**Jenkins Pipeline in Linux: (GIT, sonarqube, sonar \_runner, build on commit, Quality gates):**

**Step: 1 (Jenkins Installation)**

1. Download the Jenkins.war file and run the war file as below.

**$ Java –jar Jenkins.war - -httpPort=9090**

1. Now you can see the Jenkins is running on the port 9090.

**Step: 2 (Installing the Git**)

1. Install the Git version control version. Use apt-get git install

**Step: 3 (install the sonarqube and sonar runner)**

1. Download the sonarqube and sonar runner from the below link

<https://www.sonarqube.org/downloads/>

1. Unzip the zip file /tar file

**Zip**🡪 **$ unzip sonarfile.zip**

**Tar**🡪 **$ tar -xzvf sonarfile.tar.gz**

**Step 4: (Set the maven and java home)**

Download the maven and java and set the JAVA\_HOME and M2\_HOME

1. Download maven from the below link

<https://maven.apache.org/download.cgi> and unzip it.

**$Unzip mavenzipfile.zip**

**$export M2\_home=/maven-path/**

**$export PATH=/maven-path/bin:$PATH**

1. Download java from the below link

<http://www.oracle.com/technetwork/java/javase/downloads/java-archive-javase8-2177648.html#jdk-8u91-oth-JPR>

**$tar –xzvf javafile.tar.gz**

**$export JAVA\_HOME=/java-file-path/**

**$export PATH=/java-file-path/bin:$PATH**

**Step: 5 (creating the git repository)**

1. Create a git repository.to to do so first create a .git file example (repo.git). Then navigate to that file a

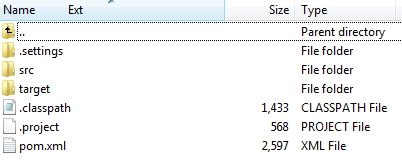
**$mkdir repo.git**

**$ cd repo.git**

Now you will be inside the git folder path and then init the git

**/path/repo.git$git init**

1. After the git init you need to add the folder (project folder to the git ).i have used winSCP to move my project to linux. As my project was in windows.



If you have your project in Linux and want to copy the file from one folder to other, use the below command.

**$cp -a /source-dir/. /dest-dir/**

ow all the folder will be copied to your new git folder.

1. The configure the git as below

**$ git config user.name "John Doe"**

**$ git config user.email “**[**johndoe@example.com**](mailto:johndoe@example.com)**”**

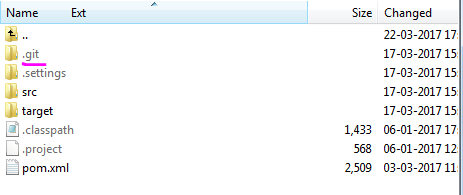
1. Then add the data which you save to the git.

**$ git add .**

1. Then you need to commit the added data to git.

**$ git commit –m “commit”**

Now you will see the .git file created in the folder (project file)



**Step: 6 (create instance in Jenkins for build the project)**

**Jenkins -> Manage Jenkins -> Manage Plugins ->advanced**

Install **Git Plugin** and dependency plugins to install software successfully

From this link <https://wiki.jenkins-ci.org/display/JENKINS/Git+Plugin>



Note:-Make sure that all the dependencies plugins must be added

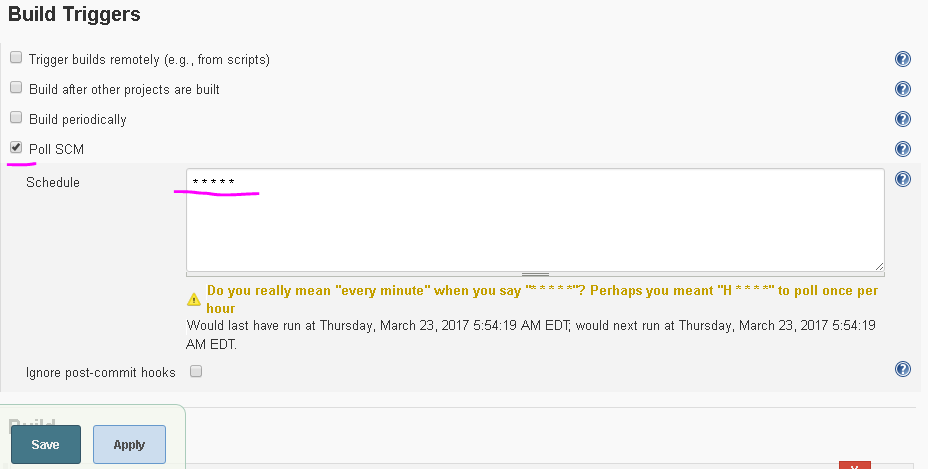
It is similar to the instance creation as windows. Refer the

1. set the Environmental variable as JAVA\_HOME and M2\_HOME as in the document.(**manage Jenkinsglobal tool configuration**)
2. Create a freestyle project and on configure (**source code management**) select the git and specify the path where the maven project is stored **(/home/sj00489487/SparkVehicle.git**).
3. BUILD—select **invoke top level maven targets** and **goal** as (clean install).
4. Save and build the project.

**Step: 7 (Build the project on commit)**

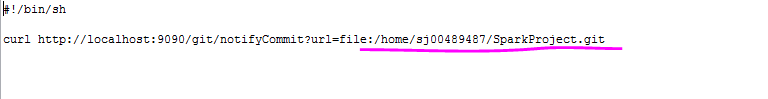
Follow the same steps as the normal build as shown above but need to configure the additional features.

1. In BUILD TRIGGERS🡪 check the poll SCM and mention \* \* \* \* \* on the box.



1. The go to the git project folder (sample.git). There you will see a .git file. Navigate inside the **.git🡪hooks** folder. Example **(/home/sj00489487/SparkProject.git/.git/hooks)**

Create a file **post-commit.sample** with below content.

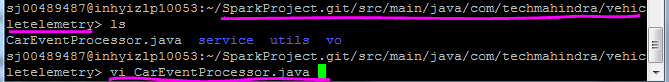


The file:/home/sj00489487/SparkProject.git is the git path where the project is.

1. After this save. Then try to make making changes to the project file sample.java) and commit it. Then you can see the auto build on commit in Jenkins.

**Note: commit process in Linux.**

1. Navigate to the file where you want to make changes.



As shown above and edit the data and save the changes in Vi editor.

1. You can view the changes what you made using the command

**$ git diff**

1. Then you need to add the changed file (CarEventProcessor.java) to git and commit it.

**$ git add CarEventProcessor.java**

**$ git commit –m “commit message”**

**Step 8: (sonarqube and sonar runner)**

Refer below for configuration

**Installing & Configuring SONAR**

1. Download from the below site (sonar-4.5.7.zip ) or new version

<http://www.sonarsource.org/downloads/>

1. Unzip the zip file (sonar-4.5.7.zip) to a folder of your choice

**$Unzip sonar-4.5.7.zip**

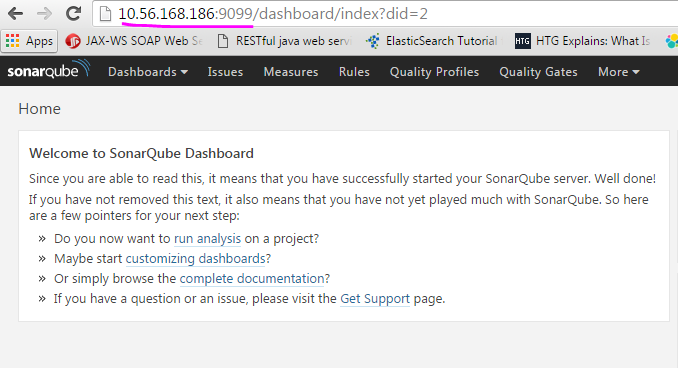
1. Edit the sonar.properties file in the conf folder. (e.g <sonar\_home>/conf/sonar.properties ) Uncomment the following lines in the sonar.properties

|  |
| --- |
| *sonar.jdbc.maxActive =20*  *sonar.jdbc.maxIdle =5*  *sonar.jdbc.minIdle =2*  *sonar.jdbc.maxWait =5000*  *sonar.jdbc.minEvictableIdleTimeMillis=600000*  *sonar.jdbc.timeBetweenEvictionRunsMillis=30000*  *sonar.web.port =9099*  *sonar.search.port =9010* |

1. Run StartSonar.bat in the <sonar\_home>/bin/windows-x86-32 folder

Example : <sonar\_home>\ bin\linux-x86-64>sonar.sh

Now in browser: localhost :9099,



**Installing & Configuring SONARSCANNER**

1. Download SONARSCANNER (sonar-scanner-2.5.zip ) or new version
2. Unzip the zip file (sonar- scanner -2.5.zip) to a folder of your choice

**$unzip sonar-scanner-2.5.zip**

1. Update the global settings (database connection, server URL) by editing the file **(<sonar\_scanner\_HOME>\ sonar-scanner-2.5\conf\ sonar-runner.properties** )

|  |
| --- |
| #Configure here general information about the environment, such as SonarQube DB details for example  #No information about specific project should appear here  #----- Default SonarQube server  #sonar.host.url=http://localhost:9000  sonar.host.url=http://localhost:9099  #----- Default source code encoding  #sonar.sourceEncoding=UTF-8  #----- Global database settings (not used for SonarQube 5.2+)  #sonar.jdbc.username=sonar  #sonar.jdbc.password=sonar  #----- PostgreSQL  #sonar.jdbc.url=jdbc:postgresql://localhost/sonar  #----- MySQL  #sonar.jdbc.url=jdbc:mysql://localhost:3306/sonar?useUnicode=true&amp;characterEncoding=utf8  #----- Oracle  #sonar.jdbc.url=jdbc:oracle:thin:@localhost/XE  #----- Microsoft SQLServer  #sonar.jdbc.url=jdbc:jtds:sqlserver://localhost/sonar;SelectMethod=Cursor |

1. Edit the sonar-runner.properties file in the conf folder

# -- Project specific configuration ------------

#---------jenkins----------------------------

sonar.projectKey=JenkinsVehicle

sonar.sources=/home/sj00489487/SparkVehicle.git

sonar.projectName=JenkinsVehicle

sonar.projectVersion=1.0

sonar. projectkey🡪give the Key for the project to show in sonarqube dashboard.

sonar. sources🡪path where the project for which you want to analysis the code (till src folder).

sonar. projectName🡪give the name for the project to show in sonarqube dashboard.

sonar.projectVersion=1.0

NOTE: change add the Highlighted data to the config file

1. Set SONAR\_RUNNER\_HOME.

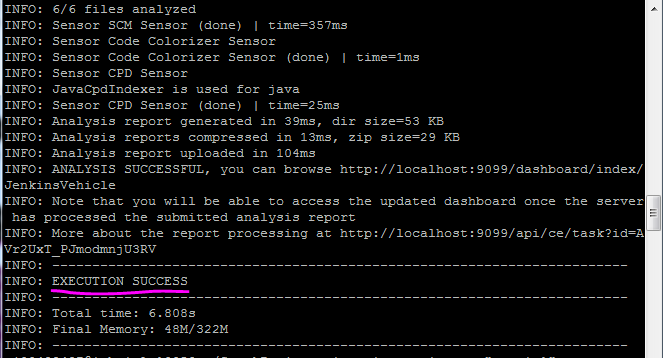
**$export SONAR\_RUNNER\_HOME=/sonar-runner-path/**

6. In the command prompt navigate to source folder

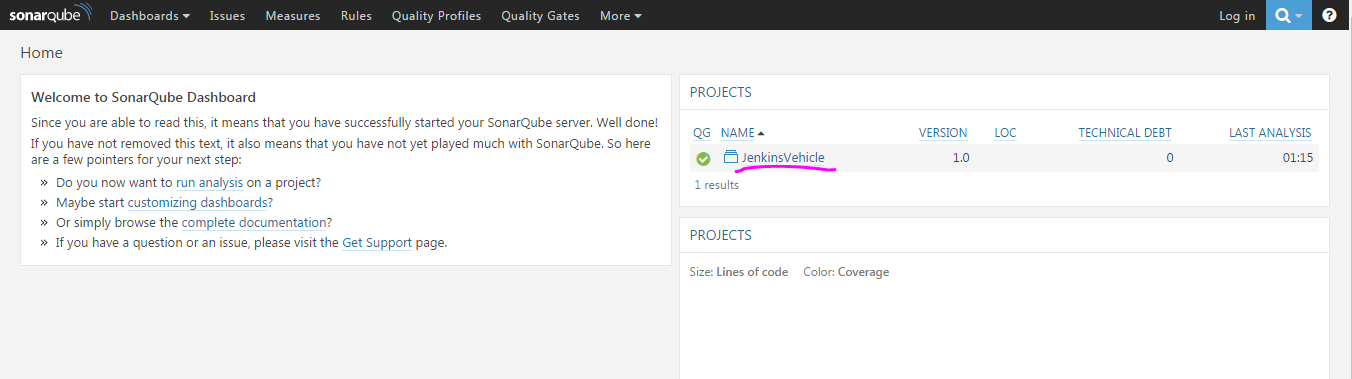


7. From there we have to navigate to <sonar\_scanner\_home>/bin/sonar-runner and run the command





Sonarscanner installed and configured successfully!!!



**Sonar Jenkins Configuration:**

**Steps:**

1. **Add the Sonar plugin and all the dependency plugin.**

**Plugins**

Jenkins -> Manage Jenkins -> Manage Plugins ->advanced

Install **SonarQube Plugin** and dependency plugins to install software successfully

From this link <https://wiki.jenkins-ci.org/display/JENKINS/SonarQube+plugin>

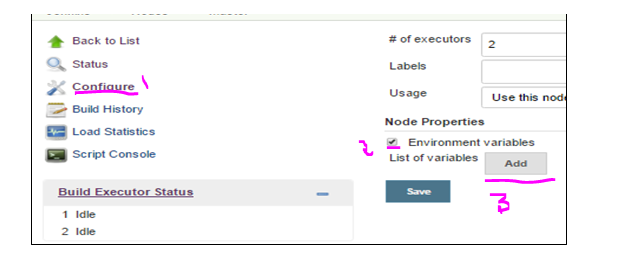


Note:-Make sure that all the dependencies plugins must be added

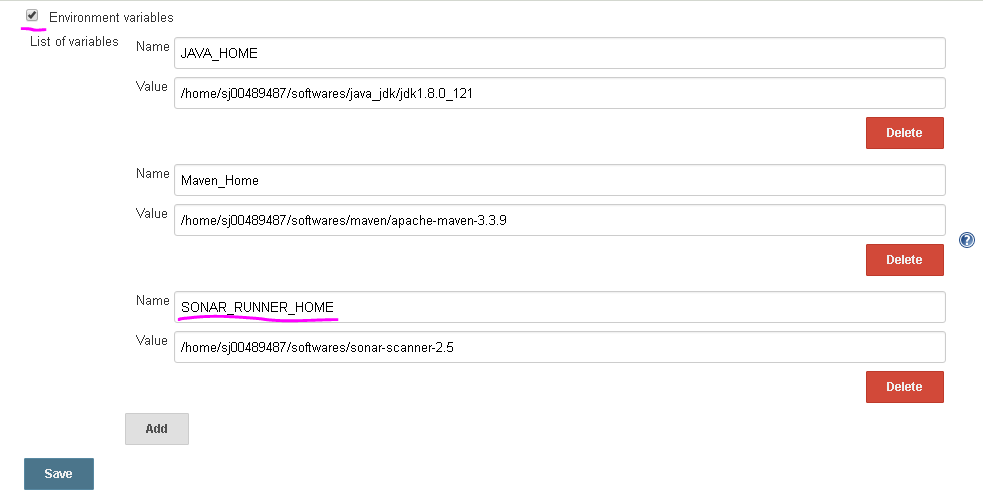
**Initial Setup**

After installation the initial steps to be done to configure the projects

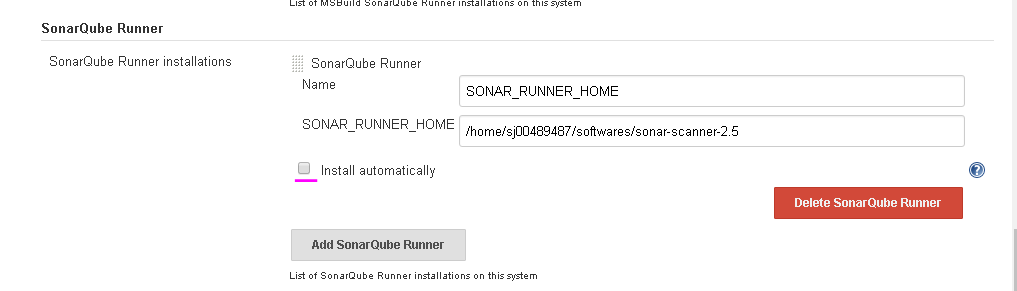
Go to Jenkins->Manage Jenkins ->Manage Nodes->Master->Configure->Add



Set Environmental variables as follows

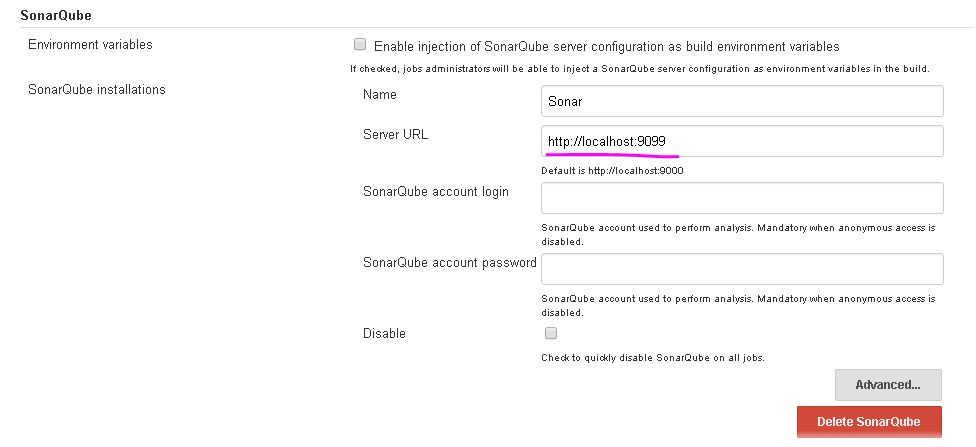


# Go to Jenkins->Manage Jenkins -> Global Tool Configuration



Set the sonar runner home path and also uncheck the install automatically check box and save.

# Go to manage Jenkins🡪configuration system



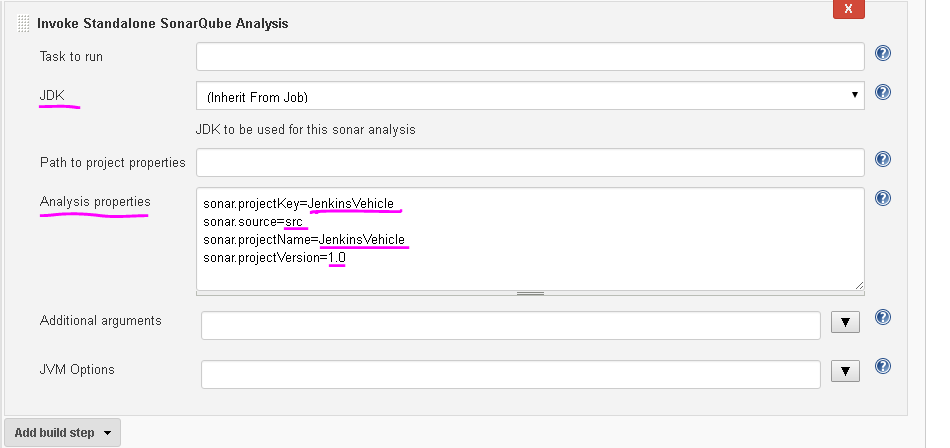
Server URL: URL in which SonarQube server is running

Note :-server should be in running condition

1. **New Project.**

Create **New item->freestyle project**

1. **Build**



Note: 1. JDk (keep it as inherit from job)

2. Analysis properties(property which is mentioned in the sonar-runner. Properties)

->Save->BuildNow



We can able to see the report by clicking the link shown above.

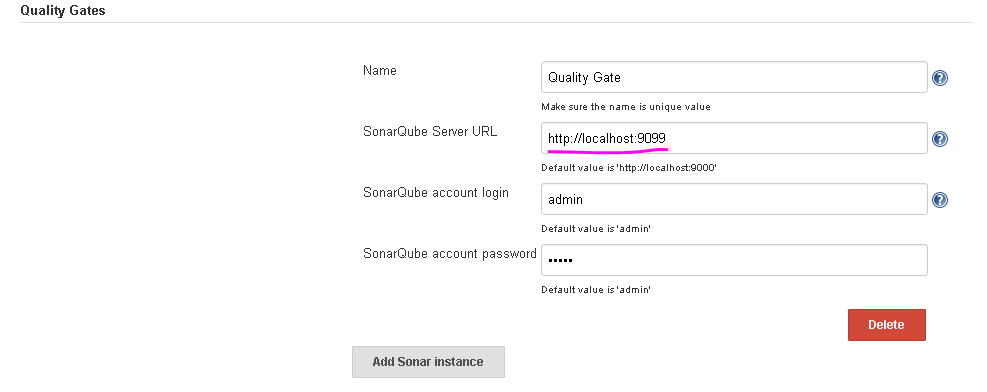
**Step: 9(quality gate in jenkins)**

1. Add the quality gate plugin to jenkins from below link.

<https://wiki.jenkins-ci.org/display/JENKINS/Quality+Gates+Plugin>



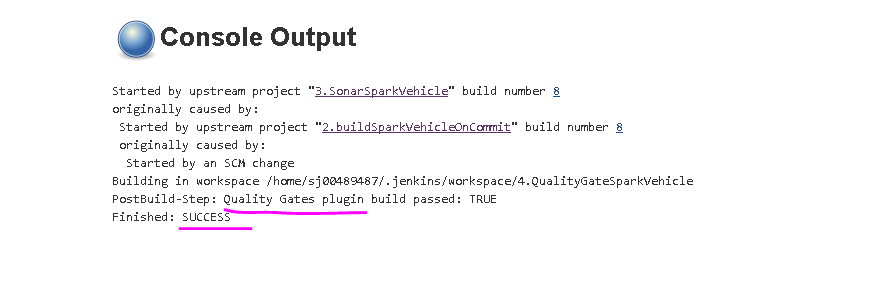
1. Configure quality gate to Jenkins (Manage Jenkins🡪system configuration)



1. Create a new instance🡪free style project and in post build action give the project key which you mentioned in the sonar-runner.property file



1. Then save and build.



**Step 10: (Jenkins pipeline)**

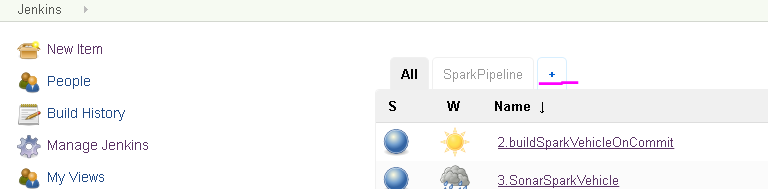
**Jenkins Pipeline**

1. To create a build pipeline add **Pipeline Plugin**

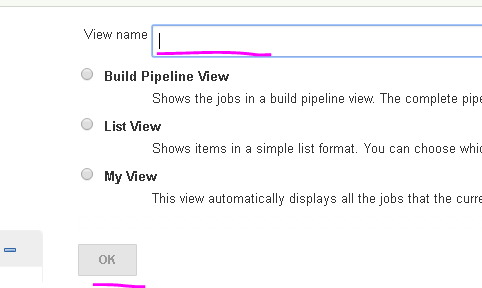
From this link <https://wiki.jenkins-ci.org/display/JENKINS/Build+Pipeline+Plugin>

In Jenkins UI,Click on ”+” sign in order to add pipeline

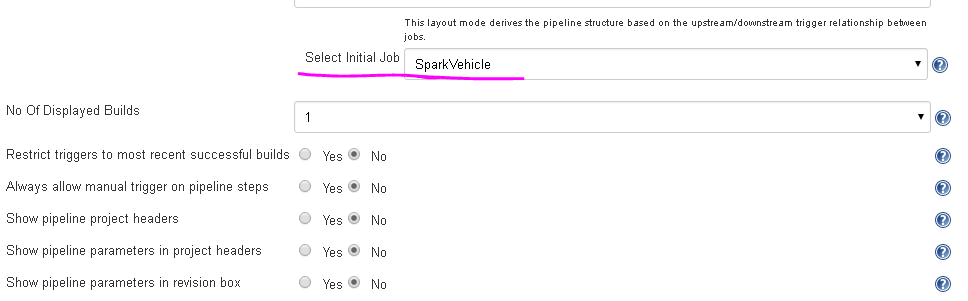
Note: add all the dependency plugin



1. Give the name and click ok



1. Give the initial job name as shown below.example(I have given the sparkvehicle project which I have already created). And give OK.

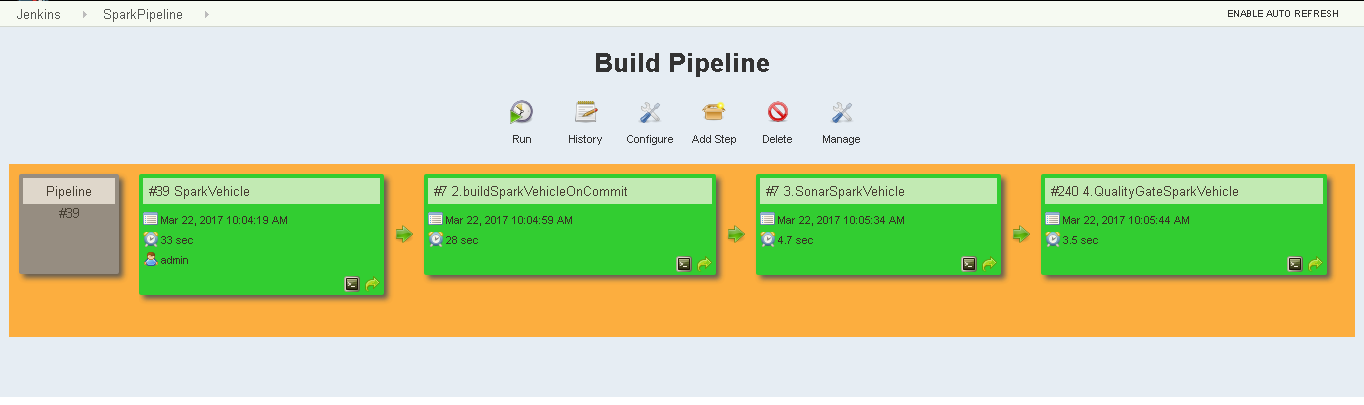


1. And in the SparkVehicle instance in POST\_BUILD action select the next project which you want to run after that project.(**SparkVehicle🡪configure🡪Post-build action**)



Like this you can add the project to the pipeline.

Sample pipeline



This is how you can create the pipeline in Jenkins.