Contents

[Initial Setup 2](#_Toc206614042)

[Setup 2](#_Toc206614043)

[Maven Plugin 2](#_Toc206614044)

[Jenkins Location 4](#_Toc206614045)

[Throttle Builds 4](#_Toc206614046)

[Project as parameterized 6](#_Toc206614047)

[Trigger builds remotely 7](#_Toc206614048)

[Poll SCM 8](#_Toc206614049)

[Webhooks 8](#_Toc206614050)

[Adding Credentials 9](#_Toc206614051)

[Jenkins Pipeline 9](#_Toc206614052)

[Common Setup 9](#_Toc206614053)

[Pipeline Syntax 11](#_Toc206614054)

[Pipeline GitHub 11](#_Toc206614055)

[SonarCloud 11](#_Toc206614056)

[Quality Profiles 13](#_Toc206614057)

[Pipeline Parallel 13](#_Toc206614058)

[Deploy to Tomcat Server 14](#_Toc206614059)

[Build using Docker instead of dedicated agent 14](#_Toc206614060)

[GitHub Actions 14](#_Toc206614061)

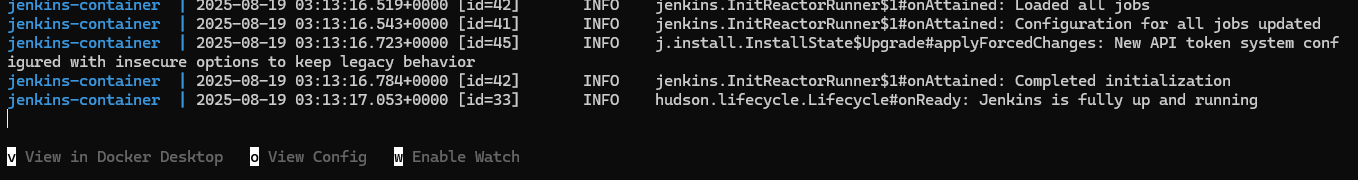
# Initial Setup

## Setup

1. Copy docker-compose file into a location



1. Go to the location and execute docker compose up (or up –d) and check if this line appears

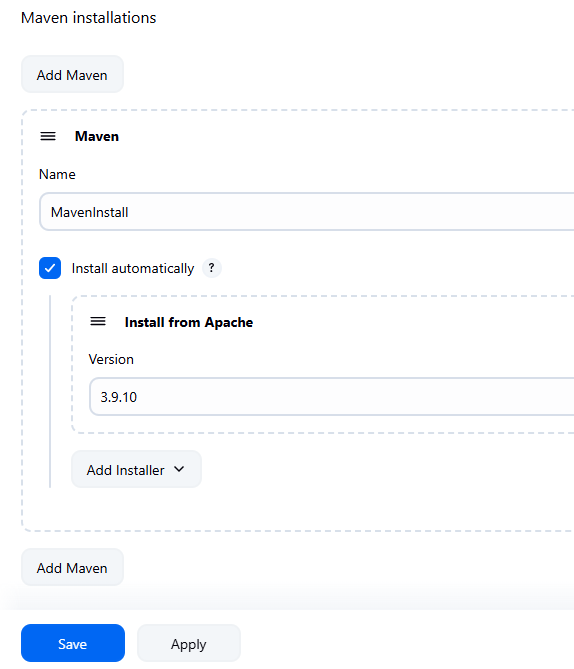


1. Execute docker ps to see list of containers
2. Once Jenkins container starts execute docker exec <container\_name> cat /var/jenkins\_home/secrets/initialAdminPassword
3. If password not there, go to user directory and find the user and password
4. deepannr/deepannr

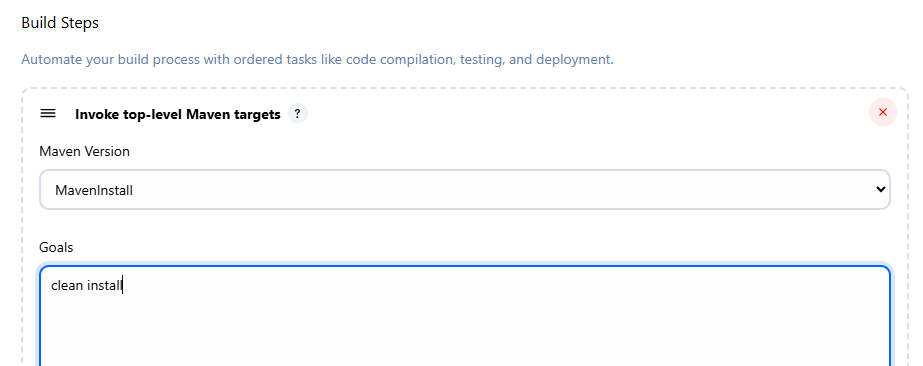
## Maven Plugin

First Install Maven Integration Plugin

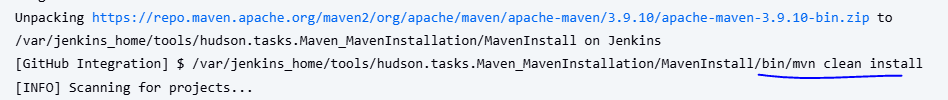
Also in Manage Jenkins >> Tools >> Maven installations, Add Maven



Then while creating a new freestyle job in Jenkins under the Build Steps, in the option **Invoke top-level Maven targets** >> Maven Install will be available.

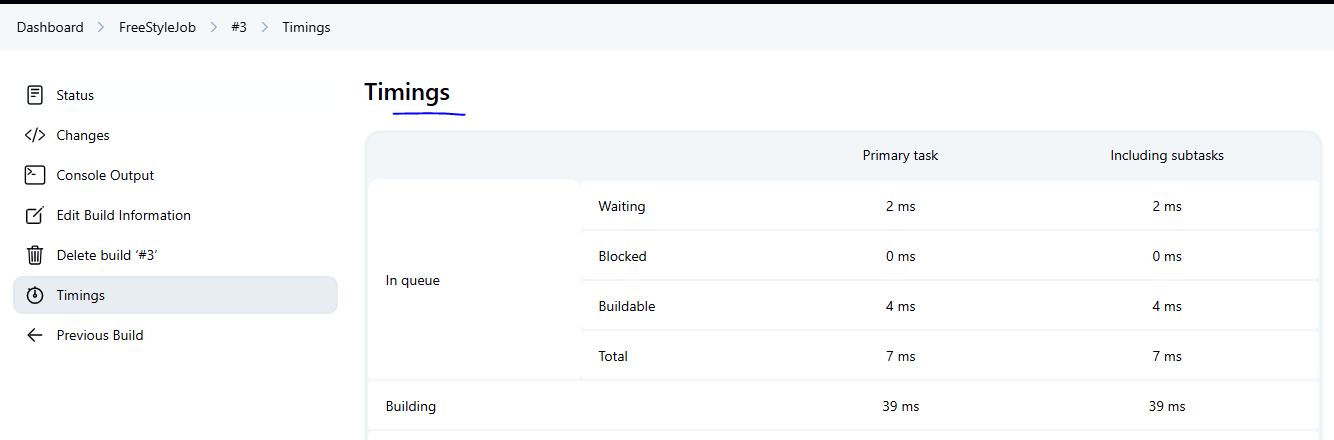


While executing, it will perform maven clean install



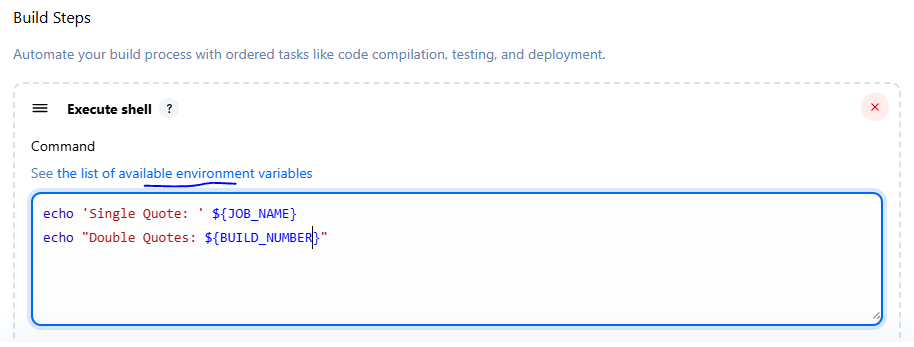
Plugin Documentation 🡪 <https://plugins.jenkins.io/> has list of plugins available with documentation

Will provide the timing to execute



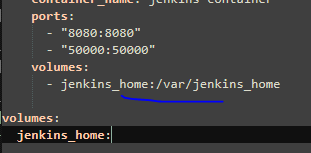
Execute shell 🡪 Help Documentation <http://localhost:8080/env-vars.html/>

Also in Build targets, can execute shell command as well.

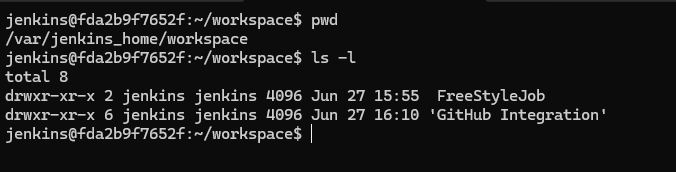


## Jenkins Location

As home is /var/Jenkins\_home from docker compose file,



/var/Jenkins\_home/workspace has all the builds

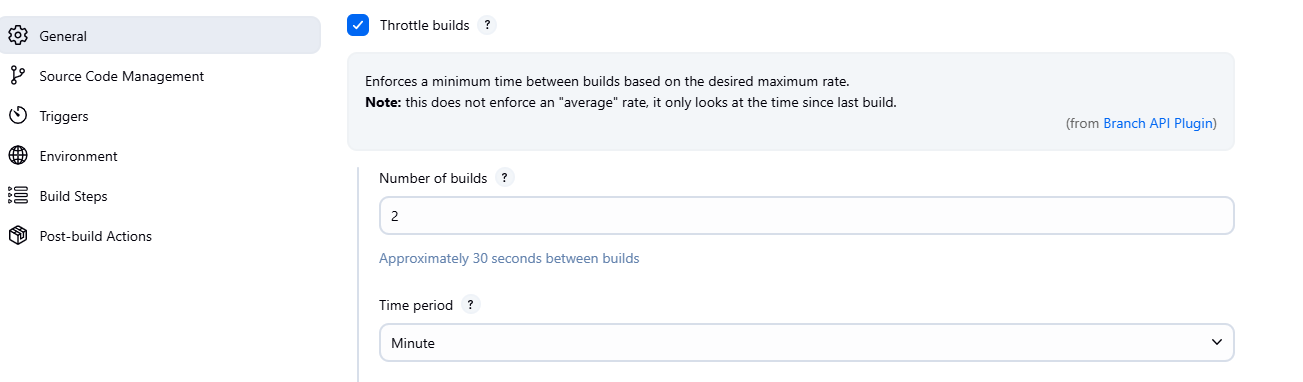


# FreeStyle Project

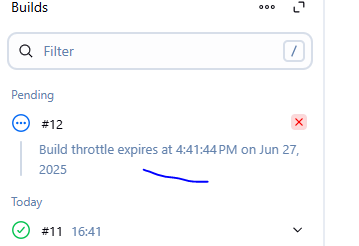
## Throttle Builds

Throttling builds limits the number of builds that can be triggered within a specific time frame, preventing excessive resource usage or unintended parallel execution

Will trigger from GitHub every certain time. Here it will trigger every 2 min.



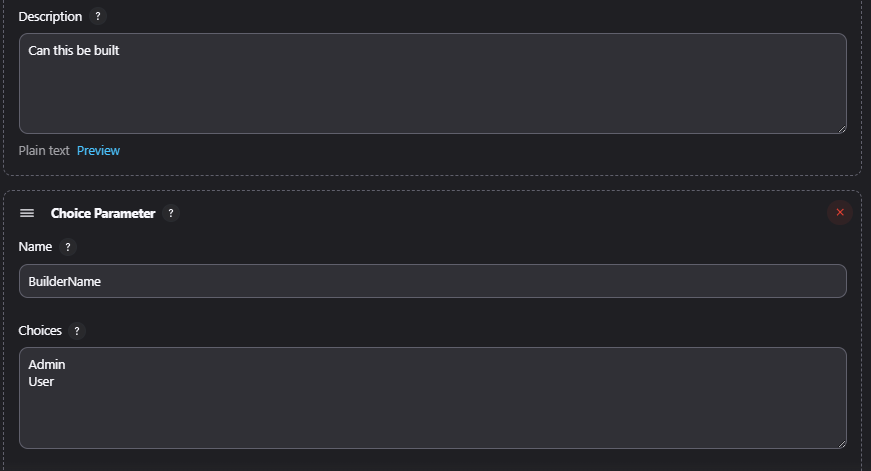
Then when another build is triggered it will wait for 2 min



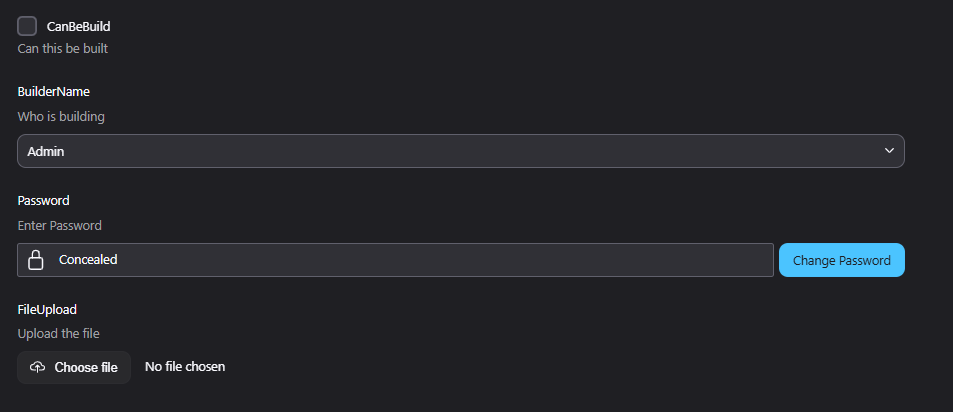
But on selecting this option, will execute build now manually, the throttle build will be overridden and applicable for checkin from git.

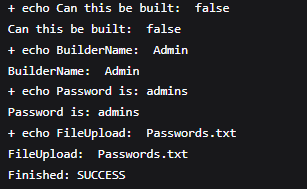


## Project as parameterized

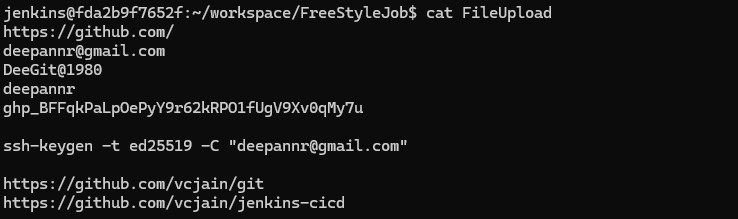
In Jenkins builds, 

While initiating build, all parameters to be filled. Checkbox, Choice, password and file upload Param are available





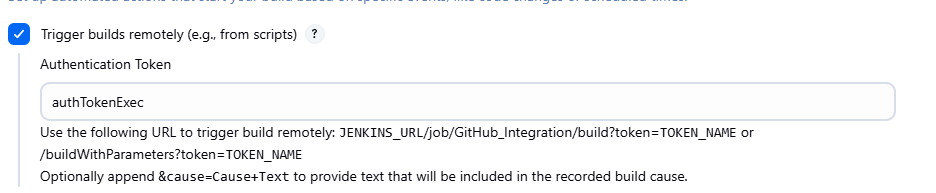
And the File will be copied to FileUpload



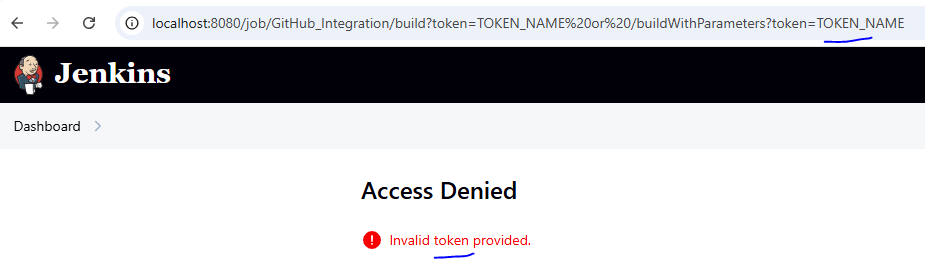
## Trigger builds remotely

Can Execute Jenkins build remotely from other place and not login into Jenkins.

Here whoever has the token authTokenExec, can execute the build without login to Jenkins.



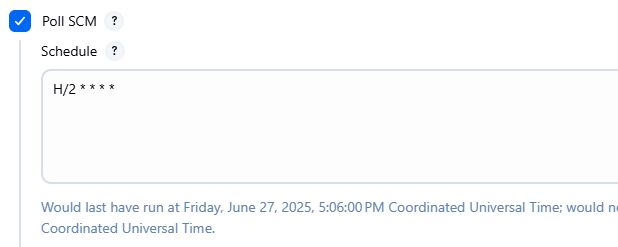
If any other token provided then will be invalid



On providing <http://localhost:8080/job/GitHub_Integration/build?token=authTokenExec> correct token, build will be triggered.

If the project is parameterized, then after the above URL, then need to provide params <http://localhost:8080/view/all/job/GitHub_Integration/build?token=authTokenExec&ApproveToBuild=Yes&Builder=Admin>

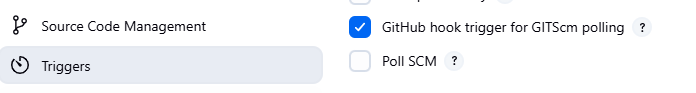
## Poll SCM

Here from the GitHub [repo, every 2 minutes if code is checked in then the build will be triggered.](http://localhost:8080/job/GitHub_Integration/configure)

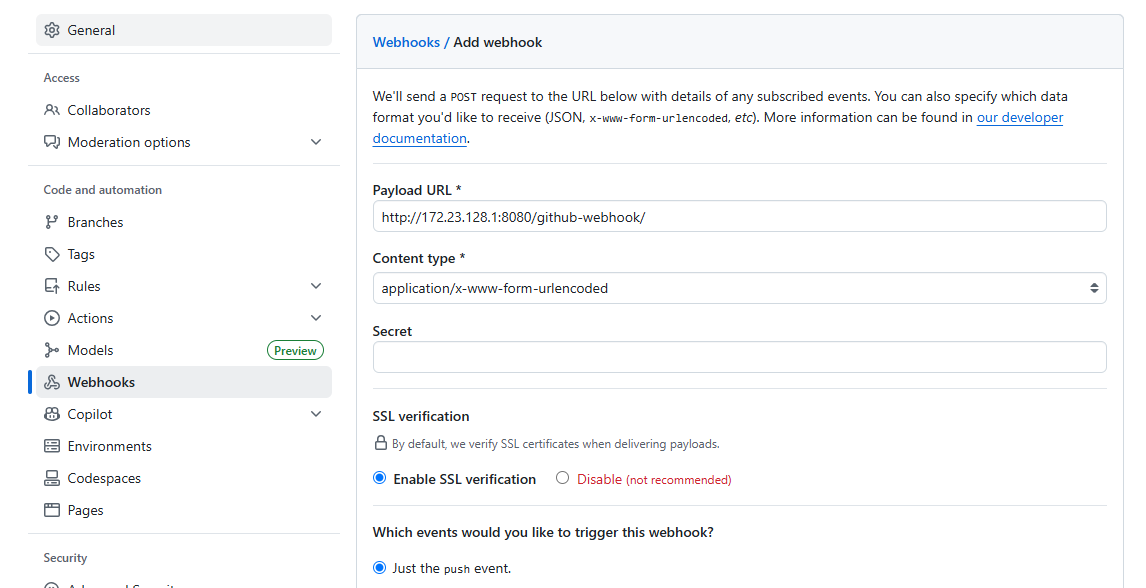
This option is a pull mechanism as every 2 min, need to check and this is not a recommended one. Instead of the push mechanism is preferred

## Webhooks

1. Select the option in build triggers



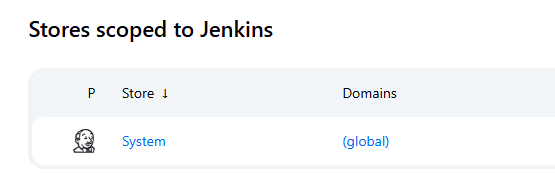
1. Go to the Github repository >> Settings >> Webhooks
2. Add public ip address



1. Whenever any code is checked in the build will be triggered

## Adding Credentials

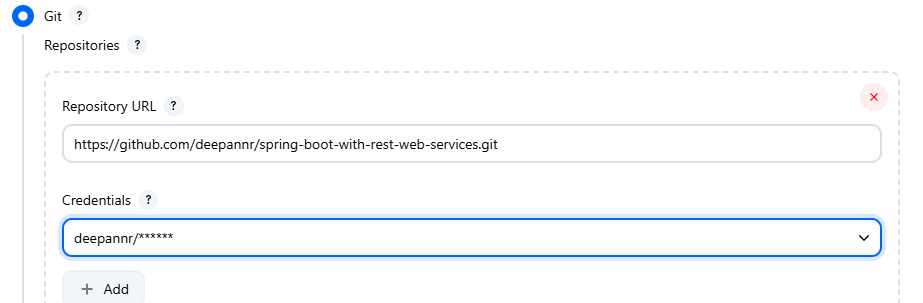
1. Manage Jenkins >> Credentials click global and click down arrow key and click add credentials



1. Add Username and password, it has multiple options



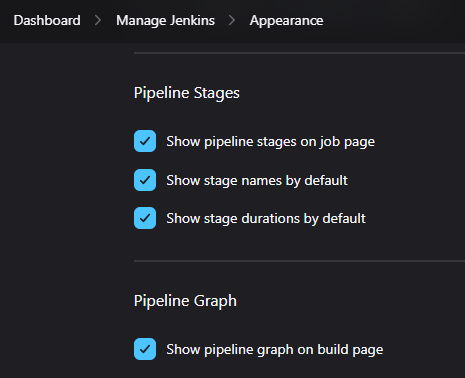
1. And in SCM Git, Credentials will be visible



# Jenkins Pipeline

## Common Setup

Check if those options are selected



Pipeline is output of one stage will be input of another. Pipeline steps can be checked into git and if Jenkins got corrupted, job can be created by pasting the Jenkins pipeline script in git

To write any pipeline script, click Pipeline Syntax and goes to Snippet generator and providing inputs, will give code.

Slack

In Manage Jenkins >> Plugins >> Available Plugins, install Slack notification

Homework 🡪 add slack notification,

If any build failed, perform Restart from Stage and select which step want to restart

Jacoco Sample Git (Code Coverage) 🡪 <https://github.com/vcjain/jenkins_pipeline_springboot_demo.git>

SonarQube Sample Git 🡪 <https://github.com/vcjain/jenkins-sonar-demo.git>

Pipeline Documentation 🡪 <https://www.jenkins.io/doc/book/pipeline/syntax/>

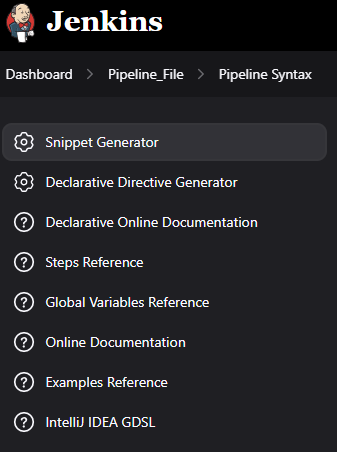
Sonar Token 🡪 6cbe4eca0949951ad5087411b6dce3d22696e4d7

Learn on SonarQube

Homework 🡪 Notification

To have any help, there is an option for Pipeline Syntax which navigates to <http://localhost:8080/job/Pipeline_File/pipeline-syntax/> where all options are available

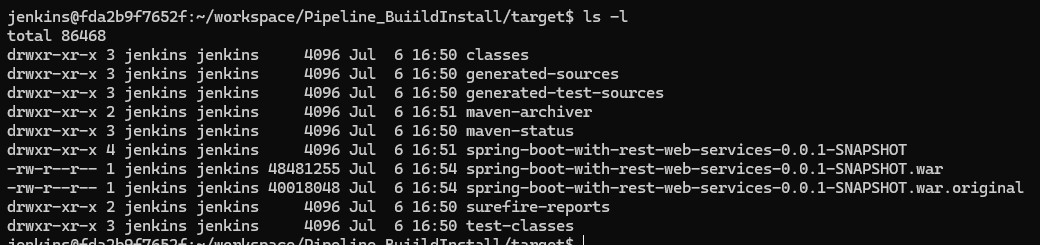
## Pipeline Syntax



## Pipeline GitHub

Look at the file <https://github.com/deepannr/spring-boot-with-rest-web-services/blob/master/Jenkinsfile-Maven>

In the Jenkins workspace



## SonarCloud

1. In Jenkins, Add Plugins Quality Gates, Sonargraph Integration, CodeSonar, SonarQube Generic Coverage.
2. Go to <https://sonarcloud.io/> and create Organization
3. <https://sonarcloud.io/account/security> and create a token and copy the token as will not be visible
4. Go to Jenkins and create credentials



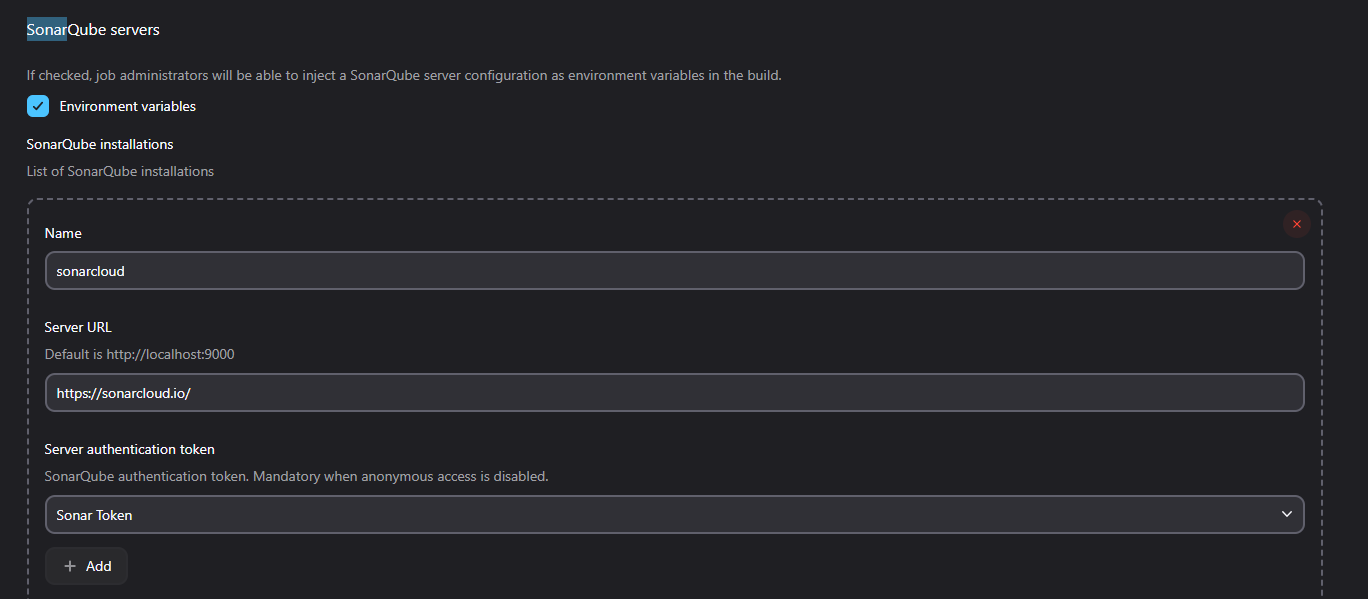
1. In Sonar.io, organizations

<https://sonarcloud.io/organizations/deepannr/quality_profiles> – sets profiles for each and every technology

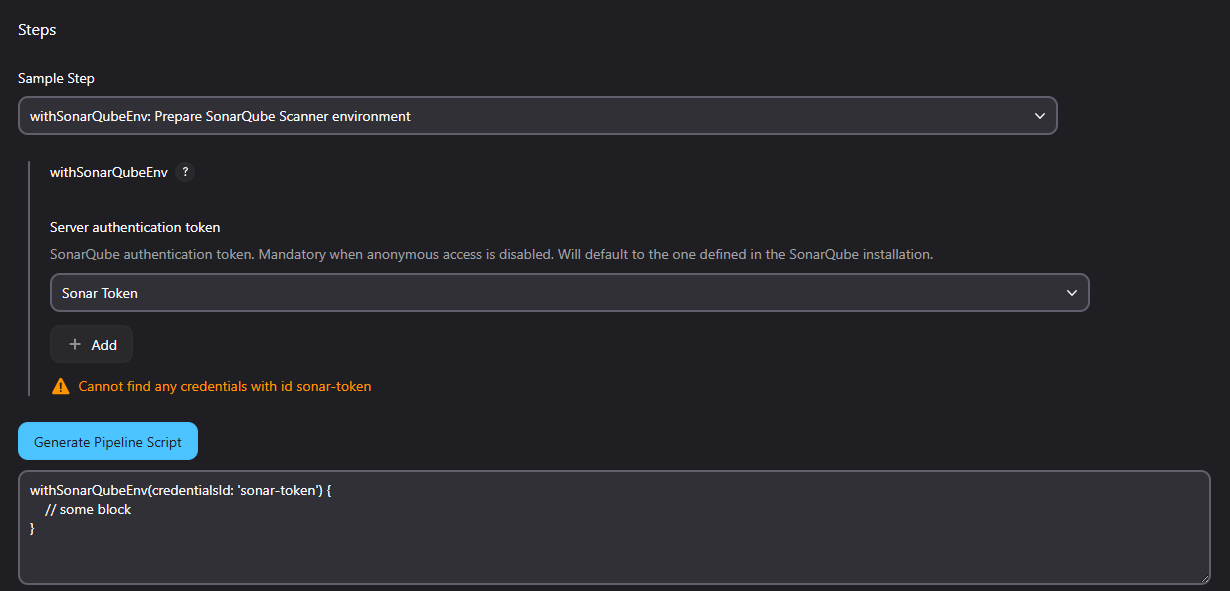
<https://sonarcloud.io/organizations/deepannr/quality_gates/> - mentions the severity

<https://sonarcloud.io/organizations/deepannr/rules> - Specify rules on severity

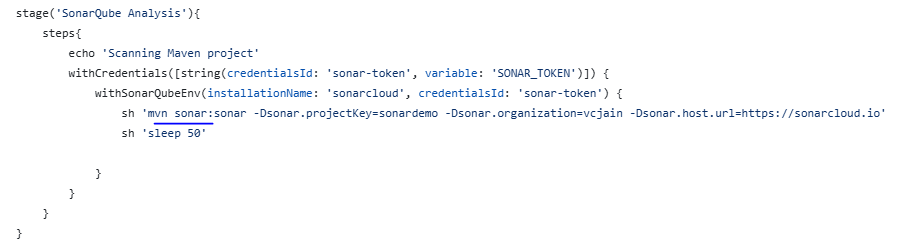
1. In manage Jenkins >> System and setup SonarQube Server



1. Generate syntax of sonar from pipeline syntax

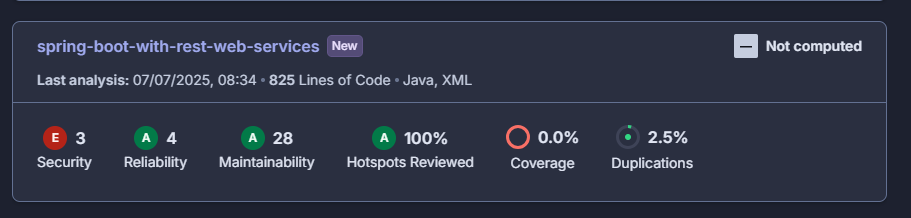


1. In the Sonar step



Mvn sonar gets the credentials from sonar-token and for the organization settings, runs the for sonardemo project

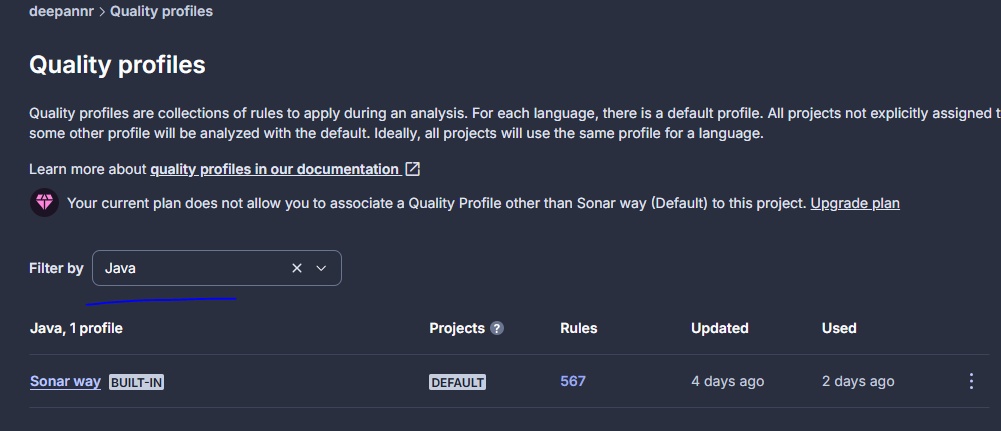
1. Execute the pipeline and after it executes successfully, navigate to <https://sonarcloud.io/organizations/deepannr/projects> to see the project



1. <https://github.com/deepannr/spring-boot-with-rest-web-services/blob/master/Jenkinsfile-Sonar> is the Jenkins Pipeline file used.

## Quality Profiles

<https://sonarcloud.io/organizations/deepannr/quality_profiles> >> Quality Profiles there can see for all technologies and can extend or copy or perform new profiles

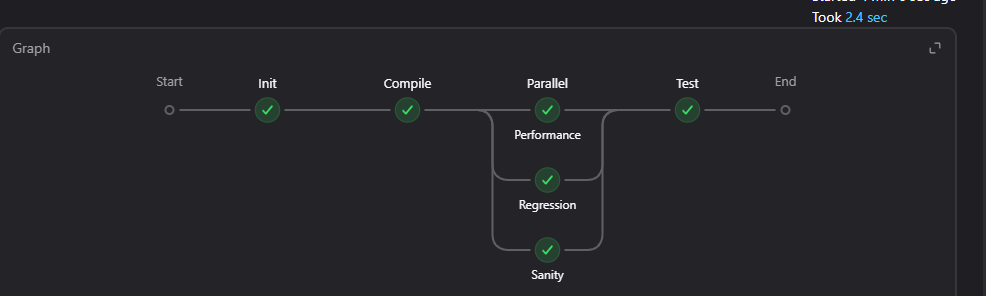


## Pipeline Parallel

Jenkins can execute stages in parallel to perform that add parallel stages inside main stage



While executing the Jenkins, all parallel stages will execute in parallel



## Deploy to Tomcat Server

1. Install tomcat by any means by docker or externally
2. Install Deploy to containerVersion Plug in Jenkins

01:37:30

<https://github.com/vcjain/jenkins_tomcat_demo> -- Build and deploy to tomcat

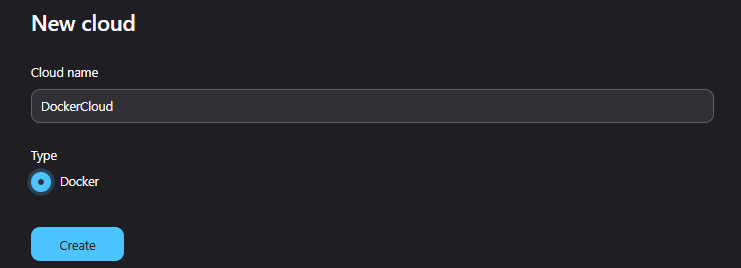
Jenkinsfile-Input-slack – continuous deployment

<https://github.com/vcjain/jenkins_tomcat_demo/blob/main/Jenkinsfile-sampleinput>

<https://github.com/vcjain/docker-agent-demo> - Adding new agent in Jenkins

## Build using Docker instead of dedicated agent

1. In Jenkins, Add Plugins Docker and Docker Pipeline
2. Manage Jenkins >> Clouds >> New Cloud and name as DockerCloud

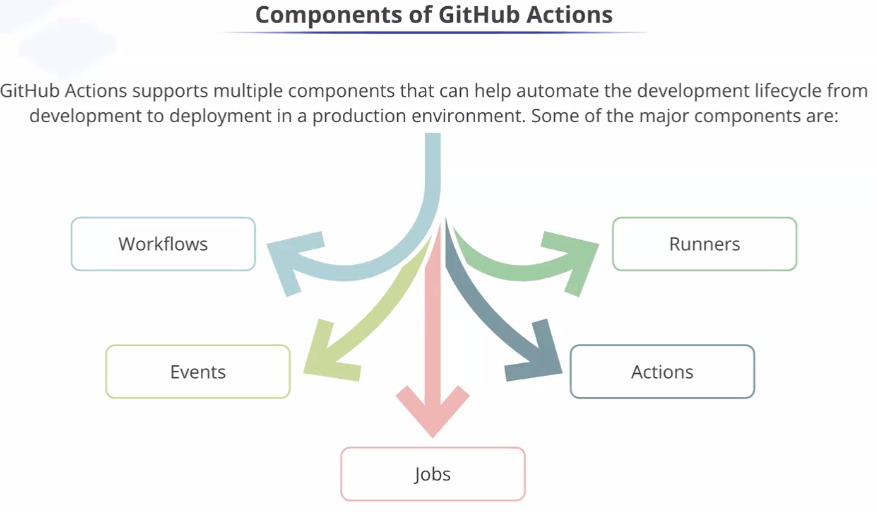


1. Follow <https://github.com/deepannr/simpli-DevOps/blob/main/docker-cloud-setup.md> to complete

# GitHub Actions

Runner 🡪 Execute the job its same as Slave in Jenkins

Actions 🡪 Like Plugins in Jenkins. Advantage is no need to install Actions as they are already available



GitHub Repo 🡪 <https://github.com/vcjain/github-action>