**Sonar + Jenkins + Git + Maven + Java**

This document will guide you to setup Jenkins and Sonar in your machine (windows). After integrating sonar and Jenkins, I’ll walk you through building a Git + Maven + Java project using Jenkins with sonar plugin.

**Step 1.** Make sure **Java JDK 1.8** and **Maven 3+** is installed and set up in windows environment variable

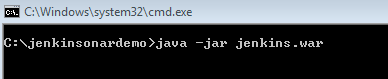
**Step 2.** Download Jenkins LTS (Long term support) version 2.46.3

Download generic Java Package WAR file

**Download Link -** <https://jenkins.io/download/>

Copy Jenkins war at a desired location. I’ll be using **c:/jenkinsonardemo/**Jenkins.war

**Step 3. Run Jenkins using command prompt –** open command prompt and change directory to  **“c:/jenkinsonardemo/”** and run the war file using “**java –jar Jenkins.war**”



**Step 4.** Run above command, this will start Jenkins, this will extract all the Jenkins files in “.jenkins” folder under “C:\Users\[username]\.jenkins” . I would prefer to have a specific directory to extract Jenkins file, if u like the same I would suggest to create “JENKINS\_HOME” and .jenkins directory will be created under “JENKINS\_HOME” directory.

**Step 5.** When installed first time, Jenkins will require an initial setup, for this Jenkins will create an initial password, initial setup data will be provided on command prompt and console both.

**Jenkins console –** <http://localhost:8080>

Following are the screen shots of command prompt and web console after first run

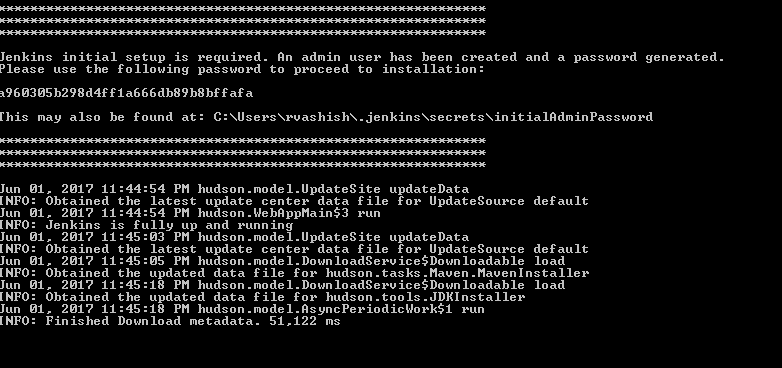


Figure Jenkins command prompt output after first run

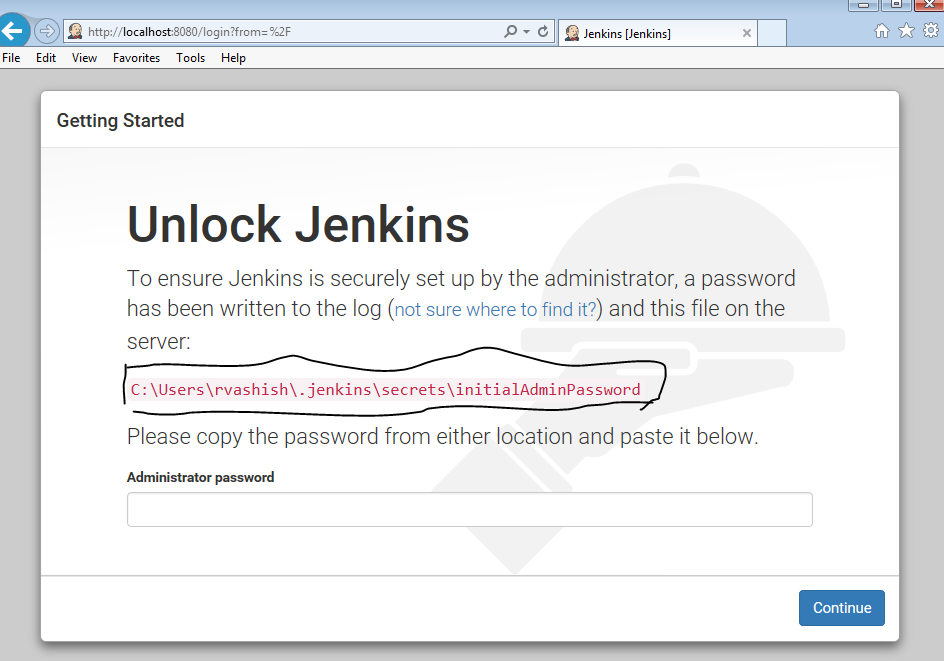
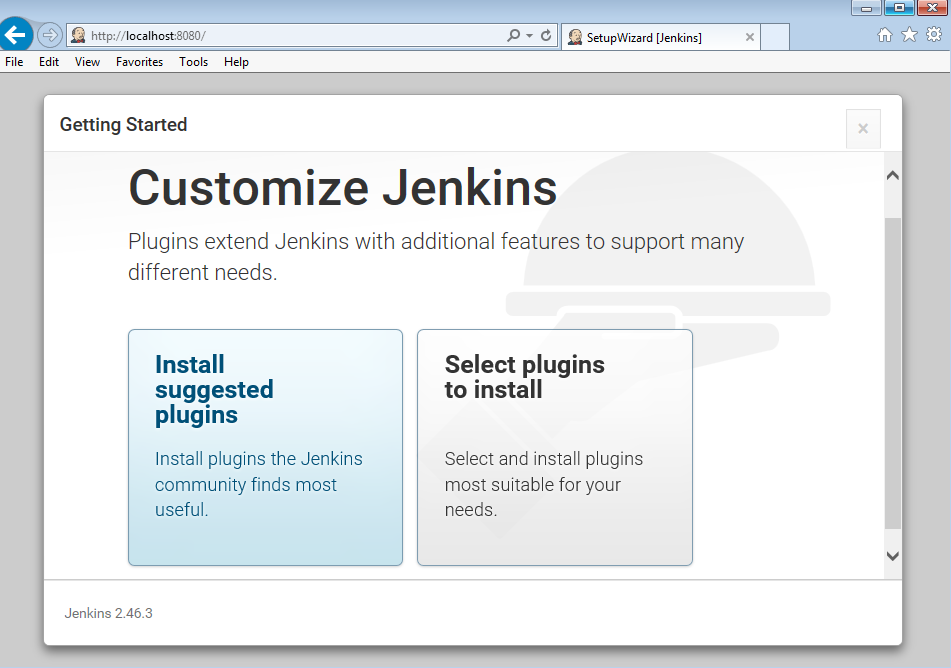
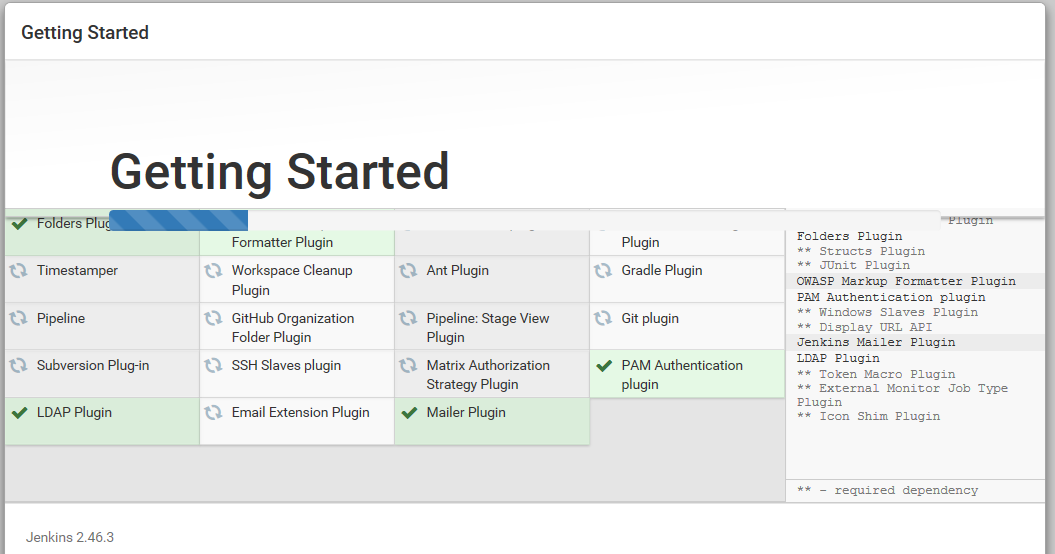


Figure Jenkins web console after first run

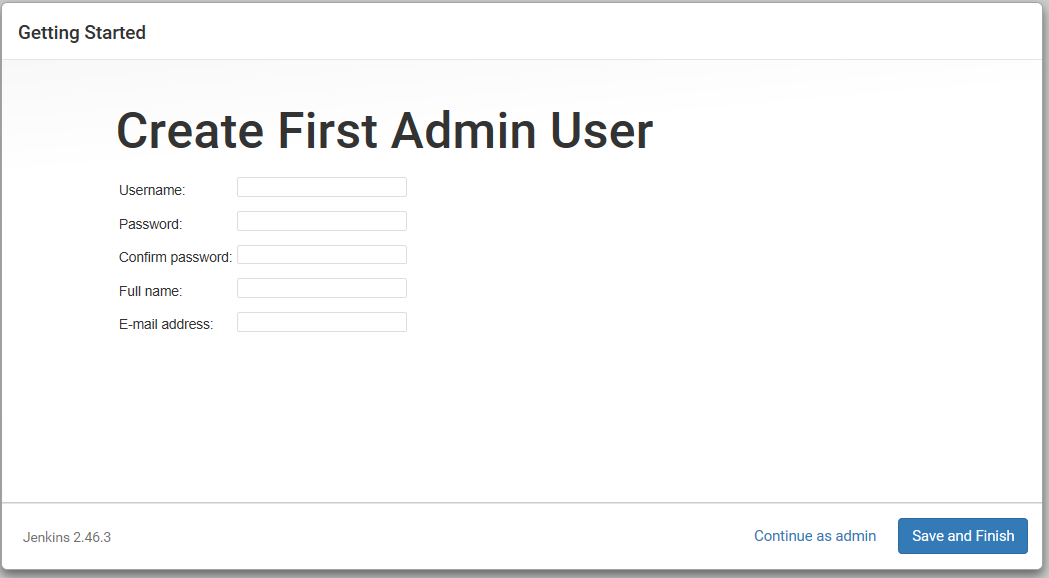
**Step 6.** Copy initial admin password and login to web console, remember this initialAdminPassword file will be removed after first login. This will take you to the following screen to customize Jenkins. I would suggest clicking on “**Install suggested plugin**” button.

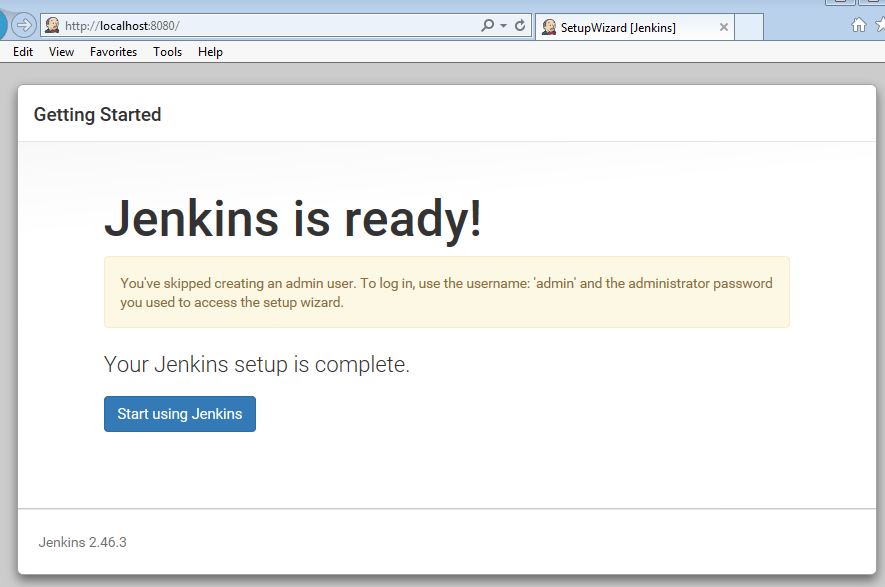


**Step 7.** Once u select install suggest plugin, it’s time you to sip your coffee, Jenkins will take few minute to install all the required plugin. Console screen may look like below one –

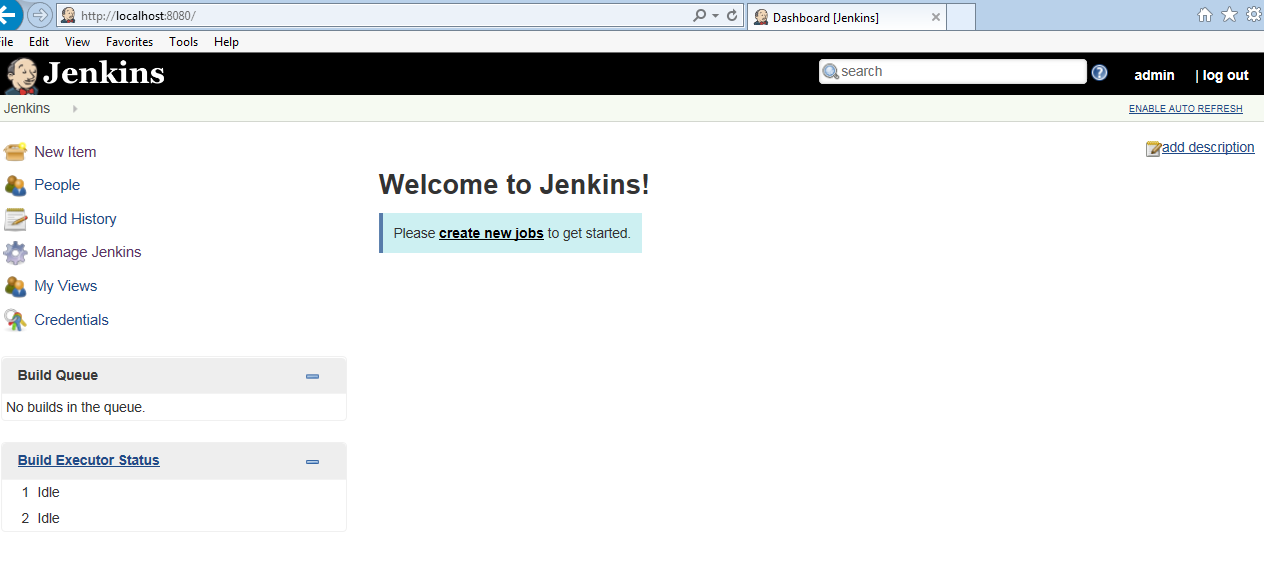


**Step 8.** Once Jenkins is done installed all the plugin it will take you to setup first admin user, form will look like below one –

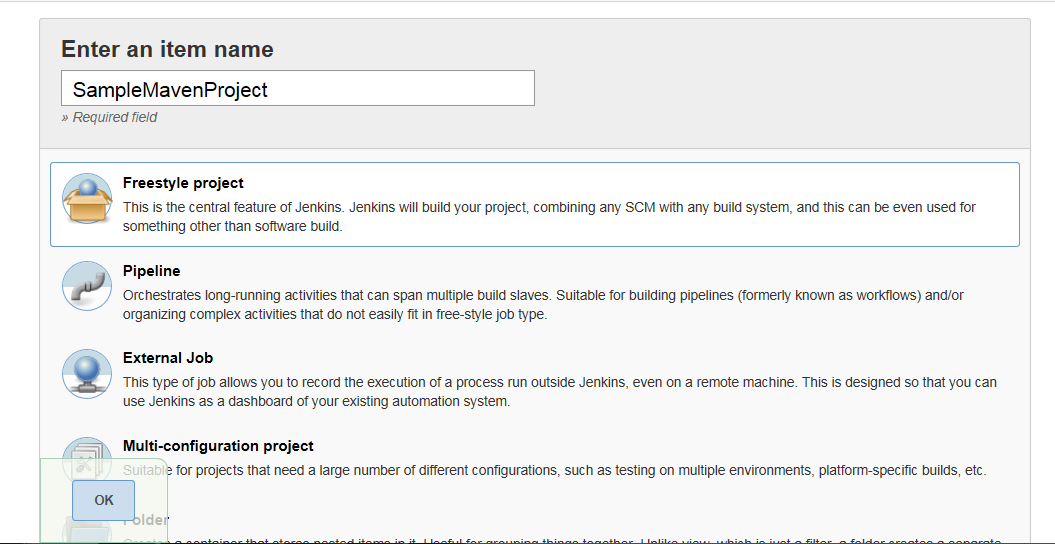
FfiFill all the required fields and hit on “continue as admin” and guess what, jenkins is ready ☺



**Step 9.** Hit on “start using Jenkins” this will take you to Jenkins home page, where we do all the magic.

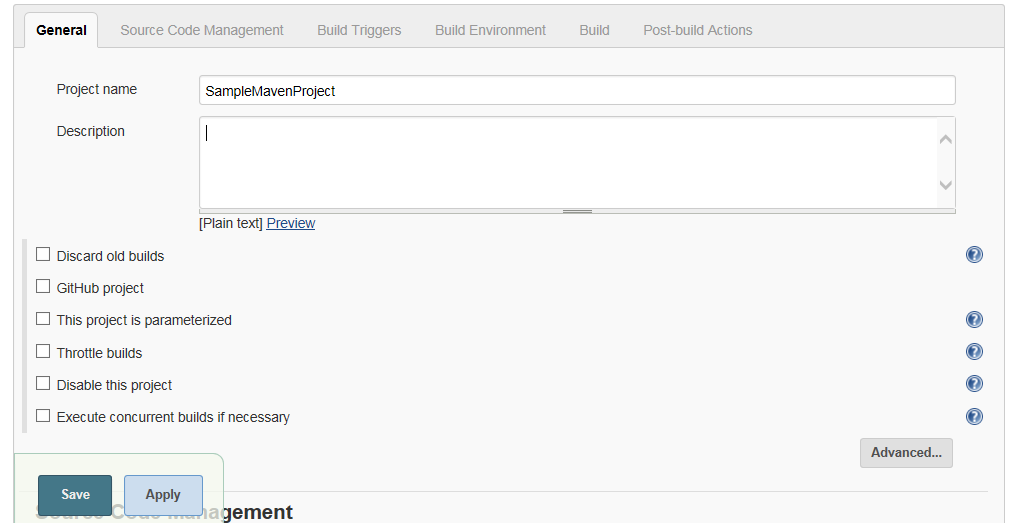


**Step 10 .** Click on create new jobs – following screen will appear



Fill in item name, select freestyle project, hit ok.

**Step 11.** This will take you to following page, where we will do all the project build related setup.



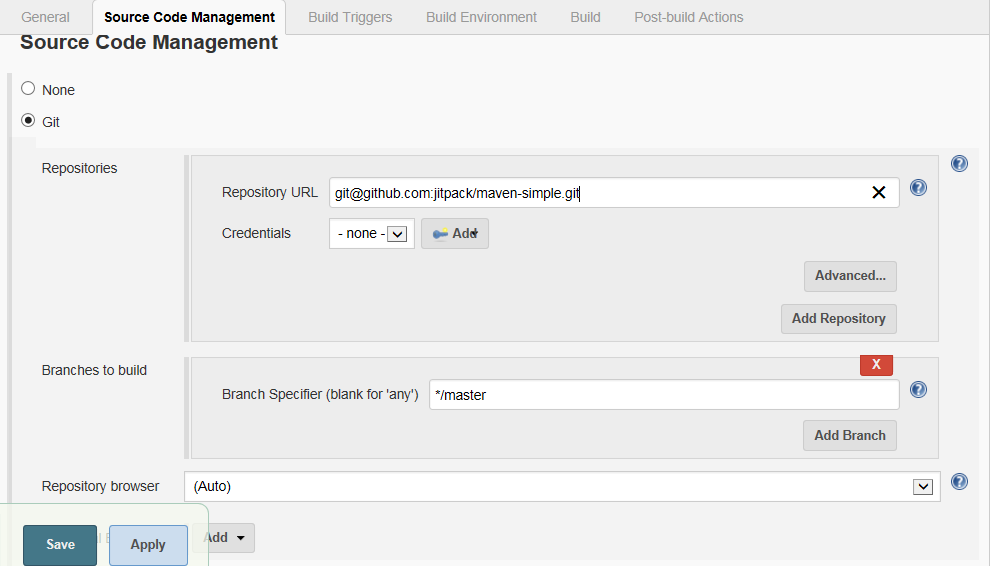
For now we are mainly interested in Source Code Management and Build section. Scroll down or select ‘Source Code Management’ tab.

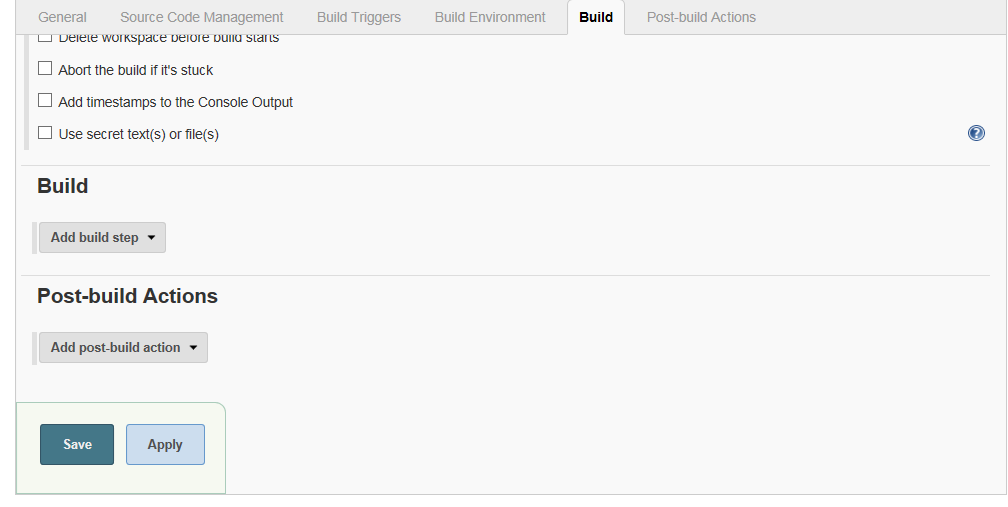
**Step 12.** Select Git radio button, and provide following git link(This is a sample maven project)

Sample Maven Java Project - [*git@github.com:jitpack/maven-simple.git*](mailto:git@github.com:jitpack/maven-simple.git)Add this to repository URL and for now keep all other fields as default. Since it’s an open repo it doesn’t require any login credentials. We are fetching repo master branch. You can also provide a separate branch. Instead of Git we can also select SVN repo.

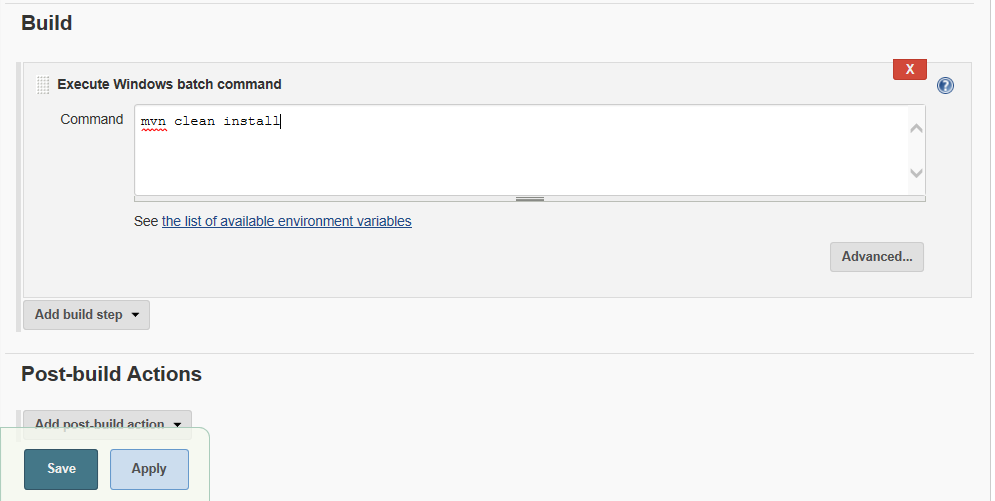
Note – User https repo url if git ssh is not setup in your machine - <https://github.com/jitpack/maven-simple.git>

Suggestion – build this project in your local first before trying to build this up using Jenkins. Also u can use any git/svn project to build using Jenkins for this tutorial.



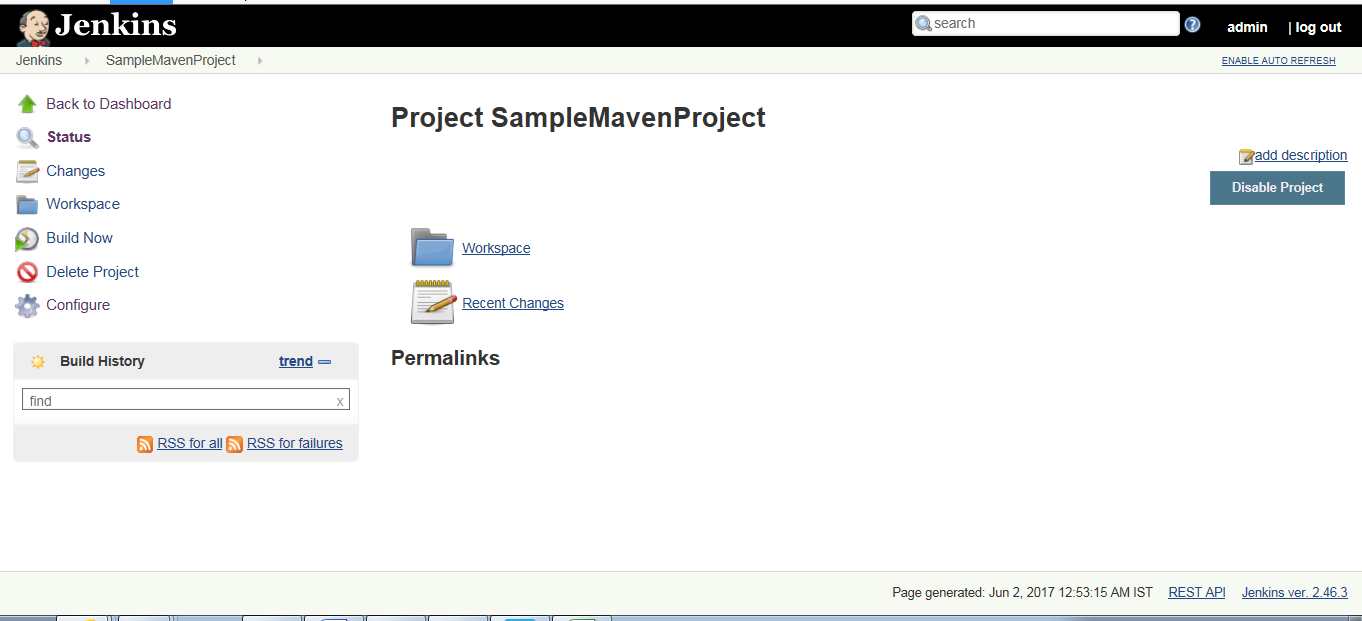
**Step 13.** Once added repo. Scroll down or select build section you will see Add build step, select ‘Add build step’ dropdown and choose “Execute windows batch command”. Screens will looks like below 

Once select build step, provide the maven build command. Since it’s a maven project use “mvn clean install”. Refer below screenshot.



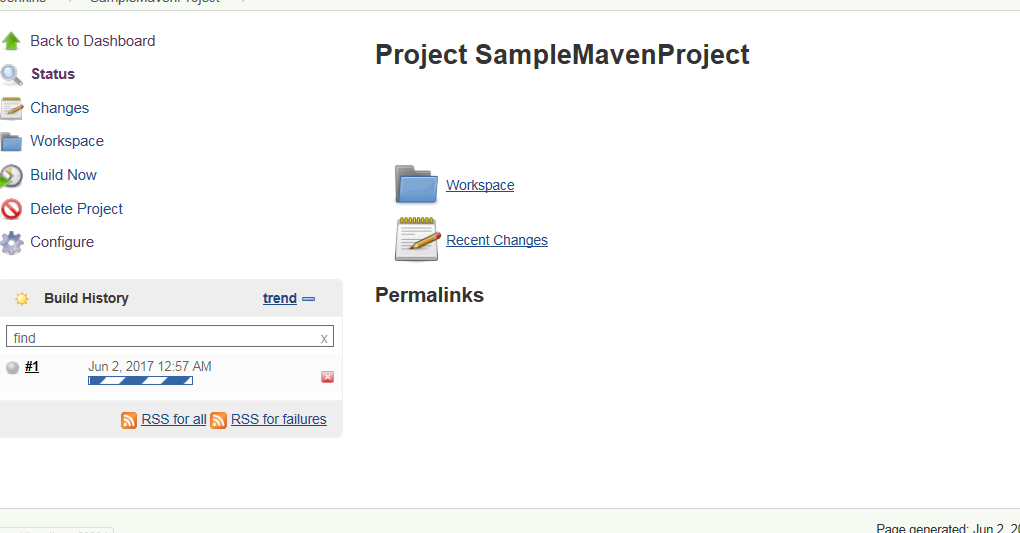
Hit “Save” and we are done with basic setup.

**Step 14.** Now we are back on Jenkins’s SampleMavenProject page. Have a look at below snapshot

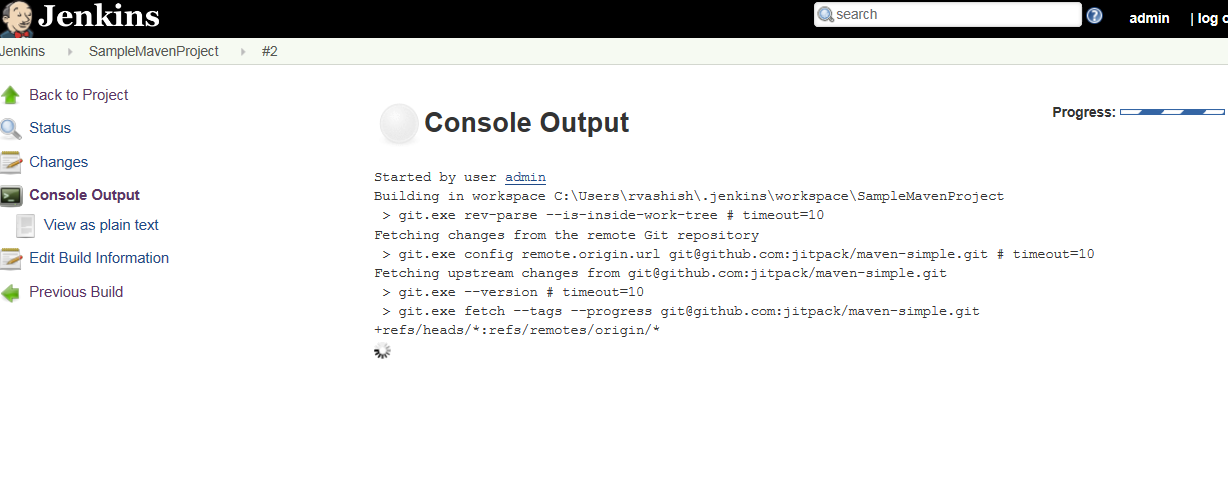


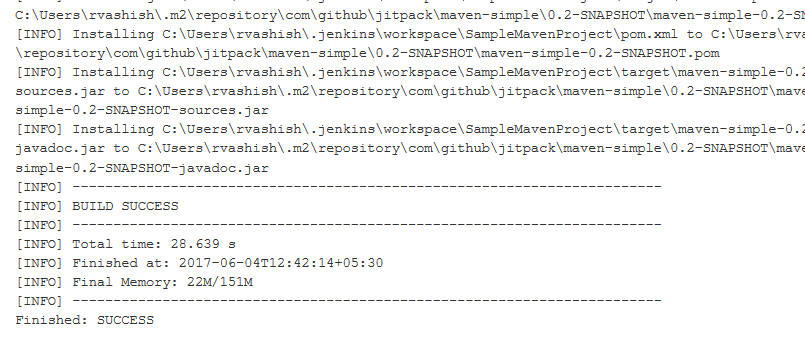
Once you click on build now, it will start the build pipeline. We can go back to build configuration anytime by selecting “Configure” gear button. Later on we can configure build now action on commit code event in git, so that after every commit Jenkins will trigger a build. For now select build now button manually.

**Step 15.** Once clicked on “build now”, a build will be scheduled and started. To see the build logs click on the build progress bar.



**Step 16.** After clicking progress bar we can see the console logs. And in few more seconds our first Jenkins build will be done.





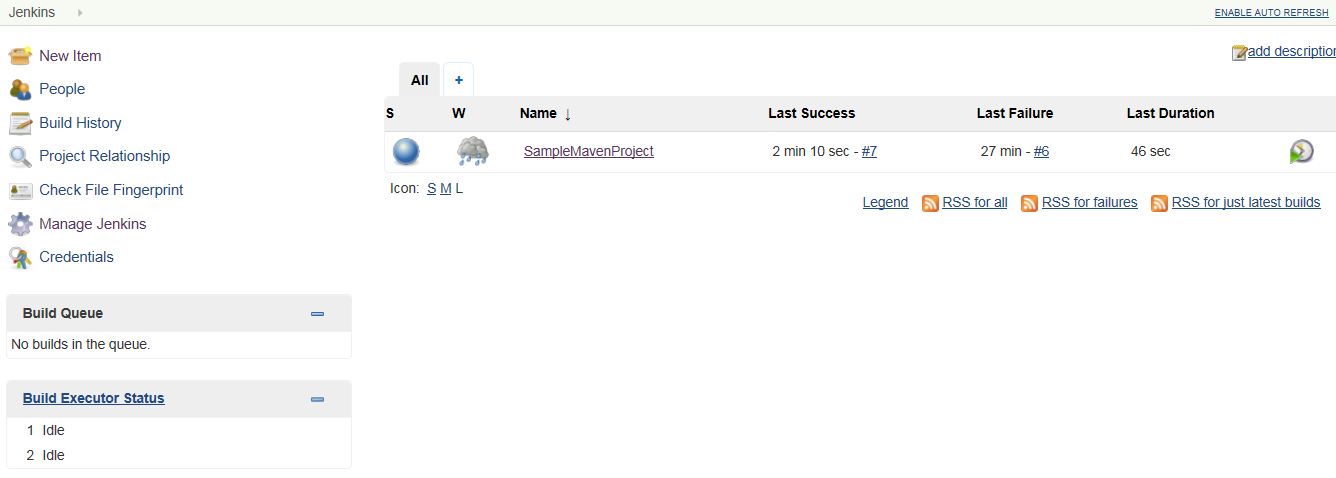
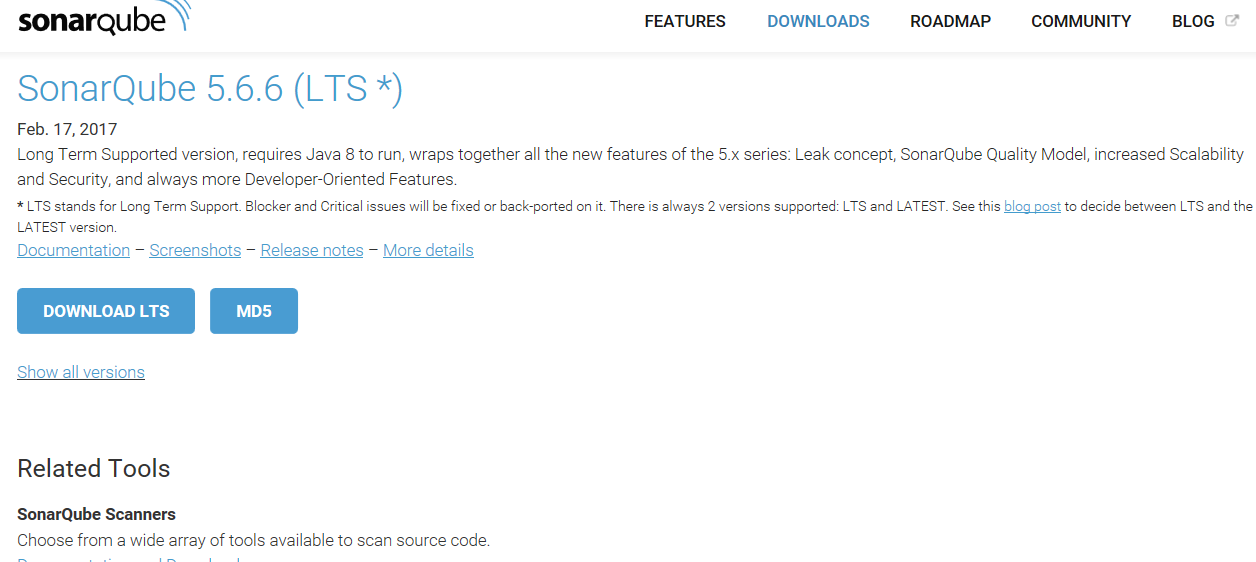


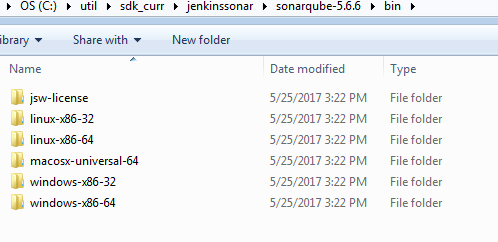
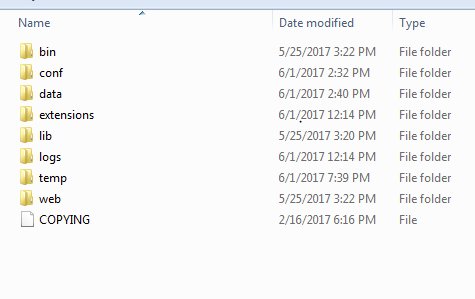
Figure Jenkins Home - having build details

Let’s have a break now. We will resume with SonarQube setup and integration with Jenkins build.

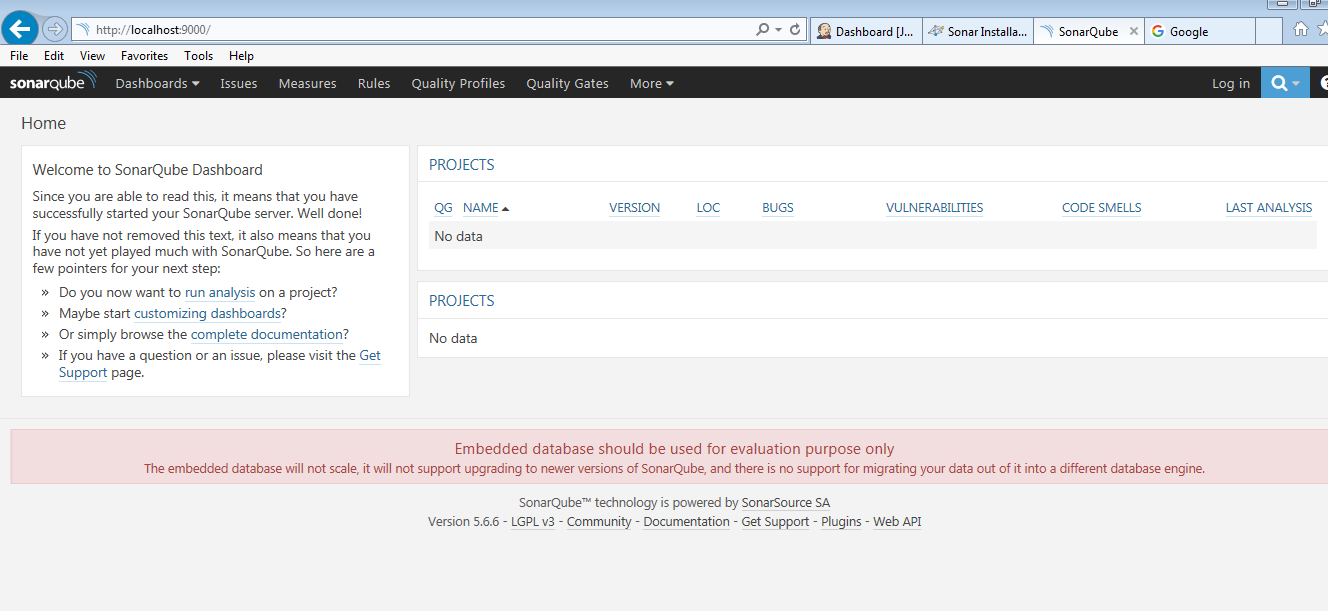
**Step 17.** Once Jenkins setup is done, let’s download the SonarQube 5.6.6 (LTS). <https://www.sonarqube.org/downloads/>



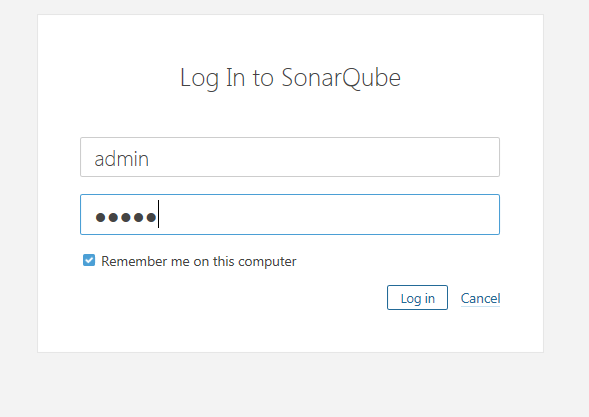
**Step 18.** Extract the file. And move to “bin/windows-x86-64” directory



**Step 19.** Run sonarServer.bat file under “sonarqube-5.6.6\bin\windows-x86-64” directory. Check the command line console output, once the process is up. SonarQube is reachable at [**http://localhost:9000/**](http://localhost:9000/) We can see public view here.

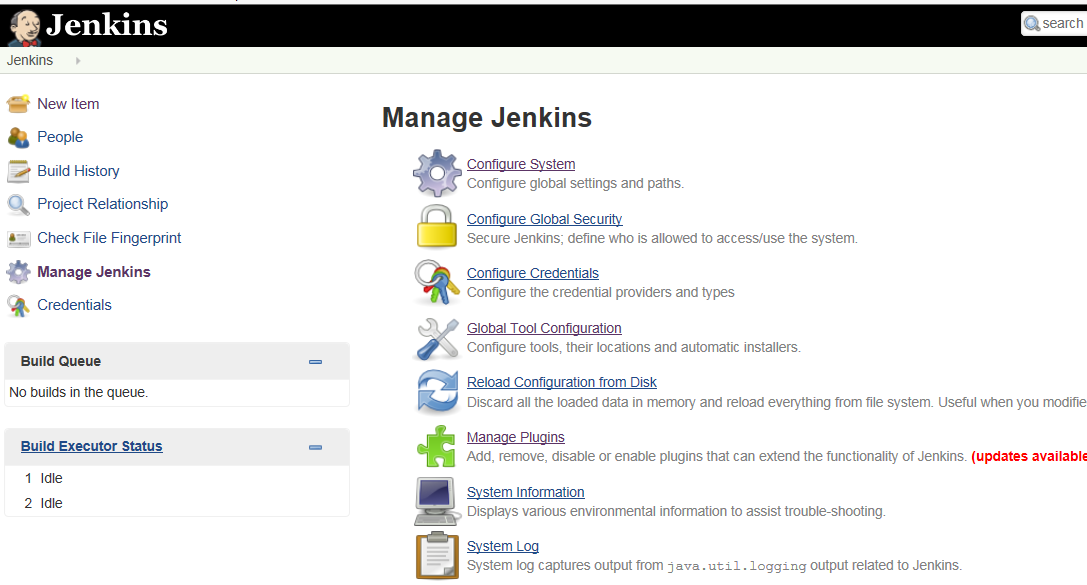


**Step 20.** Click on login – enter username – admin, password – admin to login as sonar administrator for the first time.

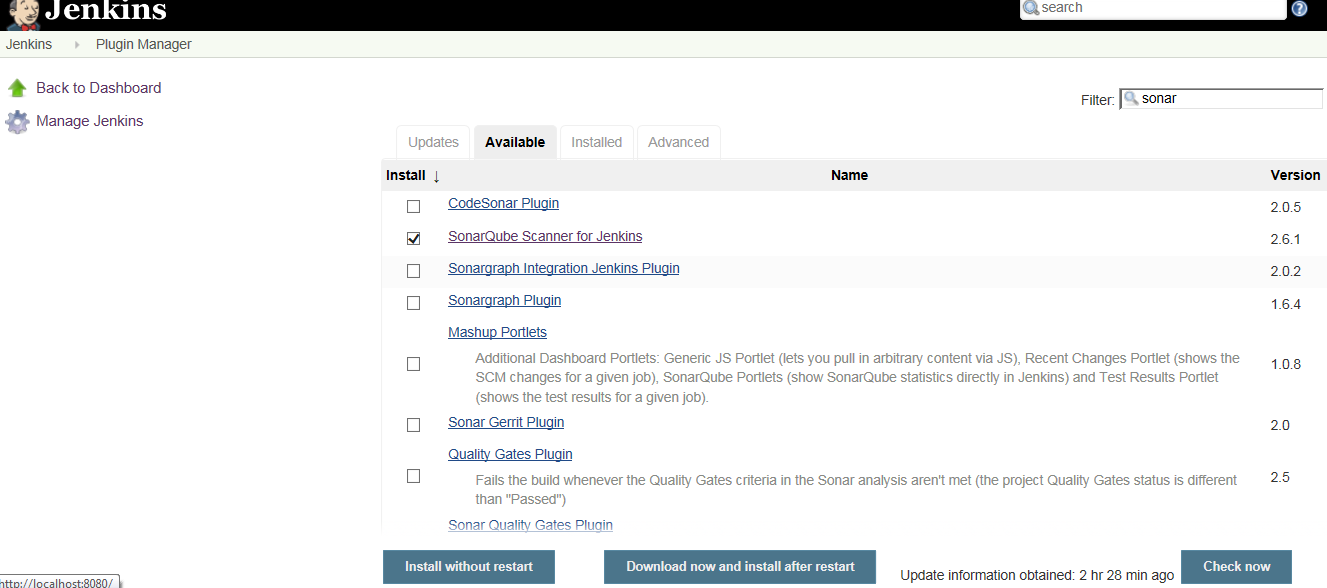


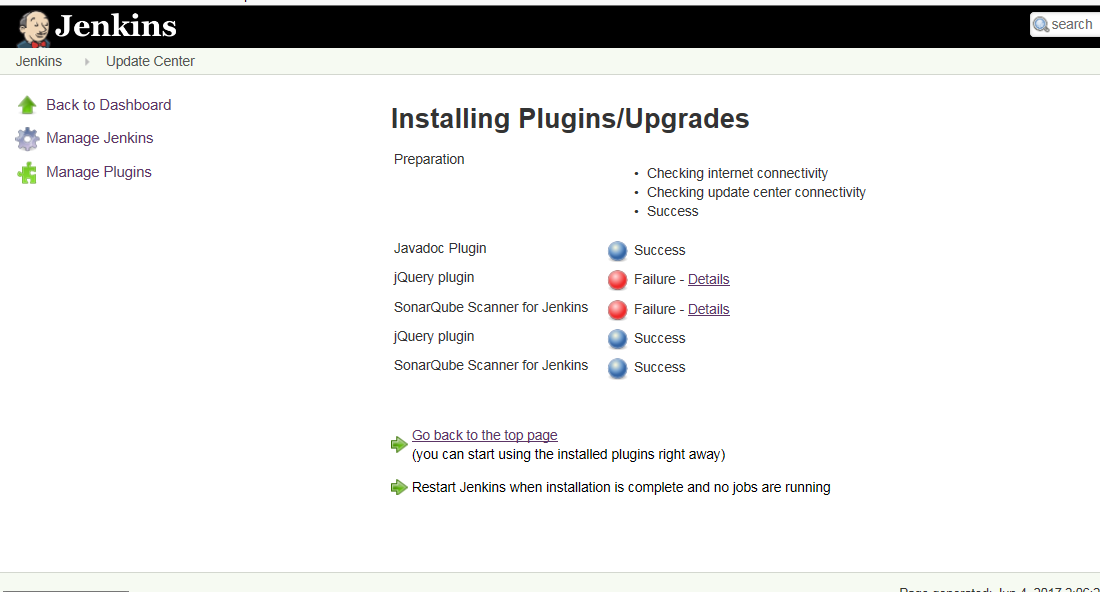
This SonarQube can be run as a standalone instance and can be configured with maven, grdle build files to run using command prompt, can be configured with IDEs. Since we are doing SonarQube and Jenkins integration, we will be skipping the above. Also this SonarQube instance is running on embedded database. You should configure a database to have persistence storage.

**Step 20.** Now we need to add SonarQube plugin in Jenkins. Go to Jenkins home -> Manage Jenkins -> Manage Plugin.

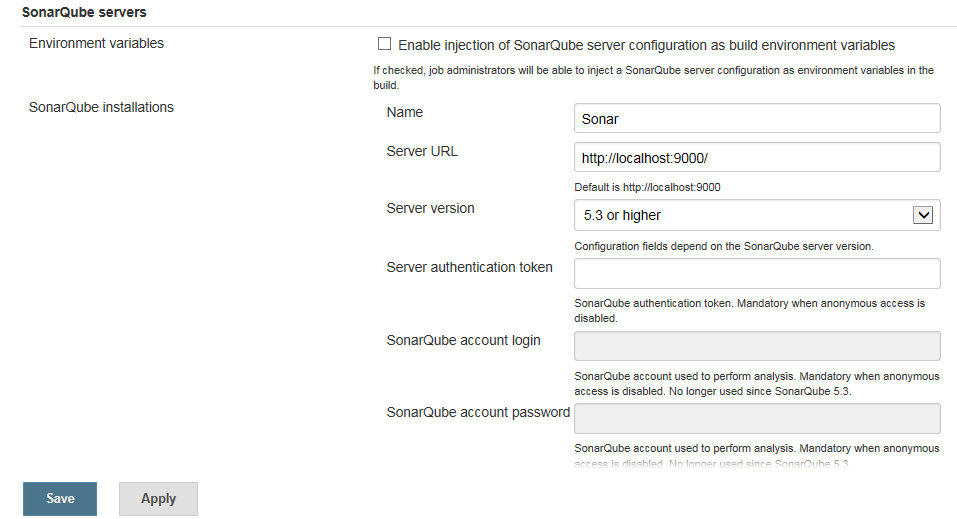


**Step 21.** Got available plugin section, search for sonar, Select “SonarQube scanner for Jenkins”. Click “install without restart”. This will install “SonarQube” plugin.

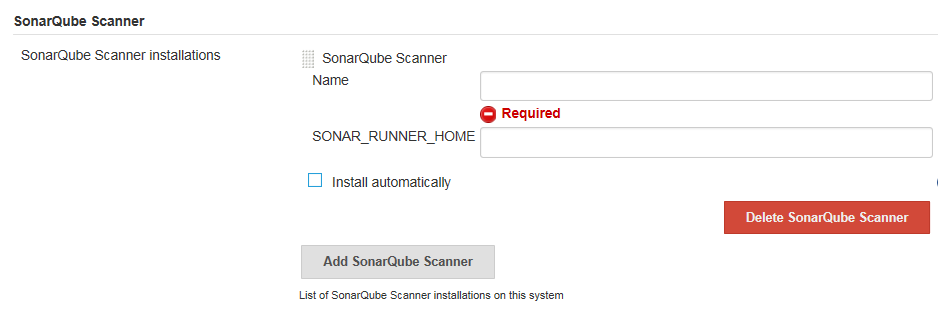




**Step 22.** Once plugin is installed, navigate to Jenkins home-> mange Jenkins -> Configure System. Scroll down to SonarQube servers. Enter SonarQube installation name and server url. Save.

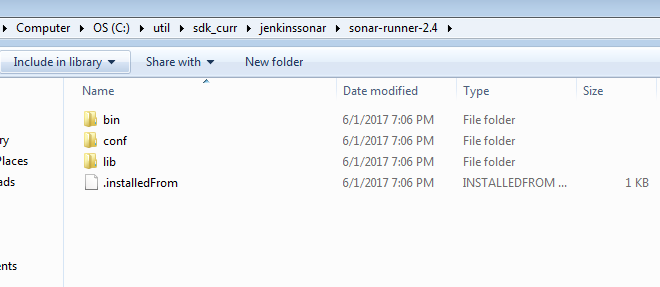


**Step 23.**  Once sonar server is setup. Navigate to jenkins home -> manage jenkins -> Global tools configuration. Scroll down to SonarQube scanner and click on Add SonarQube Scanner button.

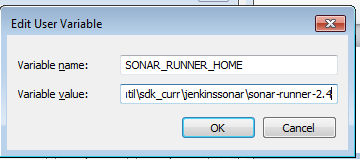


Unchek install automatically. Here we can see, this config section require “SONAR\_RUNNER\_HOME”. Let’s download the sonar runner and create above environment variable.

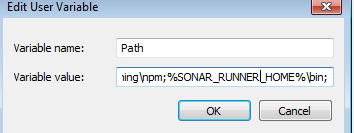
**Step 24.** Download sonar qube scanner from <https://docs.sonarqube.org/display/SONARQUBE45/Installing+and+Configuring+SonarQube+Runner> Extract the zip.



Make “SONAR\_RUNNER\_HOME” environment variable.

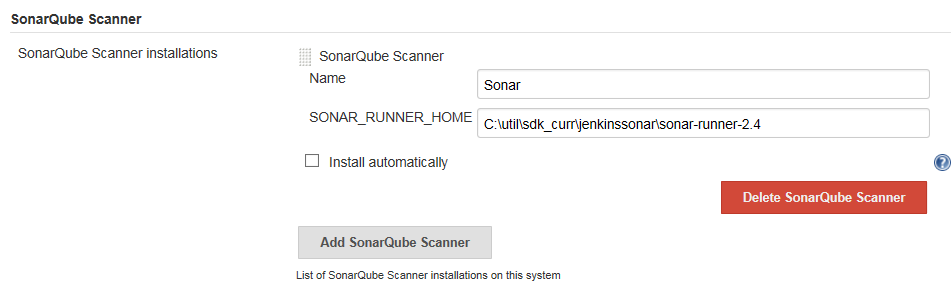


Set the path – to bin

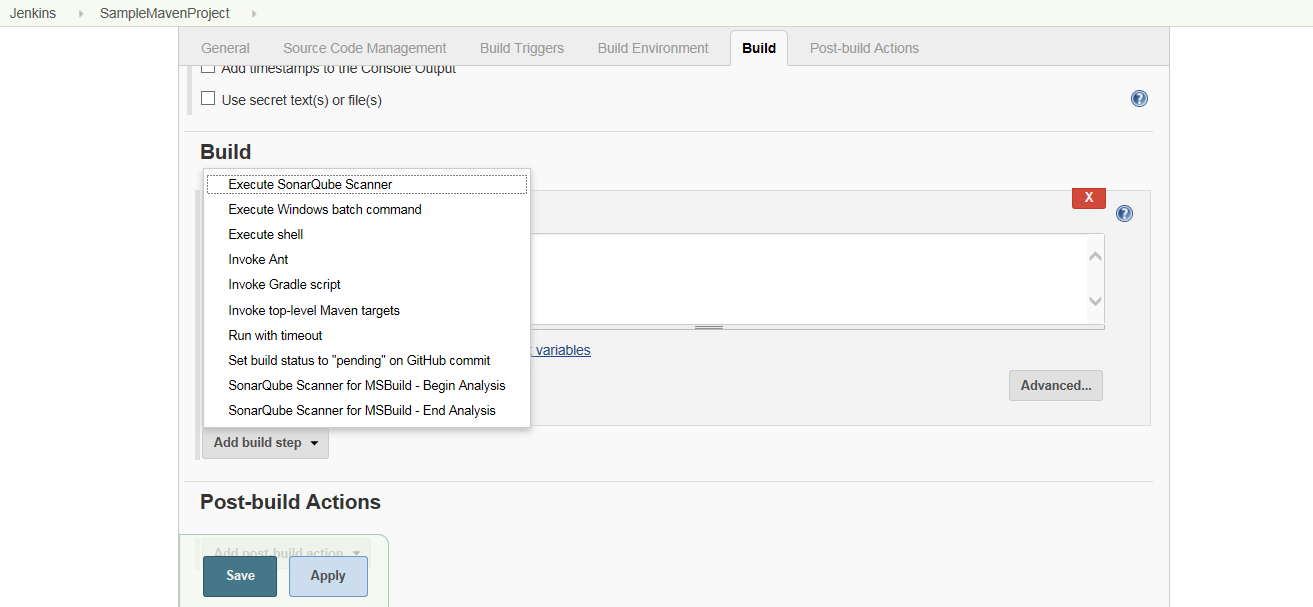


Although setting the environment variable is not required for this tutorial. Lets do this for furtur reference.

**Step 25.** Let’s move back to Jenkins Home -> Manage Jenkins -> Global Tools Configuration – SonarQube SonarQube scanner. Provide scanner home path and save



**Step 25.** Navigate to Jenkins home -> SampleMavenProject -> Configure -> Build. From the Add build Step dropdown select ‘Execute SonarQube Scanner’.



Provide the following properties and save.

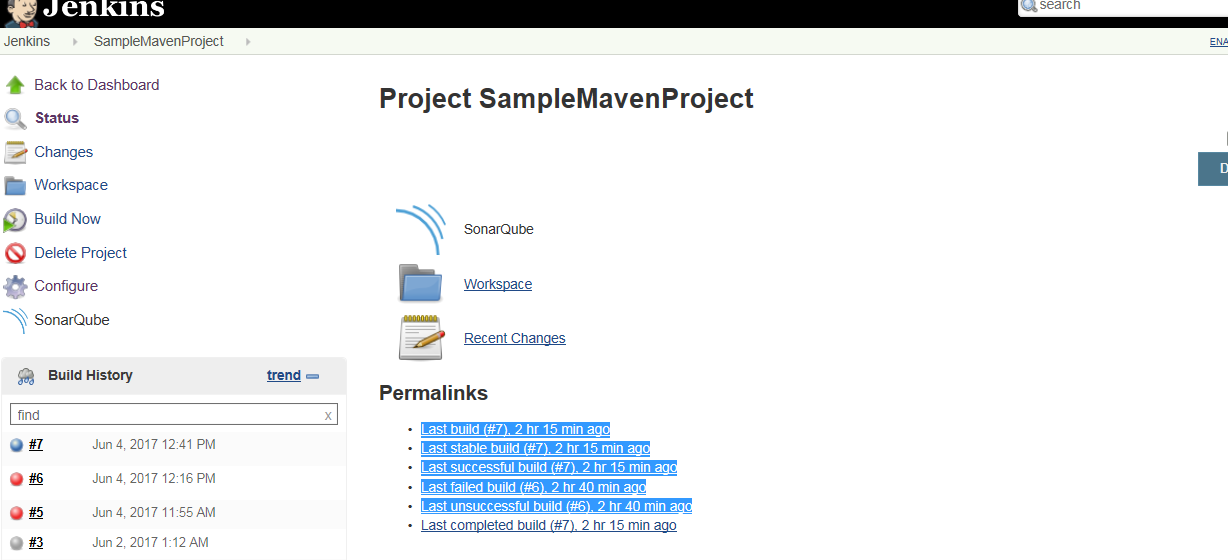
*sonar.projectKey=sample-maven-project*

*sonar.sources=src/main, src/test*

*sonar.projectName=sample-maven-project*

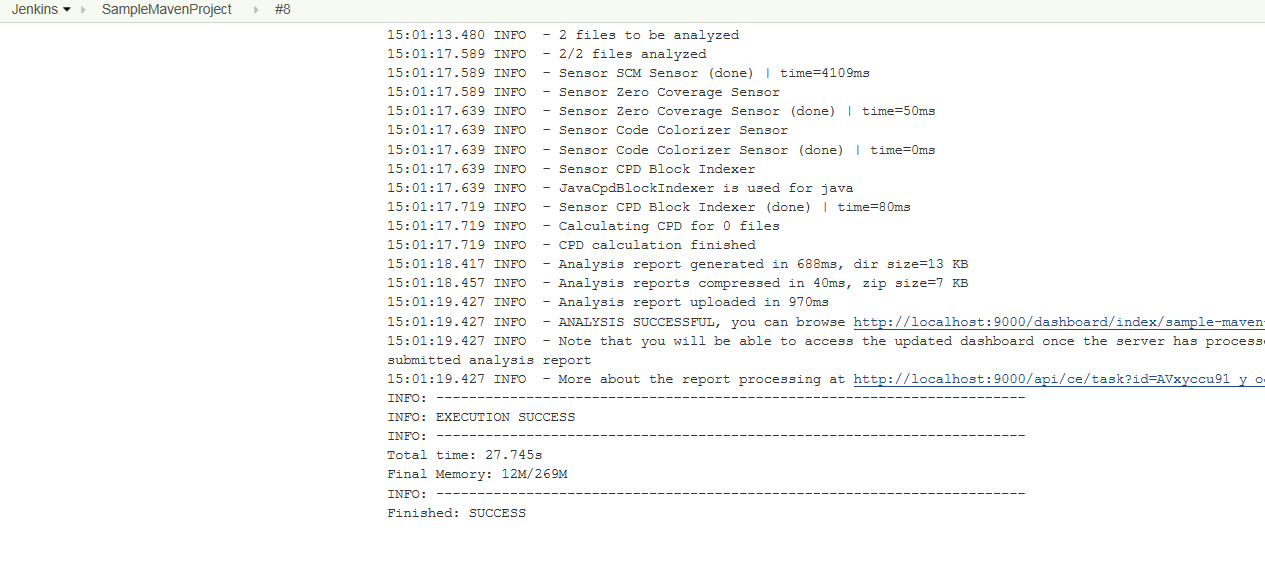
*sonar.projectVersion=1.0*

**Step 26.** Once saved. Navigate back to project home page in Jenkins. You should see SonarQube added in project as follows.

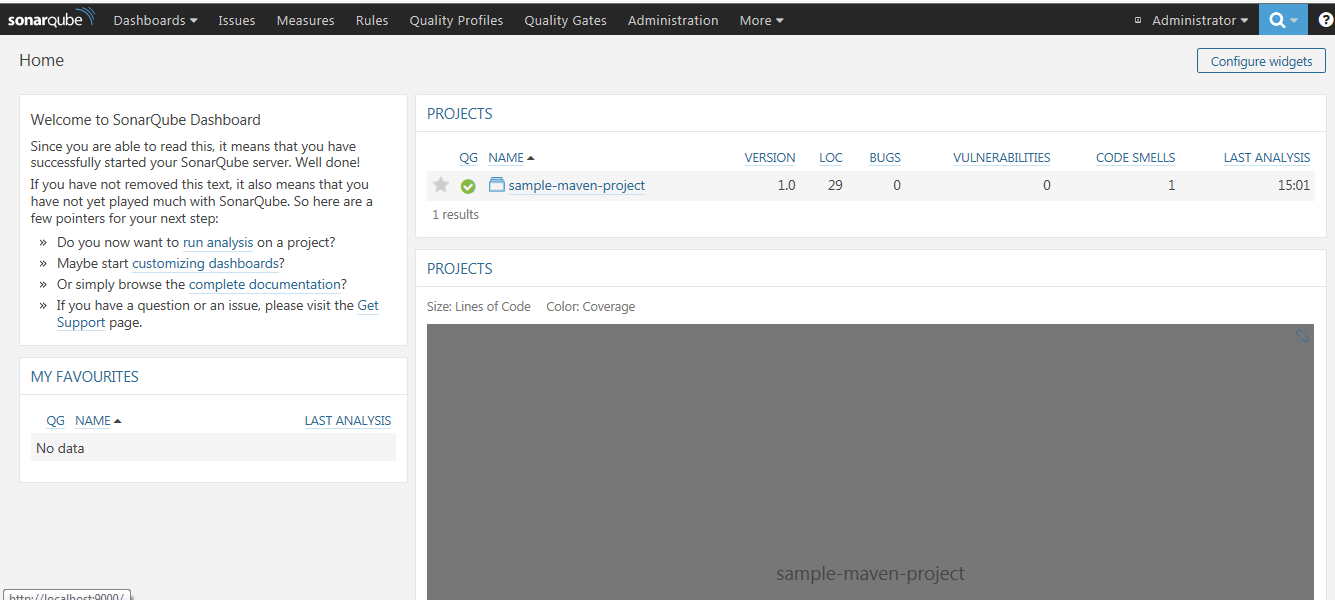


Click on BuildNow. And monitor the logs to know Jenkins sonar execution details.

**Step 27.** Once build is complete, you will see Execution success message.



Great ☺ it’s time to see the magic. Open the SonarQube console. Refresh it.



You can see sample-maven-project here with the detailed analysis report. Since it is a sample java project you won’t see many bugs here. Go ahead, analyze the code smells.

We would suggest analyzing other projects you have. Add sonar to your IDE to improve your code quality.