**Kubernetes Deployment using Jenkins**

**Praveen Kumar.M**

172-172, 5th floor Old Mahabalipuram Road

(Above Axis Bank-PTC Bus Stop)  
Thuraipakkam  
Chennai 600097

**🖂**[**praveenkumarm373@gmail.com**](mailto:praveenkumarm373@gmail.com)

**✆+91 8778697197**

# Contents

* Overviews
* Flows
* Requirements

I. Jenkins

II. Maven

III. Docker

IV. Docker Image

V. Dockerhub

VI. Kubernetes

* Jenkins Local Machine Prerequisites
* Kubernetes Local Machine Prerequisites
* Jenkins Dashboard Prerequisites
* Jenkins start build
* Result

# Overview

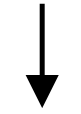
This article will guide you how I took the source code from the git and by using the Jenkins how the binary file is build in the target directories. I have install the Docker and also created the dockerhub account and in local repository I have logged in. Create the Dockerfile and build the image. After the build I have tagged the build and push it to the Dockerhub. Then in kubenetes master I have wrote the yaml file to pull the docker image and I have deployed and started the service. If the service start check it in the browser by giving the node ip and the node port we assigned.

Note:

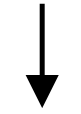
Source code url: <https://github.com/seedstack/store-webapp-sample>

# Flow

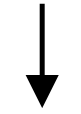
**Git**



**Jenkins**



**Kubernetes**



**Deployment**

# Requirments

Three centos minimal machine one with 2 cpu, 4GB ram and 50GB storage and remaining two with 1 cpu, 2GB ram and 30GB storage. One machine is for jenkins and another one for kubernetes master and last one act as agent for the kubernetes. Finish all the installation and keep the three machine ready.

Jenkins

Jenkins is an open-source continuous integration software tool written in the Java programming language for testing and reporting on isolated changes in a larger code base in real time. The software enables developers to find and solve defects in a code base rapidly and to automate testing of their builds.

Continuous integration has evolved since its conception. Originally, a daily build was the standard. Now, the usual rule is for each team member to submit work on a daily (or more frequent) basis and for a build to be conducted with each significant change. When used properly, continuous integration provides various benefits, such as constant feedback on the status of the software. Because CI detects deficiencies early on in development, defects are typically smaller, less complex and easier to resolve.

Maven

Maven is a build tool, in short a successor of ant. It helps in build and version control. However Jenkins is continuous integration system, where in maven is used for build. Jenkins can be used to automate the deployment process.

Docker

Docker is an open source software platform to create, deploy and manage virtualized application containers on a common operating system (OS), with an ecosystem of allied tools.

Docker Image

A Docker image is a file, comprised of multiple layers, used to execute code in a Docker container. When the Docker user runs an image, it becomes one or multiple instances of that container. Docker is an open source OS-level virtualization software platform primarily designed for Linux and Windows.

Dockerhub

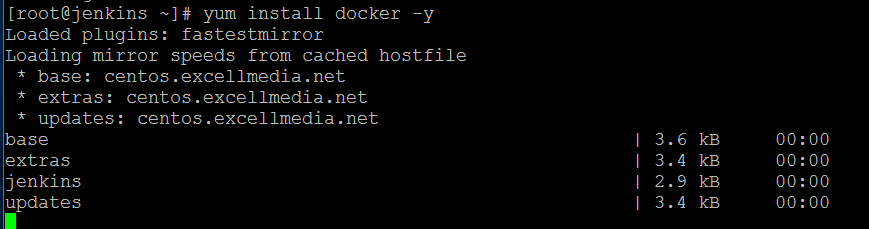
Docker Hub is a cloud-based repository in which Docker users and partners create, test, store and distribute container images. Through Docker Hub, a user can access public, open source image repositories, as well as use a space to create their own private repositories, automated build functions, webhooks and work groups

Kubernetes

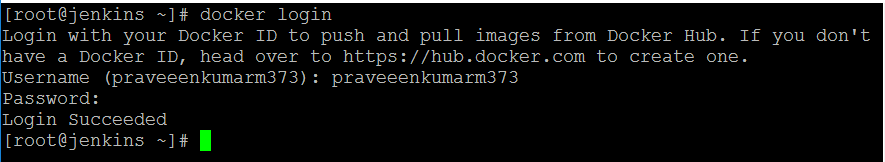
Kubernetes is a portable, extensible open-source platform for managing containerized workloads and services that facilitates both declarative configuration and automation. It has a large, rapidly growing ecosystem. Kubernetes services, support, and tools are widely available.

# Step-1 Jenkins Local Machine Prerequisites

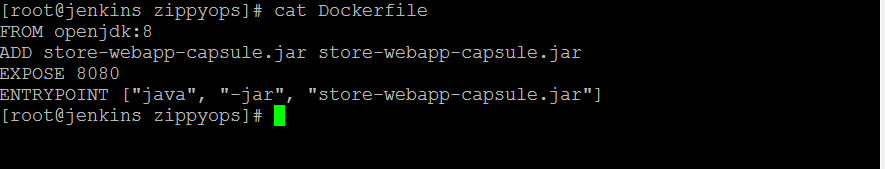
* Install the latest version of Docker.we where installing the Docker for building the image and to push the image to dockerhub.



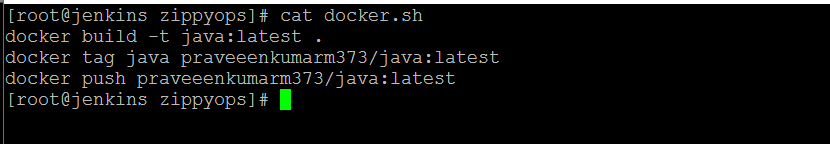
* Install the latest version of Docker.we where installing the Docker for building the image and to push the image to dockerhub. Login with your Docker ID to push and pull Images from Docker Hub. If YOU don’t have a Docker ID, Head over to https://hub.docker.com to create one.



* Create the **Dockerfile** and write the content given below.
* In that **FROM openjdk:8** pull the image from the cloud-based repository.
* **Add \*.jar \*.jar** is used to add the binary file inside the image.
* **EXPOSE 8080** is to run the image in 8080 port.
* **ENTRYPOINT** is once we run the image the given command will excute automatically

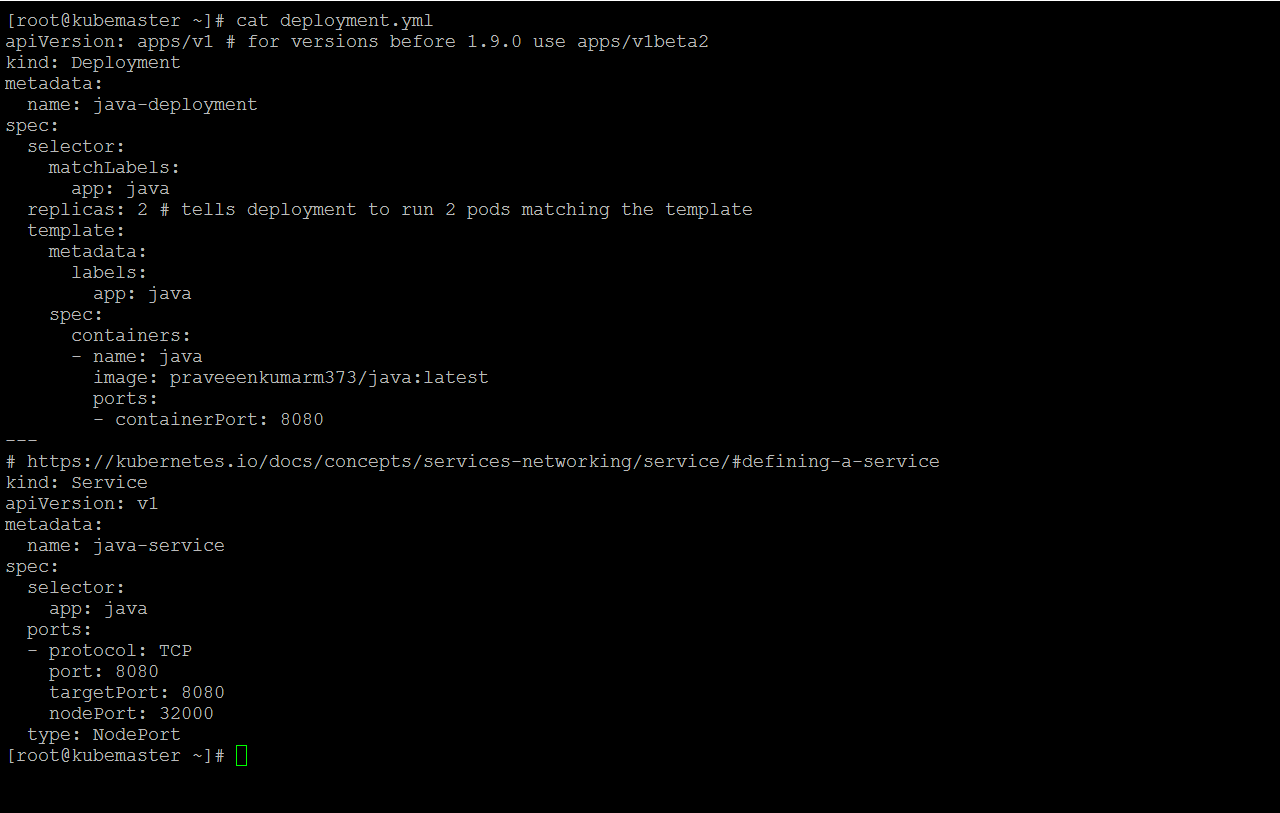


* Create the docker.sh and write the content given below.
* First command is used to build the image that we have writen in the Dockerfile
* Second command is used to tag the image you have build
* Third command is used to push the image to the cloud-based repository.

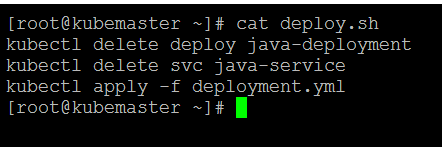


# Step-2 Kubernetes Local Machine Prerequisites

* Create the deployment.yml and write the content given below.
* In this yaml file we are going to pull the docker image from our dockerhub account which we have pushed
* And in this yml file we are going to deploy the image which we pull from the dockerhub and to start the service to run the image
* Replicas is used to scaling up the image
* We are changing the target port to node port

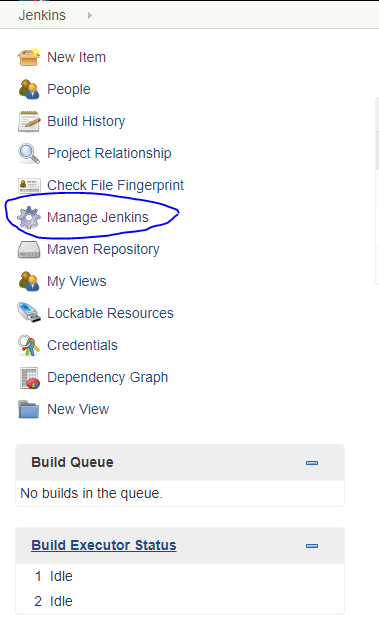


* Create the deploy.sh and write the content given below.
* First command is used to delete the deployment if any running in the same exiting name.
* Second command is used to delete the service if any running in the same exiting name.
* Third command is used to create the new deployment and new service.

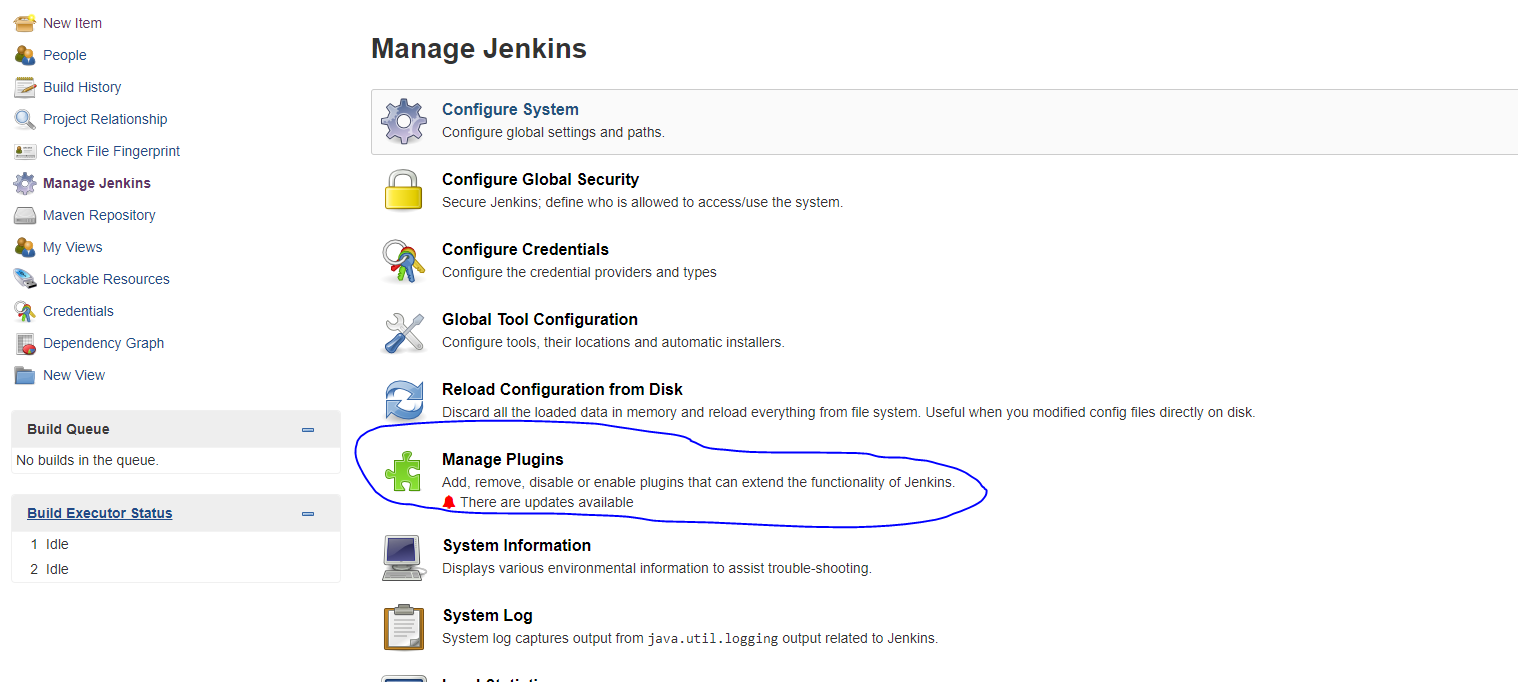


# Step-3 Jenkins Dashboard Prerequisites

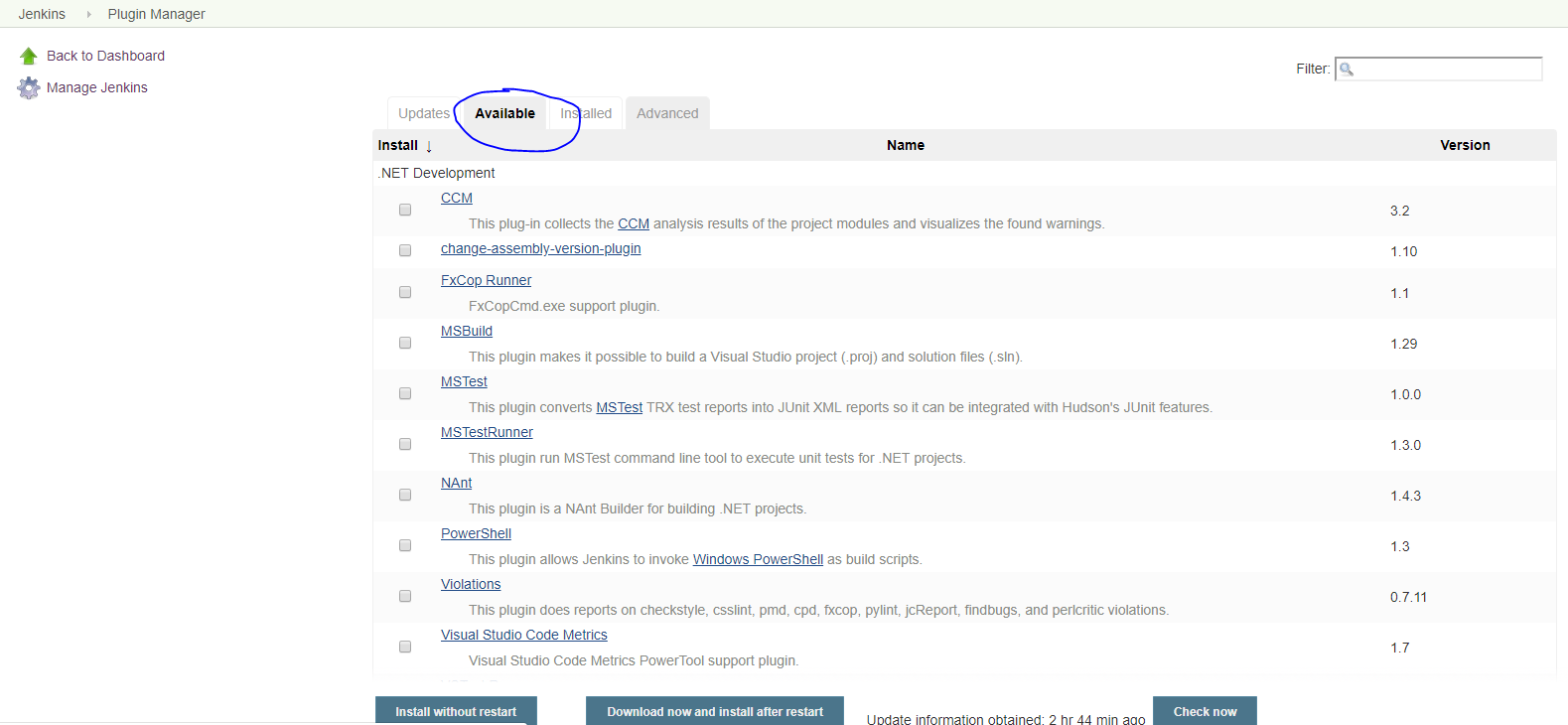
* Open the jenkins Dashboard and click the manage jenkins the set the environment for the project building process.



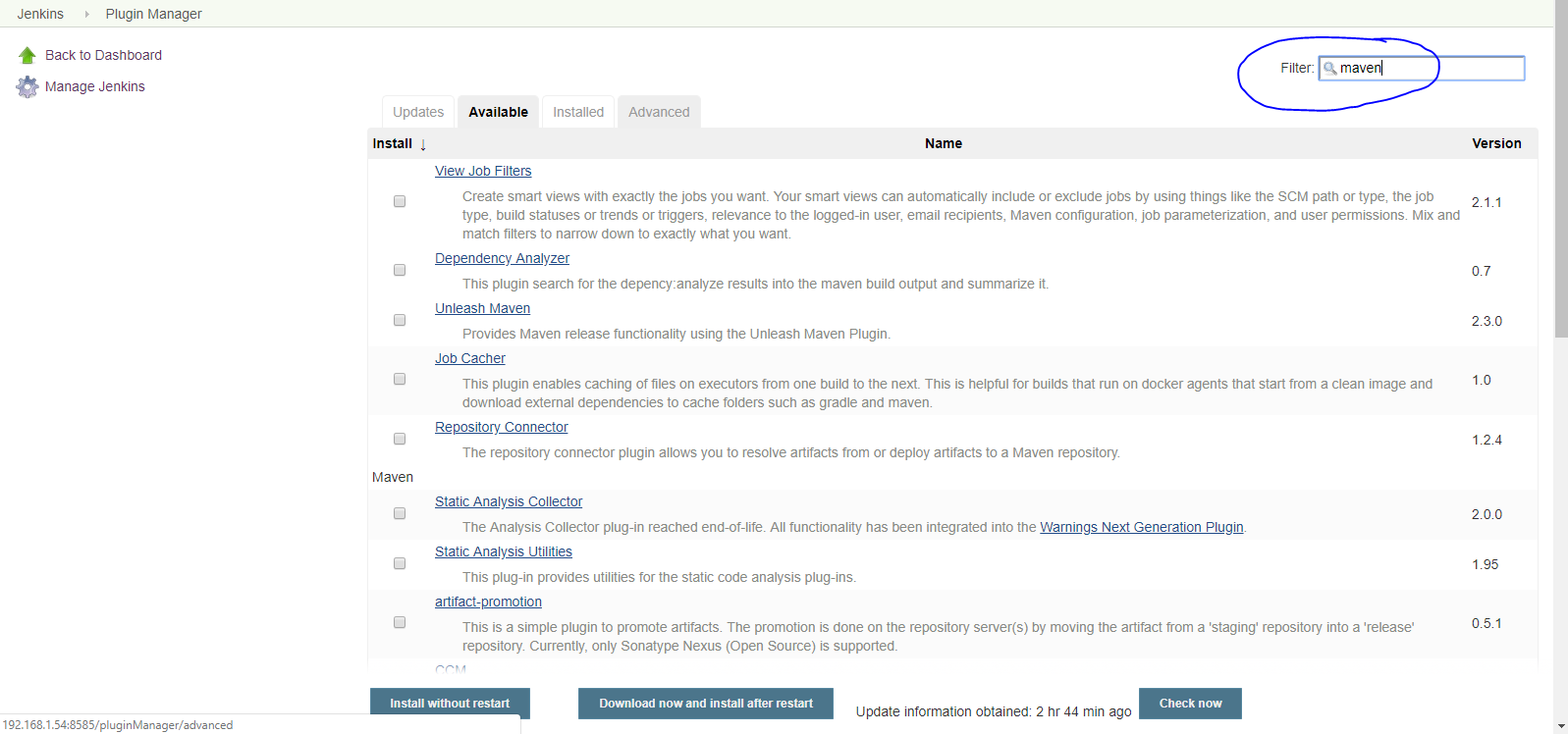
* First click the manage plugins and in manage plugins only we will get the plugins for an connectivity that for the task we going to do.

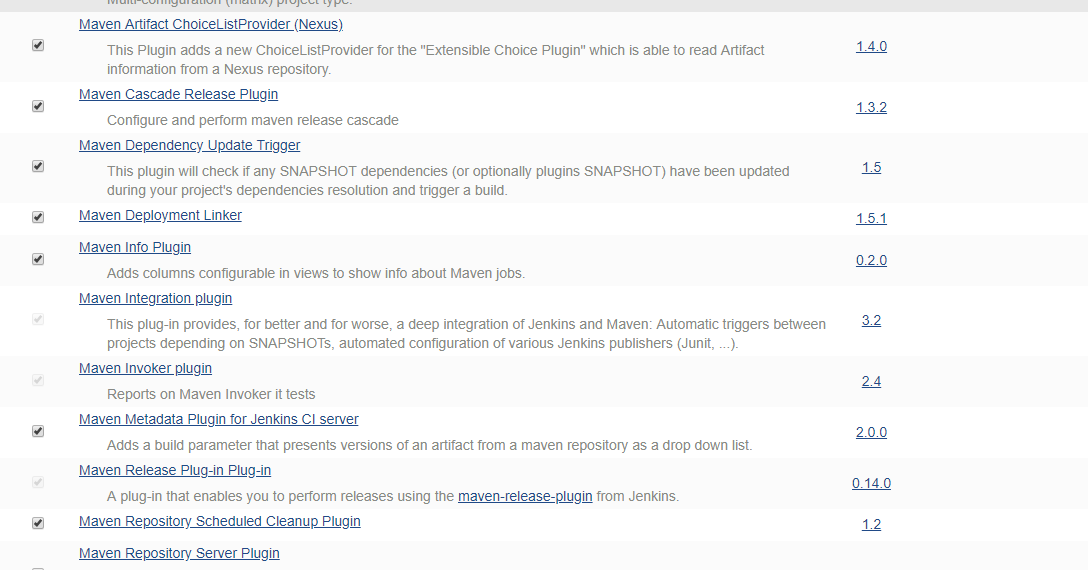


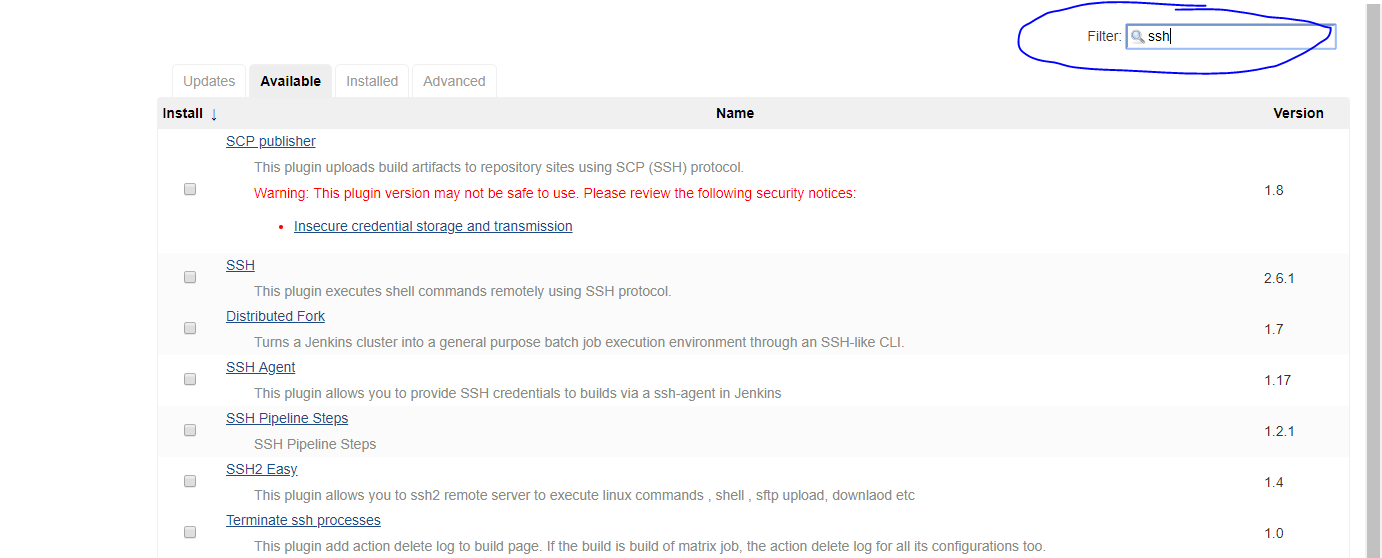
* Click the available because the only you will get the available plugins in Jenkins

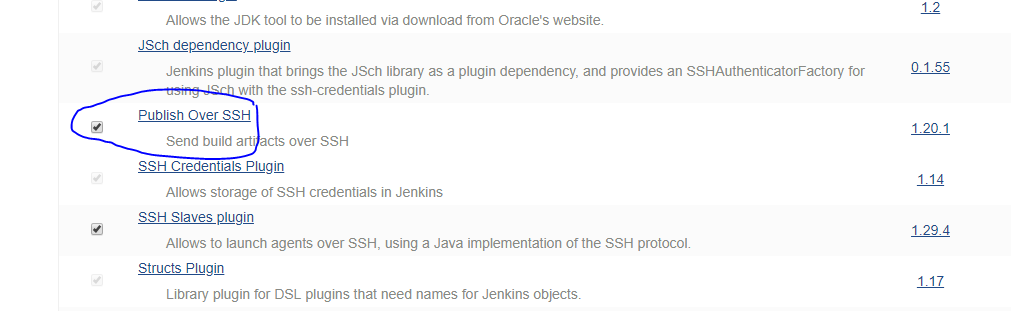


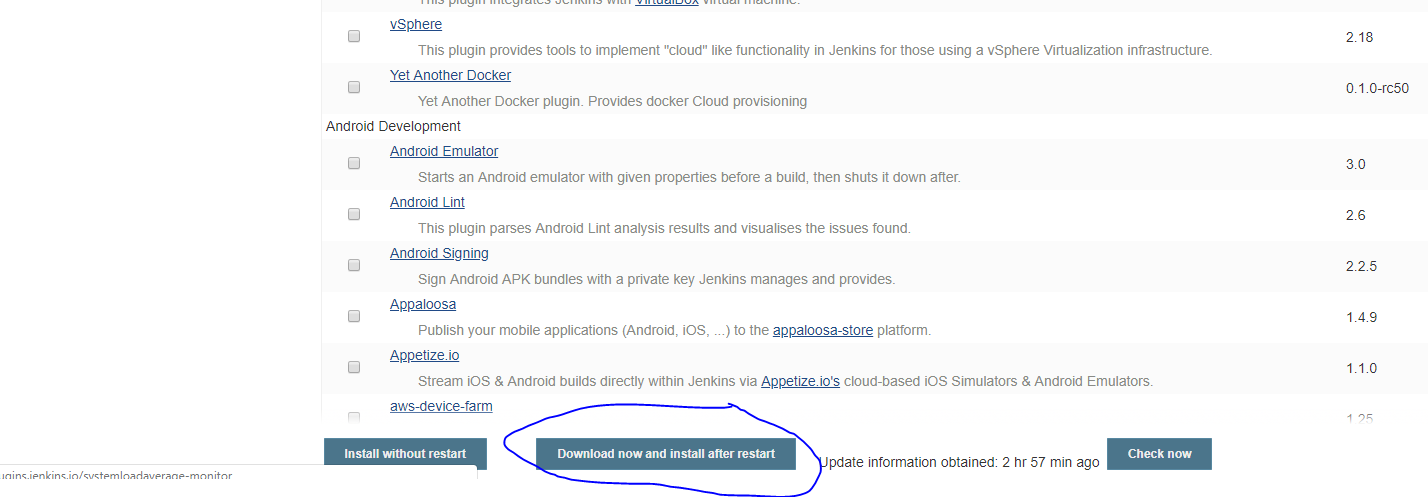
* Search the maven and ssh plugins that you have want for the task
* And select what all are the plugins you want.
* After selected click download now and install with restart
* Maven plugins is used to run the project as maven projects
* SSH plugins is used to connect the node by ssh connectivity to run the execute command in the local machine



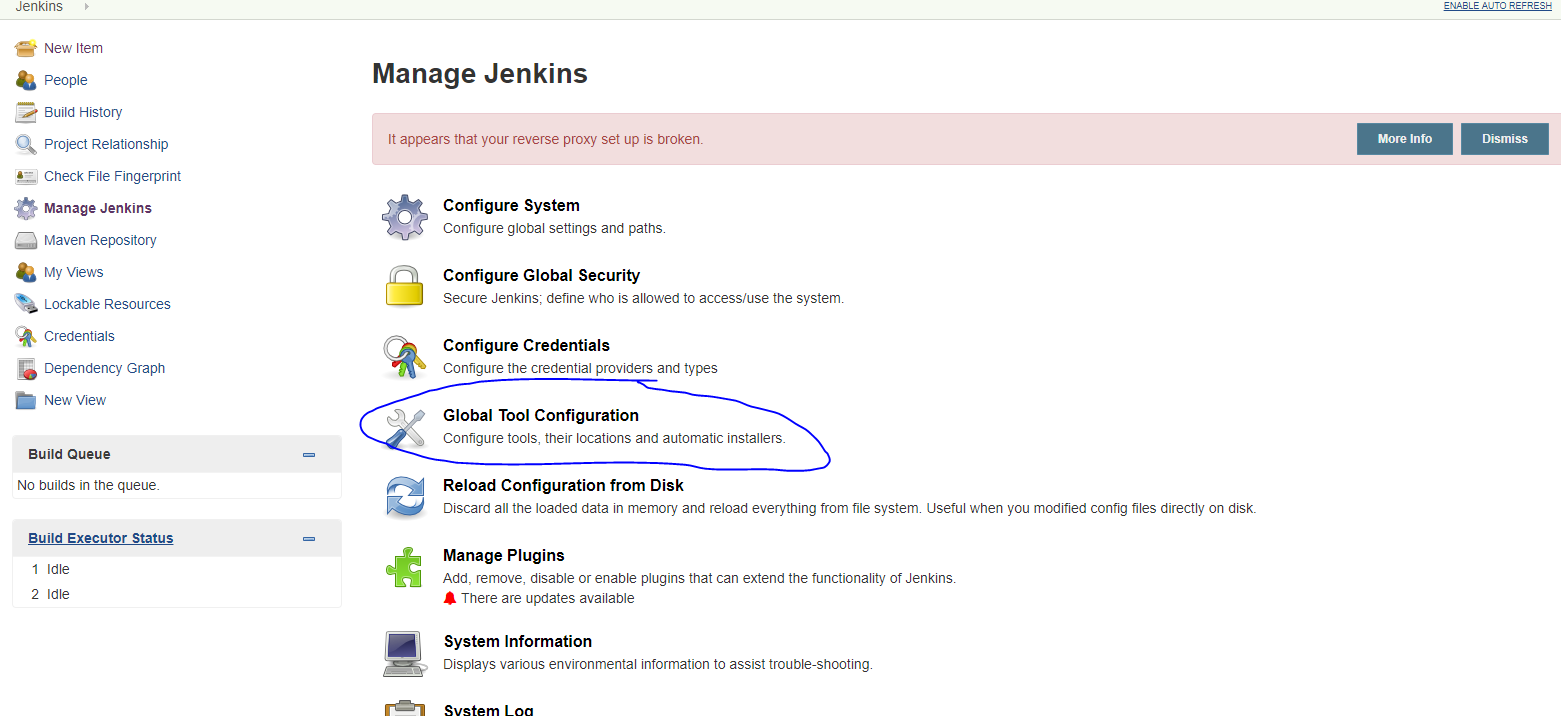
****



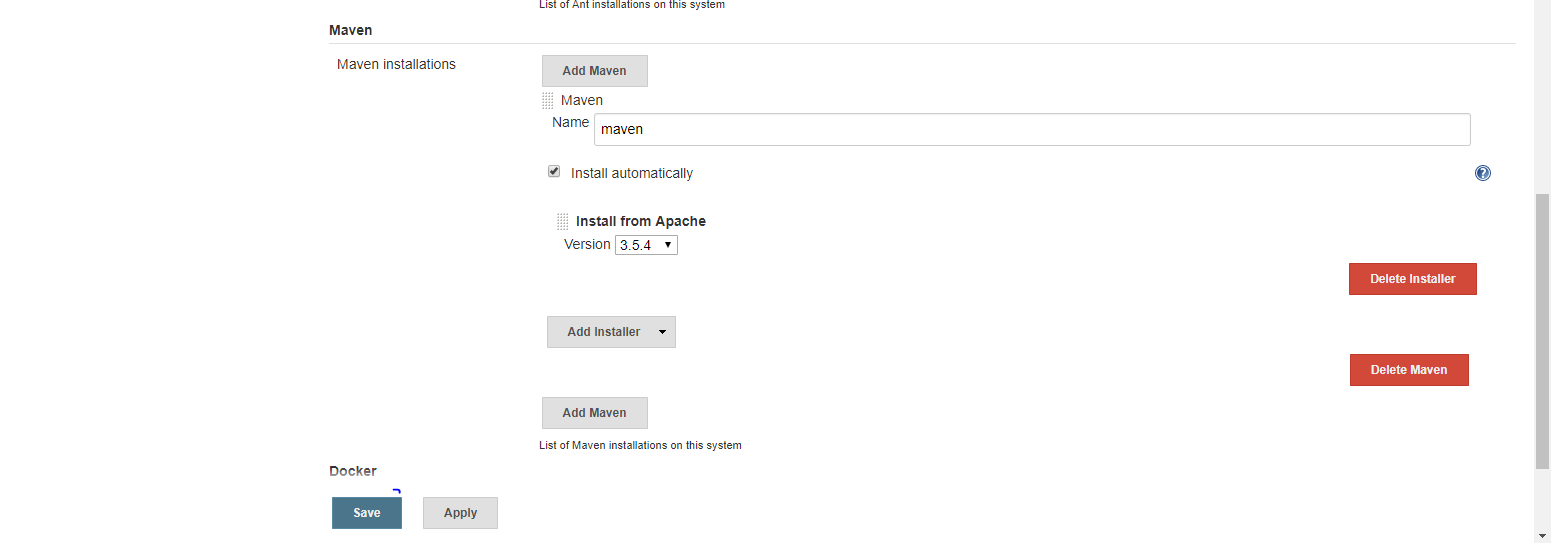
****

****

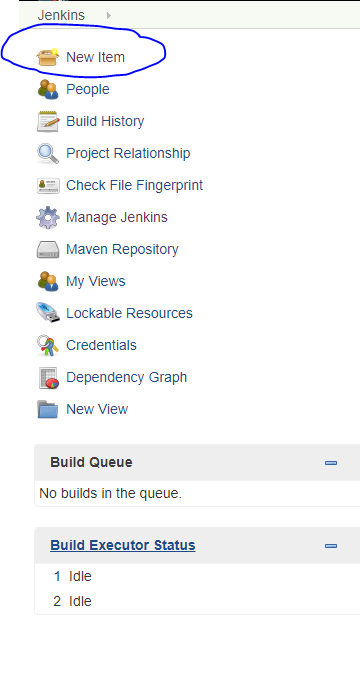
* After finishing the installation then click the Global tool configuration which is in the **Manage Jenkins**



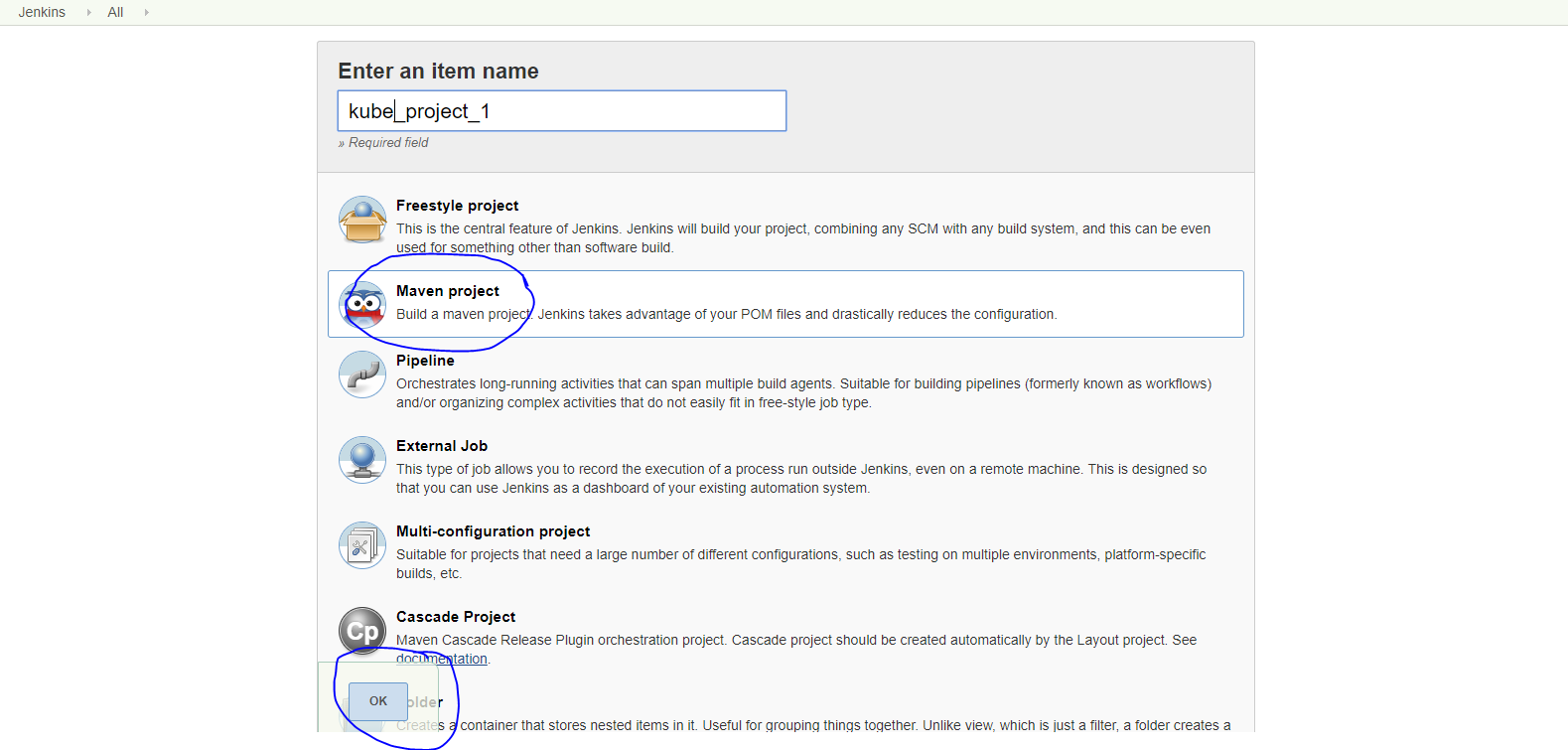
* Here set the maven environment and specify the version in which it want to work and to create the binary file and click save the save the changes.



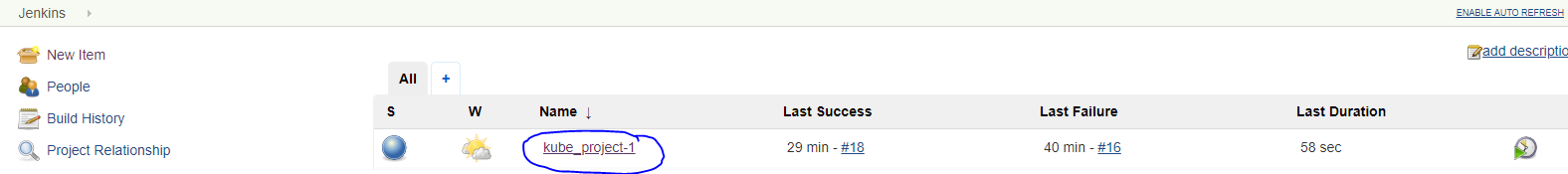
* Click the new item and here we are going to create the new project to build the binary file where the source code available in git.



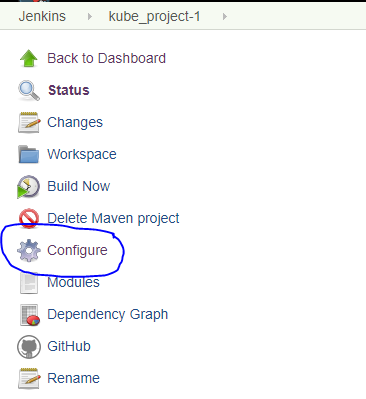
* Give the name kube\_project-1 and then click the maven project and give ok to create the maven project



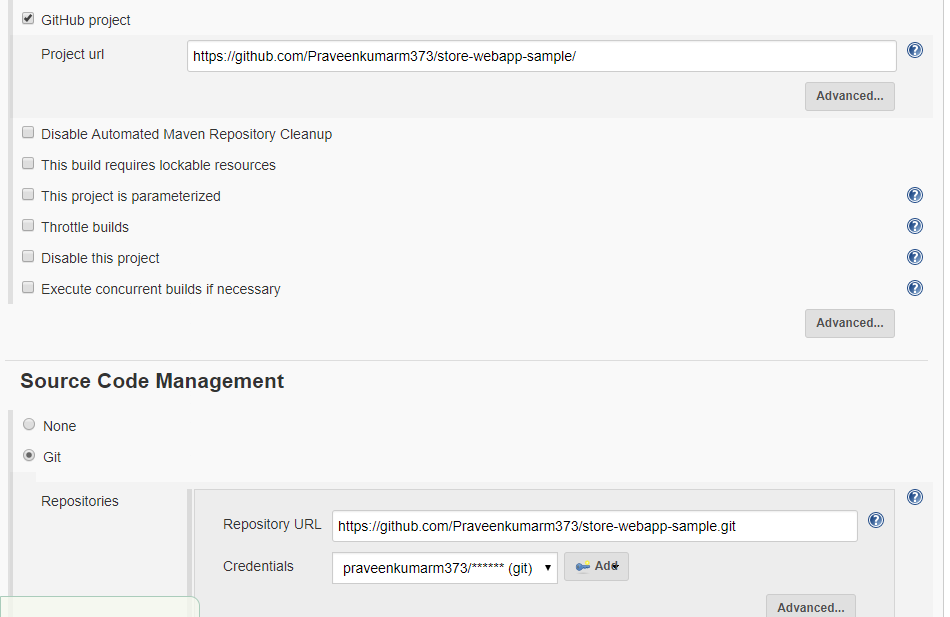
* Here we created the maven project and click on the name kube\_project-1



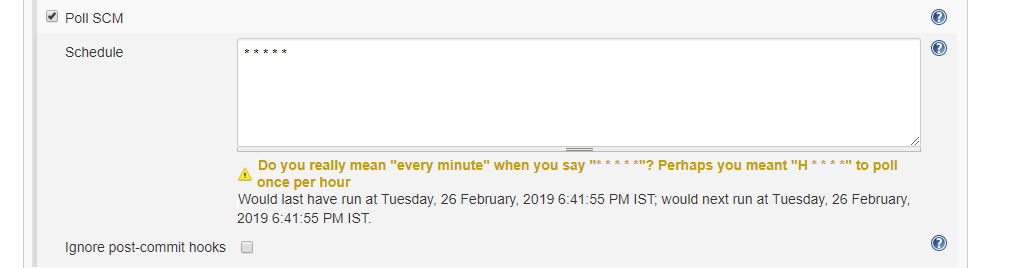
* If we go inside the project the page will be like this. Here click the configure and here only we will set the process that how it want to works.



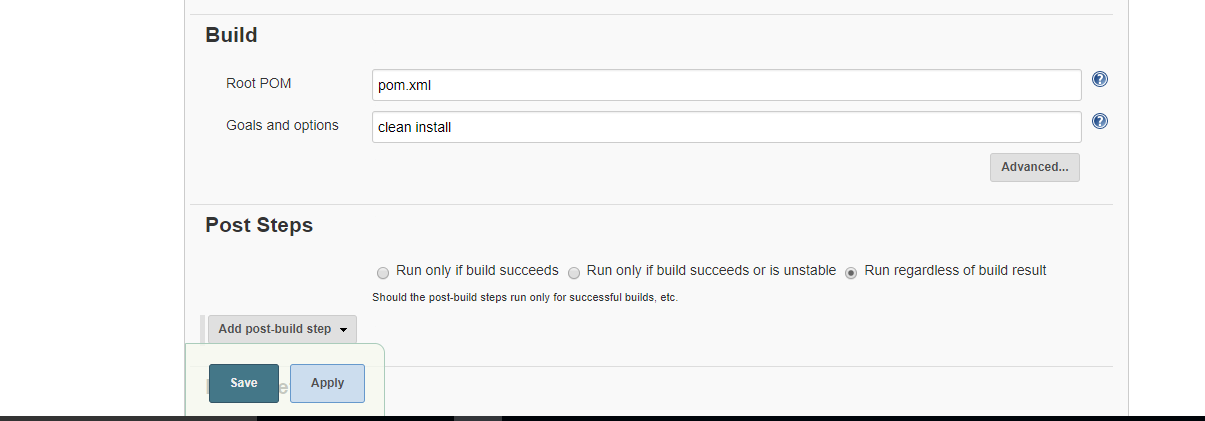
* Inside the configure give the Github URL and the Repository URL of your project that you going to build and in the Credentials add the git username and passwd



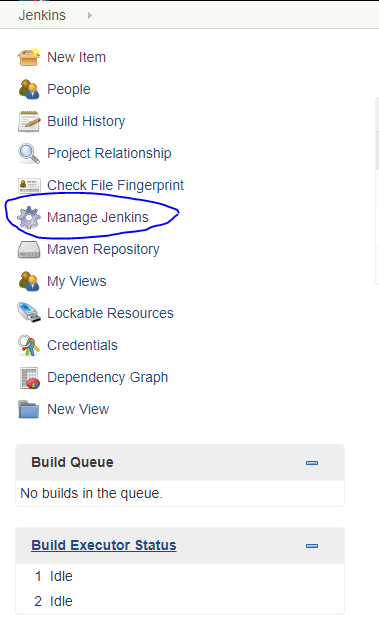
* Poll SCM just give \* \* \* \* \* 5 star to start the build when any commit change given in your Git it will start to build.



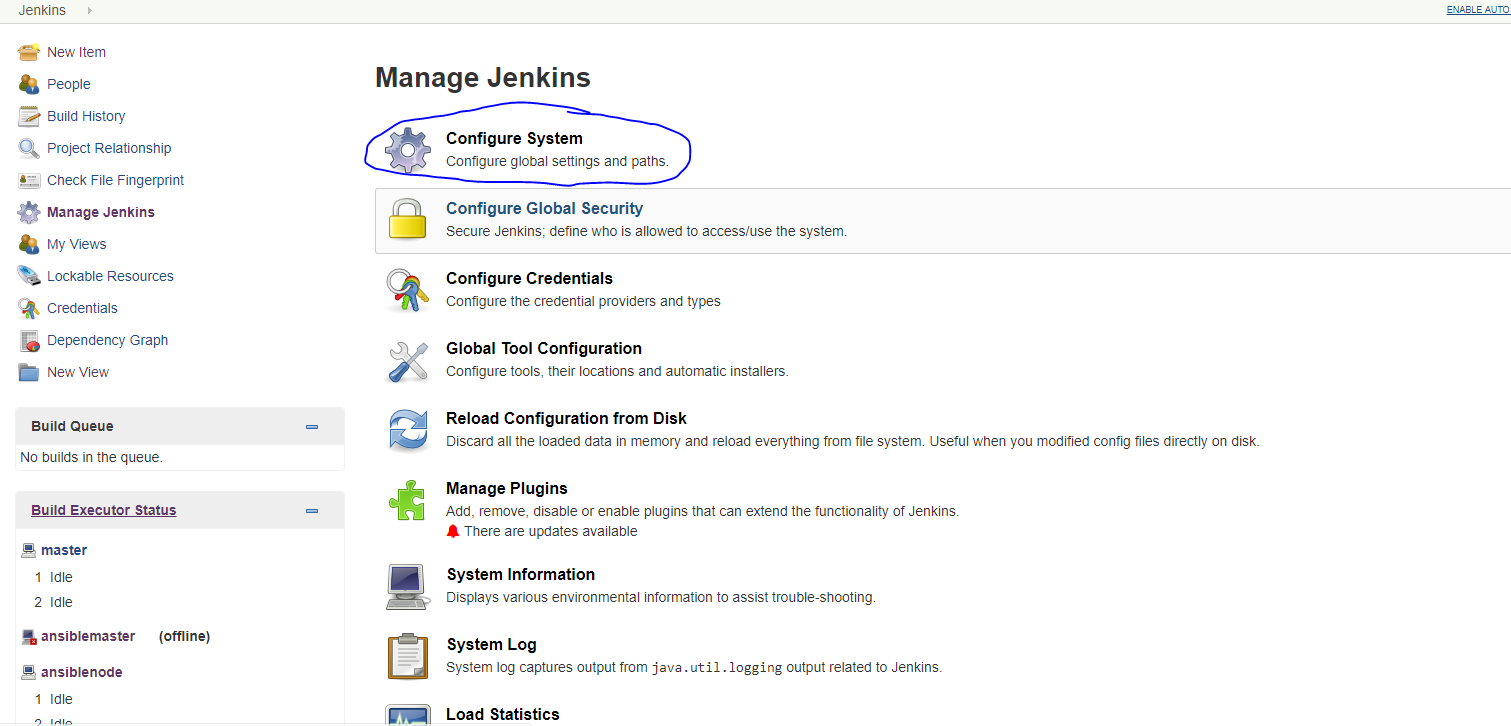
* In Goals and options give the command clean install. The command clean will clean the old build and the install will create the new binary file after the change made.
* If you not use the clean command then the change will not build.
* And click apply and save



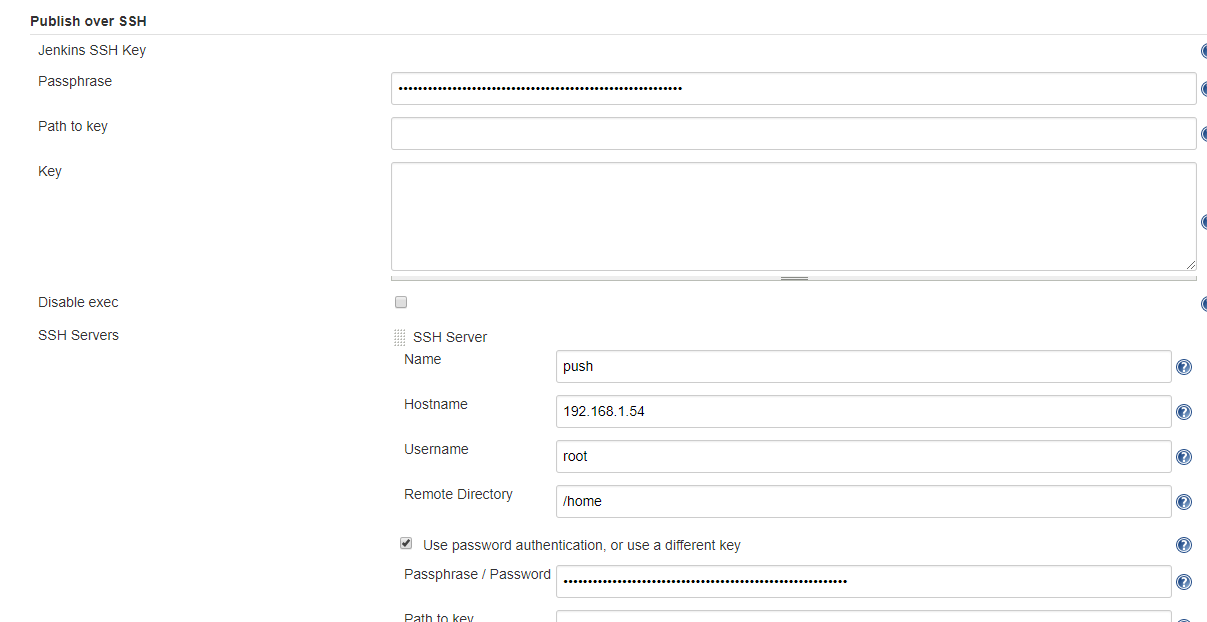
* Now kube\_project-1 set up is over
* Click the manage Jenkins



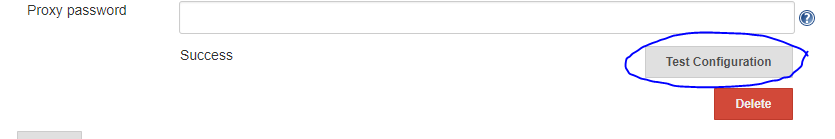
* Go to the configure system for an ssh connectivity



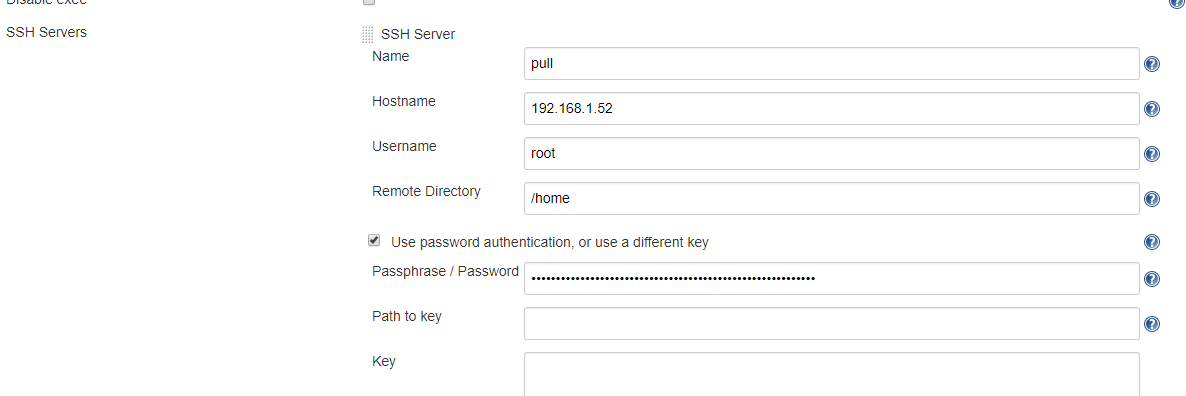
* In the Passphrase give the Jenkins password and click add ssh
* In name any name you can give for your convenient
* In Hostname give the ip of your local machine
* Username give the user of your machine and the remote directory where to command should execute
* Click the **Advance** give the password for the user
* Click the checkbox and type the user passwd



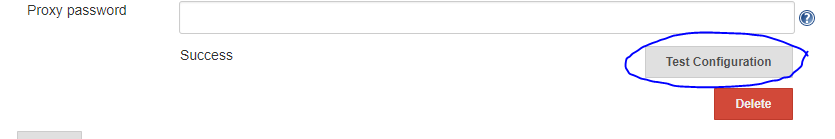
* After completed click the test Configuration to check the connectivity if success arise its ok if any Error occurs check you have mention above all are in correct



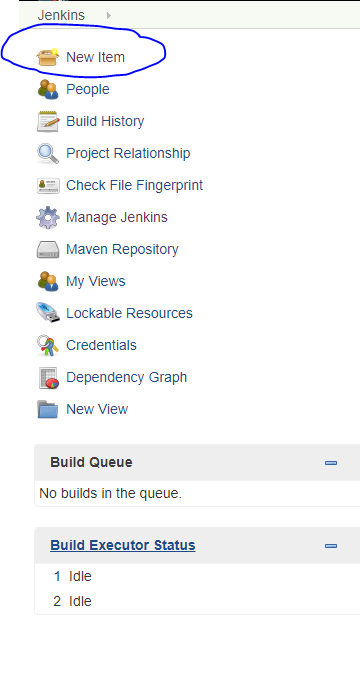
* Then click add SSH to create the another node connectivity



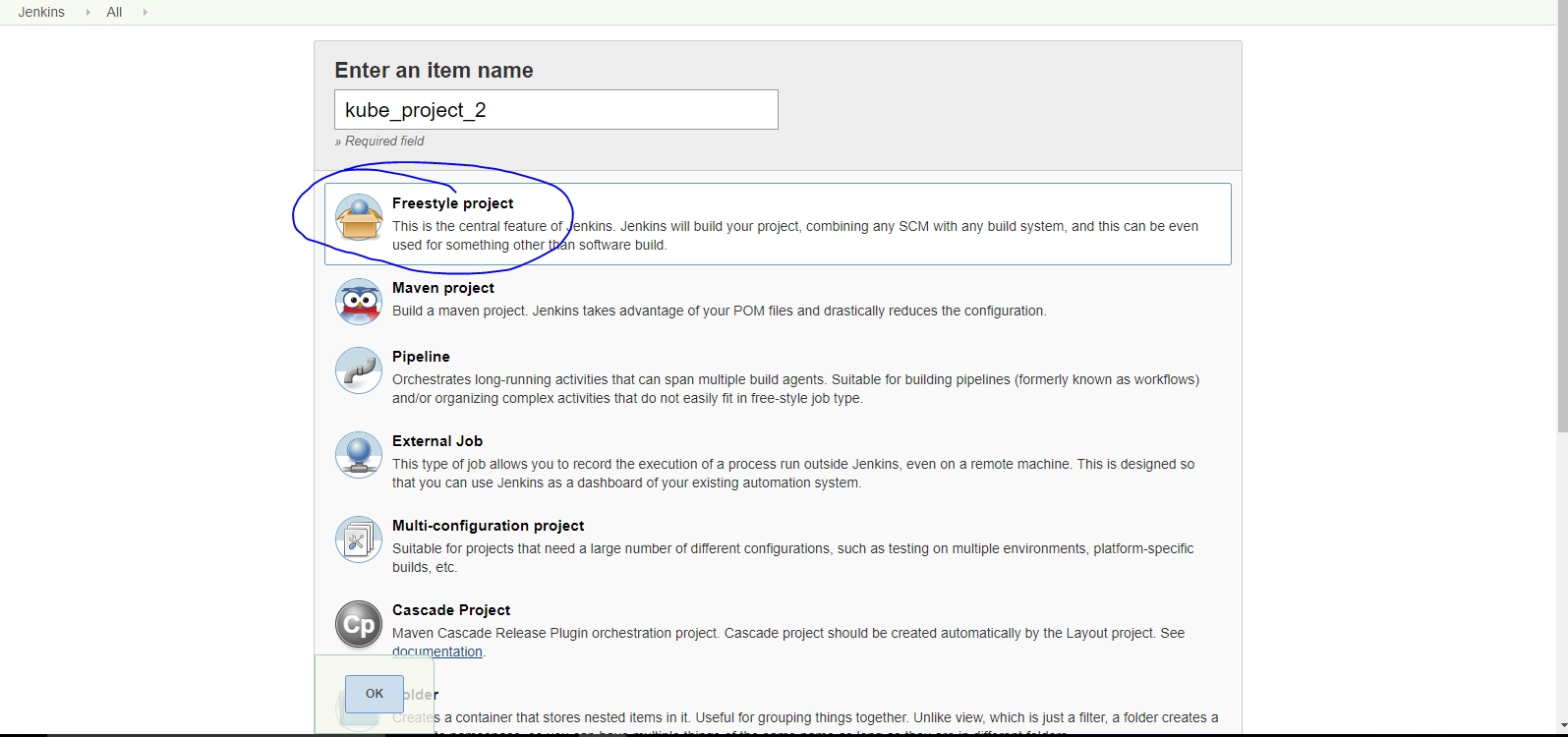
* After that check the ssh connectivity by clicking Test Configuration



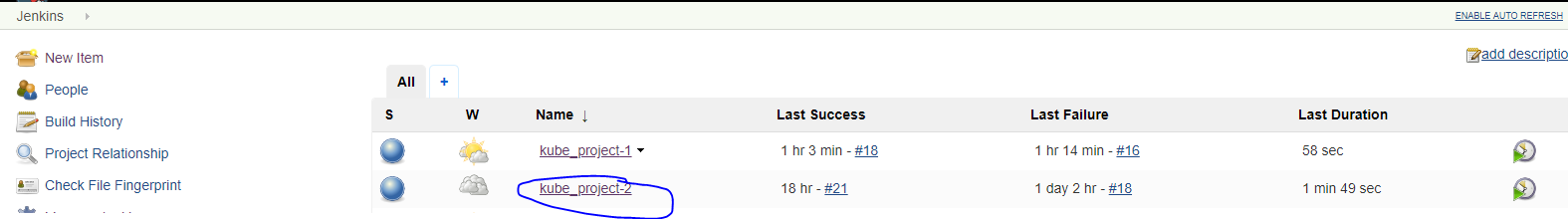
* Click apply and save.
* Create the another project in free style with the name kube\_project-2 by clicking the new item.



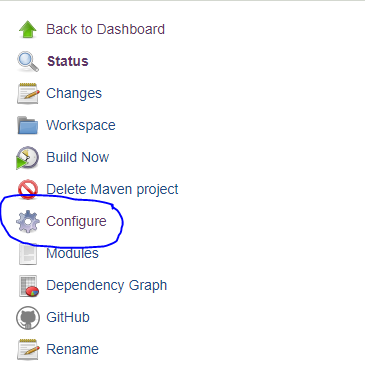
* Give the name kube\_project-2 and then click the free style project and give ok to create the free style project.



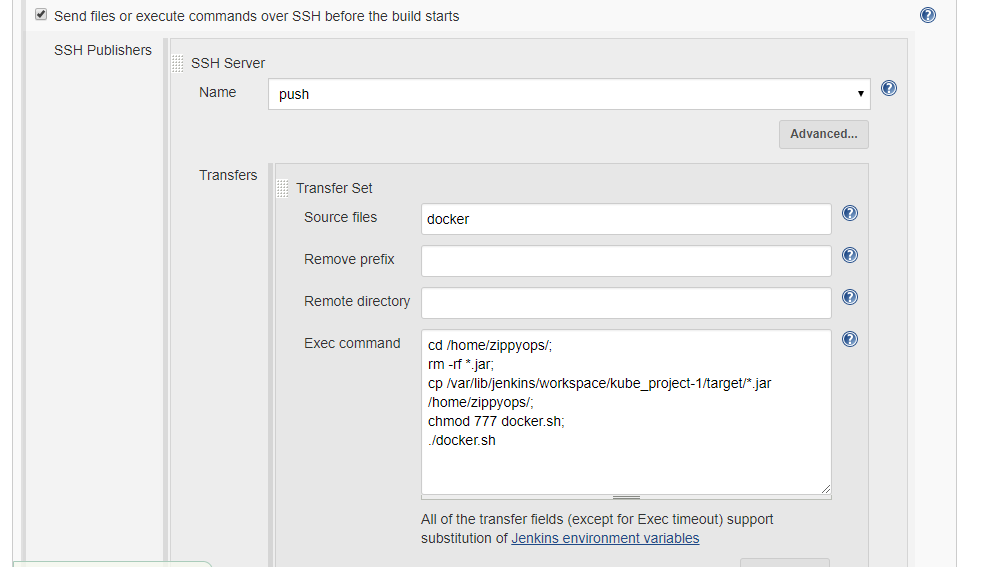
* Click the kube\_project-2 to create the environmental setup just how to perform



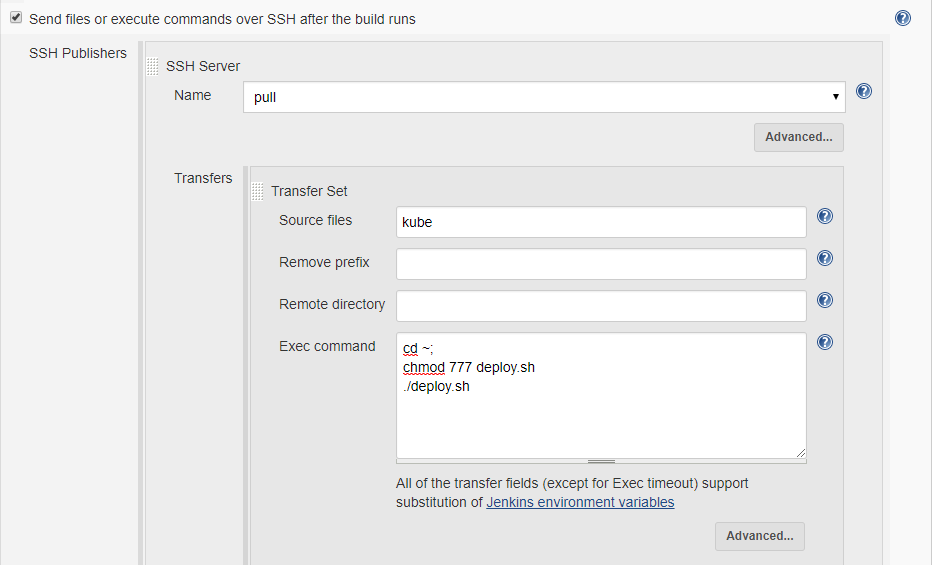
* If we go inside the project the page will be like this. Here click the configure and here only we will set the process that how it want to work.



* Click the Excute commands over SSh before the build start checkbox and mention the name that you have given there in configure system which you want to execute first.
* In source file mention any name.
* In Exec command you have to define what are the commands you want to execute.



* Click the Excute commands over SSh after the build runs checkbox and mention the name that you have given there in configure system which you want to execute second.
* In source file mention any name.
* In Exec command you have to define what are the commands you want to execute.

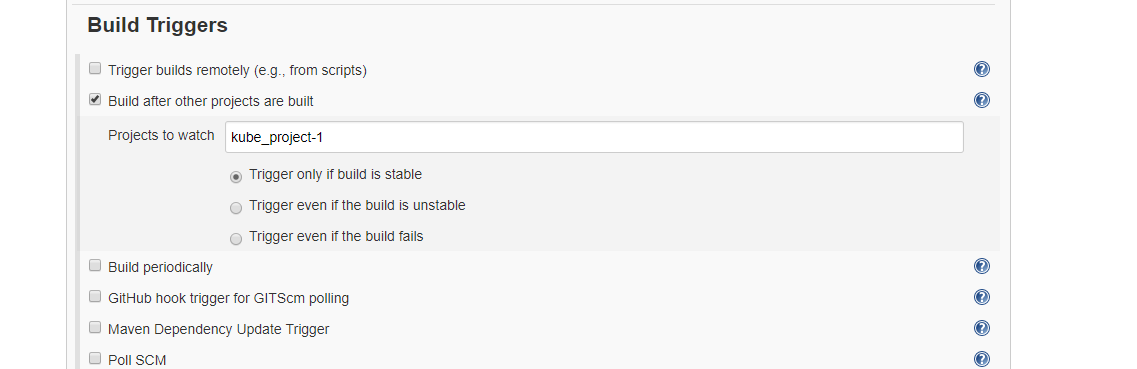


* Click the build after other projects are build checkbox in that mention the project name to make that as upstream.
* In this

**Trigger only build is stable**: If the upstream build is success then only the Downstream project will start.

**Trigger even if the build is unstable**: If the upstream build is even in unstable the Downstream project can start to build.

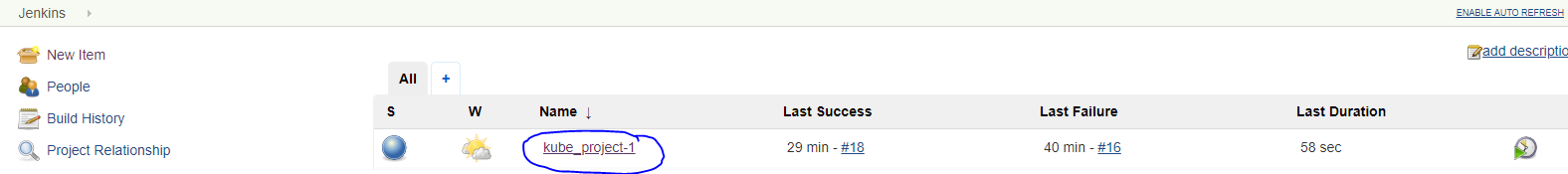
**Trigger even if the build fails:** If the upstream build even fails the Downstream project won’t bother about that it will start to build.



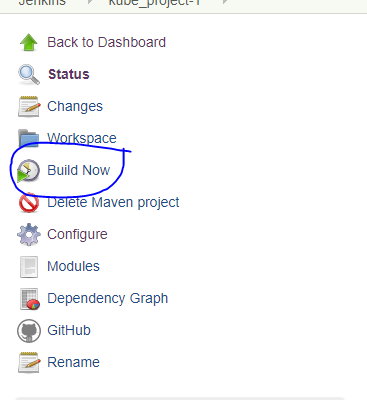
* After that click apply and save.

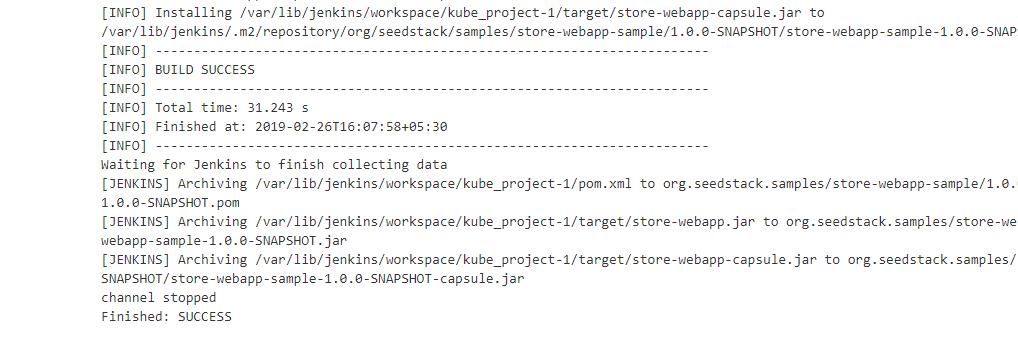
# Step-4 Jenkins start build

* Here click on the project kube\_project-1

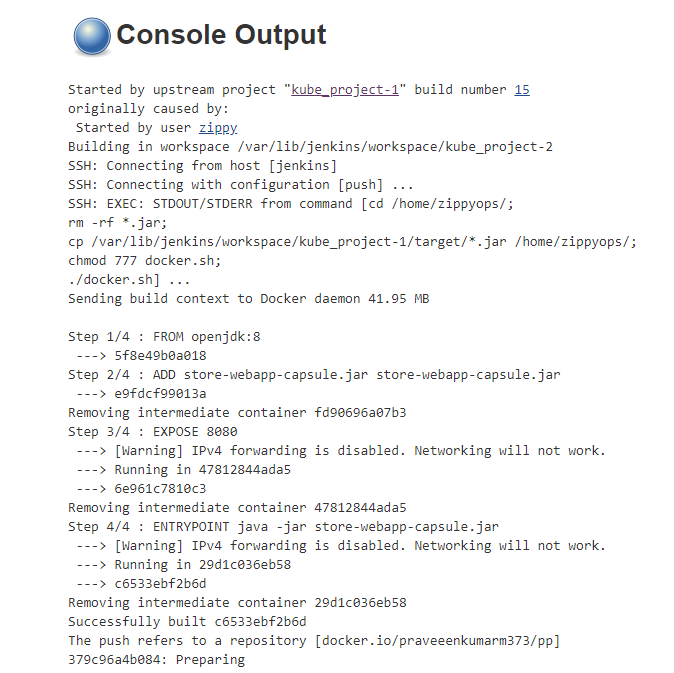


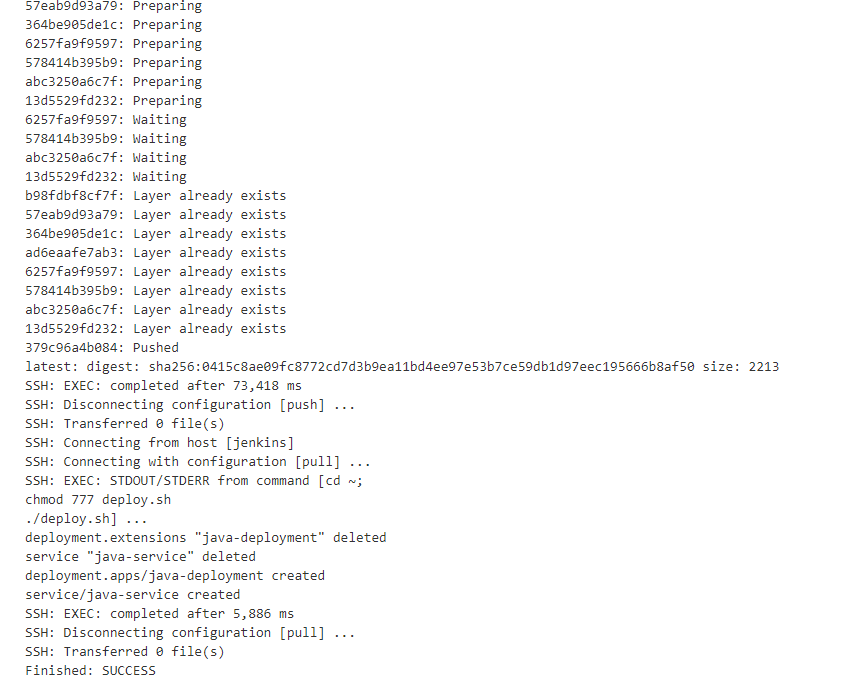
* Click the build now to start the build and the build will be process as per the configuration you have give inside





* If the first project is completed then the second project will get triggered and the second project will automatically start to build





# Step-5 Result

* If the two project get success view the binary file by using the client ip and port 31000 in the browser
* If sign in ask give the username/passwd as demo/demo

