# Publishing Artifacts to Nexus Repository using Jenkins Pipelines

**Installing Nexus Repository**

Nexus is a repository manager that allows you to store and retrieve artifacts. It enables you to host your built artifacts in a private and secure repo.

You can always pull the Nexus Docker image using the following command:

$ docker pull sonatype/nexus3

Using default tag: latest

latest: Pulling from sonatype/nexus3

cb3c77f9bdd8: Pull complete

fd8daf2668d1: Pull complete

fd1ff82b00e8: Pull complete

2a05f7b573af: Pull complete

Digest: sha256:6570855dfbc3eb094fe5cbbacec87aa8b91d16394dab627177e1deeebb5ac8ee

Status: Downloaded newer image for sonatype/nexus3:latest

docker.io/sonatype/nexus3:latest

Now it’s time to run downloaded sonatype/nexus on the default port 8081. Follow these commands:

$ docker run -d --name nexus\_repo -p 8081:8081 sonatype/nexus3

Usually it takes 1 to 2 minutes for the Nexus service to launch in your newly created Docker container. If you wish to follow the log to see if Nexus is up and ready, then run the following command:

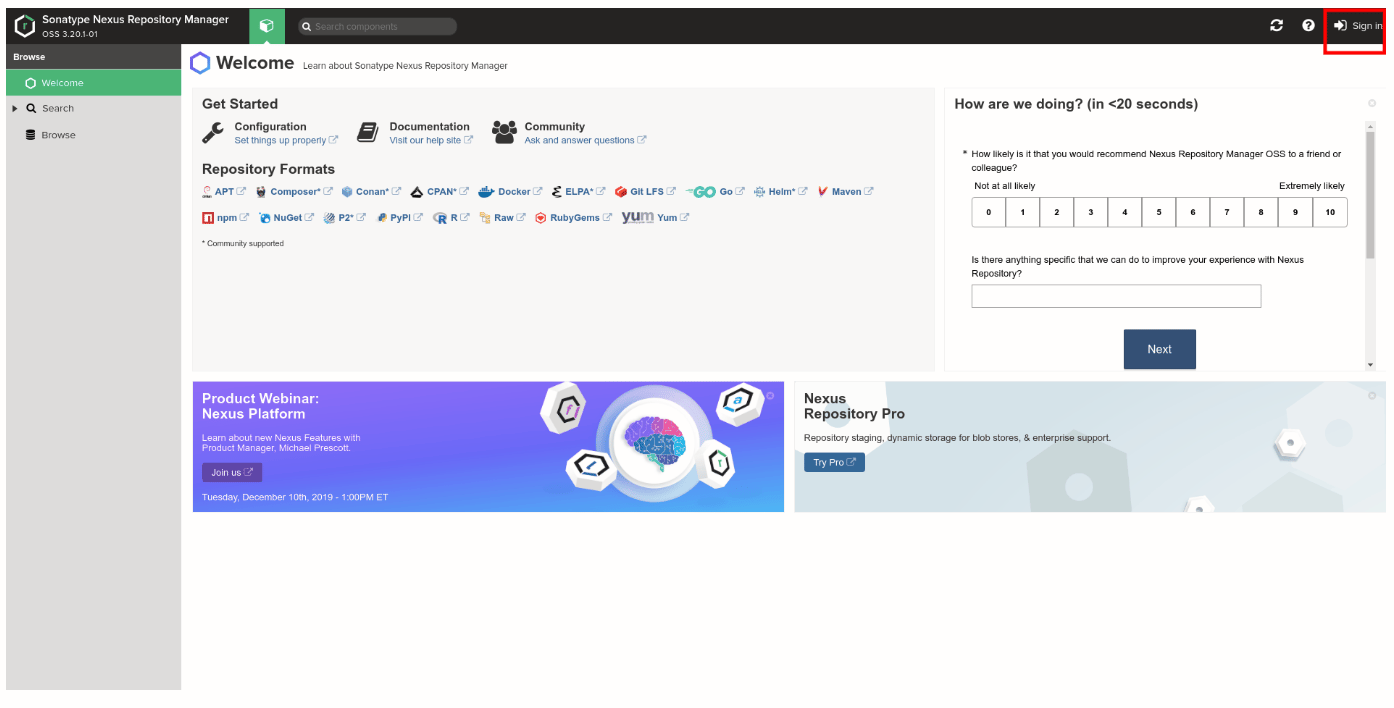
$ docker logs nexus\_repo -f

In logs you will see a message stating Started Sonatype Nexus OSS 3.20.1-01

This means your Nexus Repository Manager is ready to use. Now go to your browser and open

http://your-ip-addr:8081

Find the *Sign In* option as shown below:



The default username is admin, whereas to retrieve the password you need to run the following command:

$ docker exec -i nexus\_repo cat /nexus-data/admin.password

502ace93-5450-4f0d-97d2-9b3b3a88d149

And that’s it. Your Nexus Repository Manager is ready to use. The next step is to create a new repository.

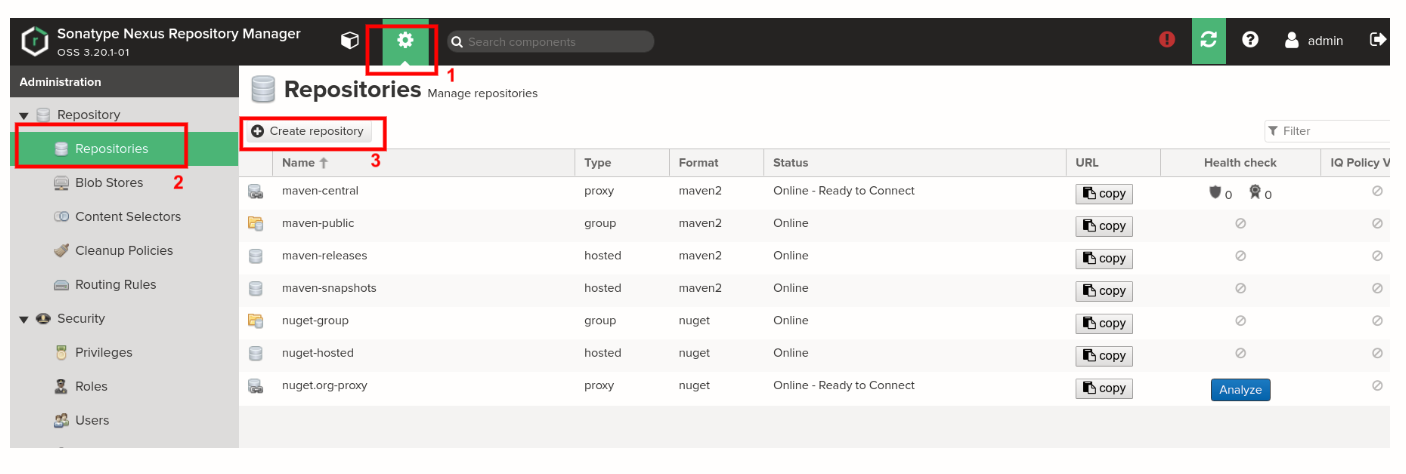
## 

## **Create a Repository in Nexus**

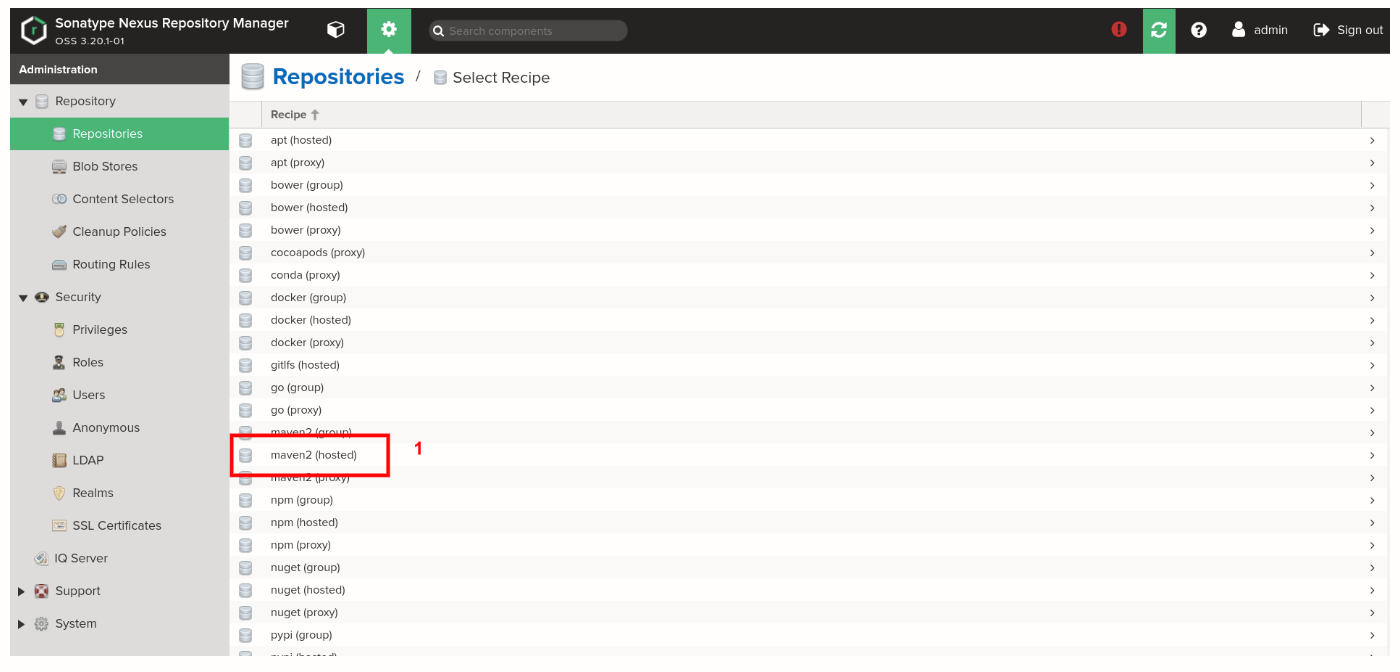
In this step, you are going to create a Maven Hosted repository in Nexus, where your Jenkins is going to upload “build” artifacts.

**Step 1:**

Follow the below-mentioned steps to create a hosted repository, name it maven-nexus-repo, which you are going to use throughout this guide.



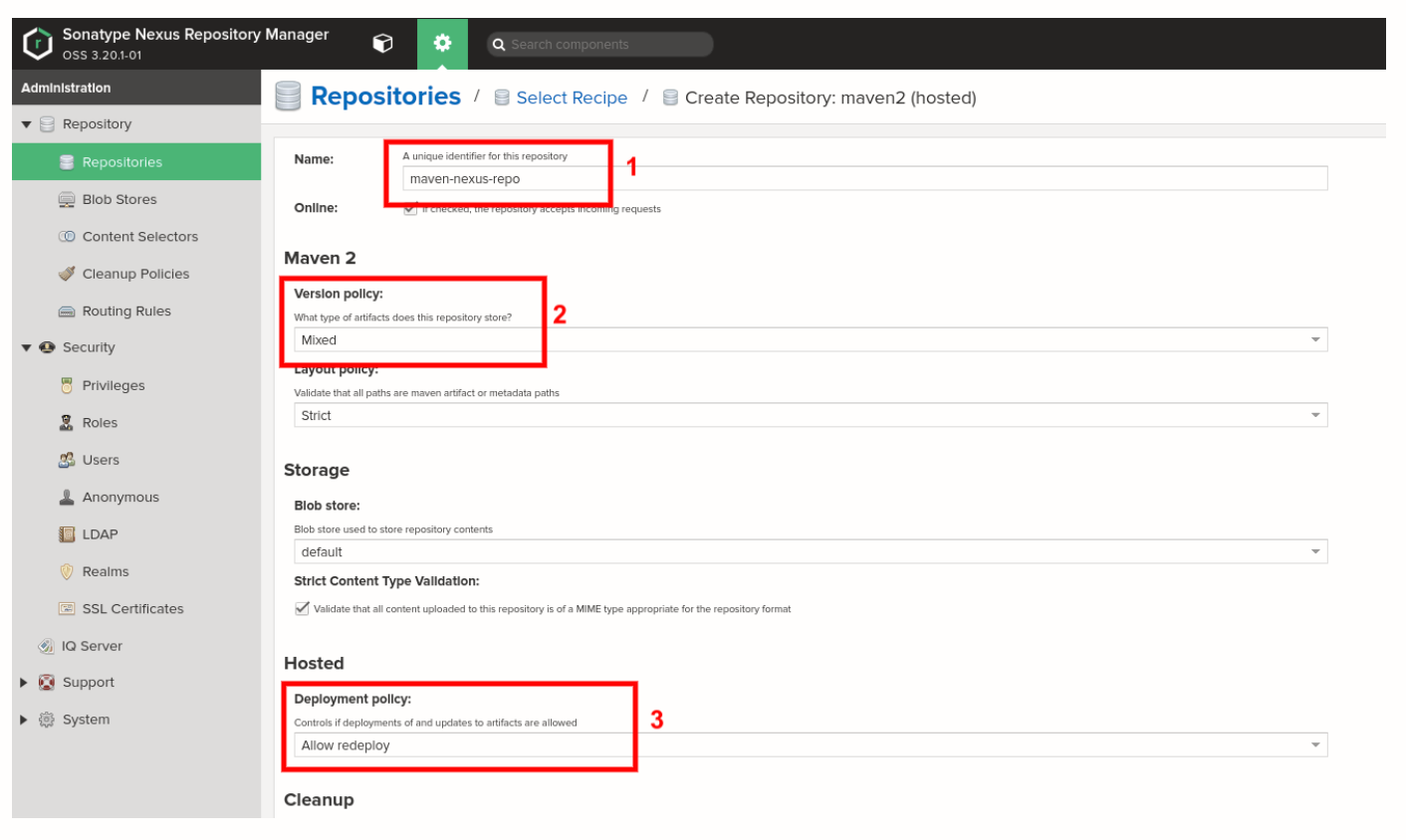
Select *maven2 (hosted)* recipe from the list as shown below:



**Step 2:**

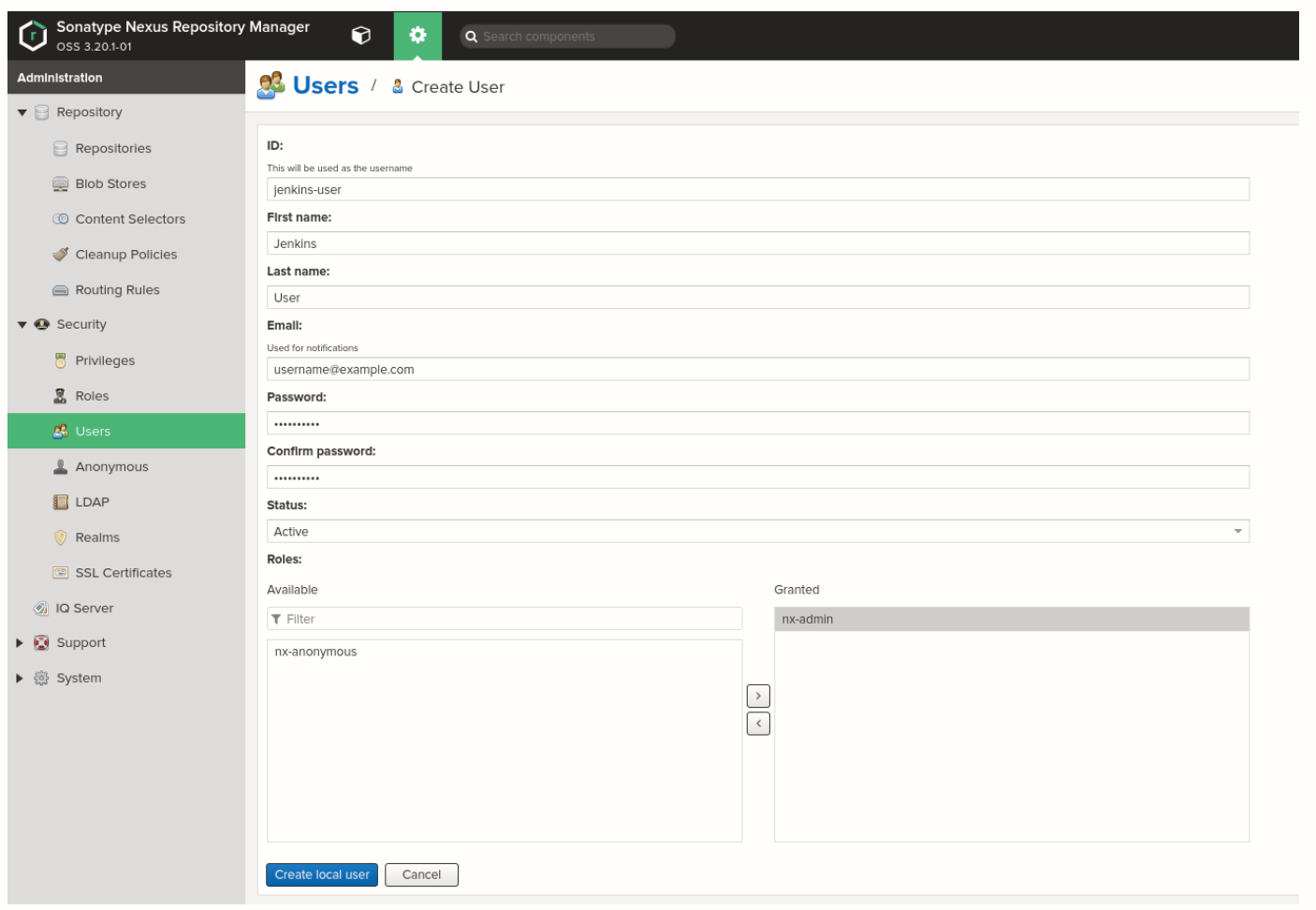
On the *Create Repository* page

* Enter the name as *maven-nexus-repo*
* In Version Policy, select the *Mixed* type of artifacts.
* Under the *Hosted* section, in *Deployment policy*, select *Allow redeploy*. It will allow you to deploy an application multiple times.



**Step 3:**

To create a new user, go to *Dashboard > Server Administrator and Configuration > User > Create user*. Select *Local* user type which happens to be the default Realm:



In the *Create User* page

1. **ID**: Enter the desired ID; in our case, it is jenkins-user.
2. **First Name**: Enter the desired first name; in our case, it is Jenkins.
3. **Last Name**: Enter the desired second name; in our case, it is User.
4. **Email**: Enter your email address.
5. **Status**: Select *Active* from your drop-down menu.
6. **Roles**: Make sure that you grant the *nx-admin* role to your user.

If you want more user creation details, [click here](https://help.sonatype.com/repomanager3/security/users?ref=hackernoon.com).

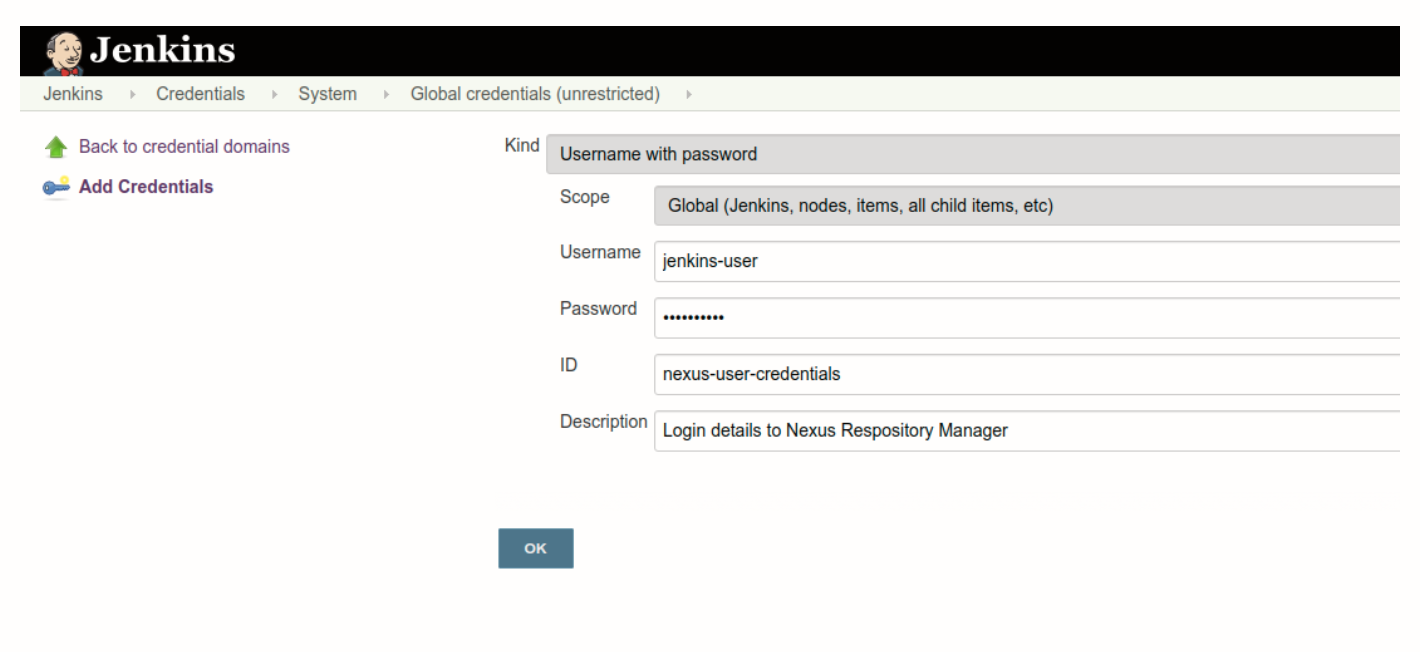
With this, we are through with the setup part of Nexus Repository Manager. Let us move to Jenkins to set up Nexus there.

## **Install and Configure Nexus Plugins in Jenkins**

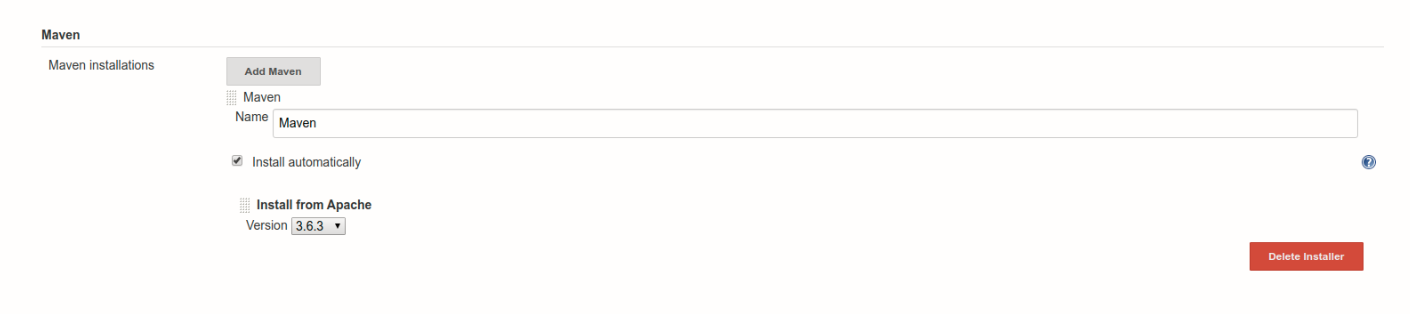
Here you are going to install and configure a few plugins for Nexus in Jenkins. For this, go to Jenkins and then *Dashboard > Manage Jenkins > Manage Plugins > Available* and search and install [Nexus Artifact Uploader](https://plugins.jenkins.io/nexus-artifact-uploader?ref=hackernoon.com) and [Pipeline Utility Steps](https://plugins.jenkins.io/pipeline-utility-steps?ref=hackernoon.com).

Add Nexus Repository Manager’s user credentials in Jenkins.

Go to *Dashboard > Credentials > System > Global credentials (unrestricted)*, as shown below:



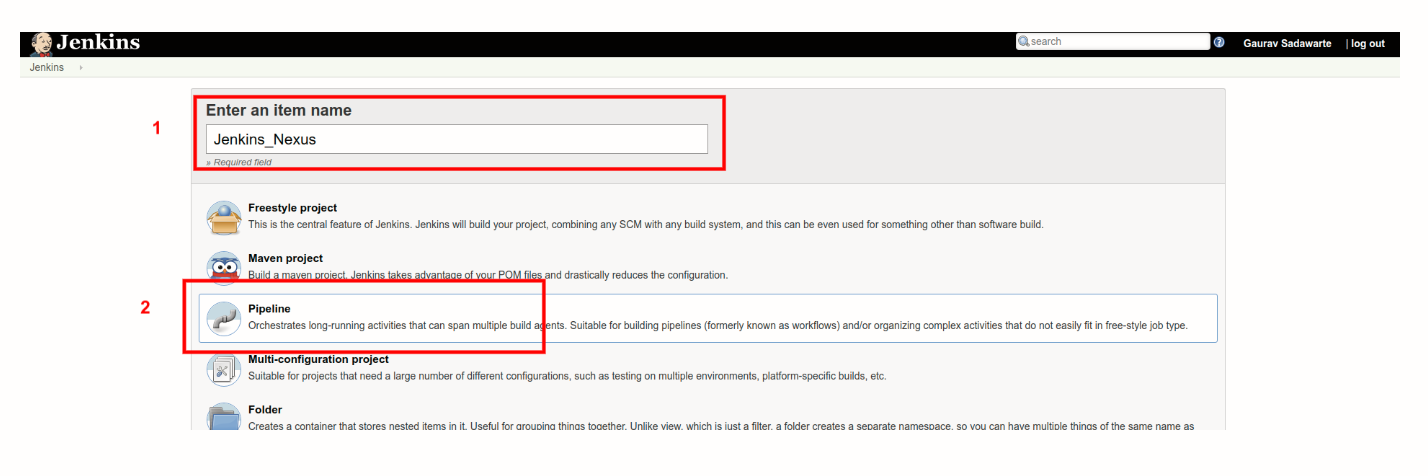
Next, set up Maven as a managed tool. Go to *Dashboard > Manage Jenkins > Global Tool Configuration* and find *Maven*. Under this section, click on the *Maven Installations* button and add *Maven*, as shown below:



Alternatively, you can also install the Maven binary directly to your container on the /var/jenkins\_home directory.

## **Create a Jenkins Pipeline**

It’s time to create a Jenkins Job. Here you are going to use Pipeline job type, named as *JenkinsNexus*, as shown below:



In the next page, find the Pipeline section and copy the below-mentioned script in the text area:

pipeline {

agent {

label "master"

}

tools {

maven "Maven"

}

environment {

NEXUS\_VERSION = "nexus3"

NEXUS\_PROTOCOL = "http"

NEXUS\_URL = "you-ip-addr-here:8081"

NEXUS\_REPOSITORY = "maven-nexus-repo"

NEXUS\_CREDENTIAL\_ID = "nexus-user-credentials"

}

stages {

stage("Clone code from VCS") {

steps {

script {

git 'https://github.com/javaee/cargotracker.git';

}

}

}

stage("Maven Build") {

steps {

script {

sh "mvn package -DskipTests=true"

}

}

}

stage("Publish to Nexus Repository Manager") {

steps {

script {

pom = readMavenPom file: "pom.xml";

filesByGlob = findFiles(glob: "target/\*.${pom.packaging}");

echo "${filesByGlob[0].name} ${filesByGlob[0].path} ${filesByGlob[0].directory} ${filesByGlob[0].length} ${filesByGlob[0].lastModified}"

artifactPath = filesByGlob[0].path;

artifactExists = fileExists artifactPath;

if(artifactExists) {

echo "\*\*\* File: ${artifactPath}, group: ${pom.groupId}, packaging: ${pom.packaging}, version ${pom.version}";

nexusArtifactUploader(

nexusVersion: NEXUS\_VERSION,

protocol: NEXUS\_PROTOCOL,

nexusUrl: NEXUS\_URL,

groupId: pom.groupId,

version: pom.version,

repository: NEXUS\_REPOSITORY,

credentialsId: NEXUS\_CREDENTIAL\_ID,

artifacts: [

[artifactId: pom.artifactId,

classifier: '',

file: artifactPath,

type: pom.packaging],

[artifactId: pom.artifactId,

classifier: '',

file: "pom.xml",

type: "pom"]

]

);

} else {

error "\*\*\* File: ${artifactPath}, could not be found";

}

}

}

}

}

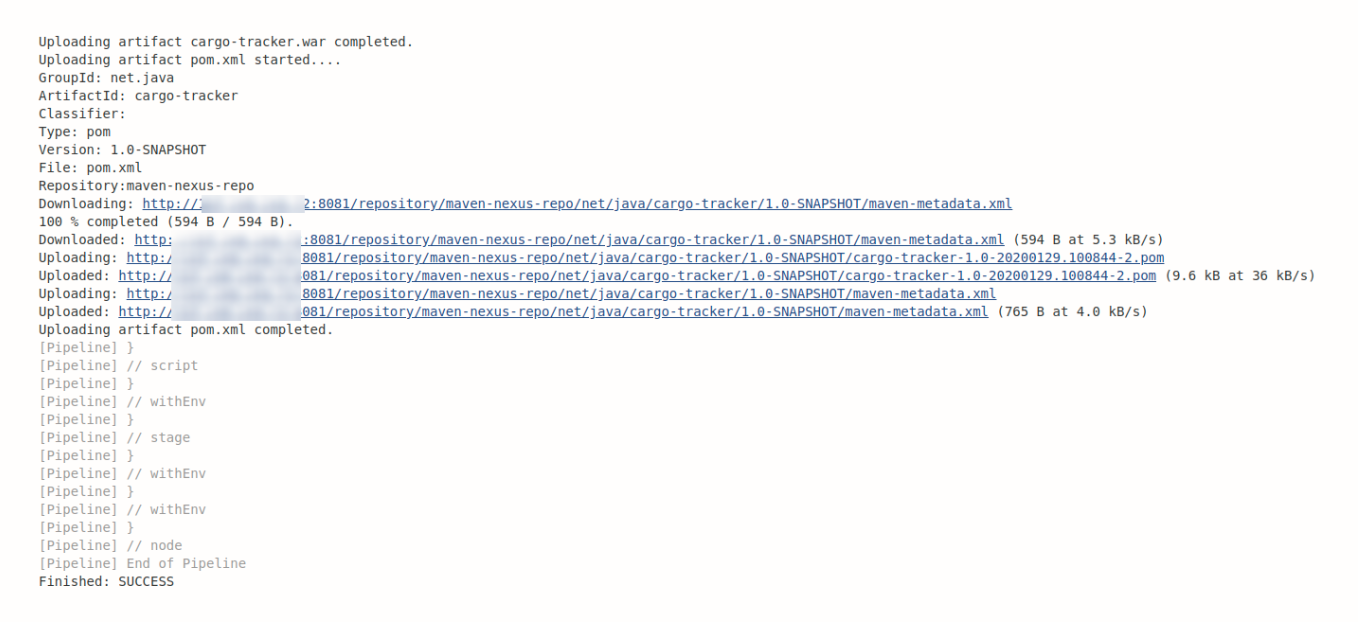
}

Let’s break down the above-mentioned parameters bit by bit:

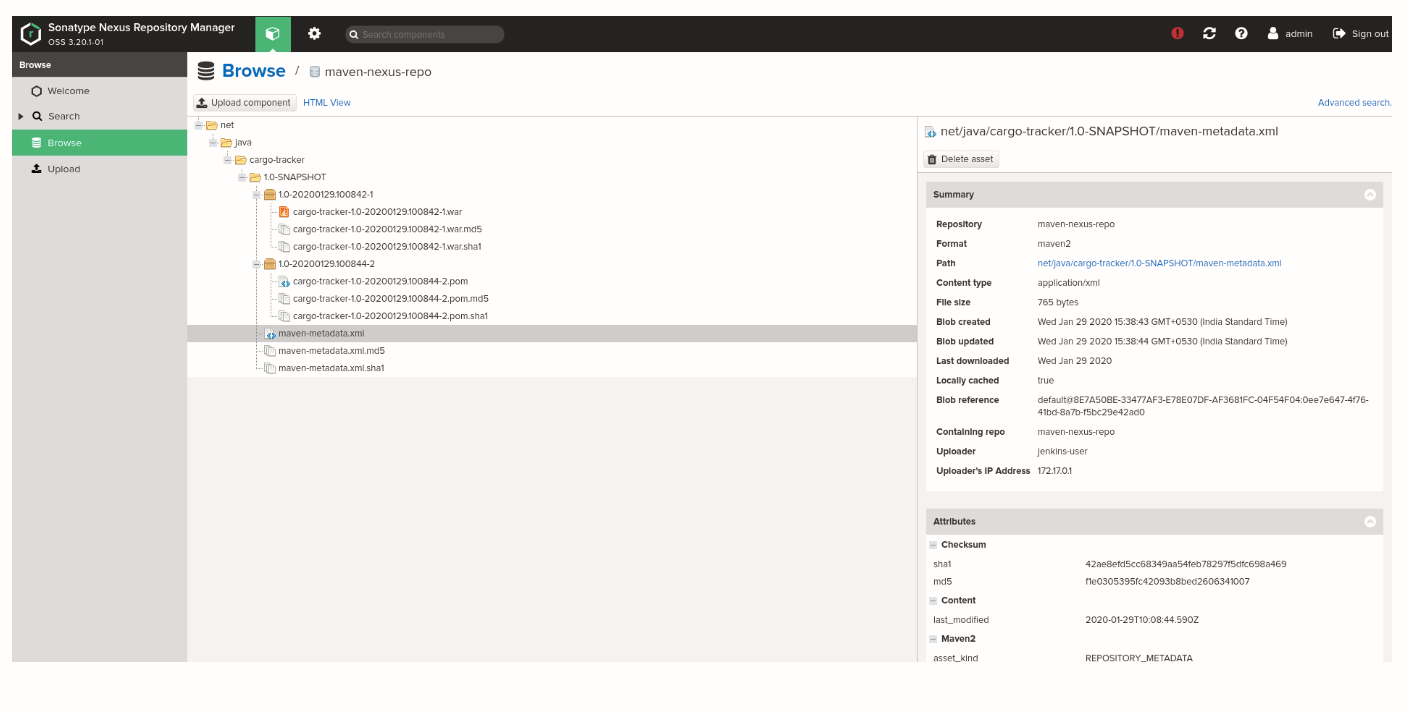
* **NEXUS\_VERSION**: Here, we have to mention the exact version of Nexus, which can be nexus2or nexus3 . In our case, it is latest version of nexus3 .
* **NEXUS\_PROTOCOL**: For this guide we have used HTTP protocol, although, in case of production, you will have to use HTTPS.
* **NEXUS\_URL**: Add your IP address and port number, where you are running Nexus. Make sure that you add Nexus instance details without mentioning protocols, i.e., https or http .
* **NEXUS\_CREDENTIAL\_ID**: Enter the user ID, which you previously created in Jenkins, which in our case is nexus-user-credentials .
* **Project Git**: Under stages, we used https://github.com/javaee/cargotracker

As you are through with the Pipeline set up, it’s time to Build our project. Go to the *JenkinsNexus* project job page and click *Build Now*. As this is your first build, it is going to take some time, so sit tight.

Once build is a success, in your Jenkins Console Output, you will see something like this:



Whereas, in your Nexus Repository Manager, you would see something similar to this:



**Conclusion**

A systematic way to distribute the project’s artifacts is critical for any organization. With the help of Jenkins Pipeline and Nexus Repository Manager, you can centralize artifact repo, which ultimately reduces the efforts of reproducing build time as well as switching CI tools without worrying about migrating our artifacts.