**Hand-on Lab CI setup**

Contents

[Logon to Jenkins 2](#_Toc22045005)

[Create a new job 2](#_Toc22045006)

[Selecting the source code for Jenkins job 2](#_Toc22045007)

[Configuring build on Jenkins Job 3](#_Toc22045008)

[Configuring sonarqube for build token 4](#_Toc22045009)

[Configuring Sonarqube scanning in Jenkins build 7](#_Toc22045010)

[Triggering the Jenkins Build 8](#_Toc22045011)

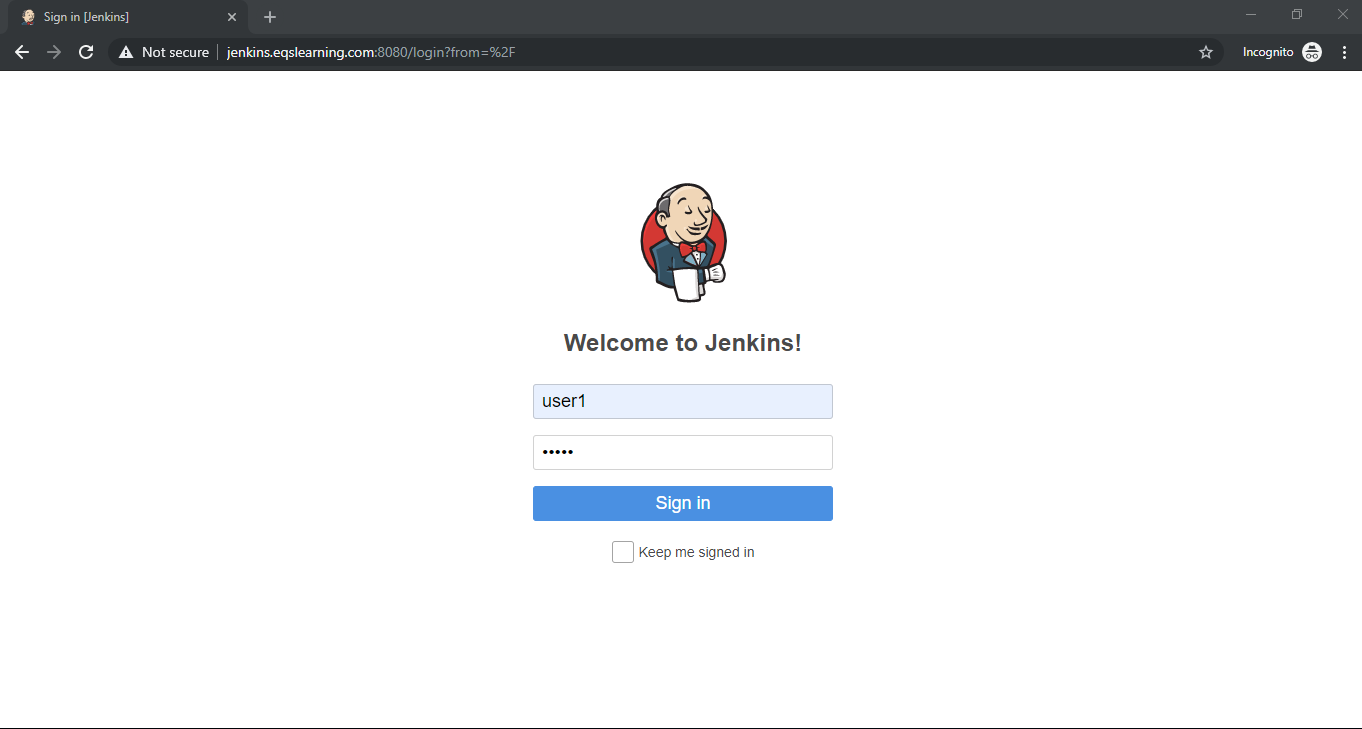
[Verify the build logs 9](#_Toc22045012)

[Verifying the Code quality status in Sonarqube 10](#_Toc22045013)

# Logon to Jenkins

Logon to Jenkins with below URL address with user id shared to you. E.g : user1. Password would same as username.

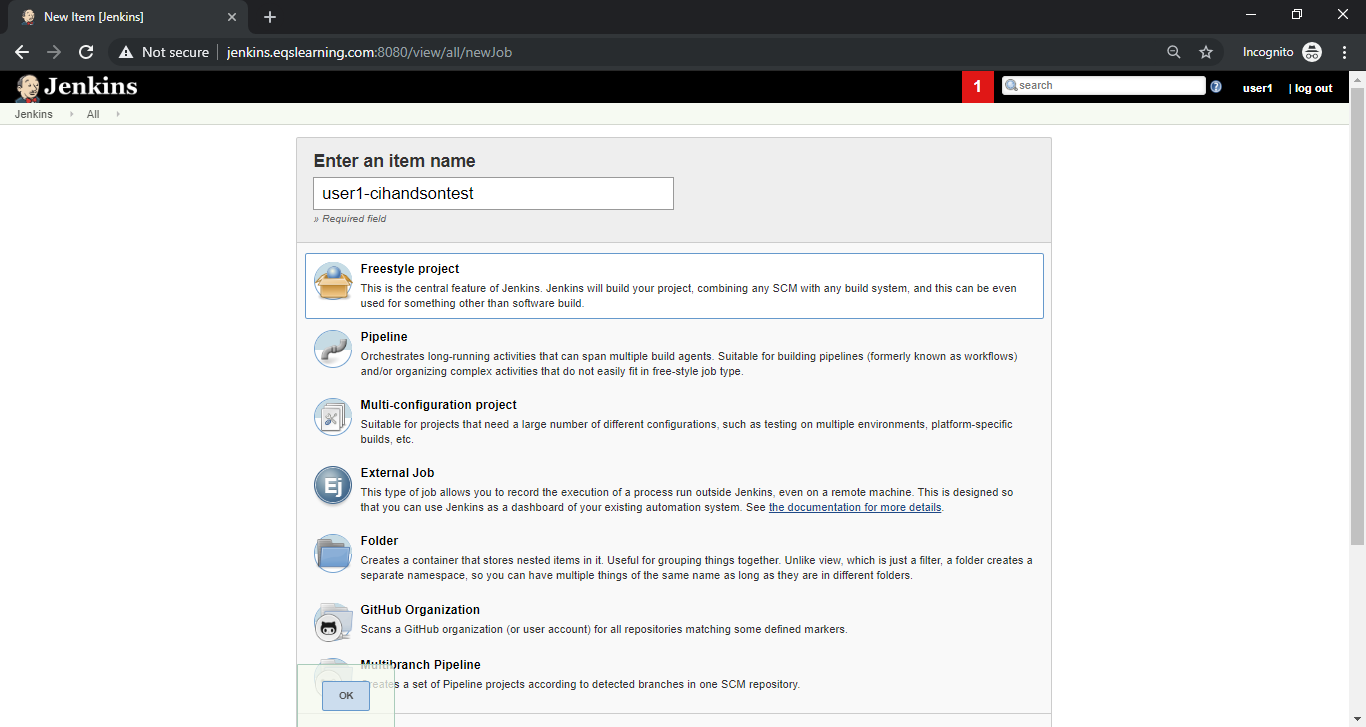
[http://jenkins.eqslearning.com:8080](http://jenkins.eqslearning.com:8080/)



# Create a new job

Click on **New Item** option on the left menu bar to create new Jenkins Job. Select job type as **Freestyle Project and** Enter the job name and click **OK** button on the bottom of the page.

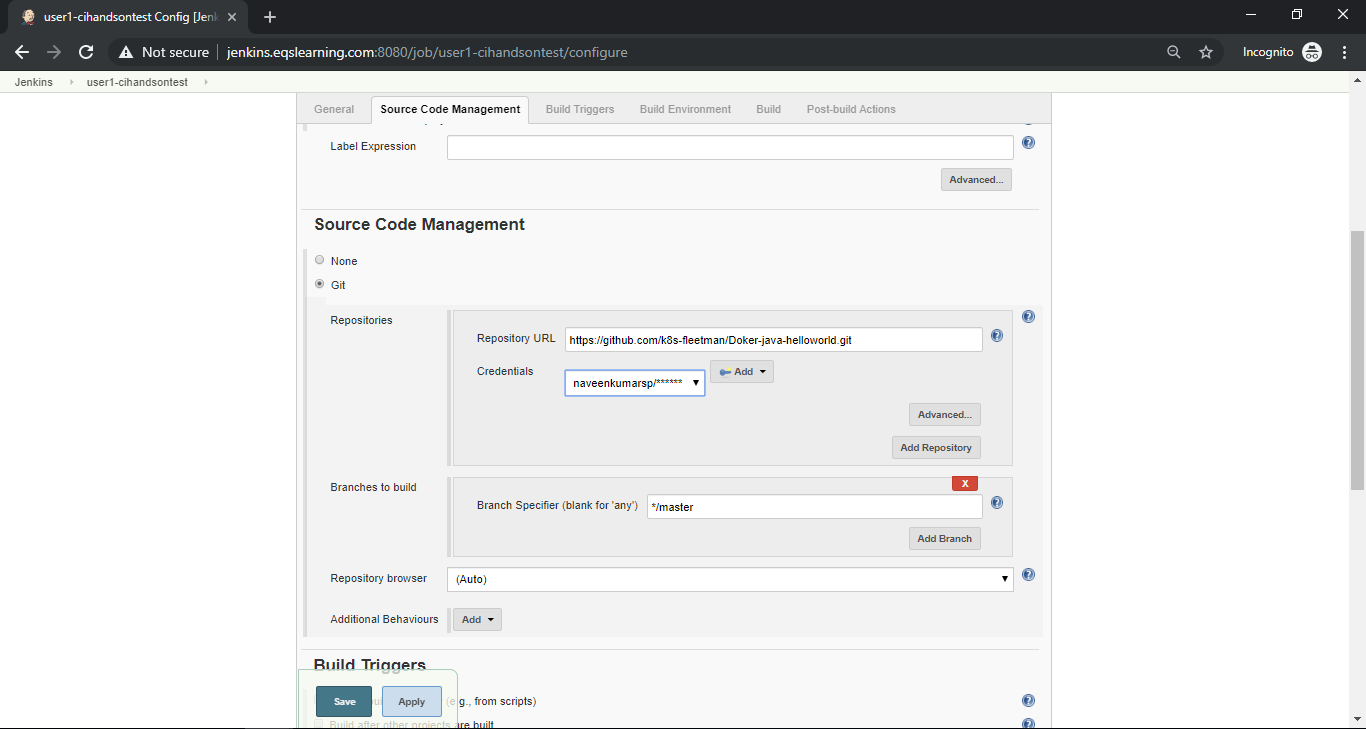
Note: For convenience of others, kindly use the naming convention as **<username>-ci-handson**



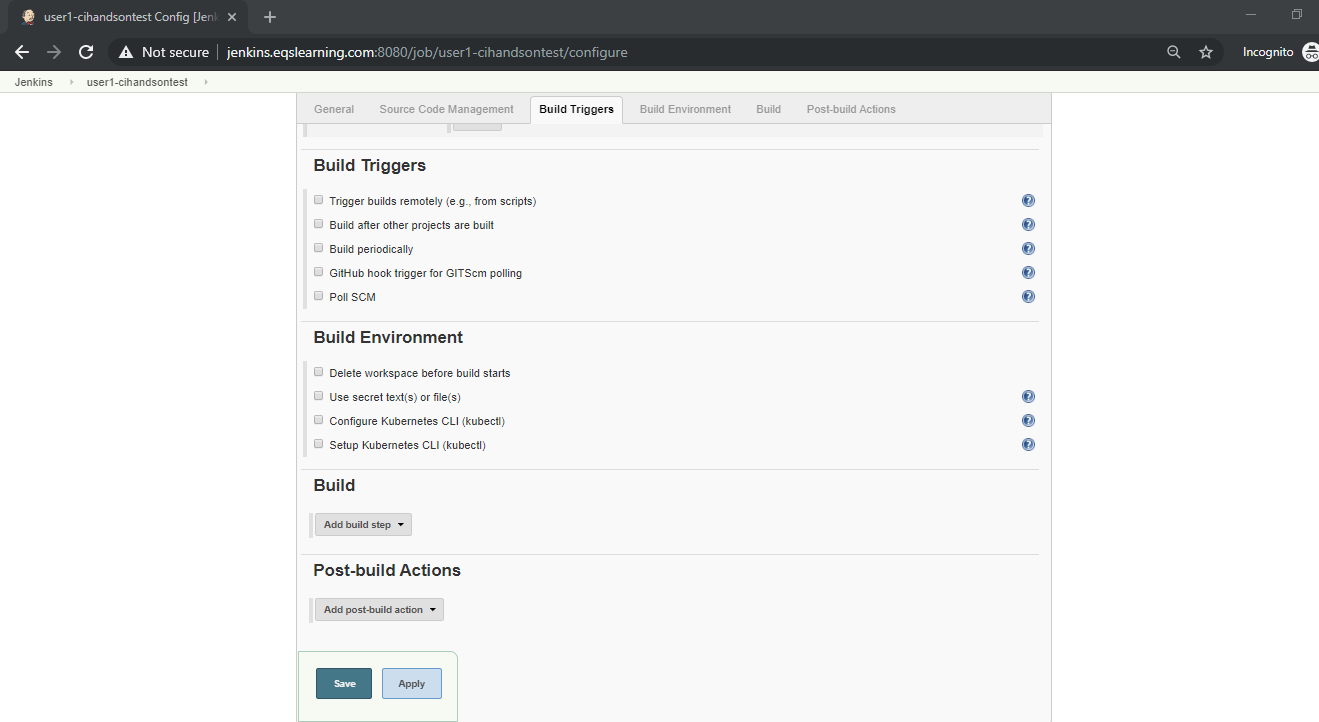
# Selecting the source code for Jenkins job

On newly created Jenkins job, go to **Source Code Management** select **Git** and update the repository address listed below and select pre-configured credential for git as shown below.

https://github.com/k8s-fleetman/Doker-java-helloworld.git



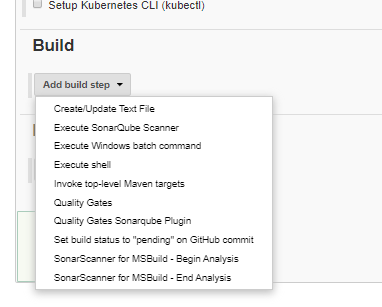
It is possible to trigger the build based on the requirement, we can choose various options including polling SCM for commits, periodic build, web hook for remote trigger etc. Since we are not committing any code to our repository, we will be skipping this option and manually trigger the build.



# Configuring build on Jenkins Job

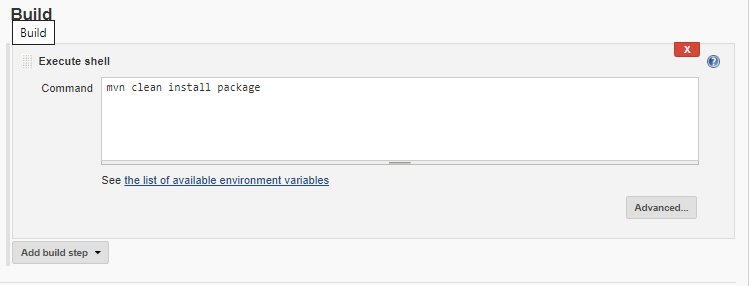
We can install various plug-in which are readily available for build configuration such as Maven build, Execute MS build etc. For our convenience, we will be using Execute shell option to build the code.

Under Build section of the newly created job, select the option “Execute shell”



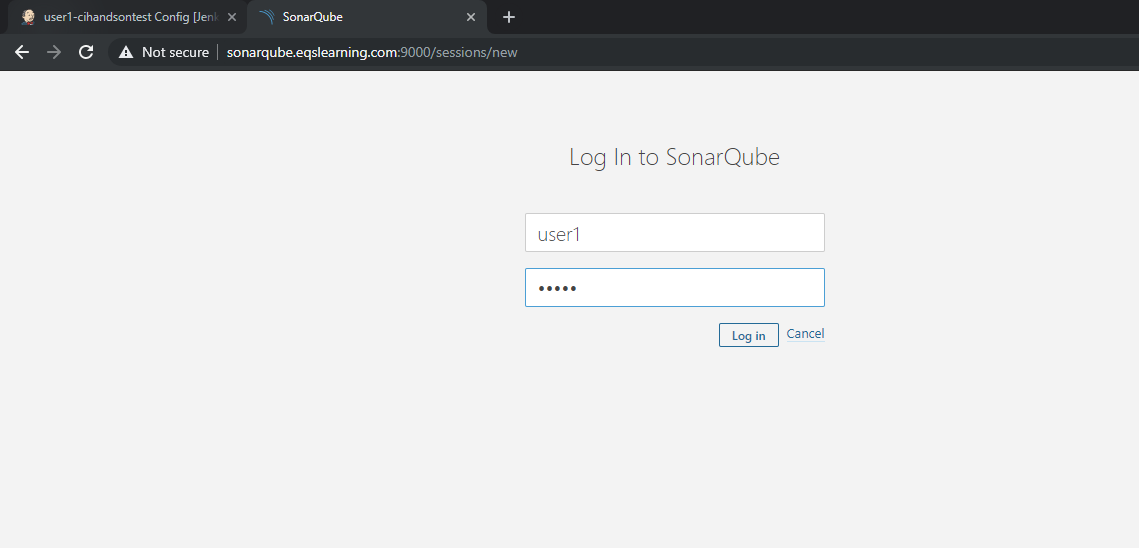
We are using Maven build tool to build our simple java project. Hence enter the below command in the shell window.

***mvn clean install package***

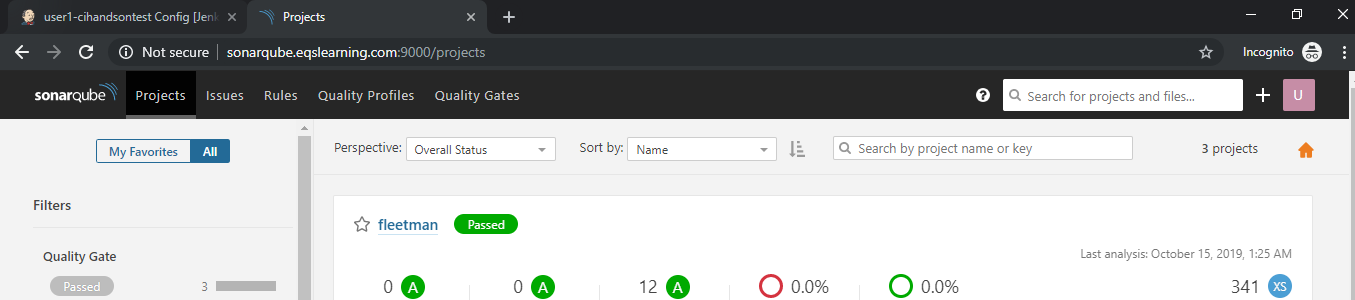


# Configuring sonarqube for build token

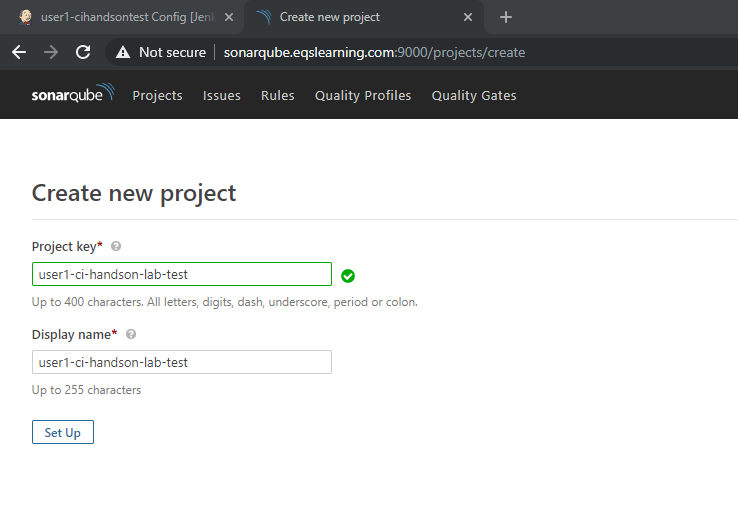
Logon to sonarqube using URL [http://sonarqube.eqslearning.com:9000](http://sonarqube.eqslearning.com:9000/) with your user name and credential which was provided for Jenkins.



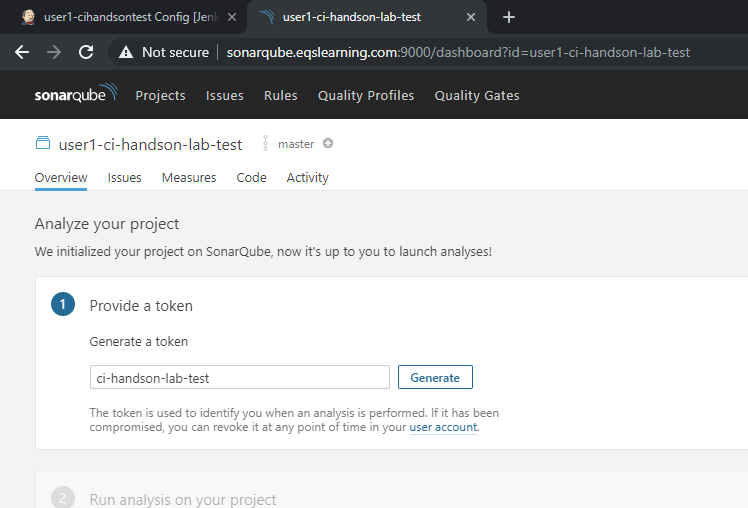
After logon, click on + symbol on the right top corner and select the option “**create new project**”.



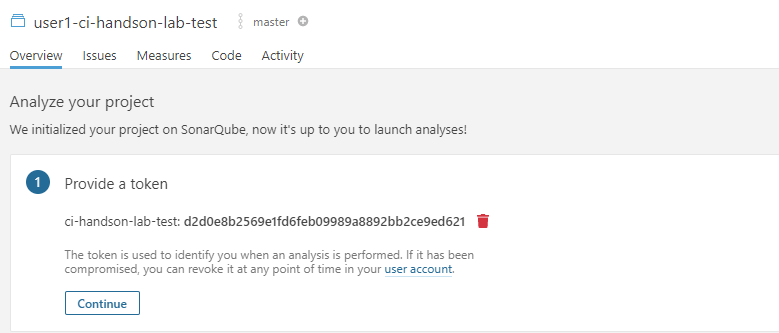
Enter the appropriate name for the project with naming convention and add a description for the project. Click on Setup to get the access token.



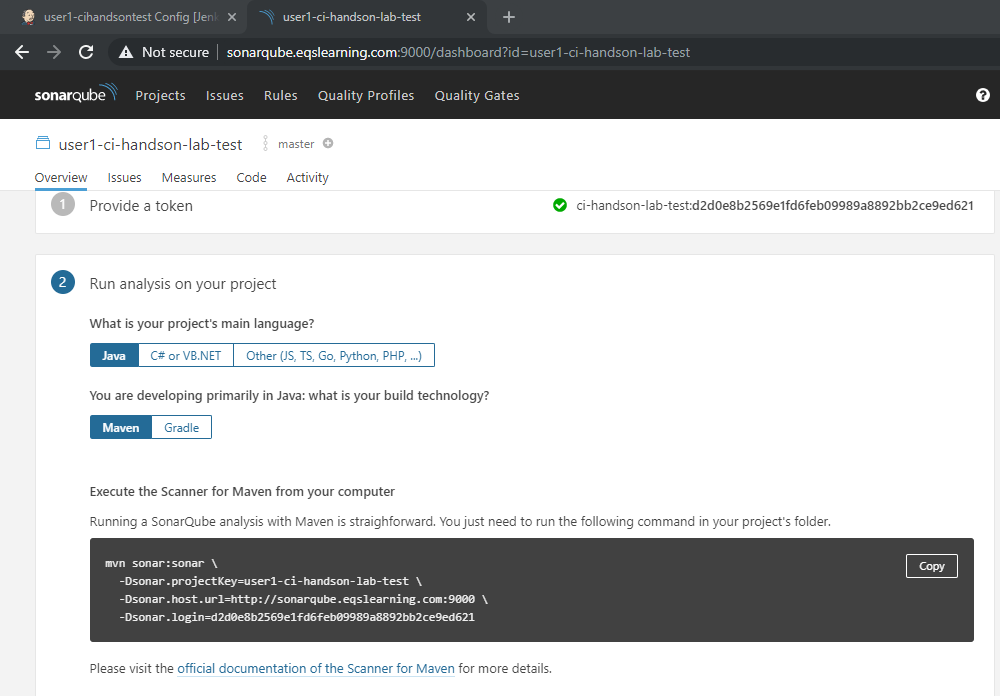
Enter a description to get a new token for the project and click on generate button.



Upon new key generated, click on Continue to get command which we need to execute in our Jenkins build.



Copy the code generated in sonarqube for maven based build on Java language as shown below.

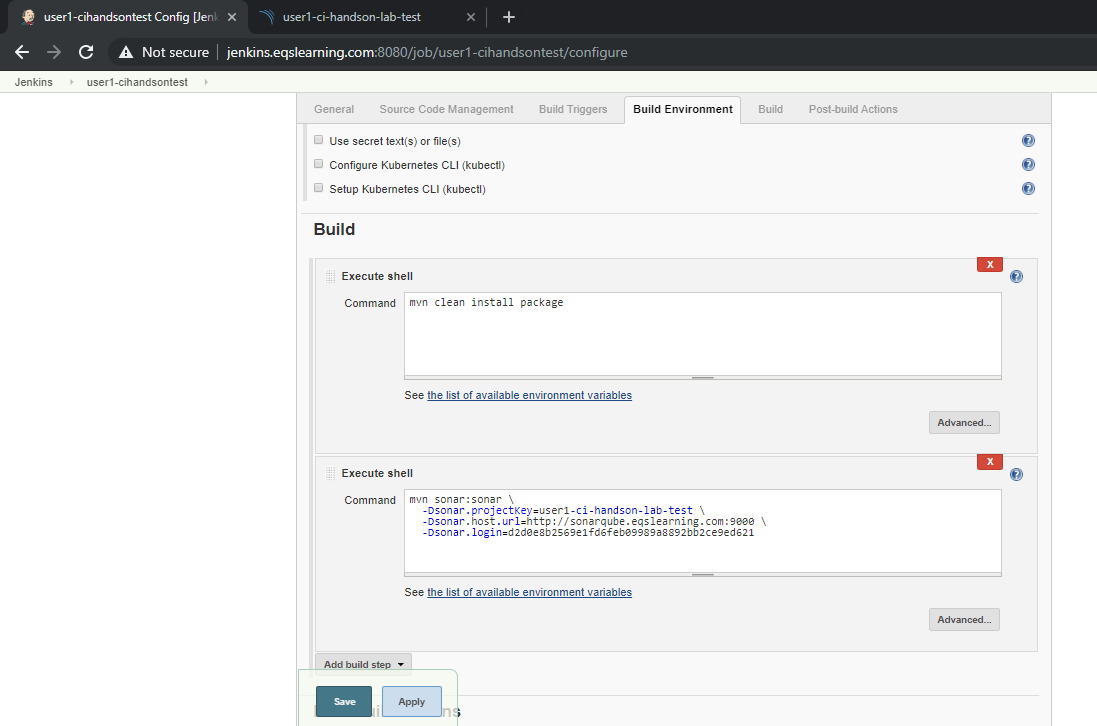


Copy the sonarqube command with toke details and save it for updating in Jenkins job.

# Configuring Sonarqube scanning in Jenkins build

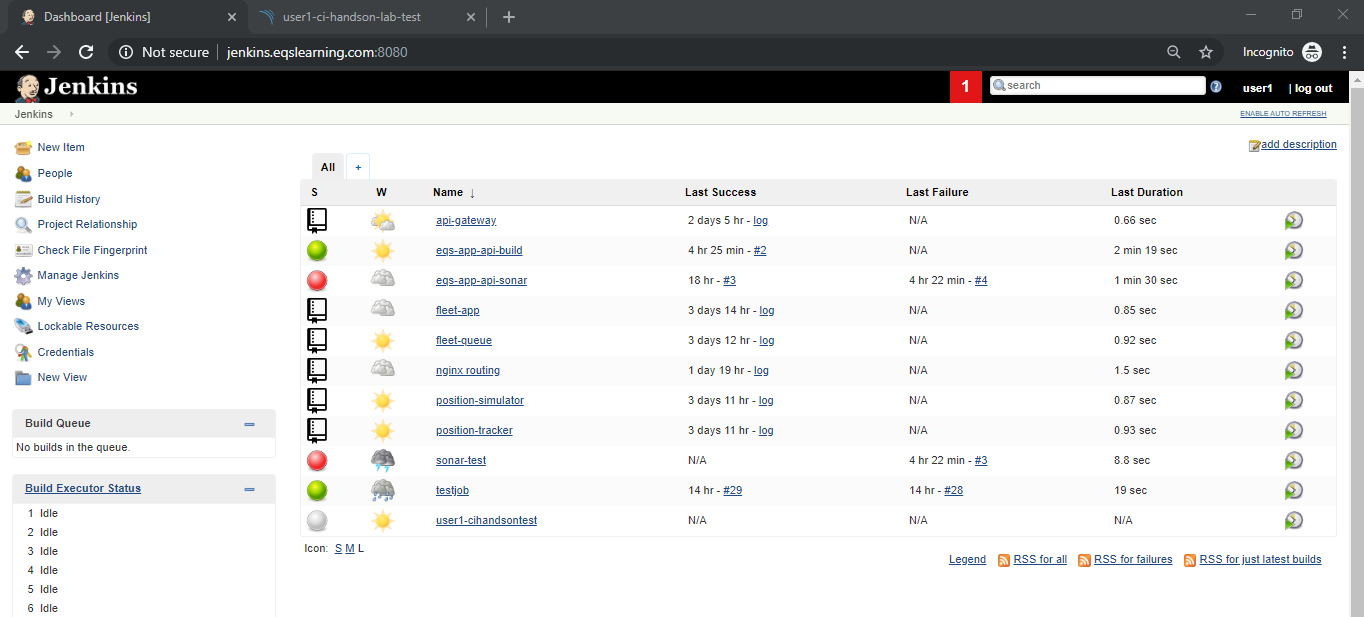
To use sonarqube for verify the code quality, click on “**Add build step**” and select execute step.

Enter the gathered command from sonarqube and click on save/apply button of the build job.

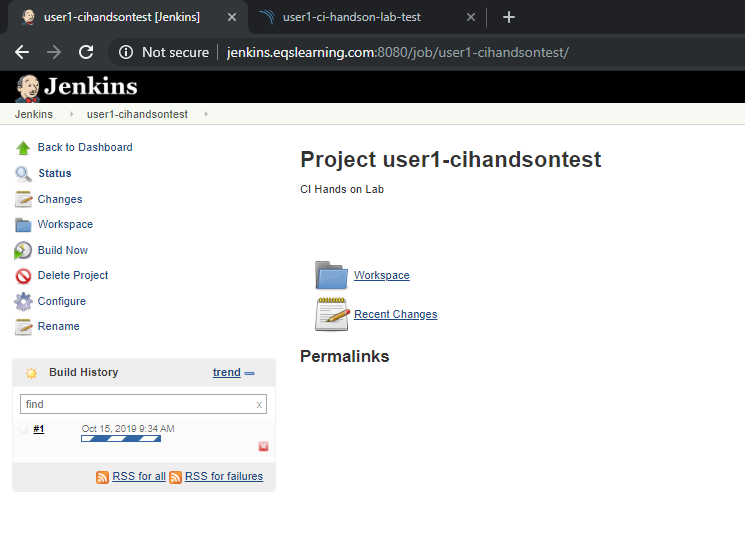


# Triggering the Jenkins Build

In order to trigger the build, Navigate the Jenkins Home page by clicking on Jenikns Icon on the top of the page. You may verify your newly created build job.

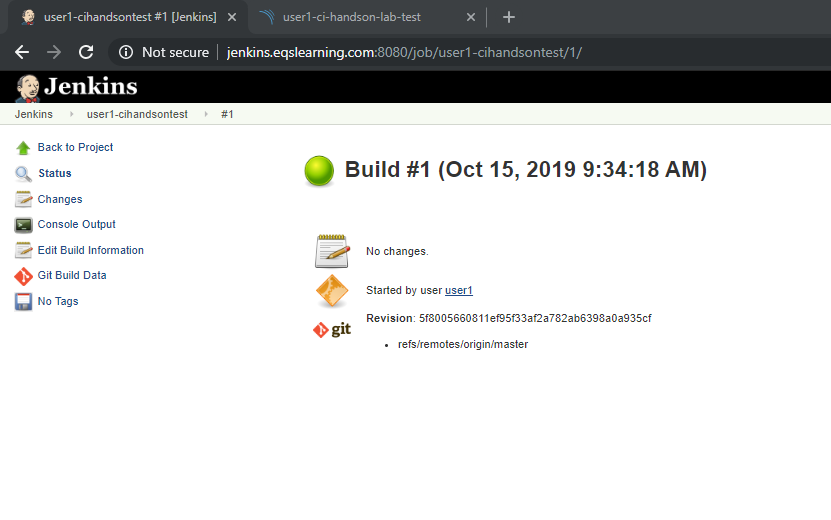


Select the name to open the build Job which was created and click on **Build now** option on the left side menu to manually trigger a new build job. Under build history of the build job, you may notice a new job running.



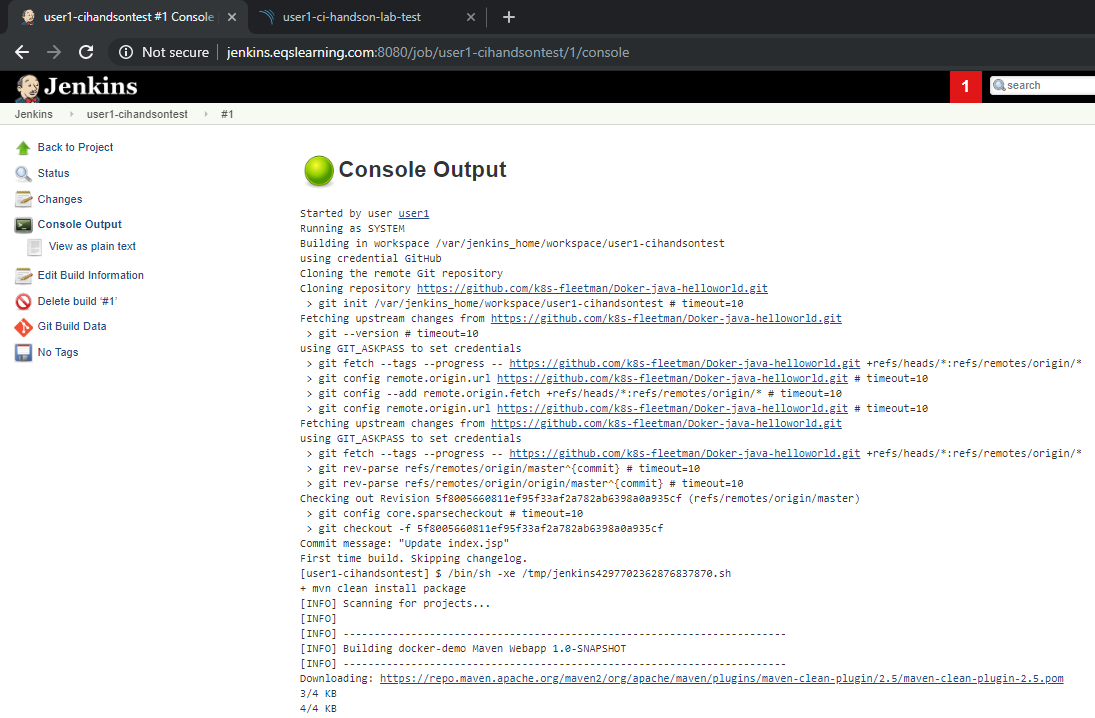
Click on the build number which was triggered to review the build job which was initiated.

Review the Build job which was triggered.



# Verify the build logs

Click on the **console output** on the left menu to review the build job history.



# Verifying the Code quality status in Sonarqube

