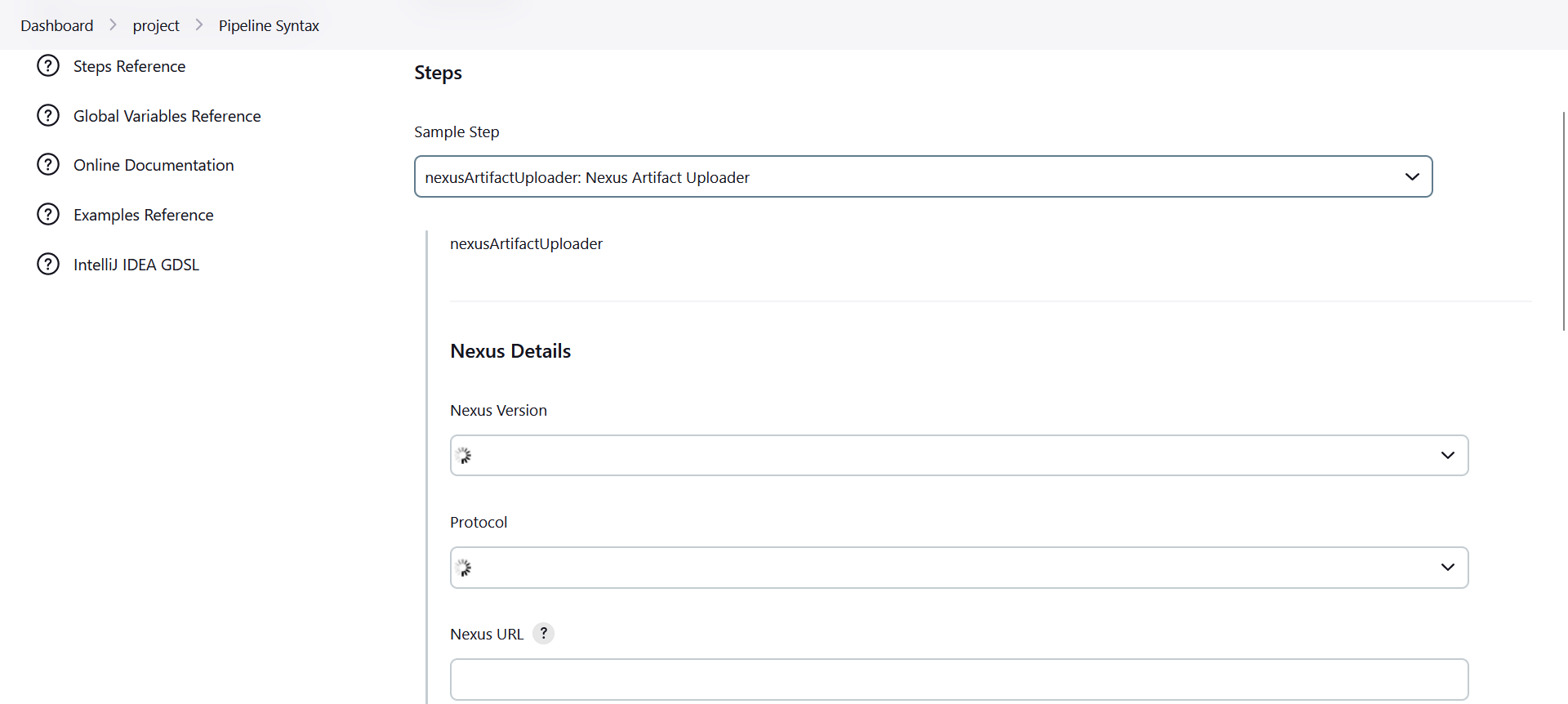
Go to dashboard🡪manage Jenkins🡪manage plugin🡪install

Plugin called ‘Nexus Artifact Uploader’.



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.



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

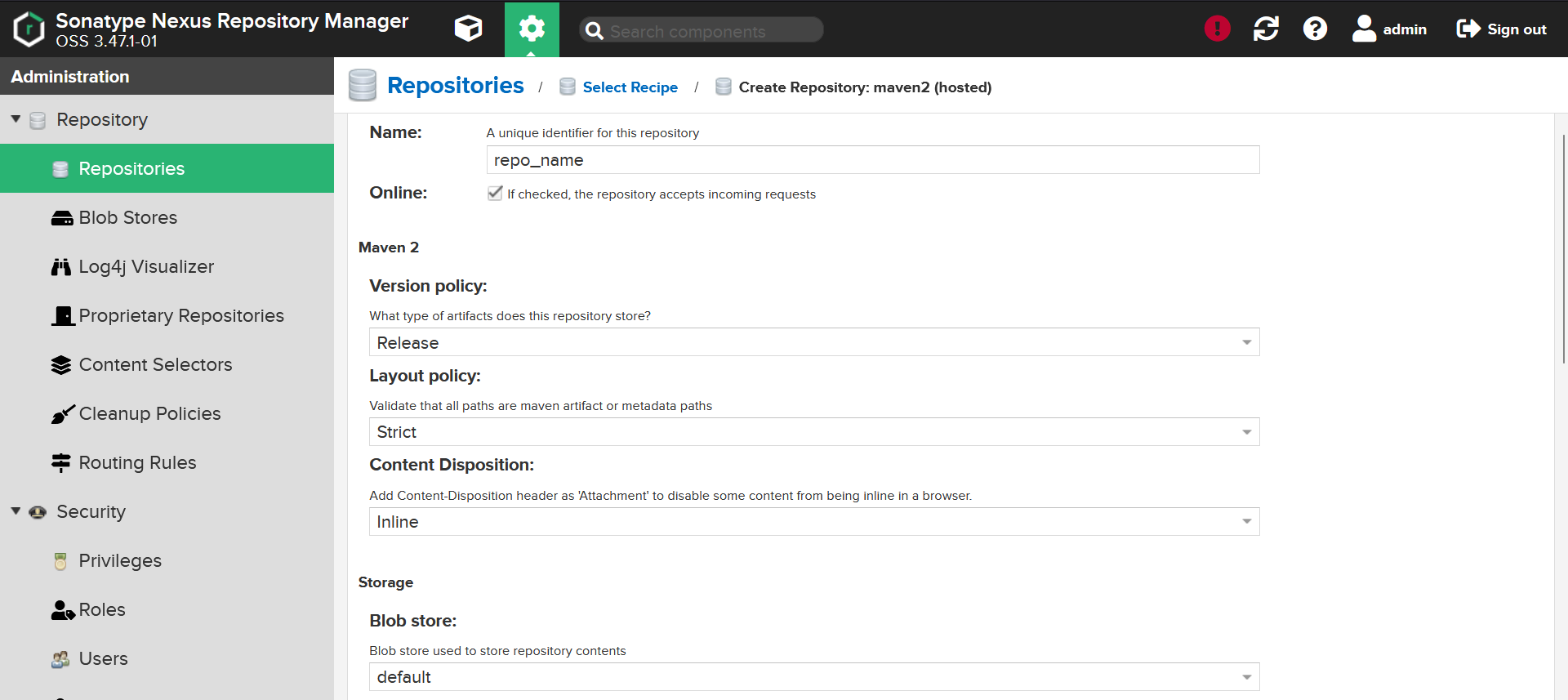
Credentials: configure credentials username & password as admin.

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.



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'){

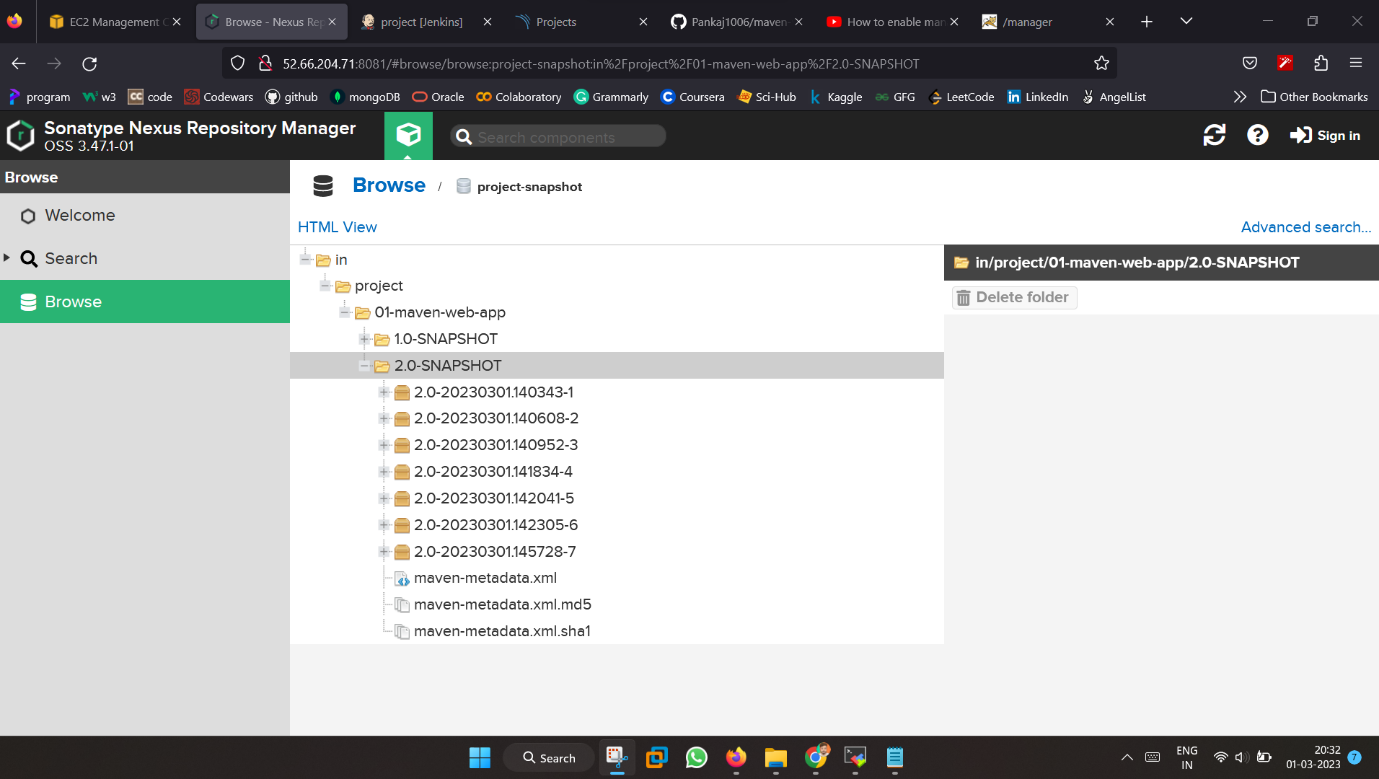
nexusArtifactUploader artifacts: [[artifactId: '01-maven-web-app', classifier: '', file: 'target/01-maven-web-app.war', type: 'war']], credentialsId: 'nexus-credentials', groupId: 'in.project', nexusUrl: '52.66.204.71:8081', nexusVersion: 'nexus3', protocol: 'http', repository: 'project-snapshot', version: '2.0-SNAPSHOT'

}

Apply & SAVE

\*if you build the stage you can get a snapshot created in Nexus server.

As shown below:

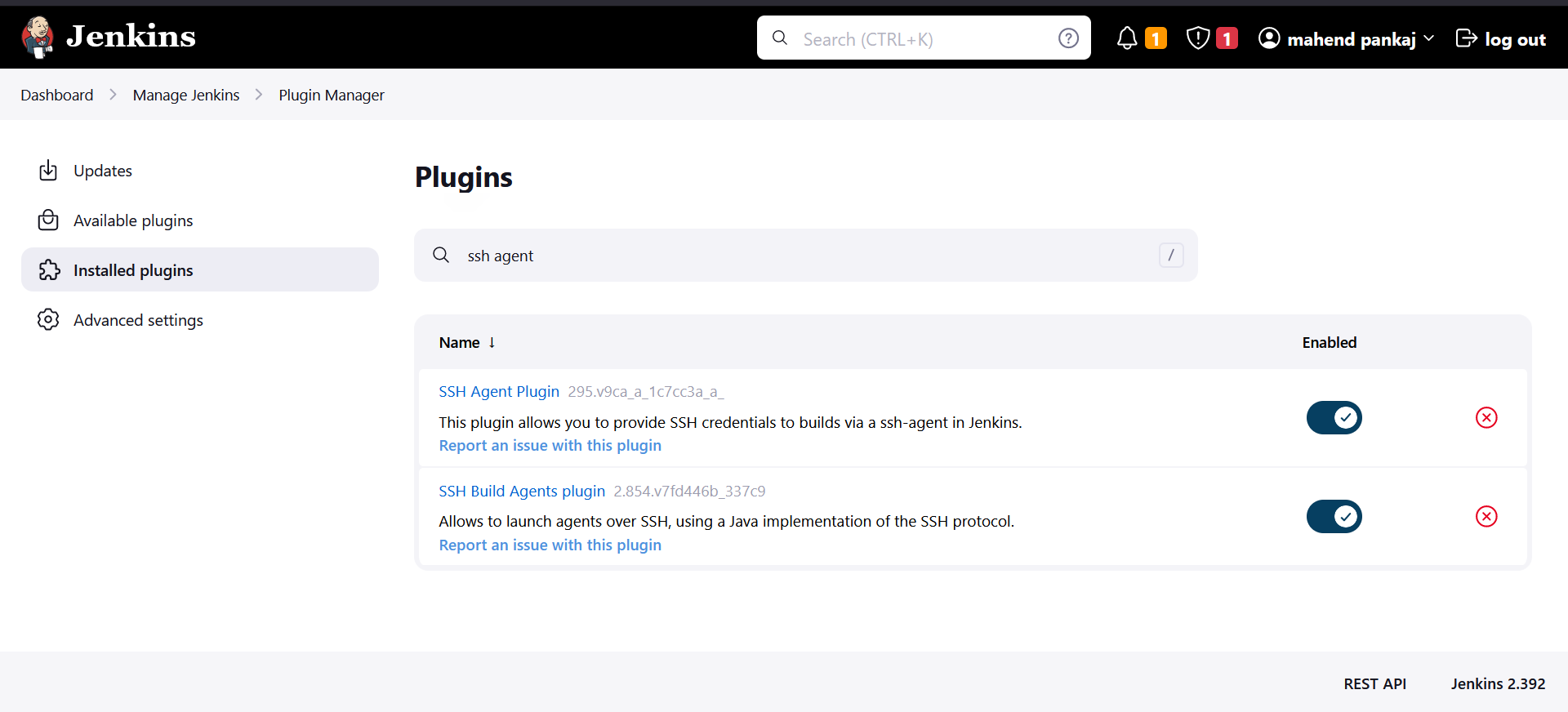


That means your nexus server being configured by Jenkins server

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’



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:

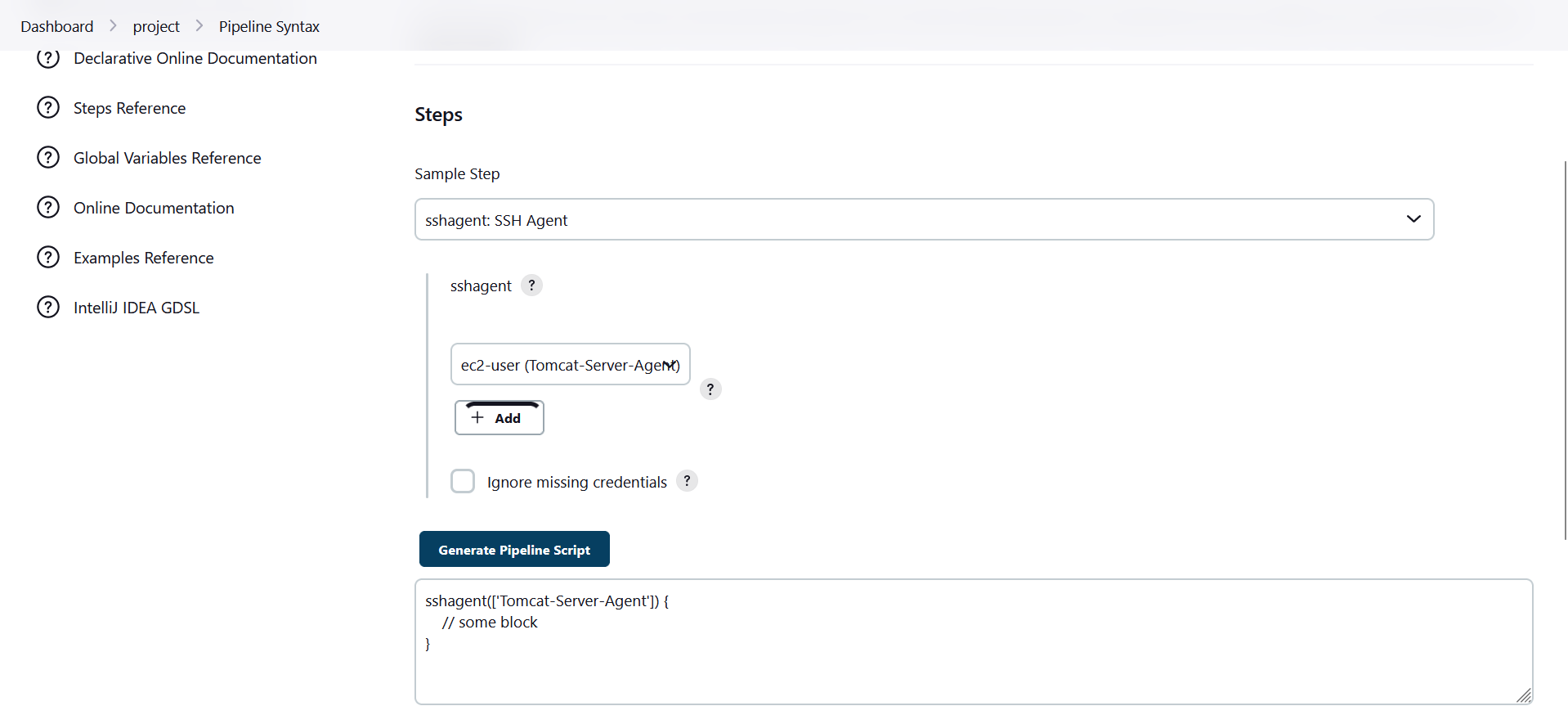
Kind: SSH username and private key

Username : ec2-user

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

Then select credentials.

Generate pipeline script



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']) {

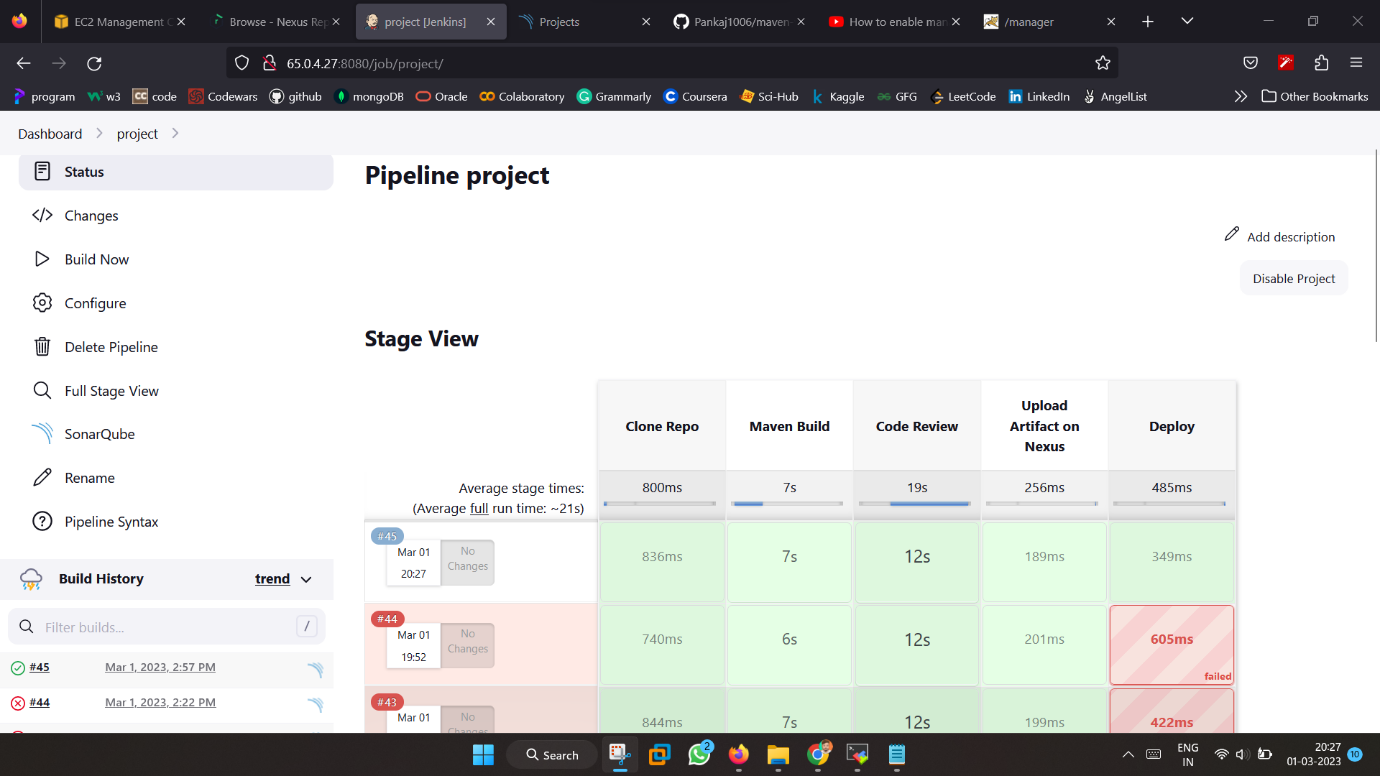
sh 'scp -o StrictHostKeyChecking=no target/01-maven-web-app.war ec2-user@65.0.32.119:/home/ec2-user/apache-tomcat-9.0.72/webapps'

}

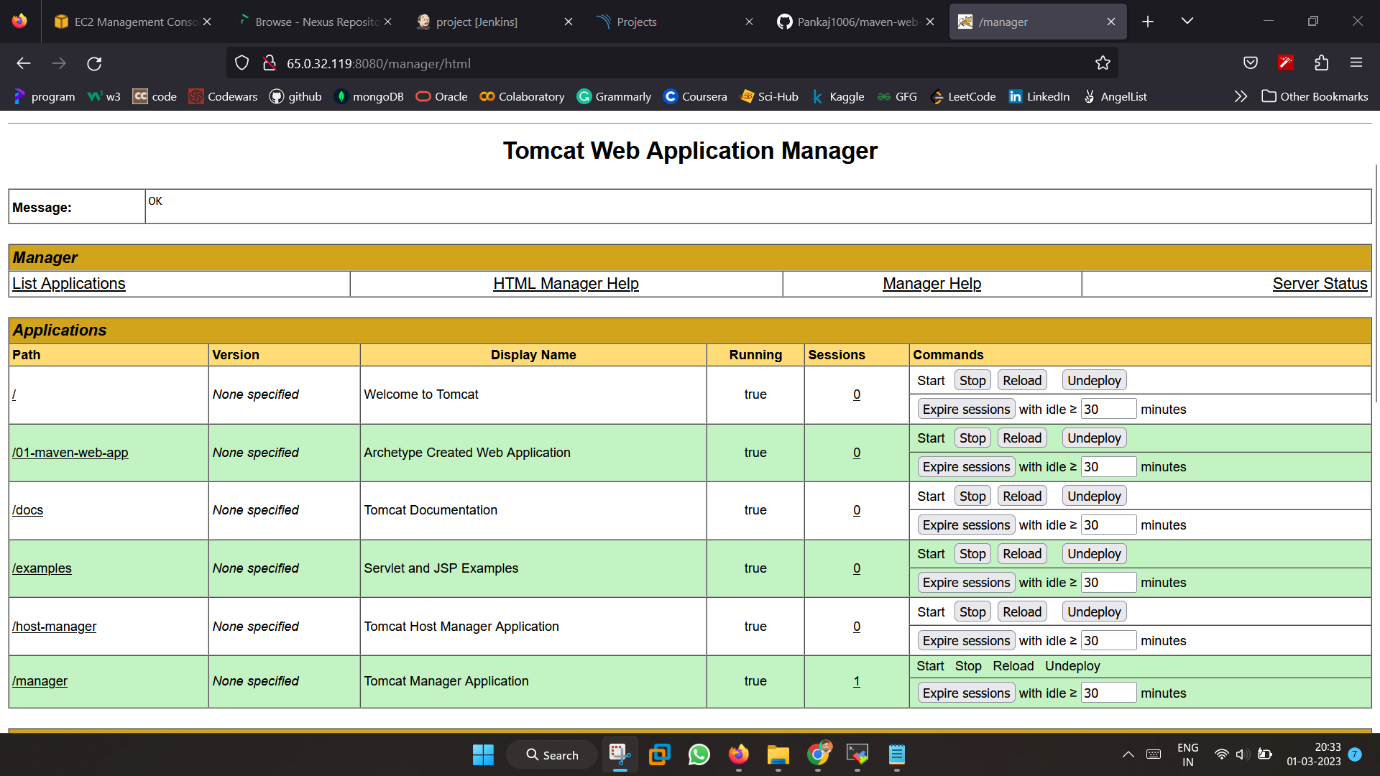
}

Apply & SAVE

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



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



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

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*