**CICD PROJECT**

**CODE INTEGRATION AND CODE DEPLOYMENT**

I have created my project using CICD flow which is Continuous integration and continuous deployment using such devops tools.

**TOOLS & PROJECT STAGING:**

SOURCE CODE MANAGEMENT- GIT

CONTINUOUS INTEGRATION- JENKINS

CODE TESTING – SONARQUBE

BUILD TOOL- MAVEN

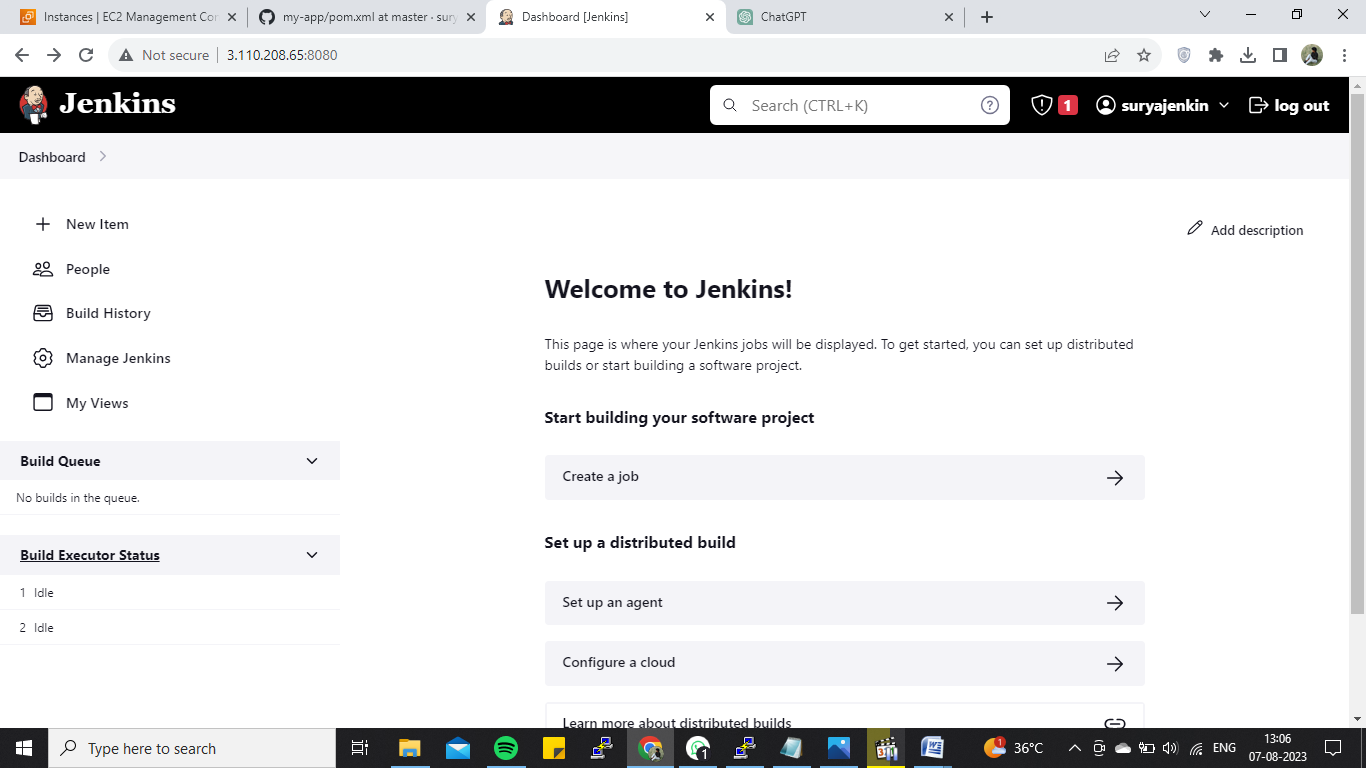
ARTIFACT REPO- NEXUS

DEPLOYMENT- DOCKER

------------------------------------------------------------------------------------------------------------------------------------------

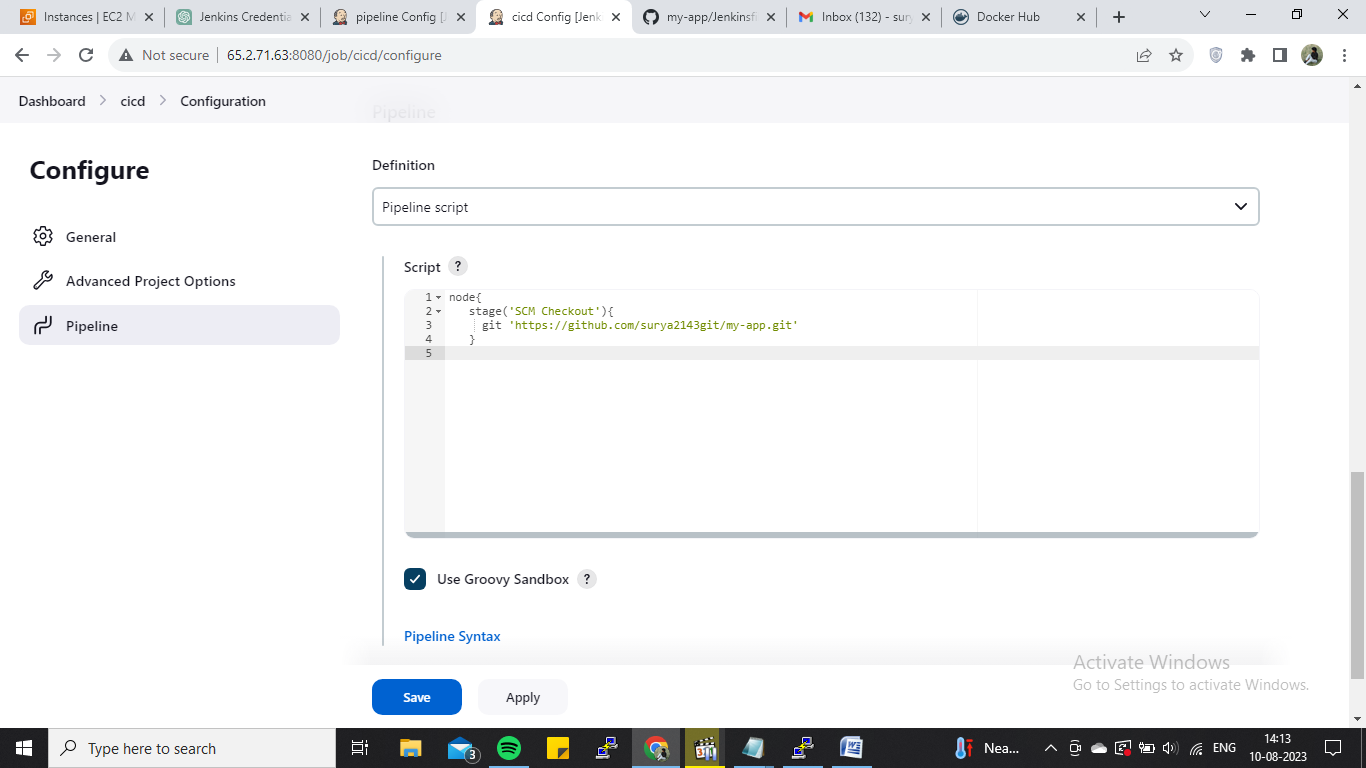
1. Creating EC2 instance with t2.medium and required protocols

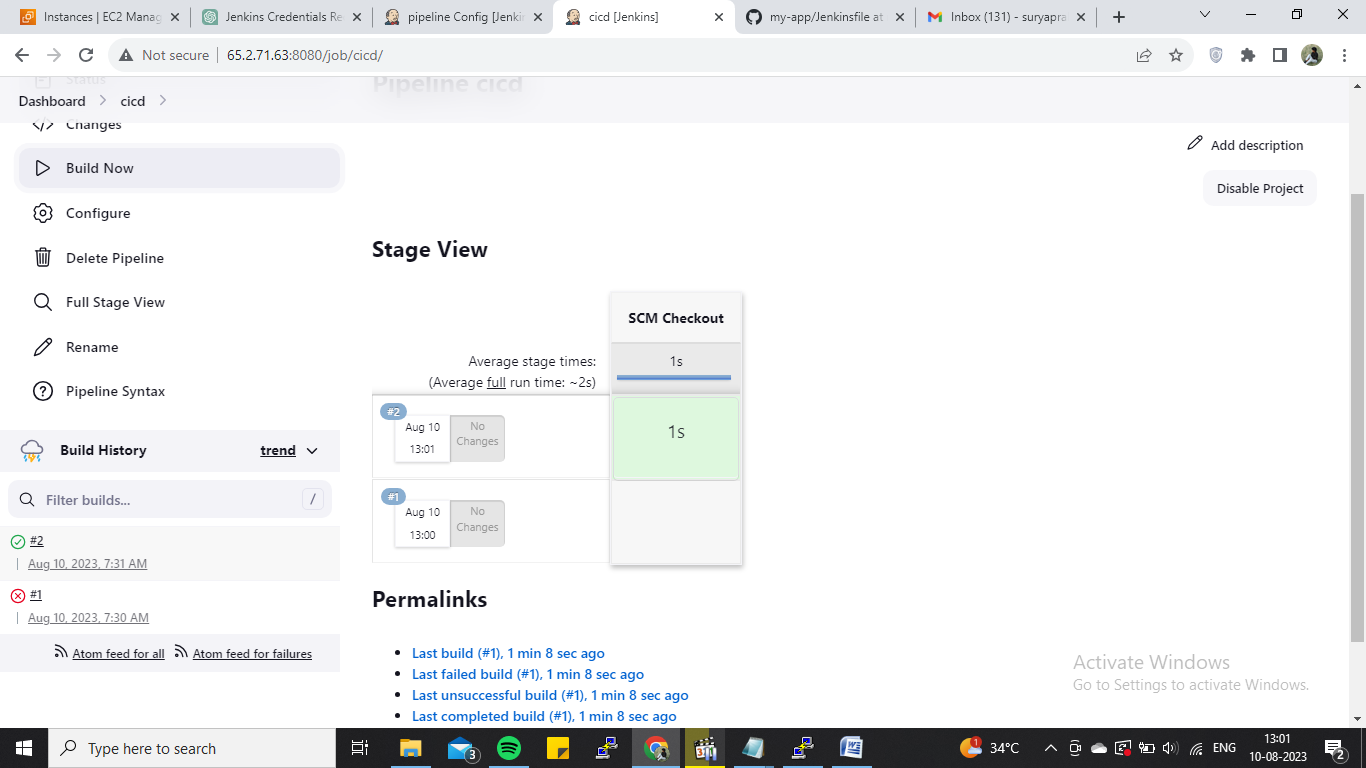
* Installing amason extras –epel.
* GIT – for the source code management.
* JENKINS – for continuous integration.
* Create the job in Jenkins using pipeline.



* Getting the Git repository link by pipeline syntax.
* Configured to create the pipeline stages.

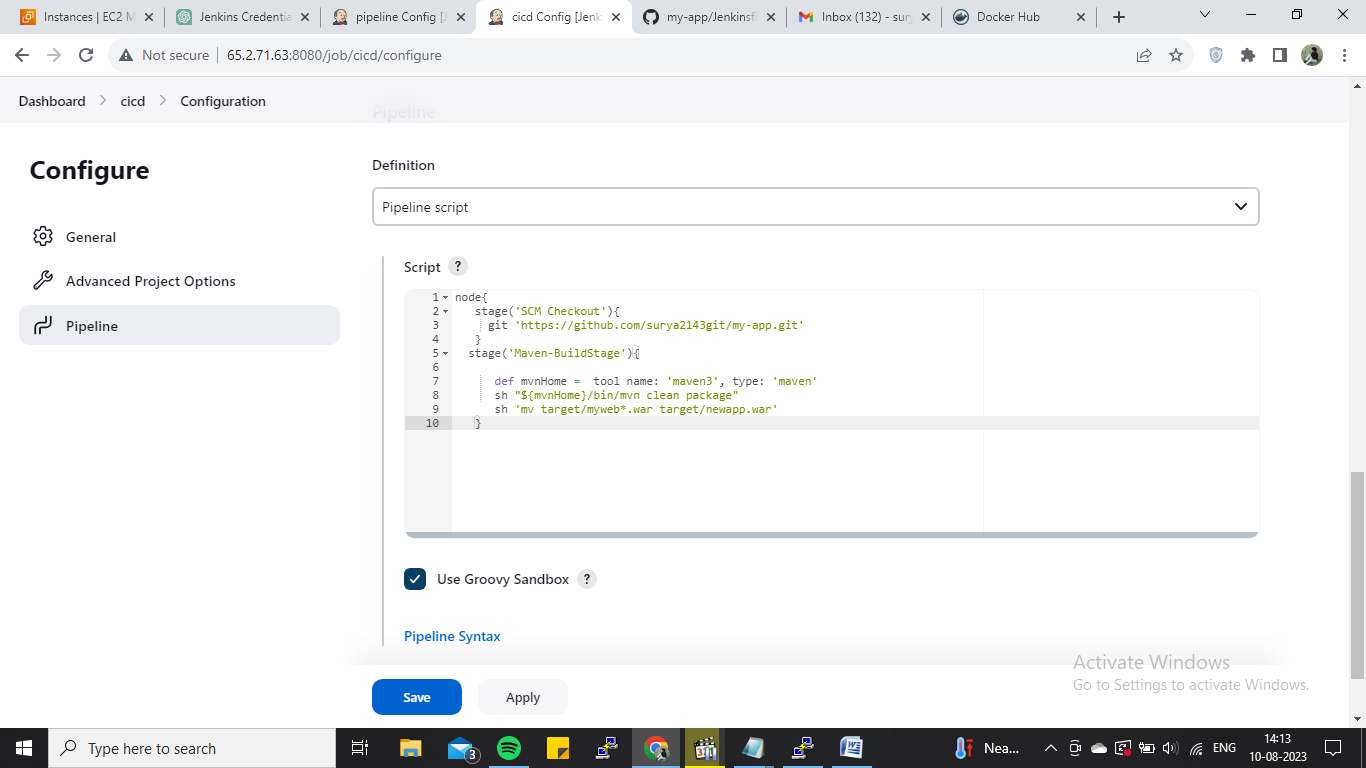
**Stage1 – Check the git repository**





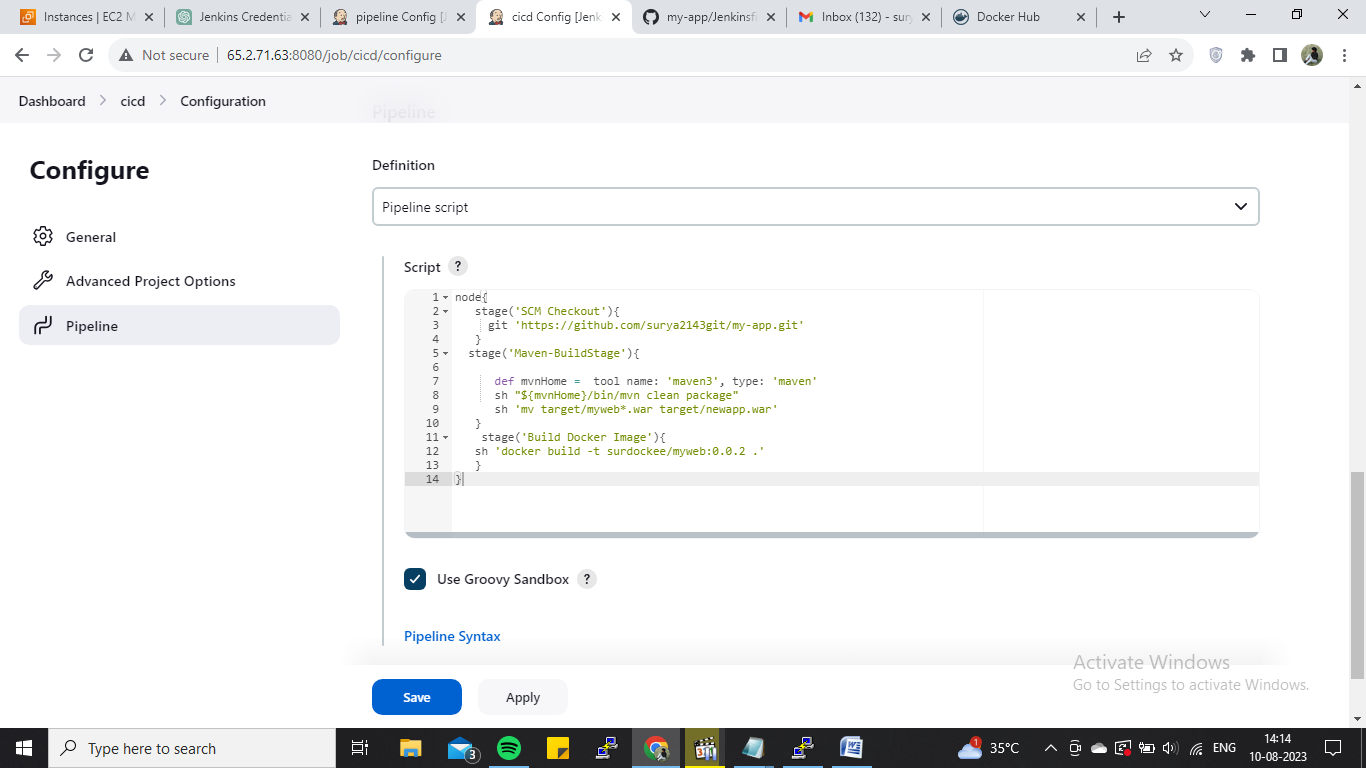
* Installed maven in Jenkins same server, in the /opt path.
* To install any tools in jenkins, we need to install plugins and configured the homepath, did it.
* Configured the MAVEN code to the stage and build it for build the code.

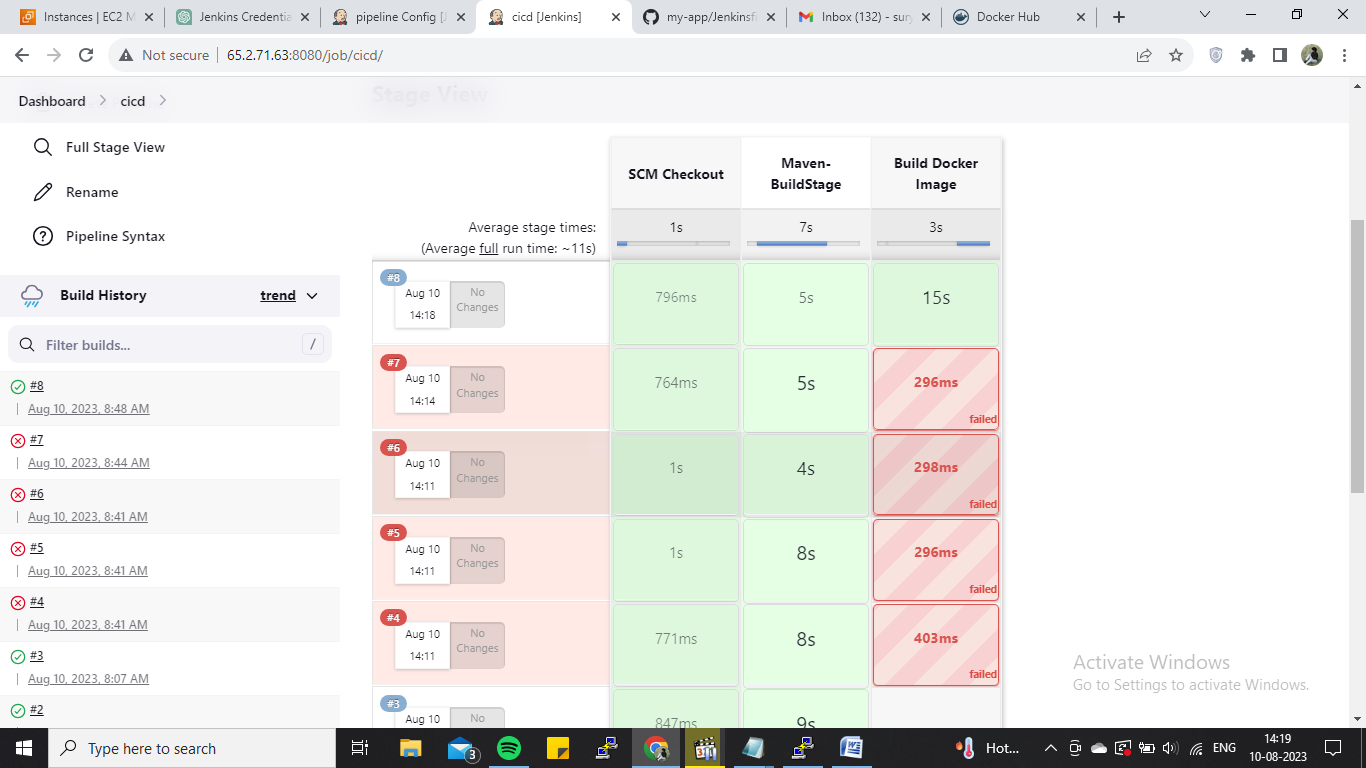
**Stage2 – Install maven**



* Installed docker to install the TOMCAT image for deploy the html file in browser.
* Configured the docker build image code to staging and builded it for deployment.

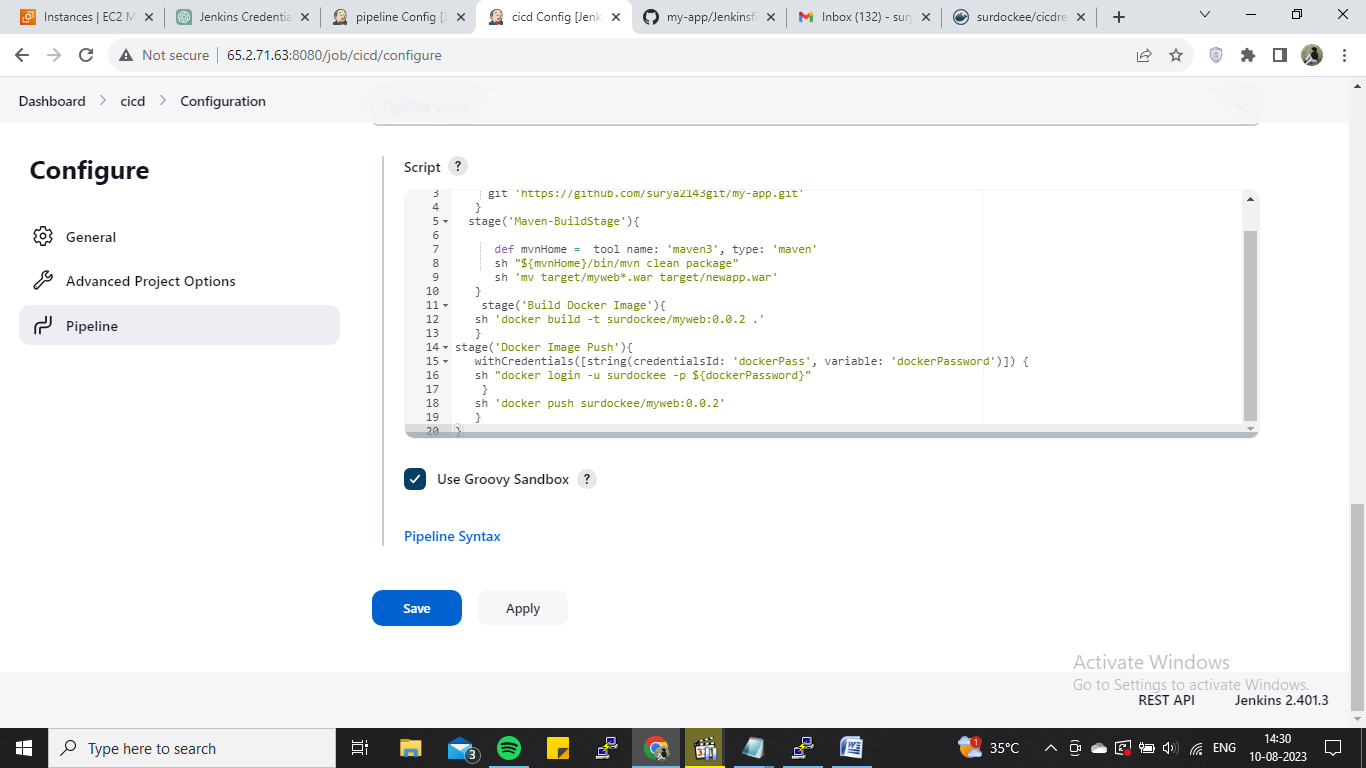
**Stage 3-Create docker image using docker file in github : version as build number**

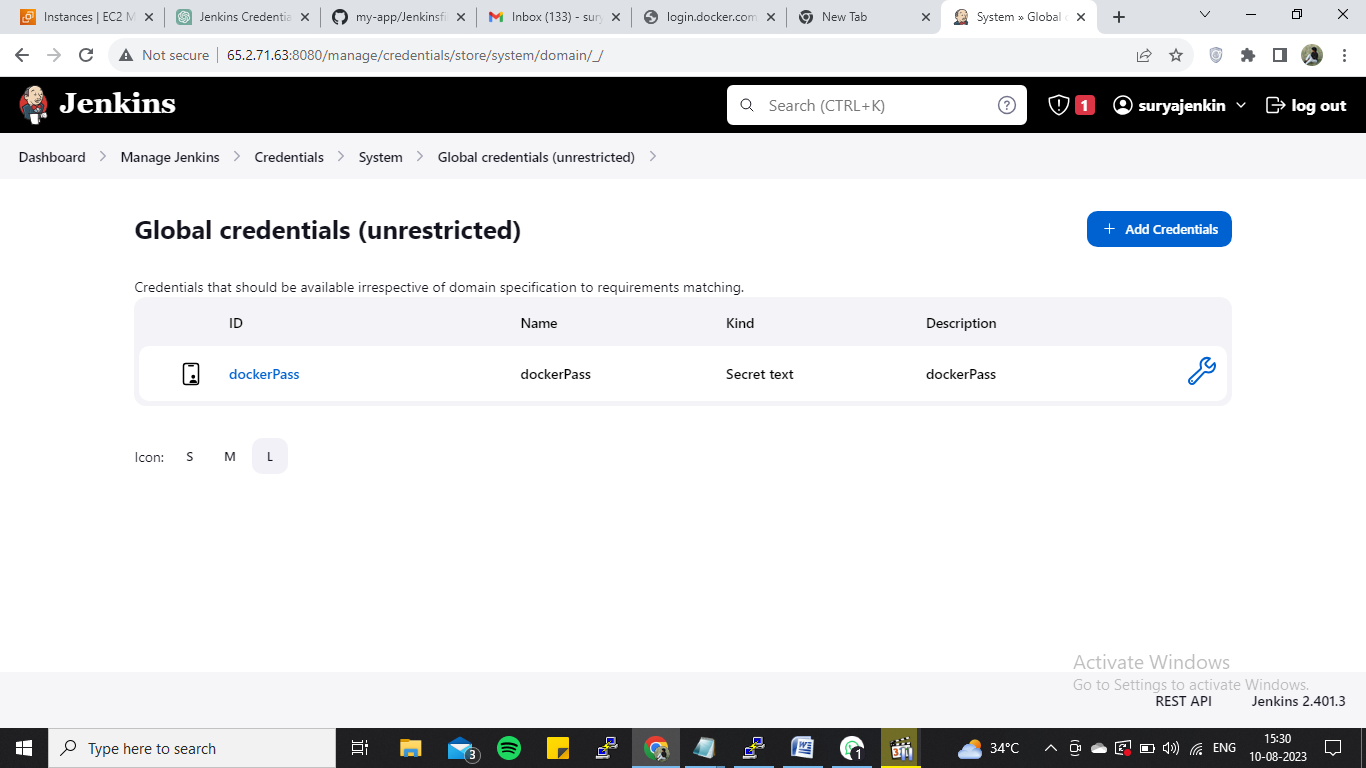


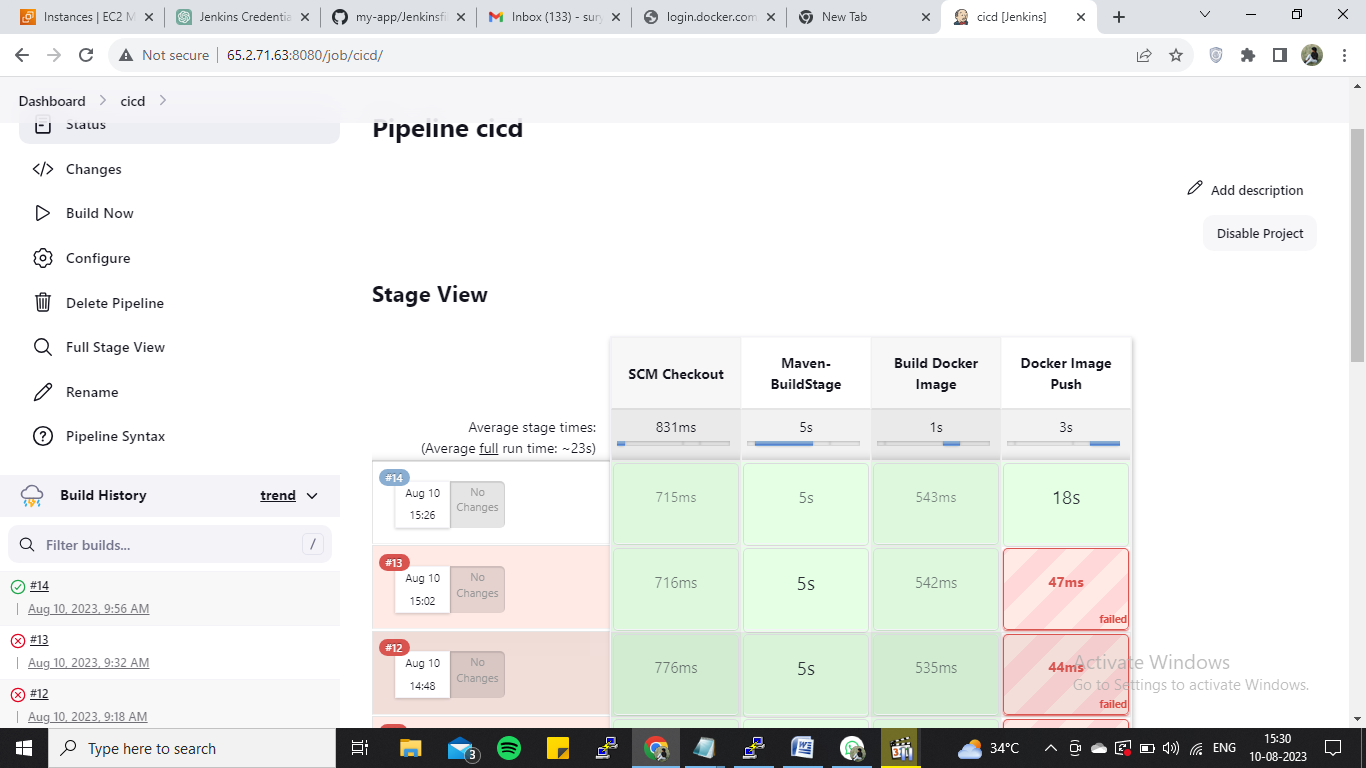


* Pushed into the Docker hub.
* For this need to give docker credentials to access in the Jenkins.
* And giving them all permissions to get the docker in.
* Given the credentials.

**Stage4 – Push image to docker hub and log in**

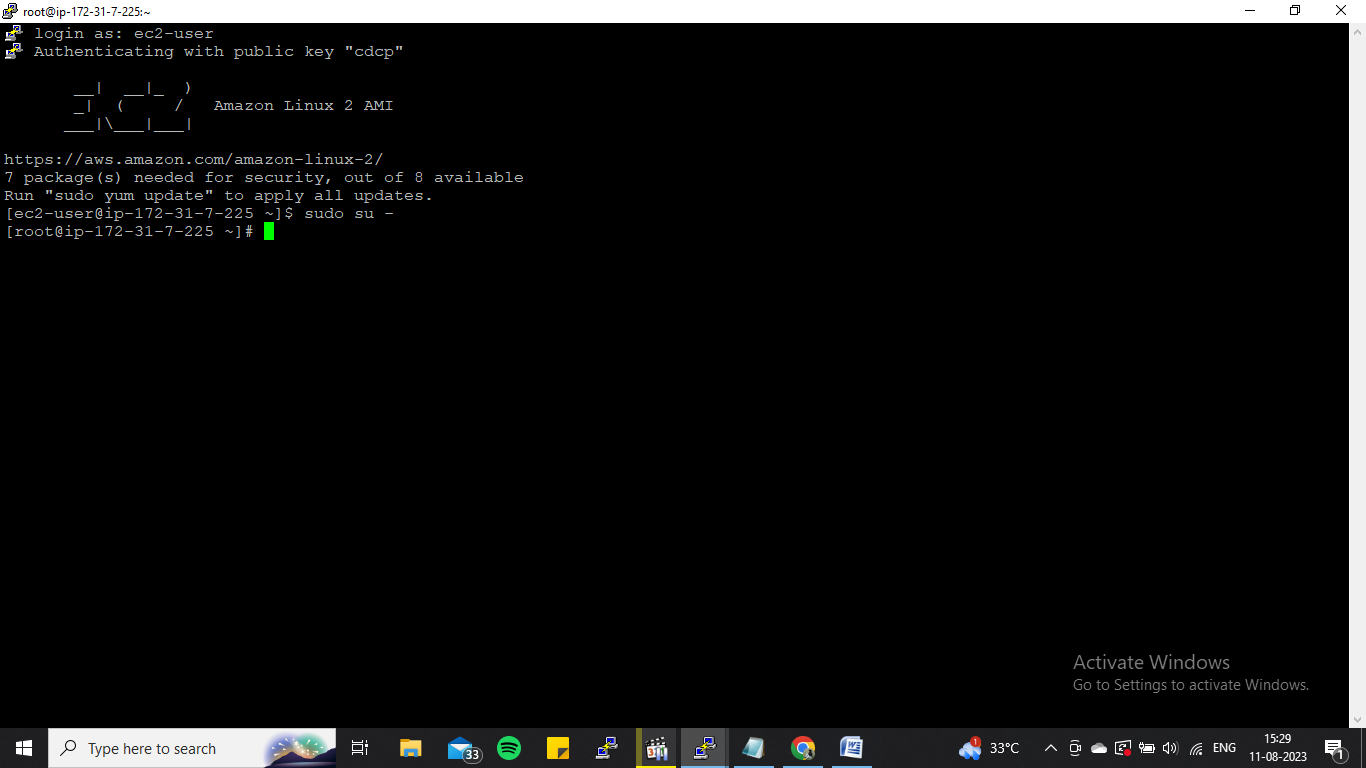




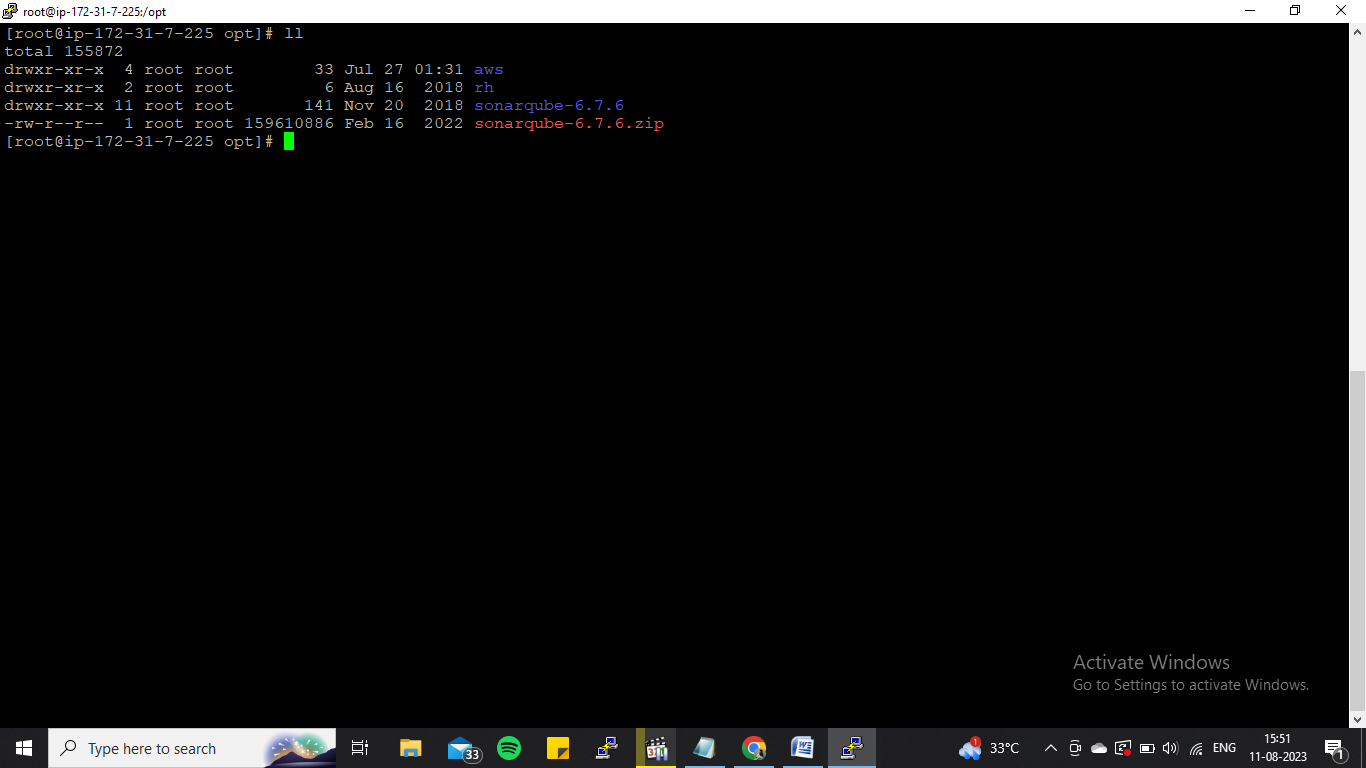


**INSTANCE FOR SONARQUBE:**

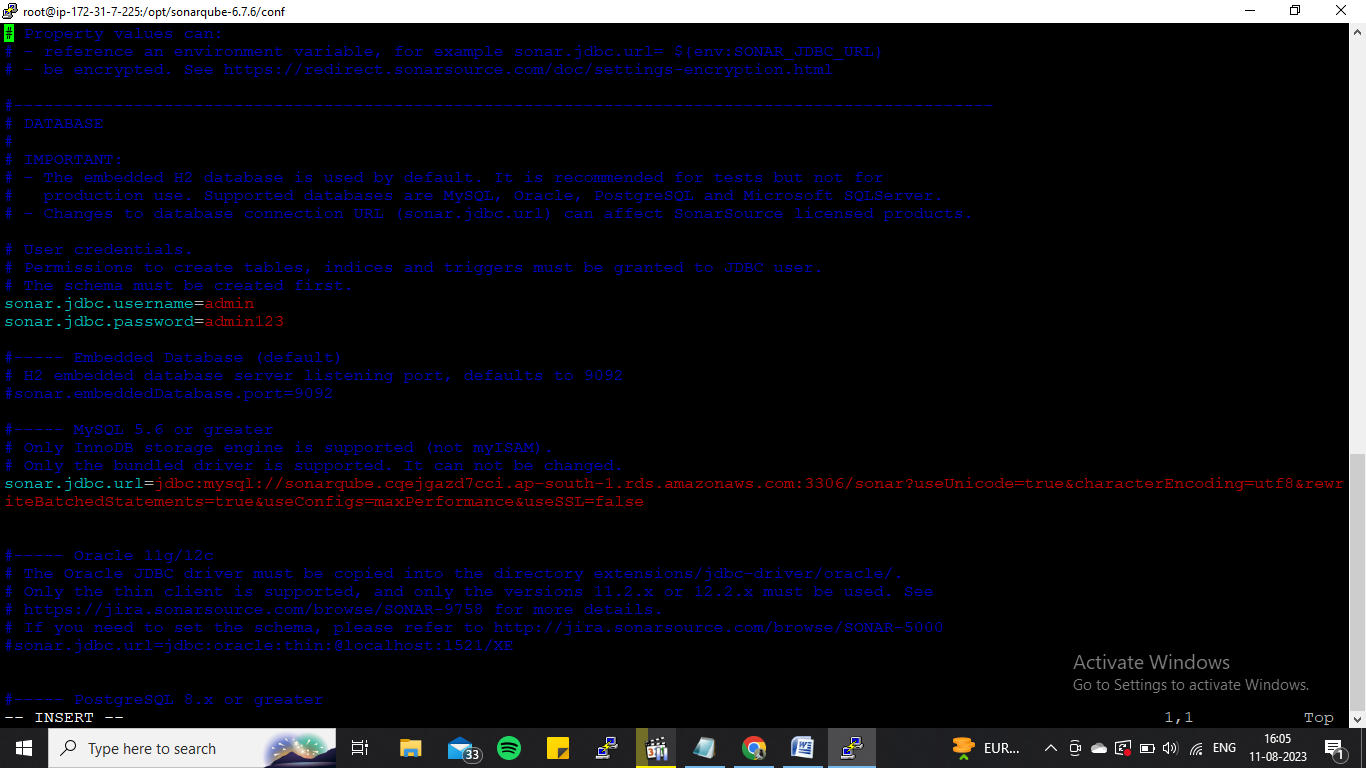
* Created another EC2 instance for installing the SONARQUBE with same aspects like previous instance which as t2.medium, All tcp, storage 30GB.

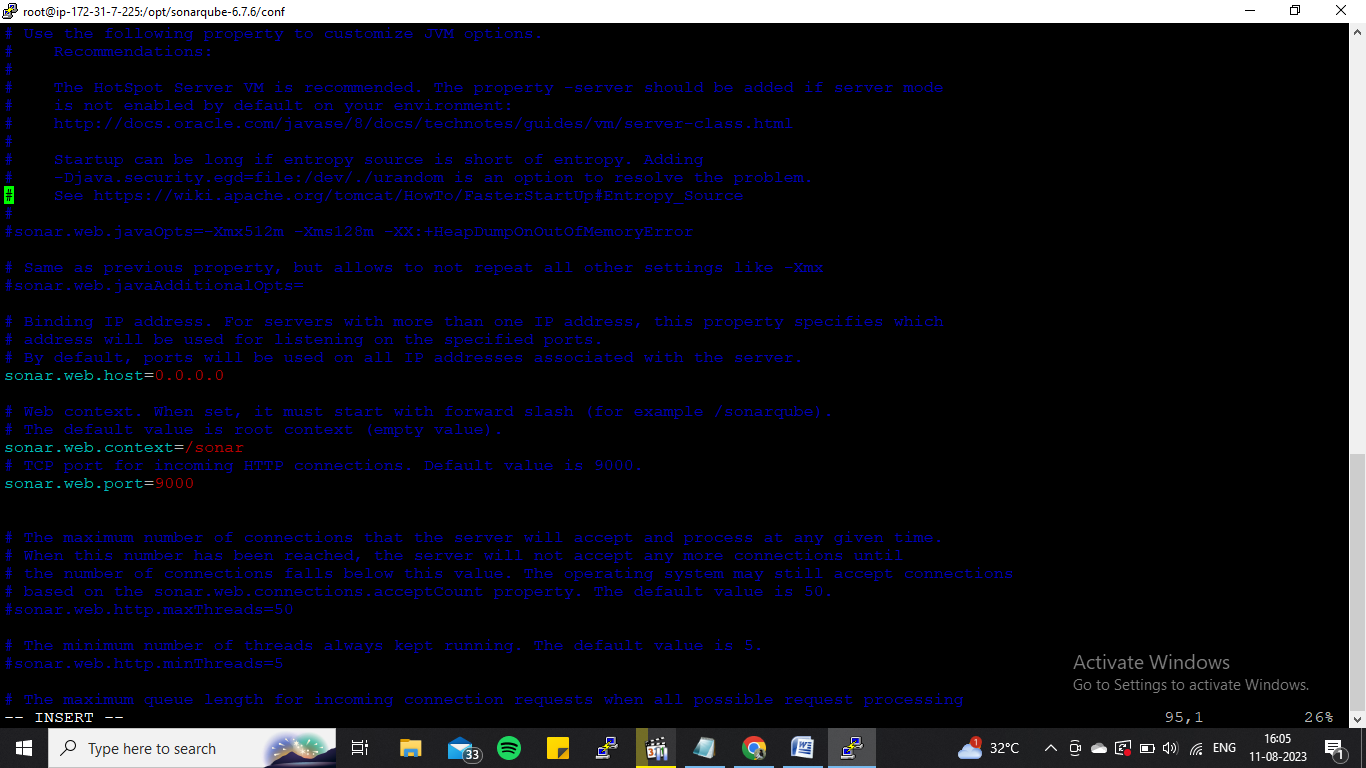


* Installed amazon extras –epel
* Created RDS in AWS console with mysql
* Installed mysql and the Sonarqube for test the code.

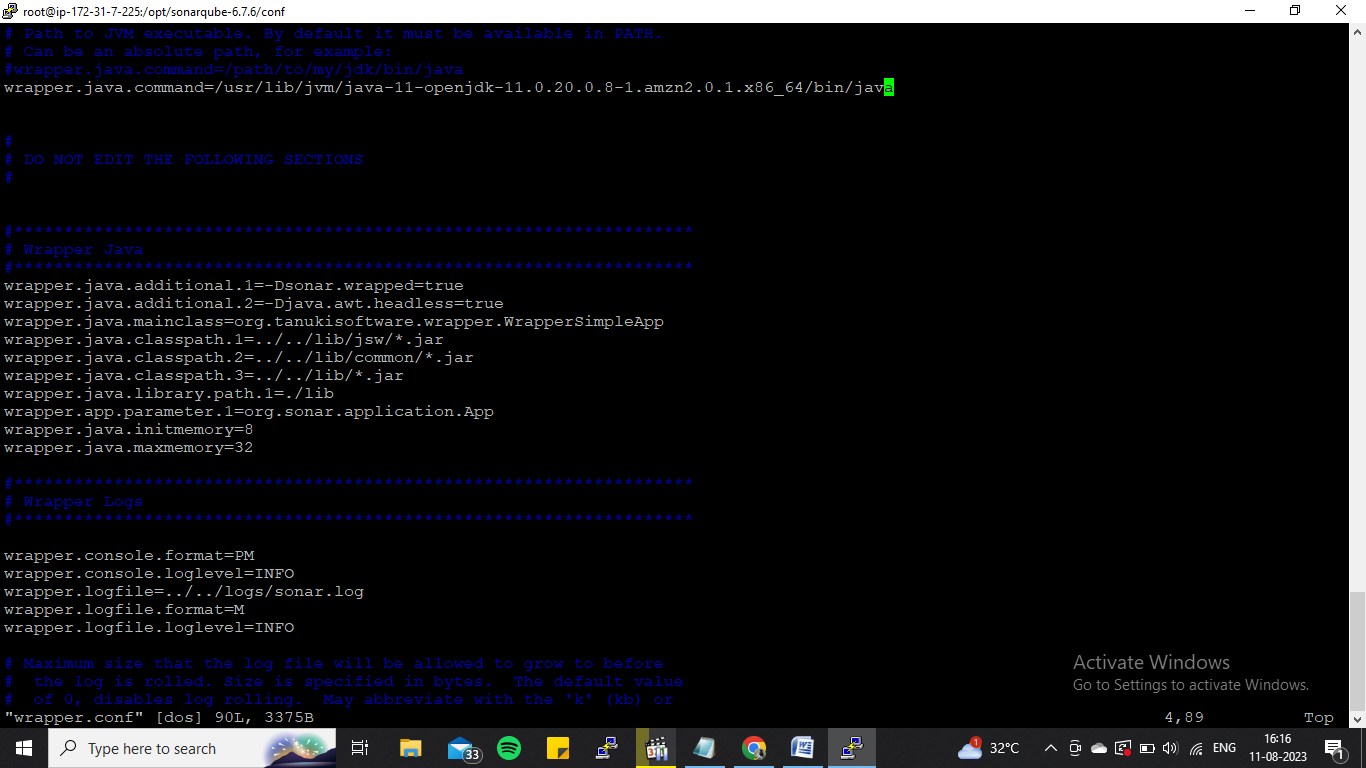


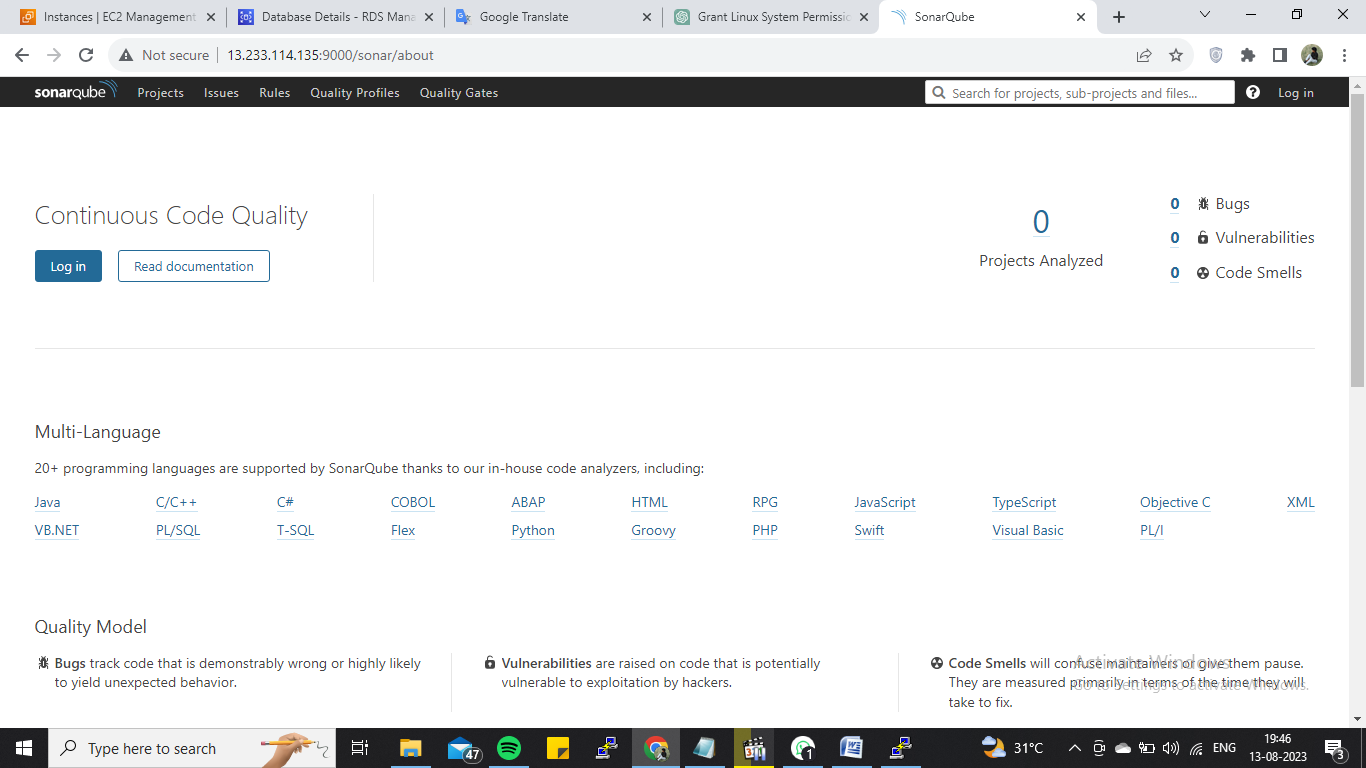
* Change such permissions and some properties in sonar conf files.



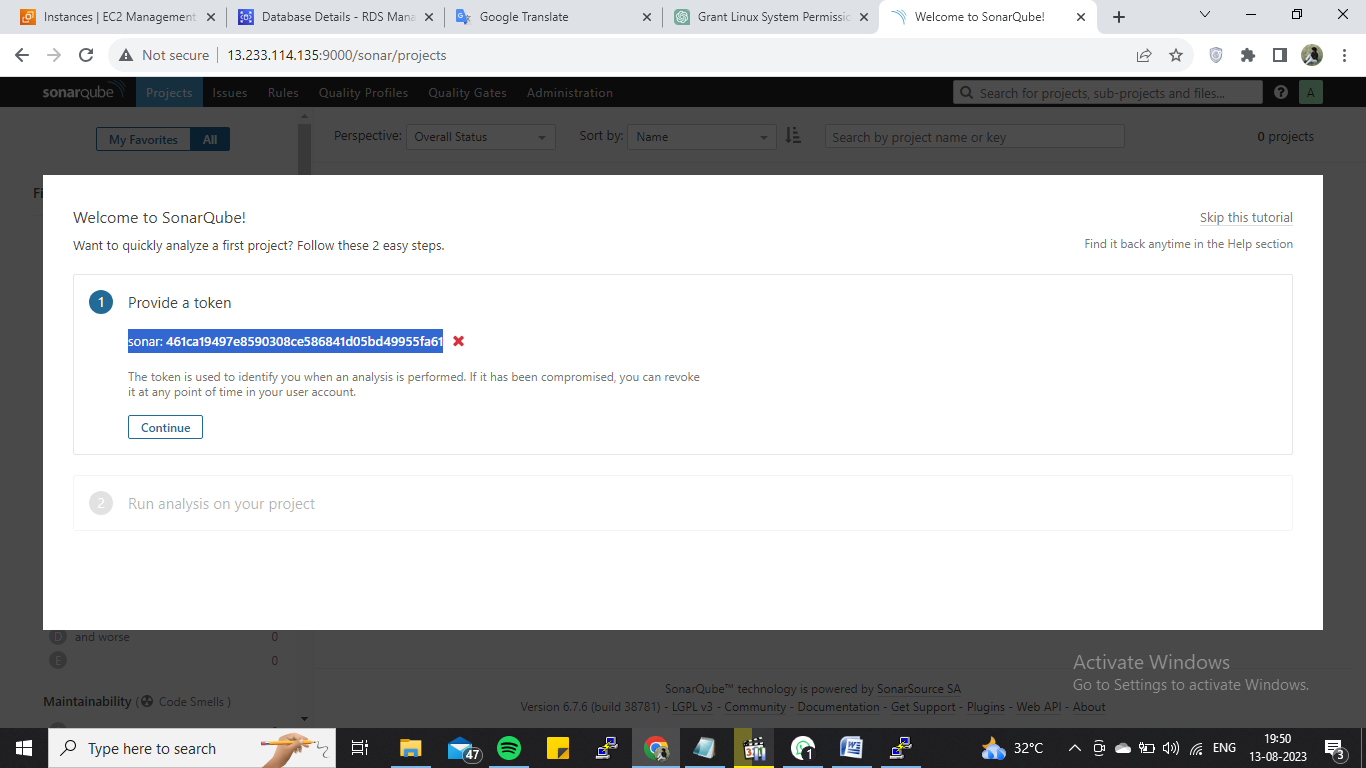


* Update the changes.
* Using sudo update alternatives.
* Also, created such mysql database using mysql command.

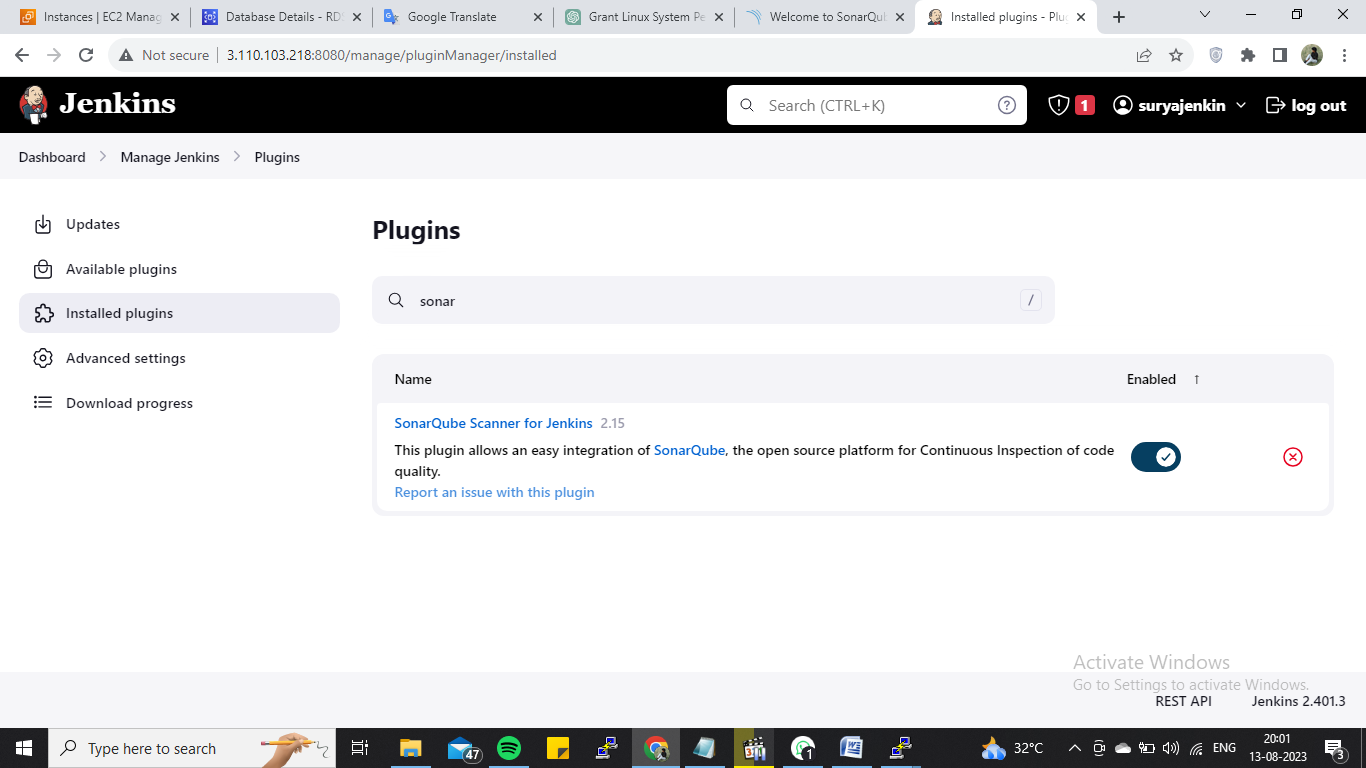




* Generating the token in sonar for get the acces to Jenkins.

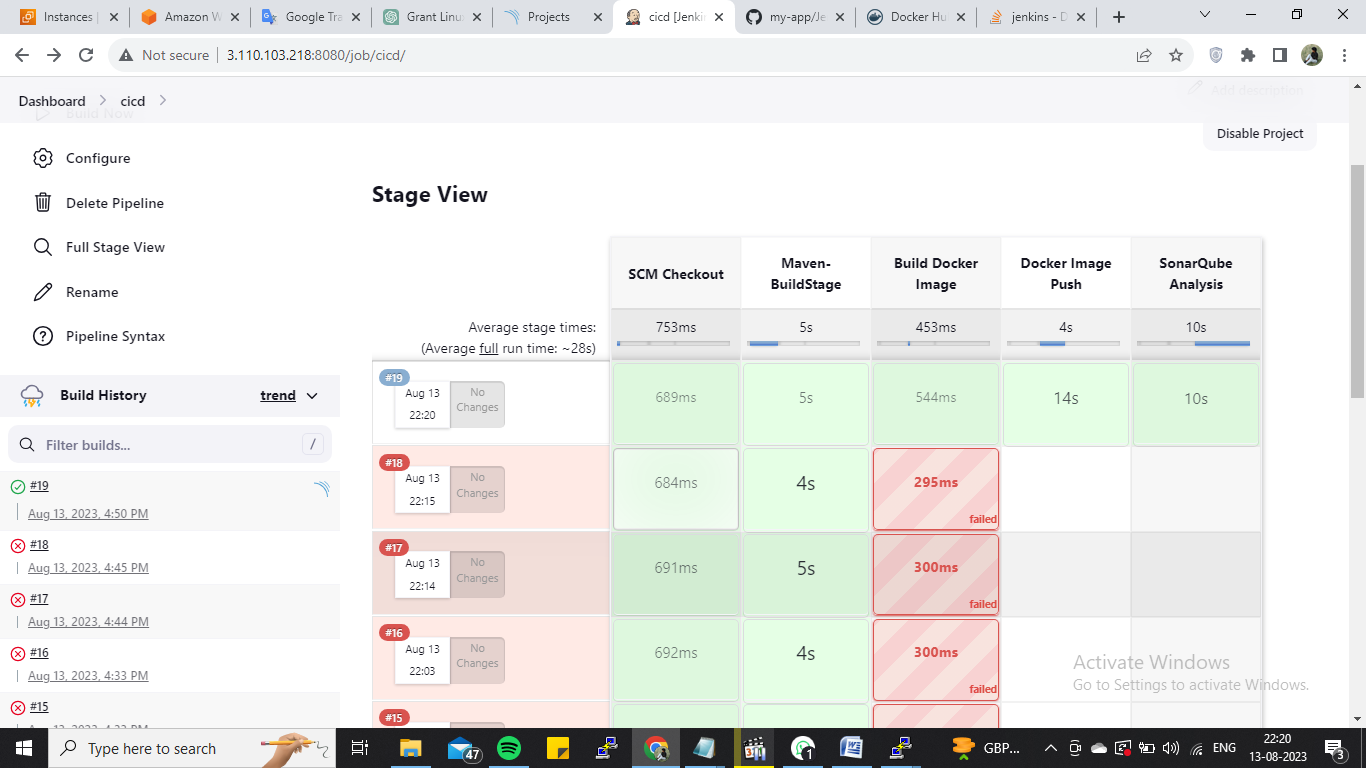


* Installing the sonar plugin and configured the homepath using the token, before this need to give secret credentials.



* Then code was configured and comes under staging successfully for testing the code quality.

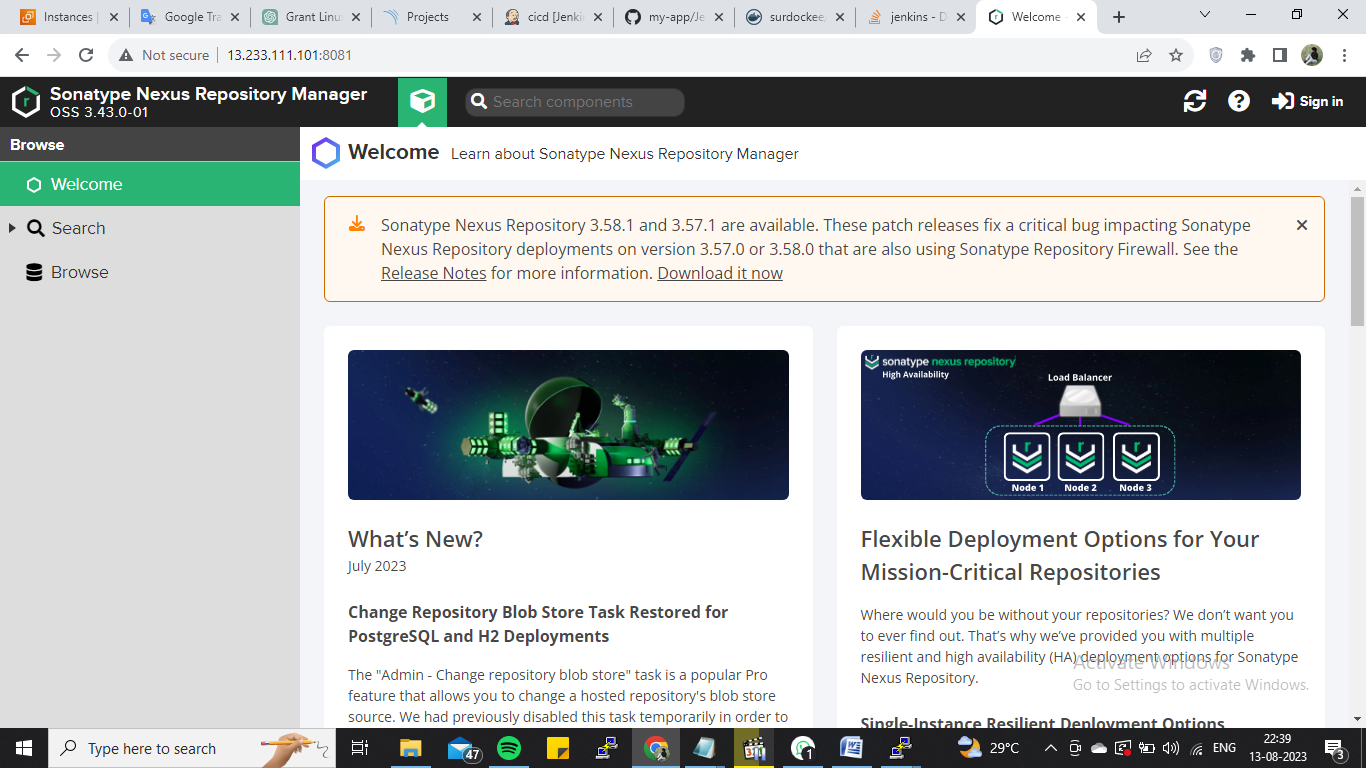
**Stage5 – for testing**



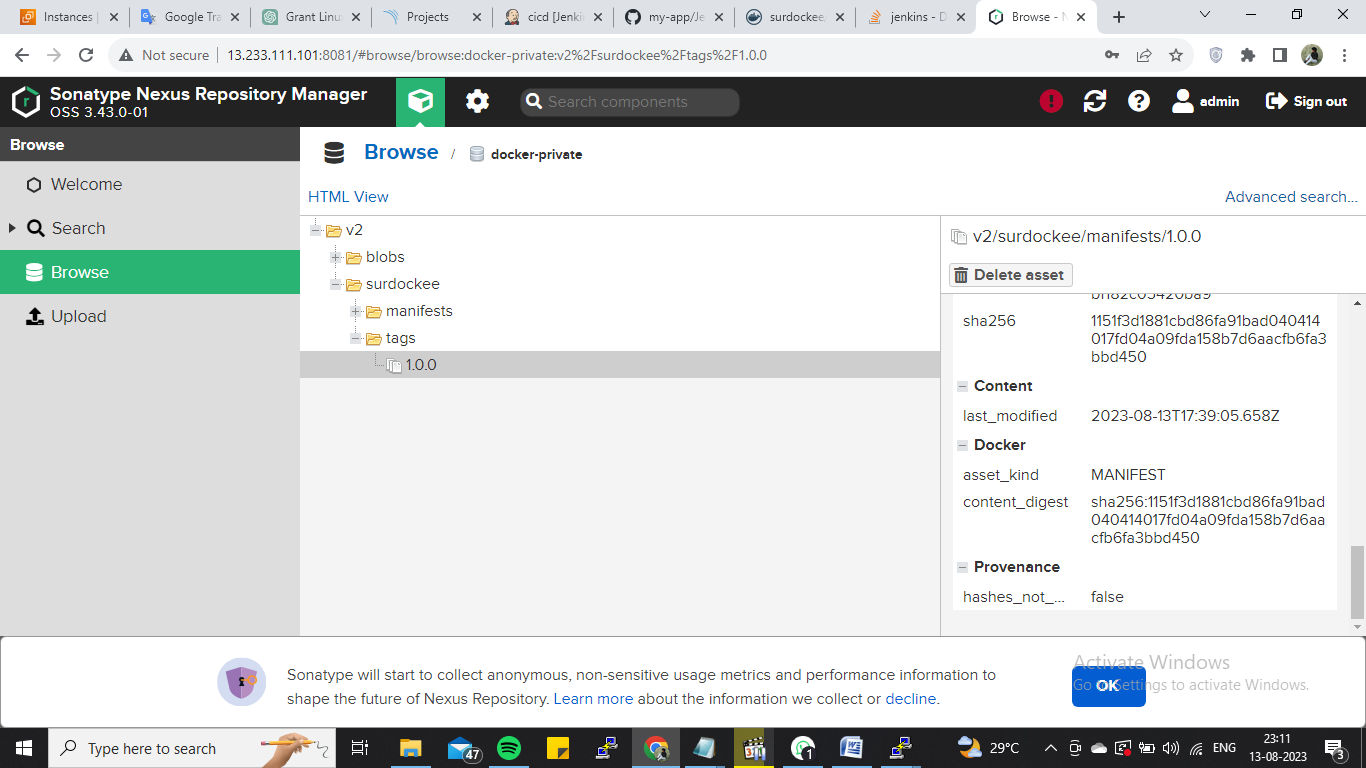
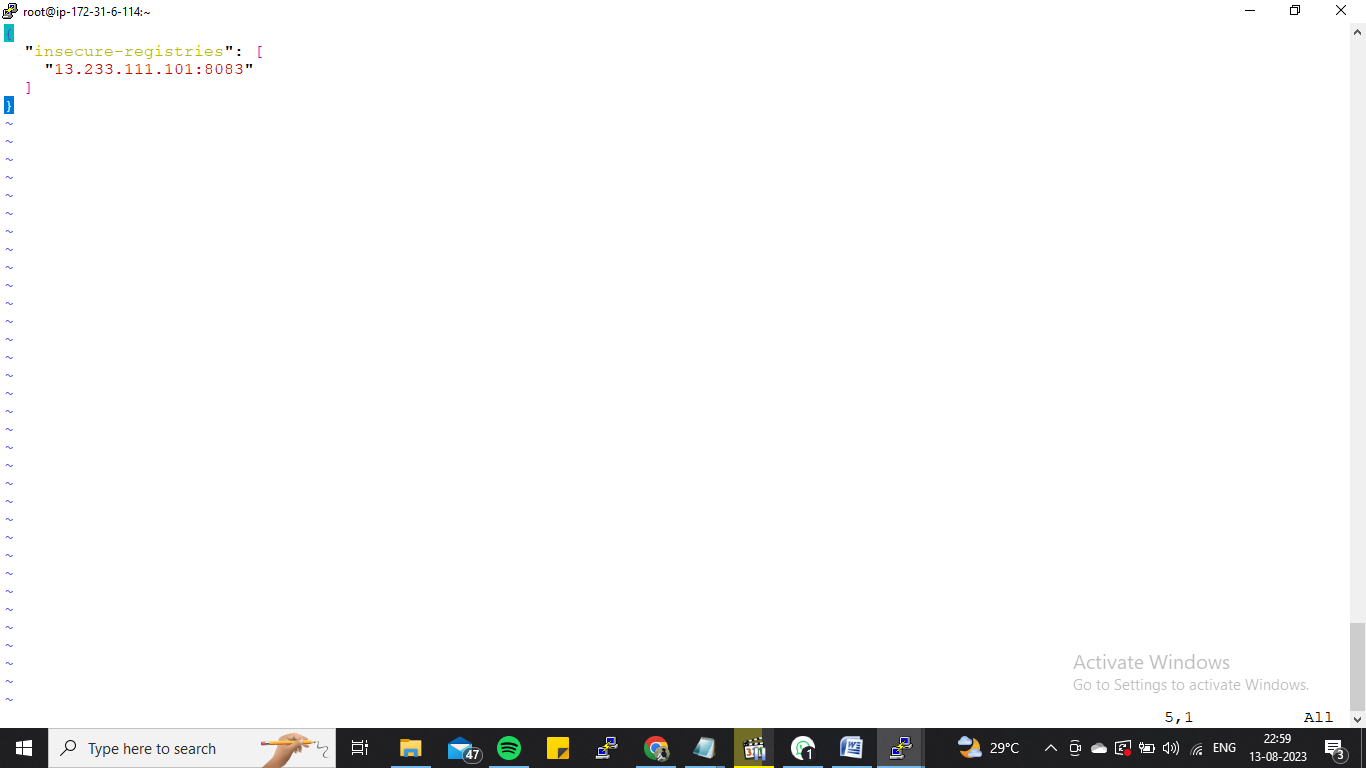
**ANOTHER INSTANCE FOR NEXUS:**

* Creating the instance with same configuration which is created previous for sonar.
* Installed java and the nexus for the artifact repository.

**Stage6 – Store Docker images in Nexus Private Registery**



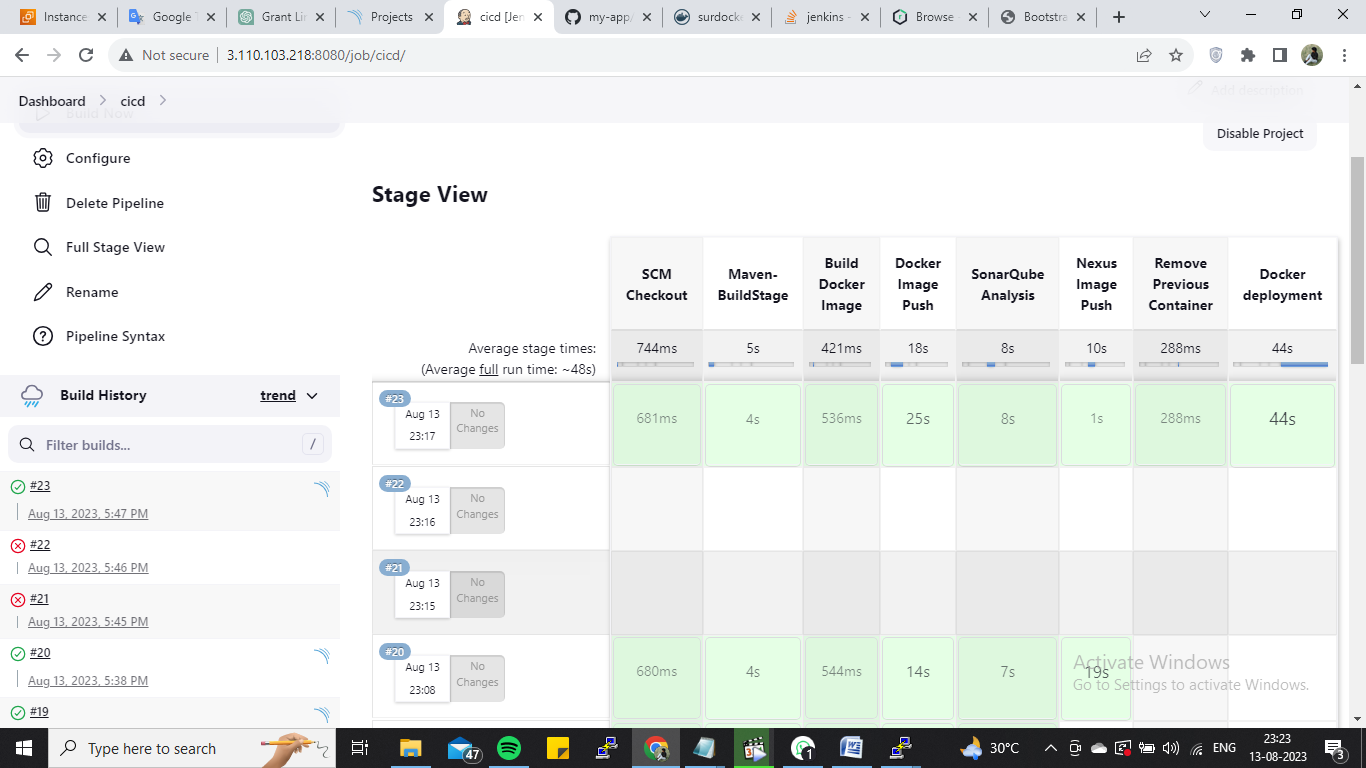
* Then finally, configured the daemon.json file in the docker installed instance for giving the nexus machine ip there with port number.
* And restarted the docker.



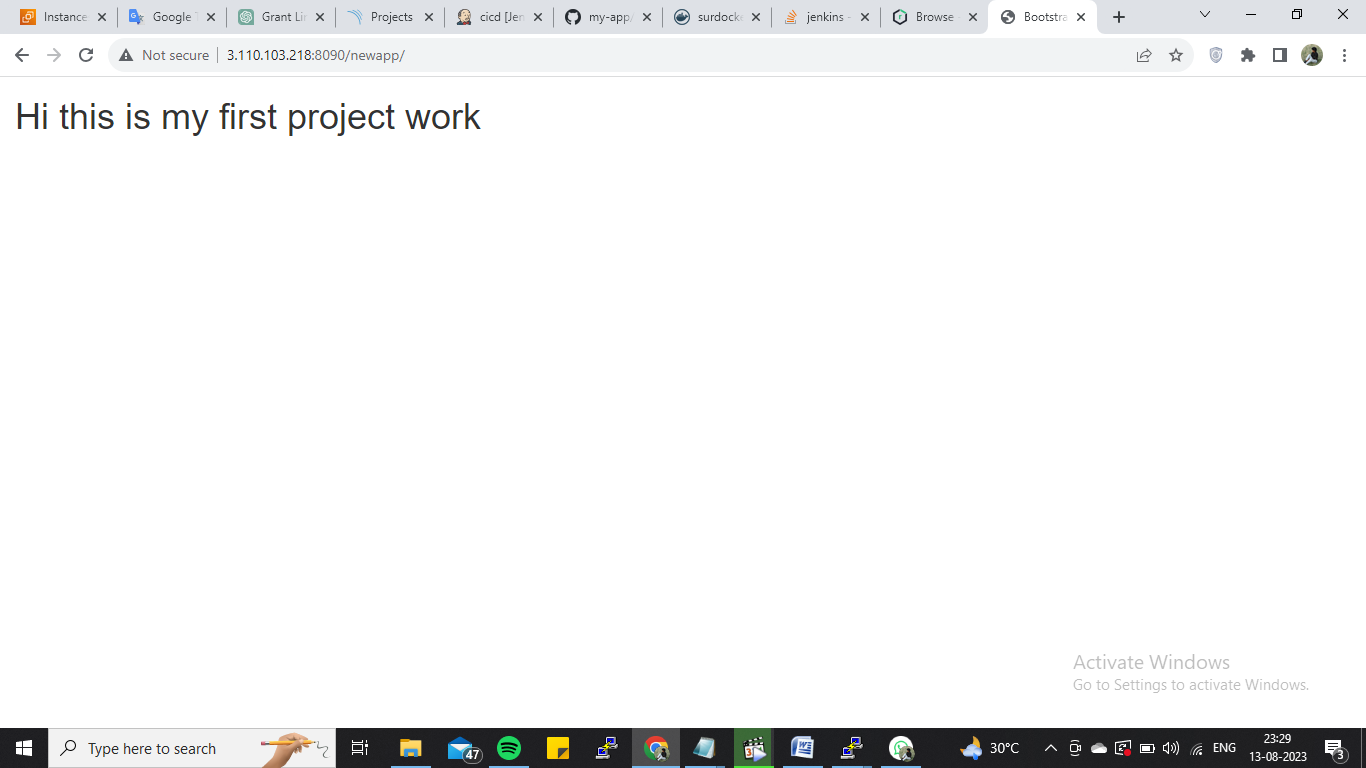
* The Final staging, for removing the old containers in docker.
* And deploy the docker in Virtual Machine.

**Stage7 – Remove Old Container bfr deploying in to VM**

**Stage8 – Deploy the docker image on VM**



* Hit the cicd ip with the given portnumber.

**OUTPUT:**

**PROJECT 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'**

**}**

**stage('SonarQube Analysis') {**

**def mvnHome = tool name: 'maven3', type: 'maven'**

**withSonarQubeEnv('sonar') {**

**sh "${mvnHome}/bin/mvn sonar:sonar"**

**}**

**}**

**stage('Build Docker Image'){**

**sh 'docker build -t saidamo/myweb:0.0.2 .'**

**}**

**stage('Docker Image Push'){**

**withCredentials([string(credentialsId: 'dockerPass', variable: 'dockerPassword')]) {**

**sh "docker login -u saidamo -p ${dockerPassword}"**

**}**

**sh 'docker push saidamo/myweb:0.0.2'**

**}**

**stage('Nexus Image Push'){**

**sh "docker login -u admin -p admin123 13.233.160.223:8083"**

**sh "docker tag saidamo/myweb:0.0.2 13.233.160.223:8083/damo:1.0.0"**

**sh 'docker push 13.233.160.223:8083/damo:1.0.0'**

**}**

**stage('Remove Previous Container'){**

**try{**

**sh 'docker rm -f tomcattest'**

**}catch(error){**

**// do nothing if there is an exception**

**}**

**stage('Docker deployment'){**

**sh 'docker run -d -p 8090:8080 --name tomcattest saidamo/myweb:0.0.2'**

**}**