DEVOPS PROJECT

CI/CD PIPELINE

CONTINOUS INTEGRATION AND CONTINOUS DEPLOYMENT

TOOLS: GIT,DOCKER,JENKINS.

Continuous deployment and continuous integration is a process.

Automation is a major one to do the automated work.

Automation-continuous process.

Devops is a methodology.

Devops is a development and operation.

Based on the code we develop the application.

Continuous integration

Bulid the code,test the code if I face the mistake in a code,then test the code,then again test the code if we face any error in the code.(it include the build and test stage) – continous integration happens in build and test stage.

Continuous delivery/ deployment

It focus on release and deploy. It involves the release of code.

Stages

1.developement

2.testing

3.quality assurance staging(pre production)

4.production

<u>If I done the deployment manually then it is a continuous delivery.</u>

Build,test,staging,deploy to a production manually is continuous delivery.

Manual deployment that occurs in production is known as continuous delivery.(deploying upto staging is continous delivery).

Automatic deployment occurs in a production is known as

Continuous deployment.

Jenkins is used for integrate with all other tools.

Jenkins is an open source tools.

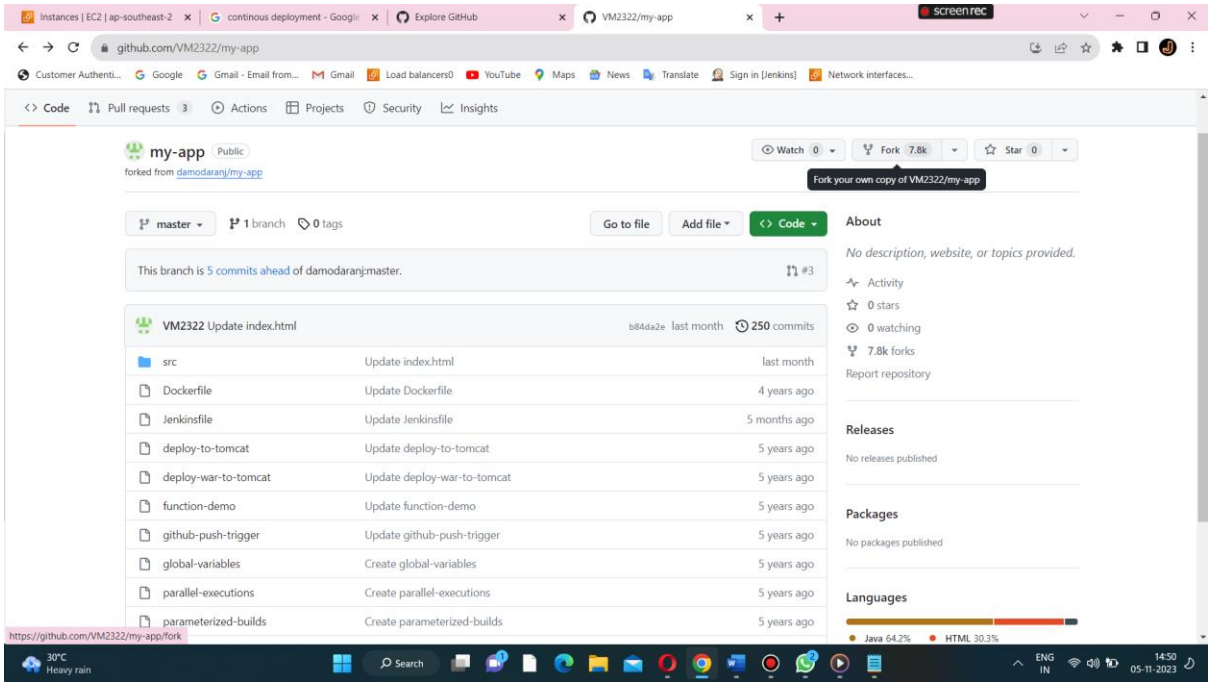
Maven is used to build the application.

Sonarqube is used for code quality check.

Nexus is used for artifact.

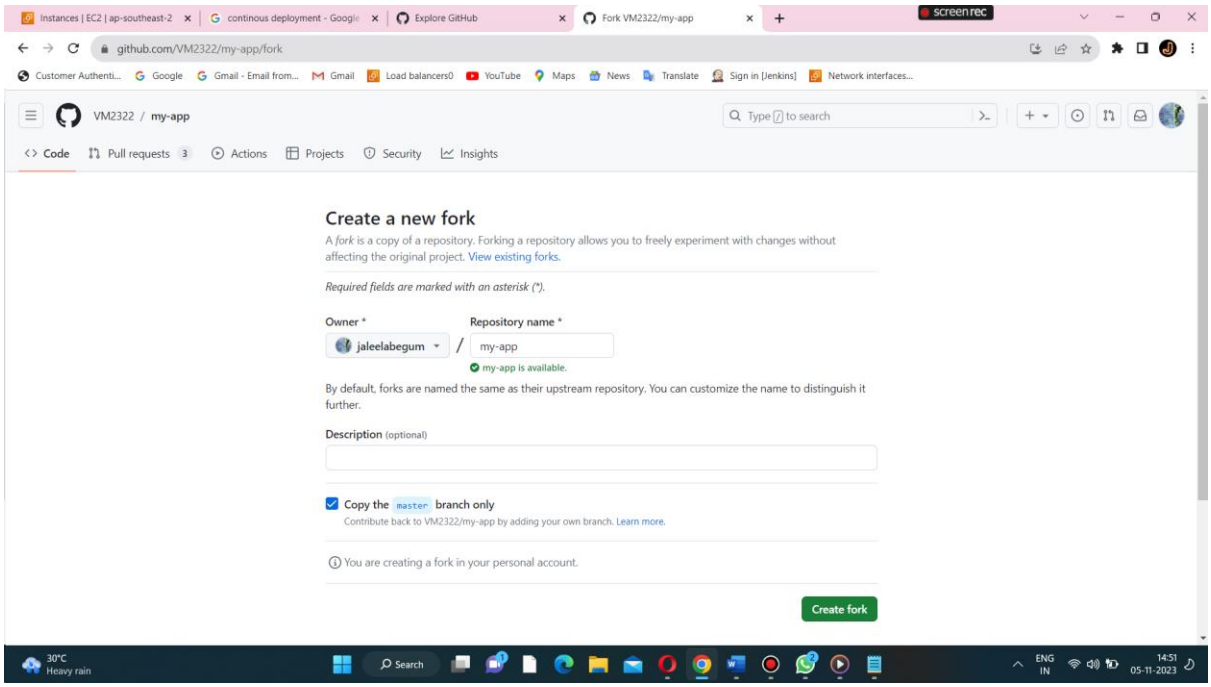
<u>STAGES IN CI/CD</u>

<u>1.source code management(SCM)</u>

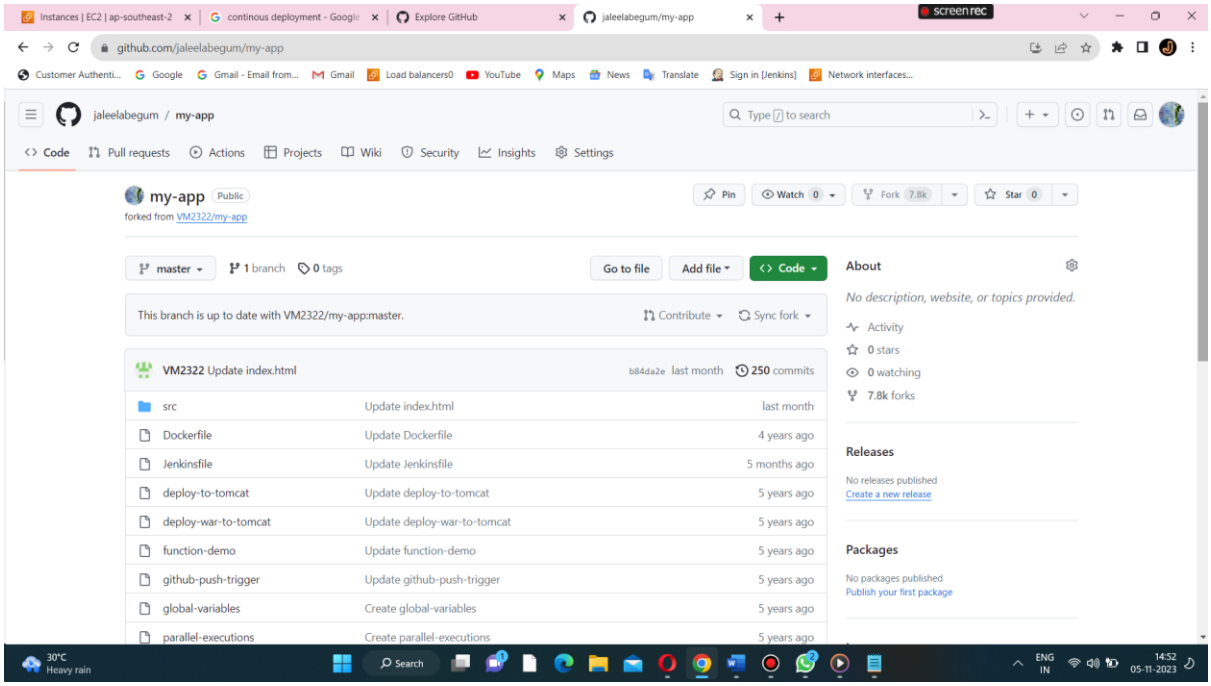


Git and github(fork from this website)

https://github.com/VM2322/my-app



Click on create fork.



That particular repository is fork into our project(to copy the from one repository into our repository).

Its present inside the master branch.

SCM CHECKOUT

STEP2:

Jenkins installation(t2.medium)-4gb ram(for installing four tools).

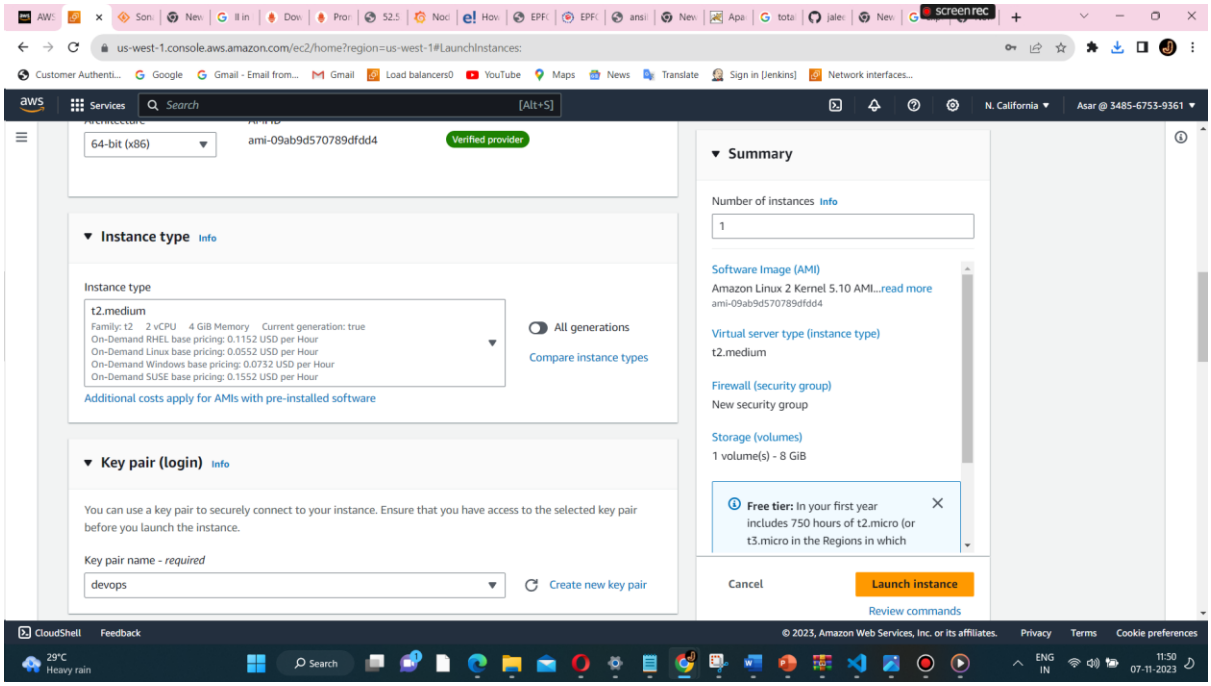
Number of servers used for ci/cd process:

Have a three typical servers for ci/cd implementation

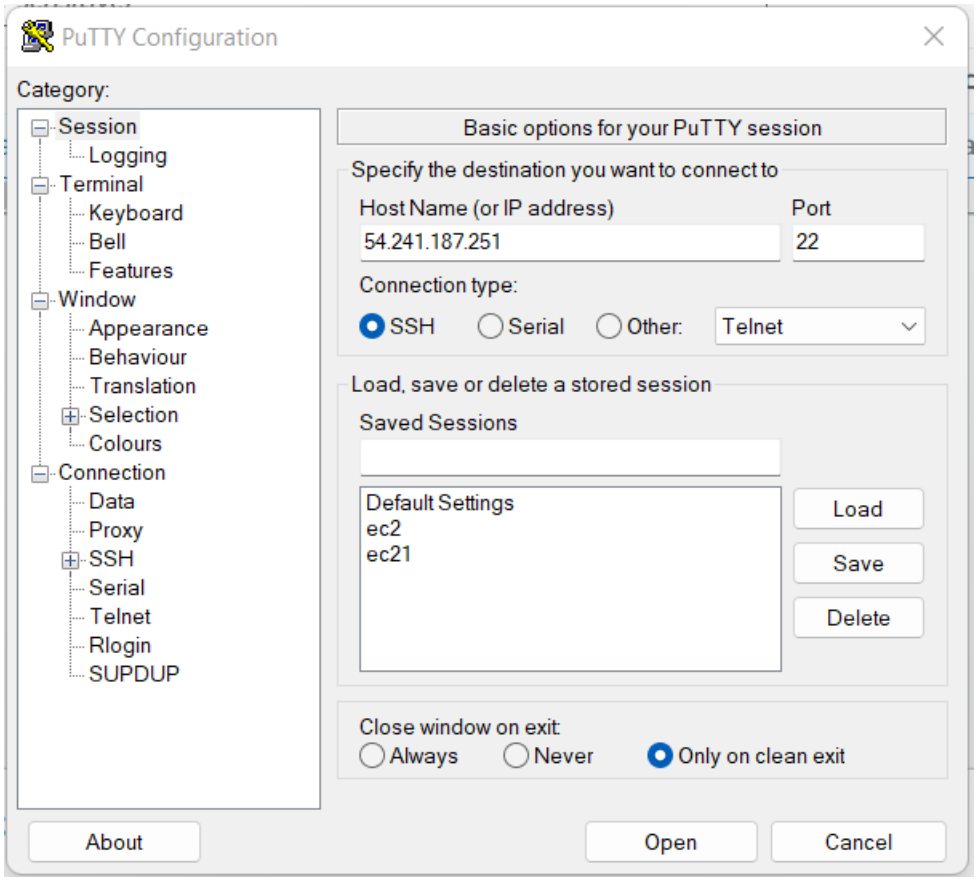
Server1: Jenkins,git,maven,docker (4GB RAM)

Server2: sonaqube

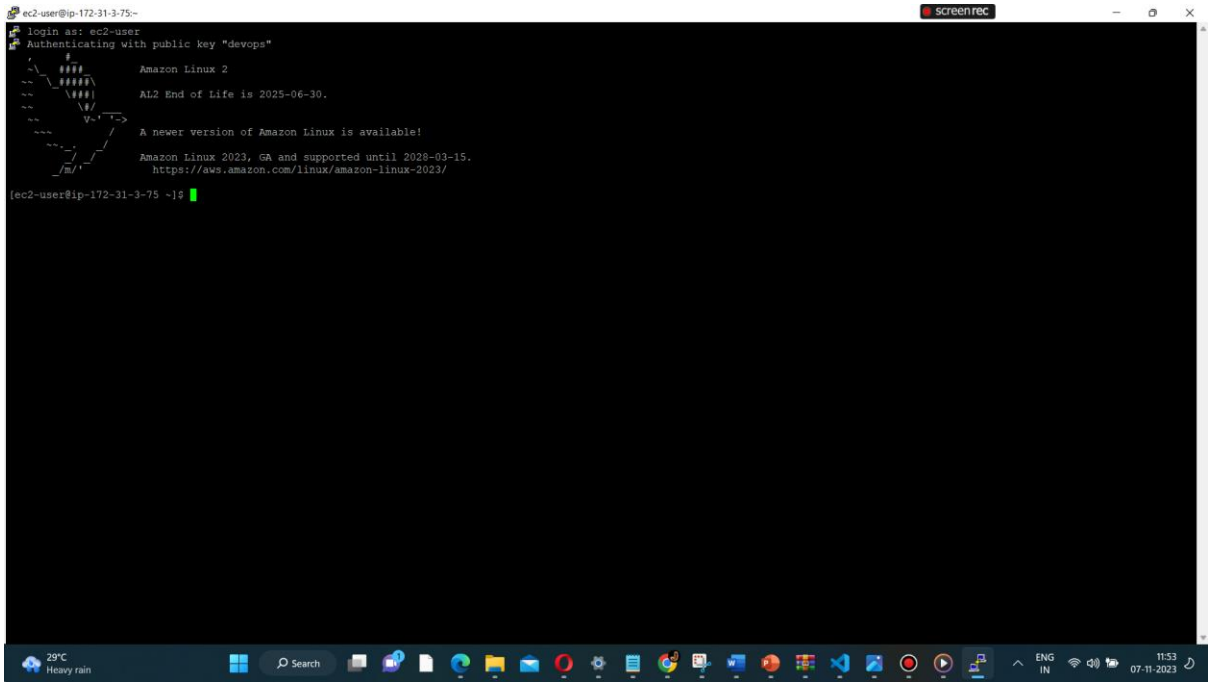
Server3:nexus(artifact).



Launch the instances in the instance type t2.medium.



Connect the instance using key pair.



Now instance is launched successfully.

<u>COMMANDS:</u>

1.amazon-linux-extras install epel (for to install the new packages)

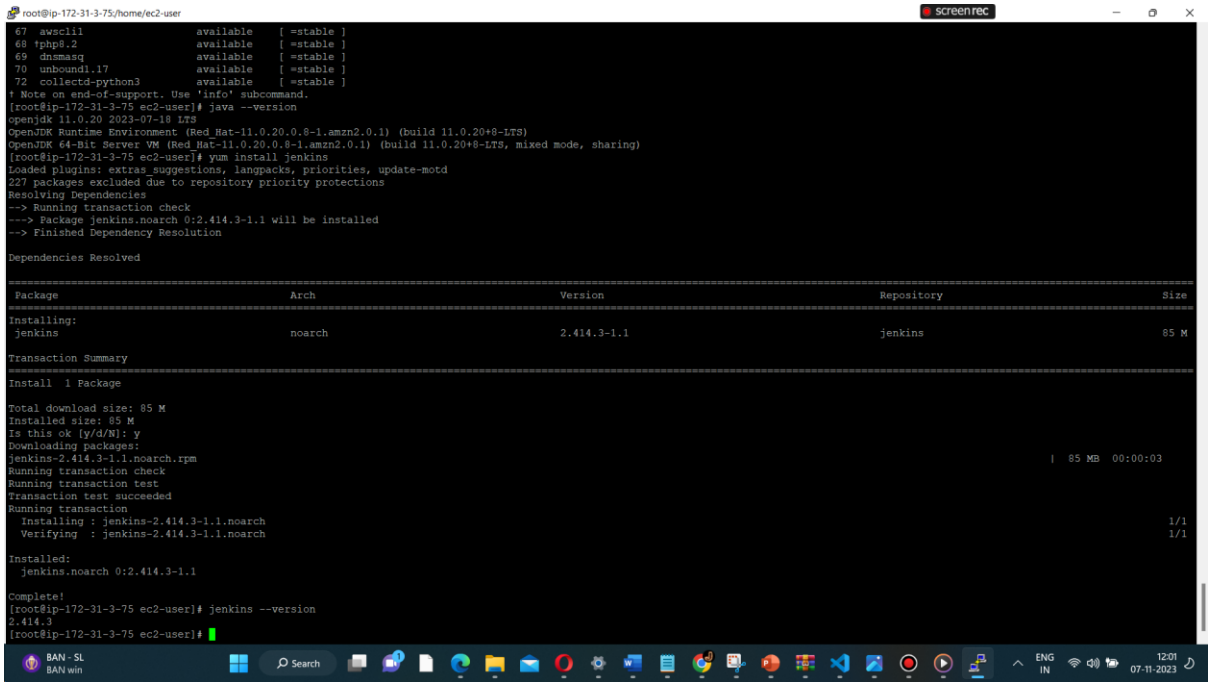
2.yum update -y

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

4. rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key

5. amazon-linux-extras install java-openjdk11

6. yum install jenkins



```
root@ip-172-31-3-75:/home/ec2-user
 67  awscli1                 available    [ =stable ]
 68  †php8.2                 available    [ =stable ]
 69  dnsmasq                 available    [ =stable ]
 70  unbound1.17             available    [ =stable ]
 72  collectd-python3        available    [ =stable ]
† Note on end-of-support. Use 'info' subcommand.
[root@ip-172-31-3-75 ec2-user]# java --version
openjdk 11.0.20 2023-07-18 LTS
OpenJDK Runtime Environment (Red_Hat-11.0.20.0.8-1.amzn2.0.1) (build 11.0.20+8-LTS)
OpenJDK 64-Bit Server VM (Red_Hat-11.0.20.0.8-1.amzn2.0.1) (build 11.0.20+8-LTS, mixed mode, sharing)
[root@ip-172-31-3-75 ec2-user]# yum install jenkins
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
227 packages excluded due to repository priority protections
Resolving Dependencies
--> Running transaction check
---> Package jenkins.noarch 0:2.414.3-1.1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=================================================================================================================================
 Package                    Arch                    Version                    Repository                                Size
=================================================================================================================================
Installing:
 jenkins                    noarch                  2.414.3-1.1                jenkins                                   85 M

Transaction Summary
=================================================================================================================================
Install  1 Package

Total download size: 85 M
Installed size: 85 M
Is this ok [y/d/N]: y
Downloading packages:
jenkins-2.414.3-1.1.noarch.rpm                                                                          |  85 MB  00:00:03
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : jenkins-2.414.3-1.1.noarch                                                                                   1/1
  Verifying  : jenkins-2.414.3-1.1.noarch                                                                                   1/1

Installed:
  jenkins.noarch 0:2.414.3-1.1

Complete!
[root@ip-172-31-3-75 ec2-user]# jenkins --version
2.414.3
[root@ip-172-31-3-75 ec2-user]#
```

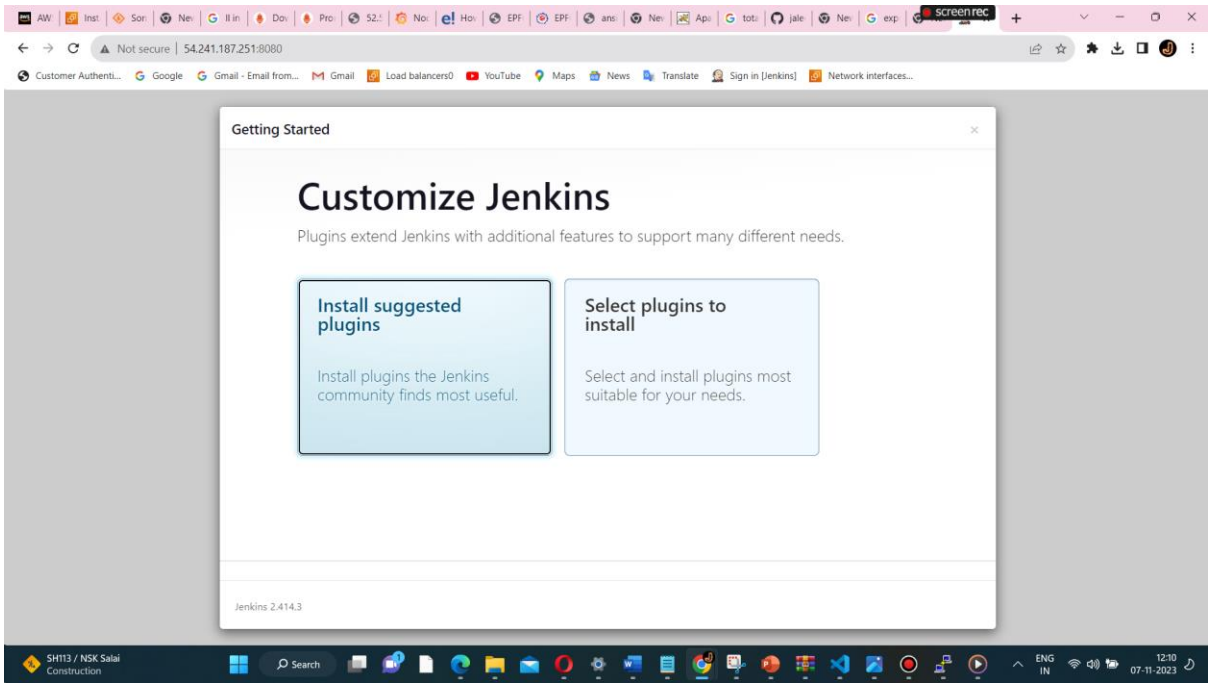
Install the Jenkins inorder to integarate with git.

To start the Jenkins using this commands.

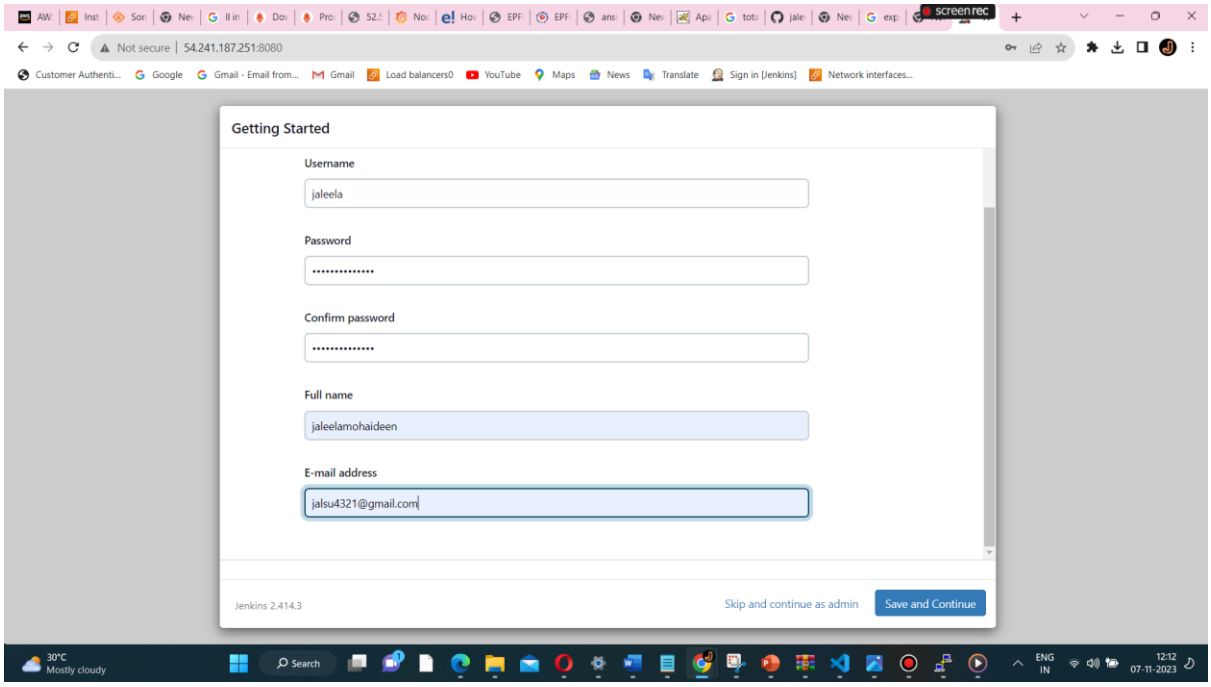
1.sudo systemctl enable Jenkins

2.sudo systemctl start Jenkins

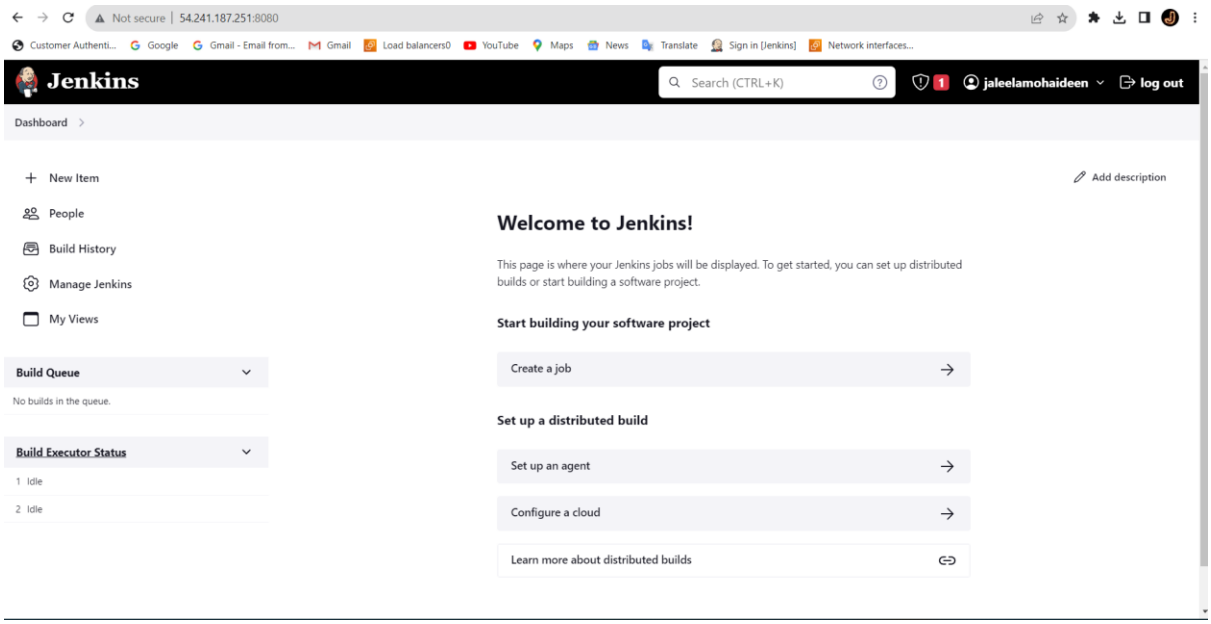
3.sudo systemctl status Jenkins



Jenkins installation to install the suggested plugins.



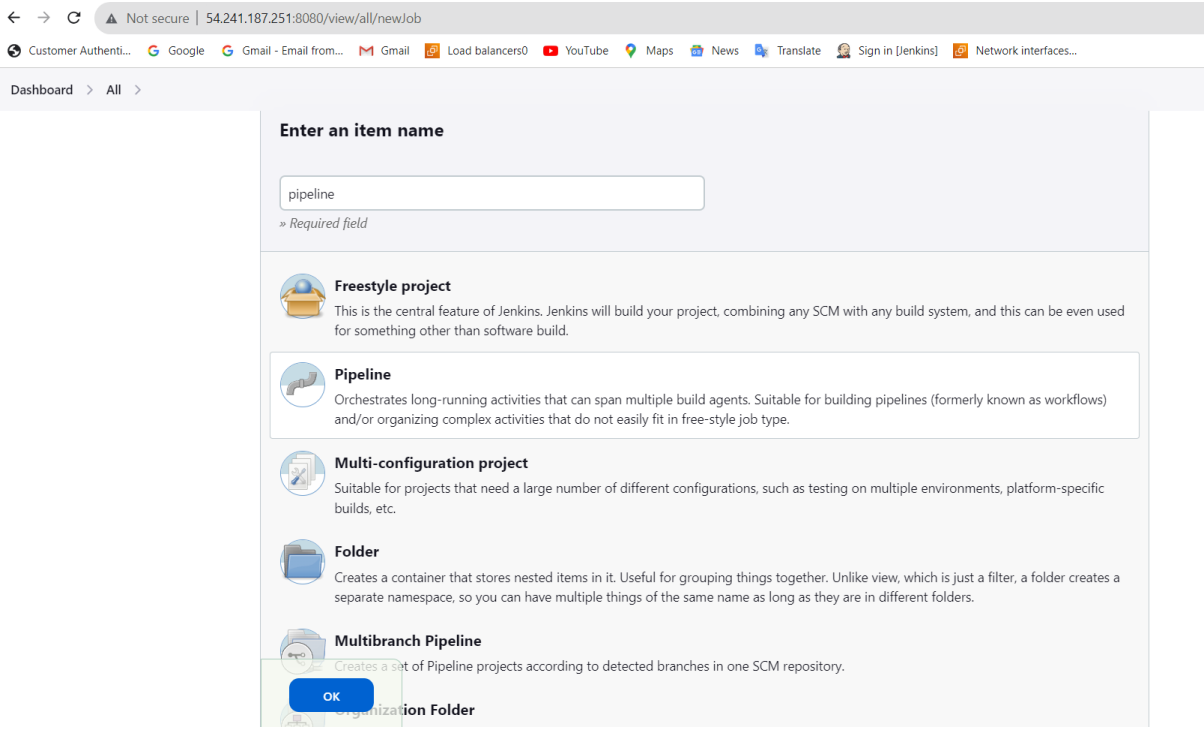
Click on save and continue.

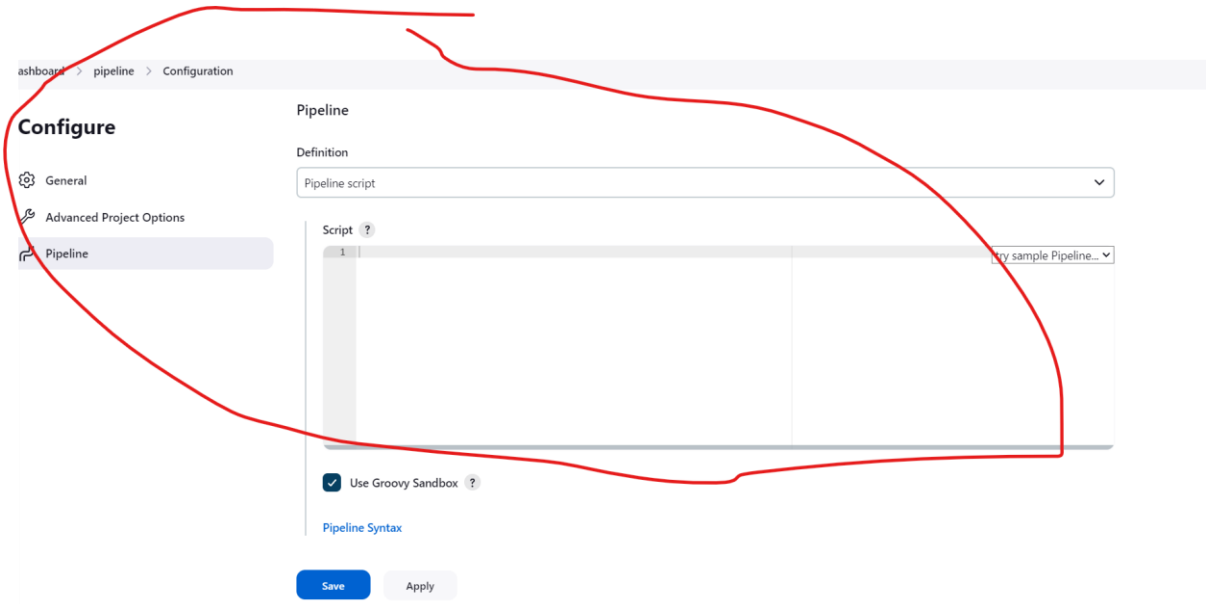


Then the Jenkins dashboard got displayed.

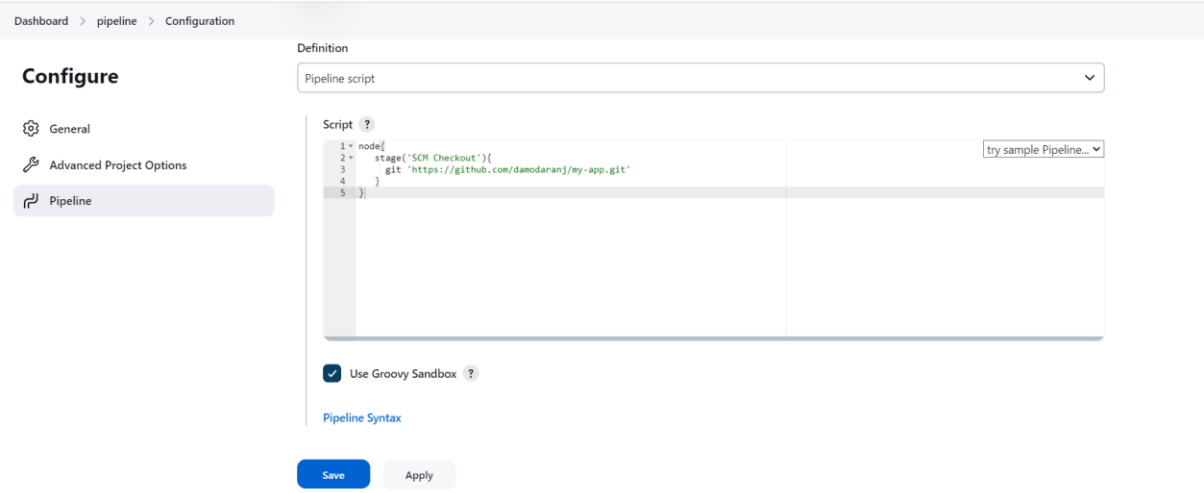
STEP3: STAGING

Once we need to integrate with Jenkins we need to install Jenkins and configure the home path.

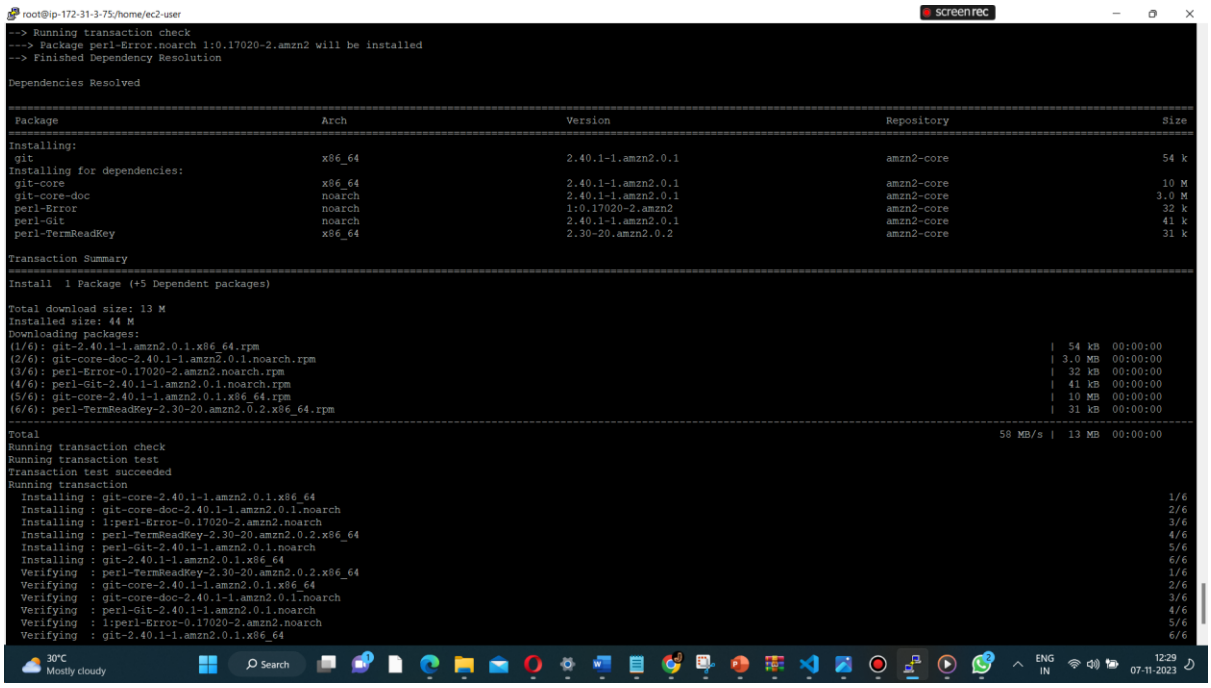




Need to write the script here.



To add the staging file here



Need to install the git in a server.

Not secure | 54.241.187.251:8080/job/pipeline/2/console

ipeline > #2

Console Output

utput

plain text

Information

ld '#2'

Data

:eps

es

uild

```
Started by user jaleelamohaideen
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (SCM Checkout)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/damodaranj/my-app.git
 > git init /var/lib/jenkins/workspace/pipeline # timeout=10
Fetching upstream changes from https://github.com/damodaranj/my-app.git
 > git --version # timeout=10
 > git --version # 'git version 2.40.1'
 > git fetch --tags --force --progress -- https://github.com/damodaranj/my-app.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/damodaranj/my-app.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 59bdda88fa3c53787e0ef508f7f2fc77cf02aa41 (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 59bdda88fa3c53787e0ef508f7f2fc77cf02aa41 # timeout=10
 > git branch -a -v --no-abbrev # timeout=10
```

Now I completed the git checkout by adding the scripted pipeline.

To view the pipelines in a Jenkins using

cd /var/lib/Jenkins/workspace

```
[root@ip-172-31-3-75 ec2-user]# cd /var/lib/jenkins/workspace
[root@ip-172-31-3-75 workspace]# ls
pipeline
[root@ip-172-31-3-75 workspace]# cd pipleline
bash: cd: pipleline: No such file or directory
[root@ip-172-31-3-75 workspace]# cd pipeline
[root@ip-172-31-3-75 pipeline]# ls
deploy-to-tomcat  deploy-war-to-tomcat  Dockerfile  function-demo  github-push-trigger  global-variables  Jenkinsfile  parallel-executions  parameterized-builds  pom.xml  src
```

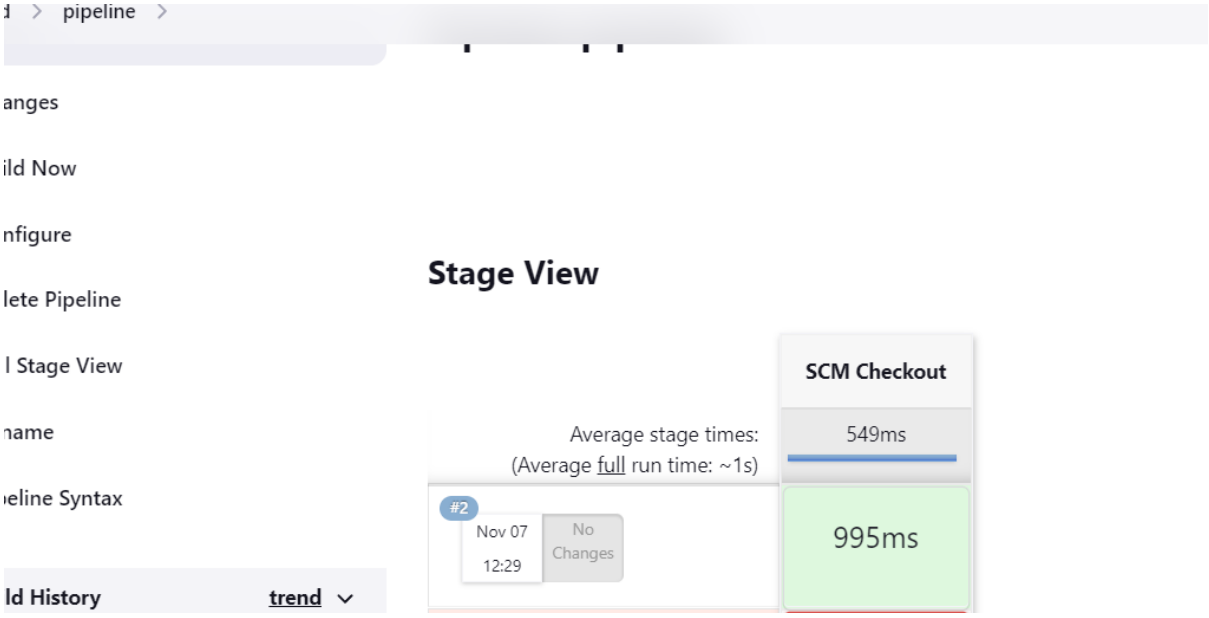
```
[root@ip-172-31-3-75 pipeline]# ll
total 40
-rw-r--r-- 1 jenkins jenkins  824 Nov  7 06:59 deploy-to-tomcat
-rw-r--r-- 1 jenkins jenkins  938 Nov  7 06:59 deploy-war-to-tomcat
-rw-r--r-- 1 jenkins jenkins  109 Nov  7 06:59 Dockerfile
-rw-r--r-- 1 jenkins jenkins 1108 Nov  7 06:59 function-demo
-rw-r--r-- 1 jenkins jenkins  234 Nov  7 06:59 github-push-trigger
-rw-r--r-- 1 jenkins jenkins  339 Nov  7 06:59 global-variables
-rw-r--r-- 1 jenkins jenkins 1245 Nov  7 06:59 Jenkinsfile
-rw-r--r-- 1 jenkins jenkins  311 Nov  7 06:59 parallel-executions
-rw-r--r-- 1 jenkins jenkins  328 Nov  7 06:59 parameterized-builds
-rwxr-xr-x 1 jenkins jenkins 1822 Nov  7 06:59 pom.xml
drwxr-xr-x 4 jenkins jenkins   47 Nov  7 06:59 src
```

2.ls

3.cd pipeline

4. ls

All the repository files are present in the Jenkins now.



STEP2:

Build the file(here java packages is there so we can build using maven).

MAVEN

If we integrate with Jenkins we need to install plugins and to set the path.

Build the application suing maven.

To convert a code into a package.

(executable format is .war or .jar file)

Other than maven-ant or gradel is used for building the application.

Maven is used to build the application only for java based application.

Maven file is a pom.xml file

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
        <modelVersion>4.0.0</modelVersion>
        <groupId>in.javahome</groupId>
        <artifactId>myweb</artifactId>
        <packaging>war</packaging>
        <version>0.0.5</version>
        <name>my-app</name>
```

For a my-app application I deploy the myweb.war file

Install the maven using

wget https://mirrors.estointernet.in/apache/maven/maven-3/3.8.5/binaries/apache-maven-3.8.5-bin.tar.gz

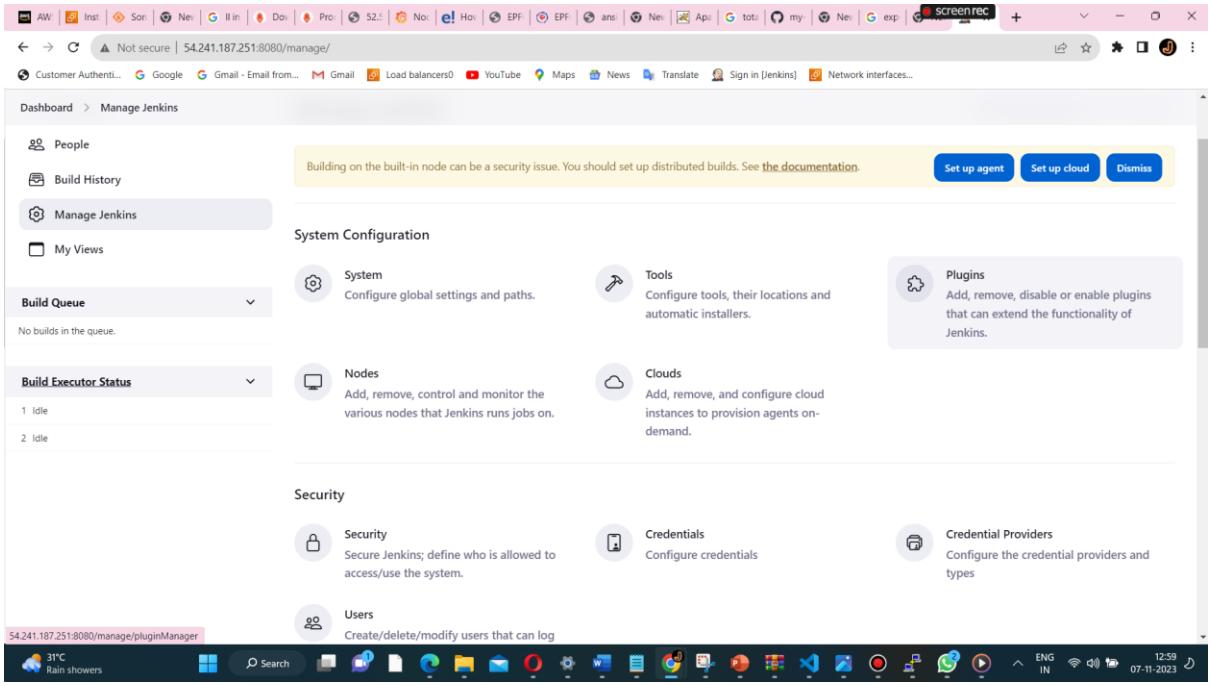
using wget command I can install the maven.

Unzip the tar file

ls

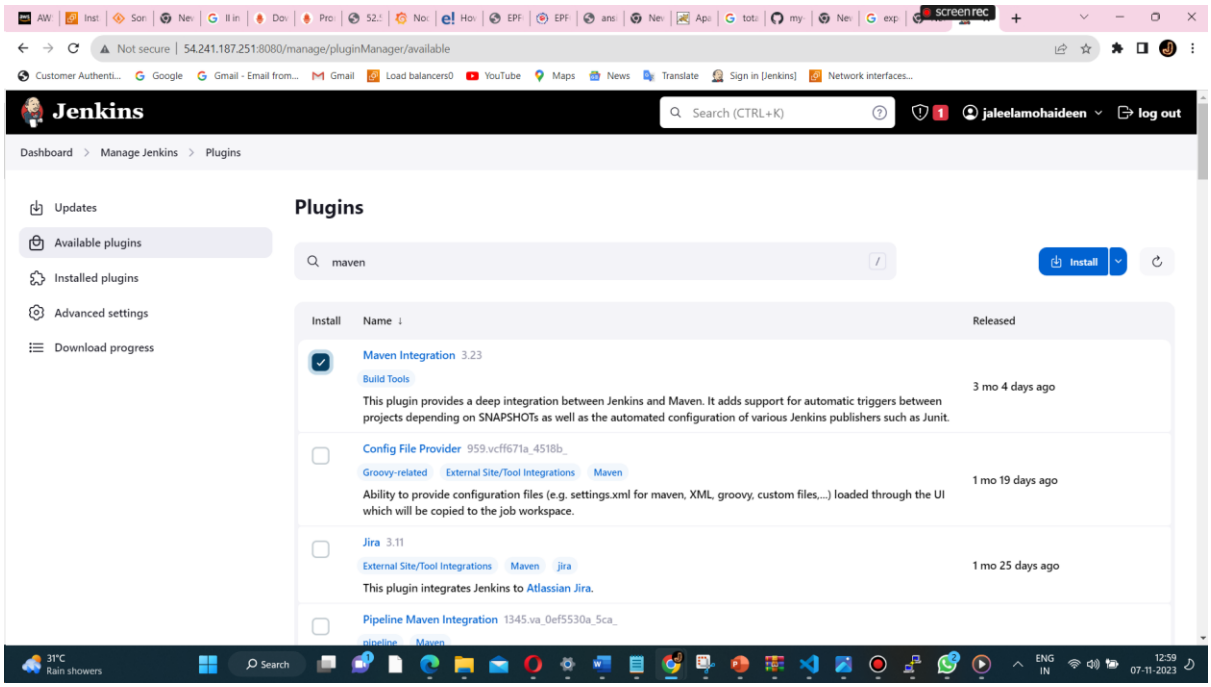
tar -xvzf apache-maven-3.8.5-bin.tar.gz (to extract the file).

Then install the plugins in jenkins.



Click on manage plugins choose plugins

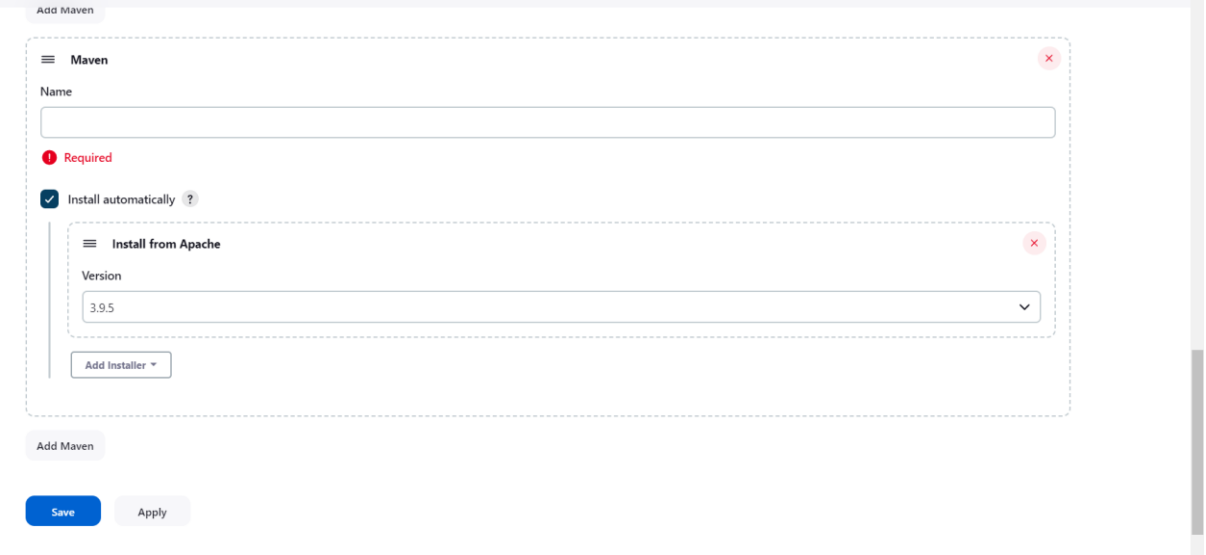
Click on available plugins



Select maven integration for installing.

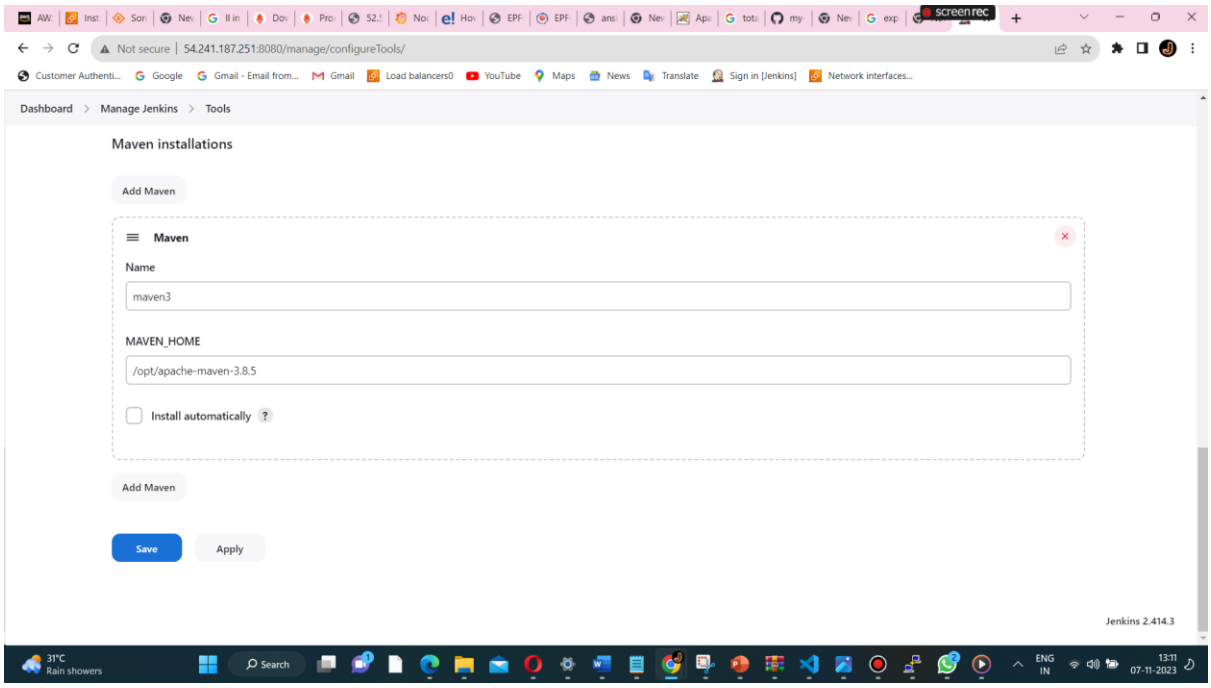
Click on install then maven plugins is installed successfully.

Add the maven integration to a Jenkins.



Click on manage Jenkins, choose tools for configure to a Jenkins.

Here we choose maven.



In tools we are adding completed the maven.

Definition

Pipeline script

Script ?

```
node{
   stage('SCM Checkout'){
     git 'https://github.com/damodaranj/my-app.git'
   }
}
stage('maven-buildstage'){

      def mvnHome =  tool name: 'maven3', type: 'maven'
      sh "${mvnHome}/bin/mvn clean package"
      sh 'mv target/myweb*.war target/newapp.war'
   }
```

To add the maven staging in the pipeline.

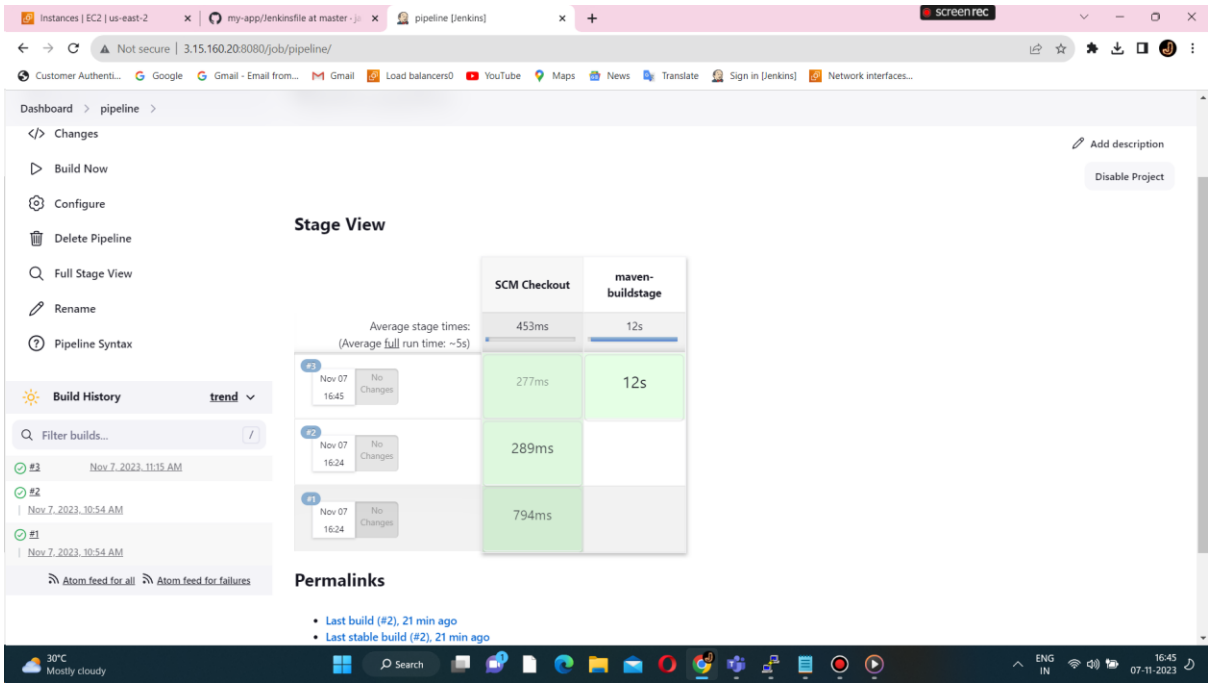
This script indicates- need to build the code using maven.

Tool name: maven3

Using sh mvnhome indicates /opt/apache-maven-3.8.5 /bin/mvn clean package

Next line is about create a target folder then create a myweb.war inside the folder.

newapp.war - to provide the unique name to deploy the war file.



Definition

Pipeline script

Script

```
node{
   stage('SCM Checkout'){
     git 'https://github.com/jaleelabegum/my-app'
   }
    stage('maven-buildstage'){

      def mvnHome =  tool name: 'maven3', type: 'maven'
      sh "${mvnHome}/bin/mvn clean package"
      sh 'mv target/myweb*.war target/newapp.war'
   }
}
```

Use Groovy Sandbox

Pipeline Syntax

Save Apply

To add the maven staging to build the files.

```
apache-maven-3.8.5-bin.tar.gz  deploy-war-to-tomcat  function-demo        global-variables  parallel-executions   pom.xml  target
deploy-to-tomcat               Dockerfile            github-push-trigger  Jenkinsfile       parameterized-builds  src
[root@ip-172-31-28-90 pipeline]# ll
total 8512
-rw-r--r--  1 root    root    8673123 Mar  5  2022 apache-maven-3.8.5-bin.tar.gz
-rw-r--r--  1 jenkins jenkins     824 Nov  7 10:54 deploy-to-tomcat
-rw-r--r--  1 jenkins jenkins     938 Nov  7 10:54 deploy-war-to-tomcat
-rw-r--r--  1 jenkins jenkins     109 Nov  7 10:54 Dockerfile
-rw-r--r--  1 jenkins jenkins    1108 Nov  7 10:54 function-demo
-rw-r--r--  1 jenkins jenkins     234 Nov  7 10:54 github-push-trigger
-rw-r--r--  1 jenkins jenkins     339 Nov  7 10:54 global-variables
-rw-r--r--  1 jenkins jenkins    1243 Nov  7 10:54 Jenkinsfile
-rw-r--r--  1 jenkins jenkins     311 Nov  7 10:54 parallel-executions
-rw-r--r--  1 jenkins jenkins     328 Nov  7 10:54 parameterized-builds
-rwxr-xr-x  1 jenkins jenkins    1822 Nov  7 10:54 pom.xml
drwxr-xr-x  4 jenkins jenkins      47 Nov  7 10:54 src
drwxr-xr-x 10 jenkins jenkins     199 Nov  7 11:15 target
[root@ip-172-31-28-90 pipeline]#
```

After inside the pipeline path target folder is created

Inside the folder.

```
-rw-r--r--  1 jenkins jenkins     234 Nov  7 10:54 github-push-trigger
-rw-r--r--  1 jenkins jenkins     339 Nov  7 10:54 global-variables
-rw-r--r--  1 jenkins jenkins    1243 Nov  7 10:54 Jenkinsfile
-rw-r--r--  1 jenkins jenkins     311 Nov  7 10:54 parallel-executions
-rw-r--r--  1 jenkins jenkins     328 Nov  7 10:54 parameterized-builds
-rwxr-xr-x  1 jenkins jenkins    1822 Nov  7 10:54 pom.xml
drwxr-xr-x  4 jenkins jenkins      47 Nov  7 10:54 src
drwxr-xr-x 10 jenkins jenkins     199 Nov  7 11:15 target
[root@ip-172-31-28-90 pipeline]# cd target
[root@ip-172-31-28-90 target]# ls
classes  generated-sources  generated-test-sources  maven-archiver  maven-status  myweb-0.0.5  newapp.war  surefire-reports  test-classes
```

newapp.war file is created. Now the code is converted into a war package. Unique name to a file.(newapp.war)

	SCM Checkout	maven-buildstage
Average stage times: (Average full run time: ~5s)	453ms	12s
#3 Nov 07 16:45 No Changes	277ms	12s
#2 Nov 07 16:24 No Changes	289ms	
#1 Nov 07 16:24 No Changes	794ms	

Permalinks

TO CONVERT A WAR FILE TO A DOCKER IMAGE AND PUSH INTO A DOCKER HUB

COMMANDS

<u>Install the docker on same server</u>

yum install docker -y

Pipeline script

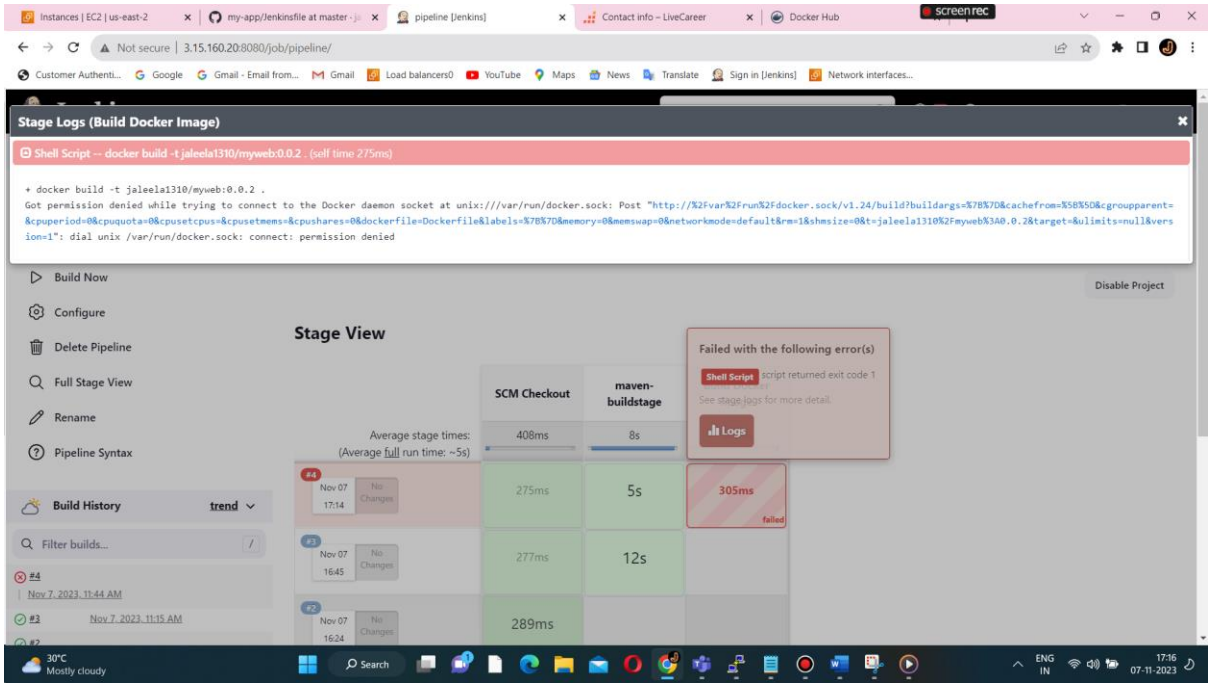
Script ?

```
node{
   stage('SCM Checkout'){
     git 'https://github.com/jaleelabegum/my-app'
   }
    stage('maven-buildstage'){

      def mvnHome =  tool name: 'maven3', type: 'maven'
      sh "${mvnHome}/bin/mvn clean package"
      sh 'mv target/myweb*.war target/newapp.war'
   }
    stage('Build Docker Image'){
   sh 'docker build -t jaleela1310/myweb:0.0.2 .'
   }
}
```

After installing the docker,we need to built the docker file to add the stage here.

Already Dockerfile is present in the pipeline directory so we just build the image in a script.

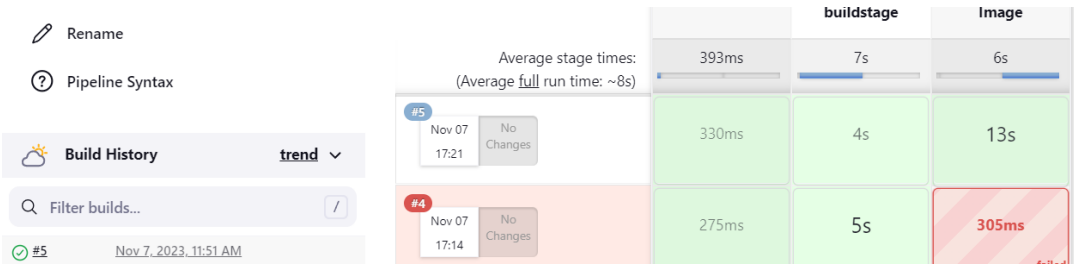


to provide the permission to a Jenkins to build a docker file otherwise it throws an error.

To run the docker file provide the permission to a Jenkins.

chmod 0777 /var/run/docker.sock

command provide in a server.

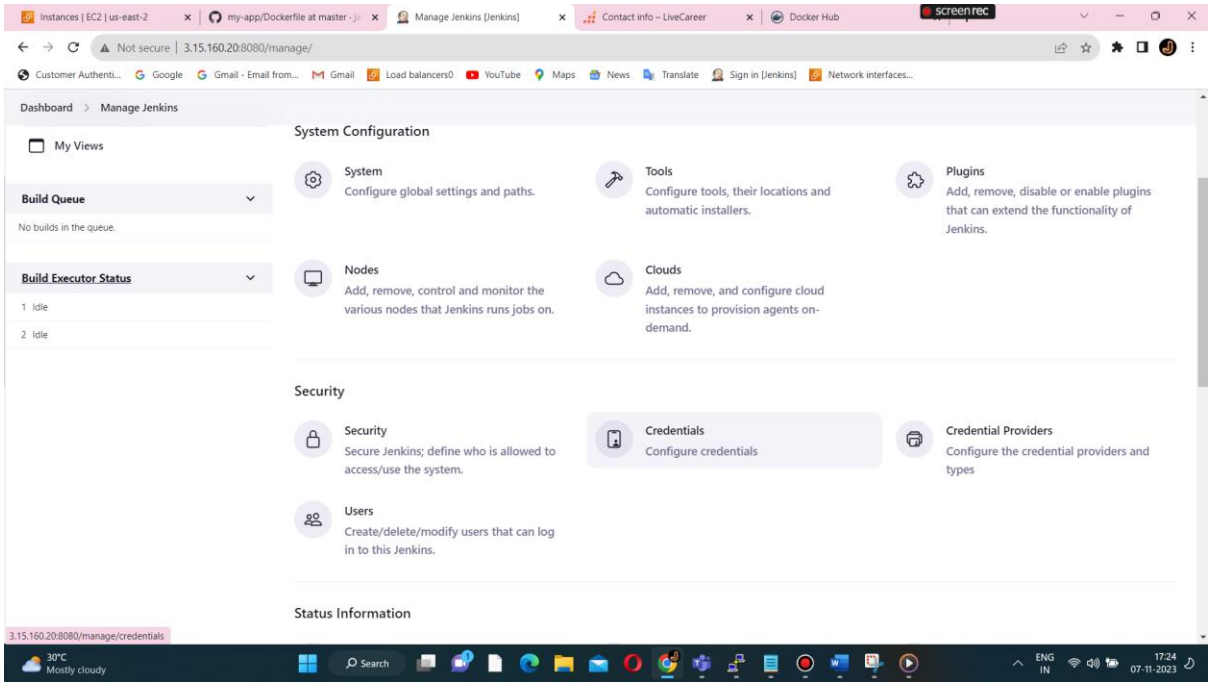


Now it image is built successfully.

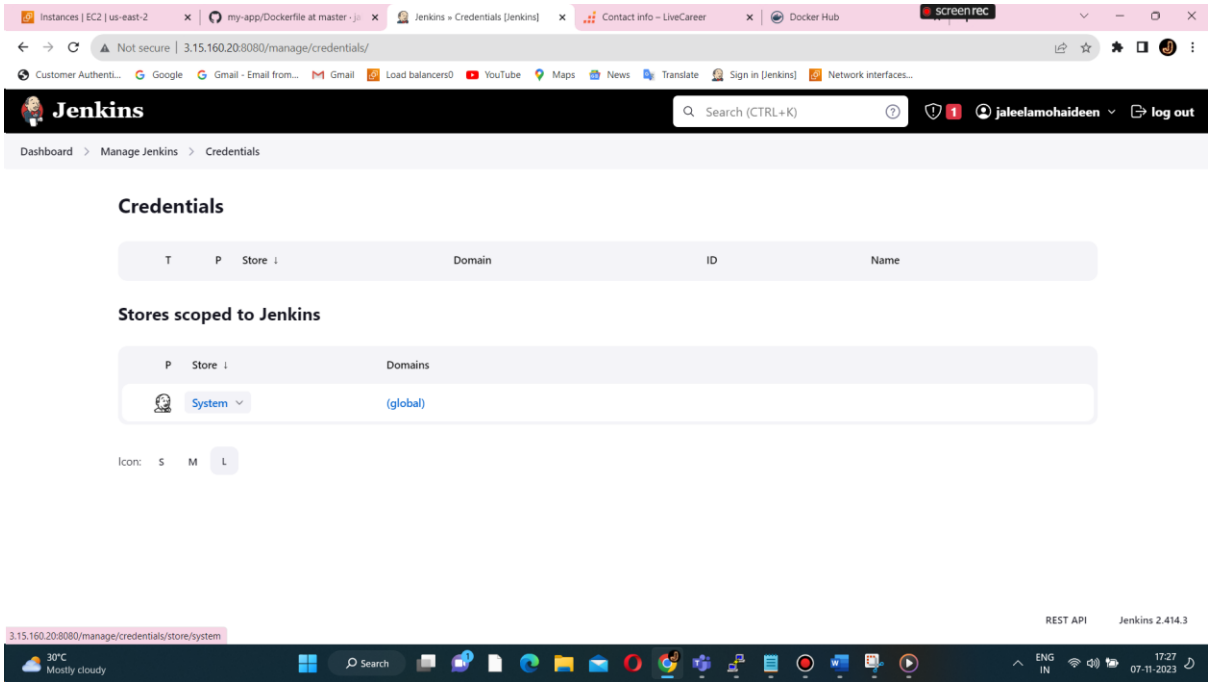
To push the image to a docker.

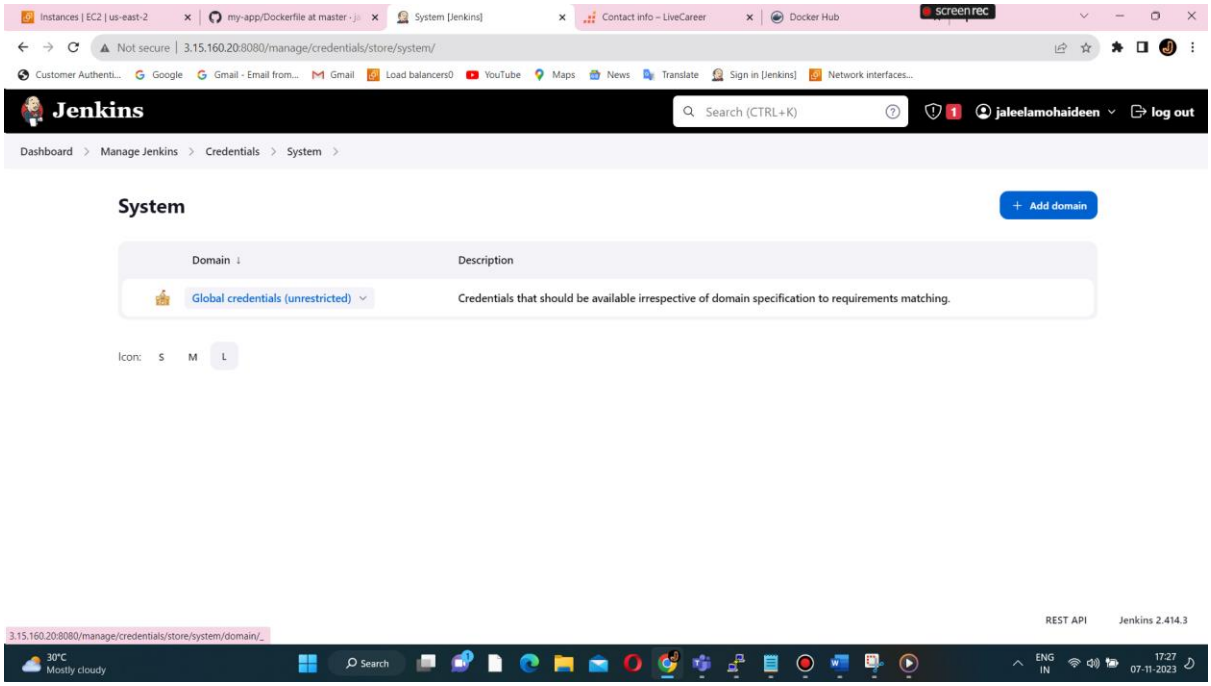
```
[root@ip-172-31-28-90 target]# docker images
REPOSITORY          TAG       IMAGE ID       CREATED             SIZE
jaleela1310/myweb   0.0.2     b7a2382ed514   About a minute ago  462MB
tomcat              8         3ae2bbab2686   7 days ago          461MB
[root@ip-172-31-28-90 target]#
```

To pass the sensitive information to a secrets.

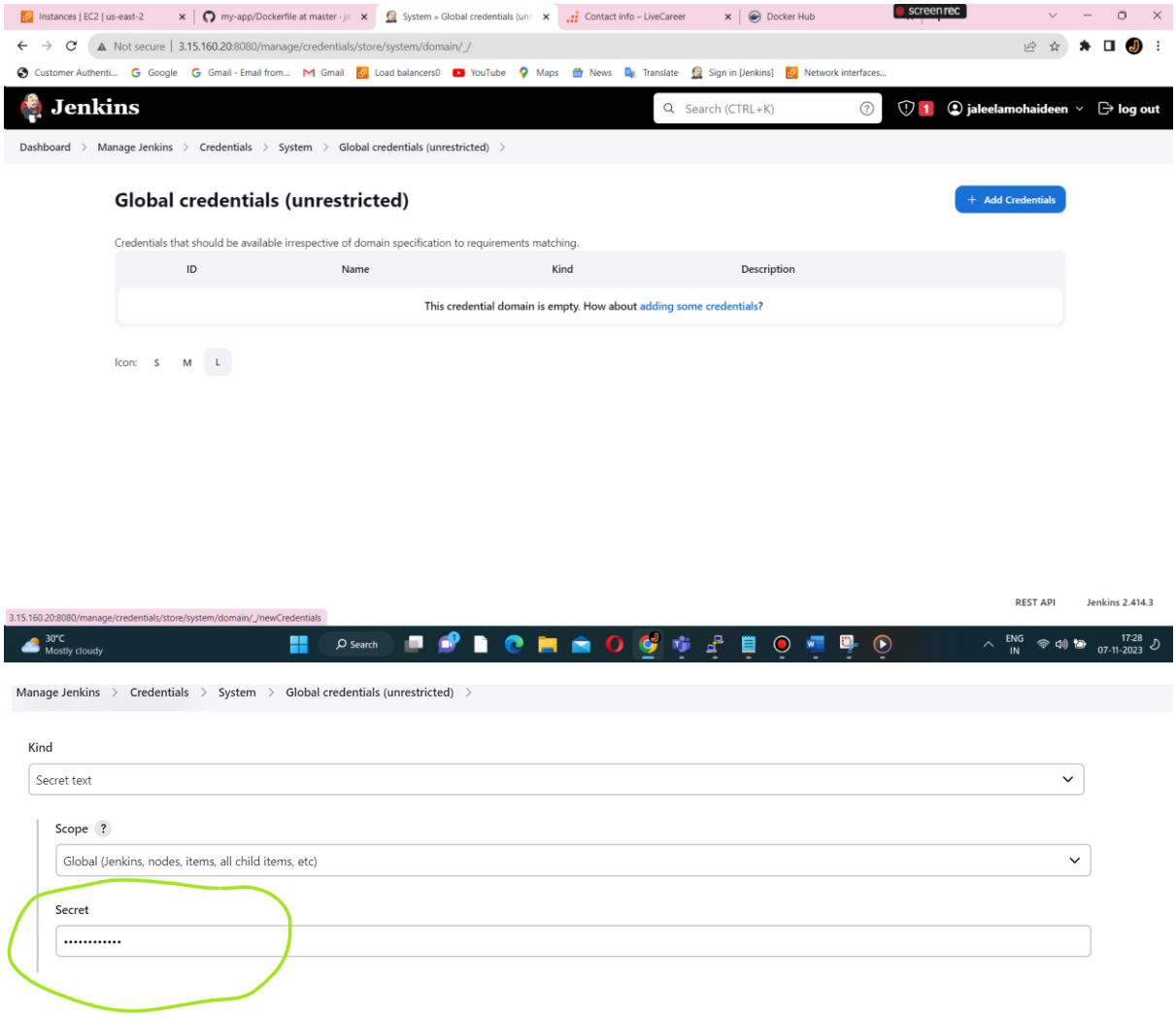


Click on manage Jenkins->credentials->system->globalcredentails

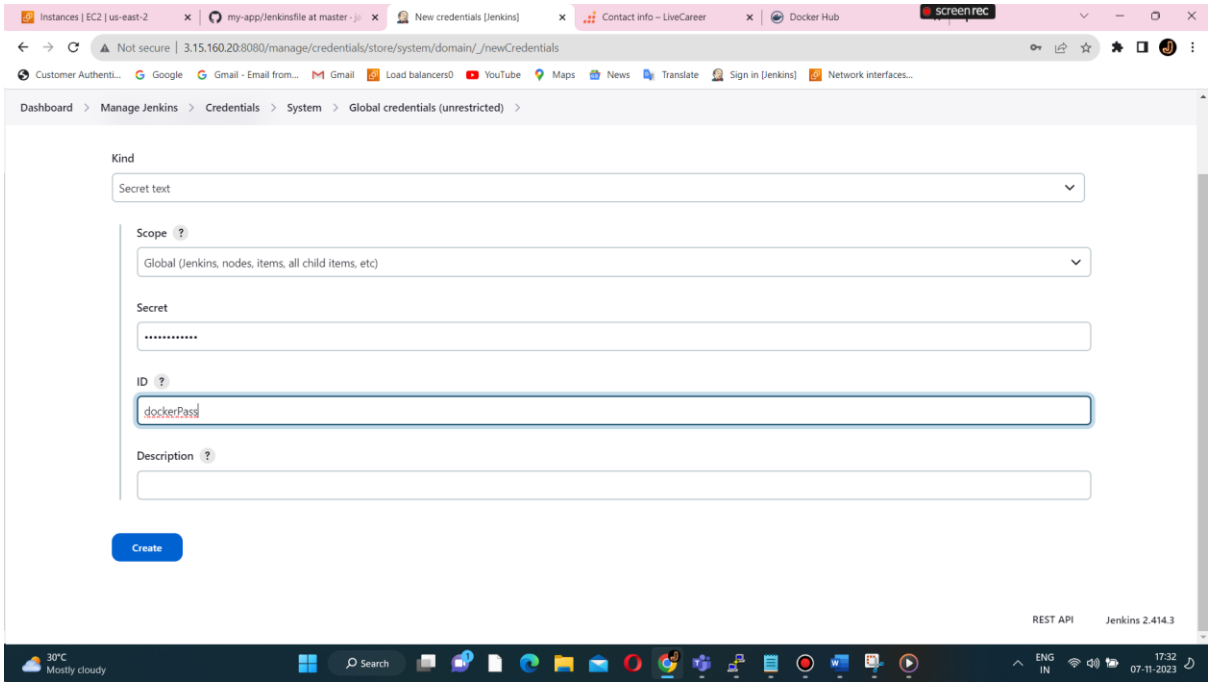




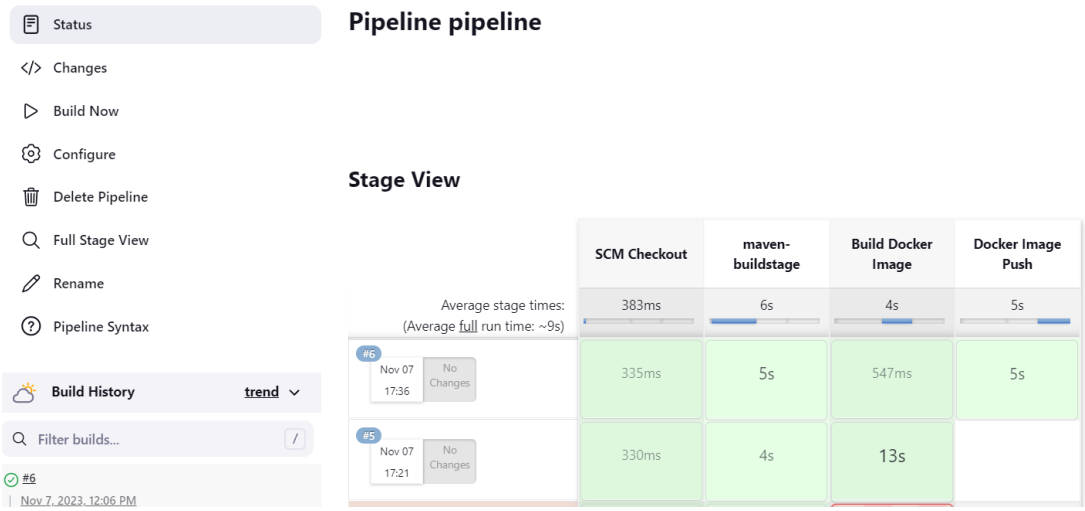
Then add credentials.



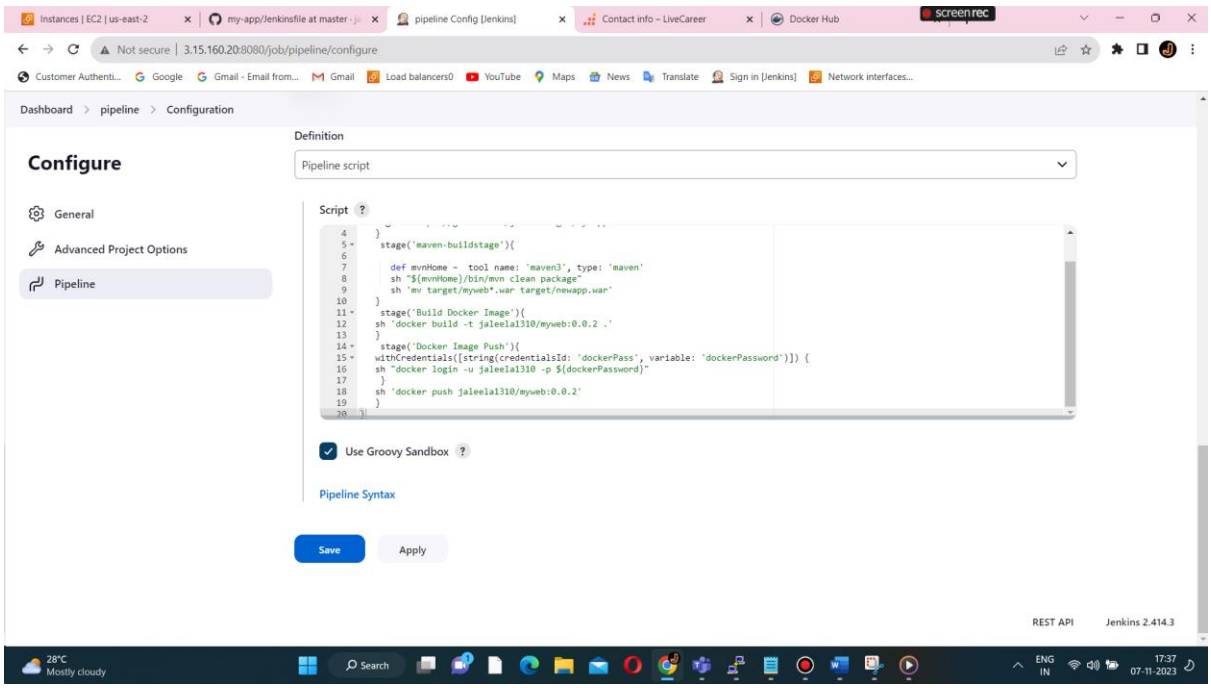
In the secret give the dockerhub password.

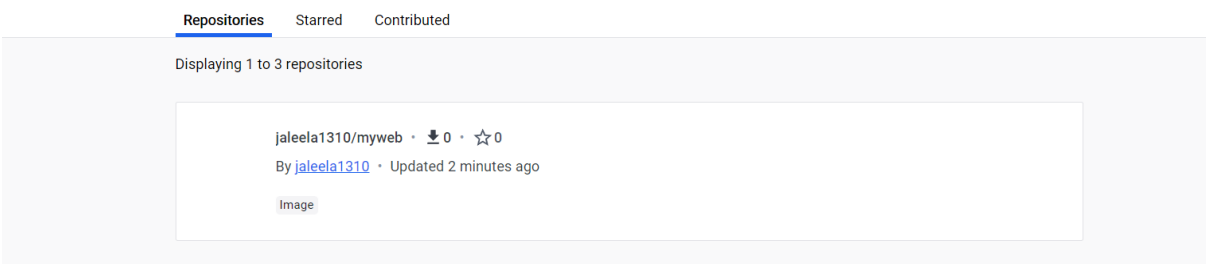


Id is dockerPass that should be mentioned in a script file.



Now the docker image is pushed into our docker hub.





Myweb got pushed into a docker hub.