Jenkins-SonarQube Integration

> **Product Name:** Integration jenkins with sonarqube.
> **Platform** : Nodejs.
> **Phase:** Deployement
> **Date:** 🗓 Nov 10, 2022
> **Creator** : **Fayaz Shaik**

Requirements :

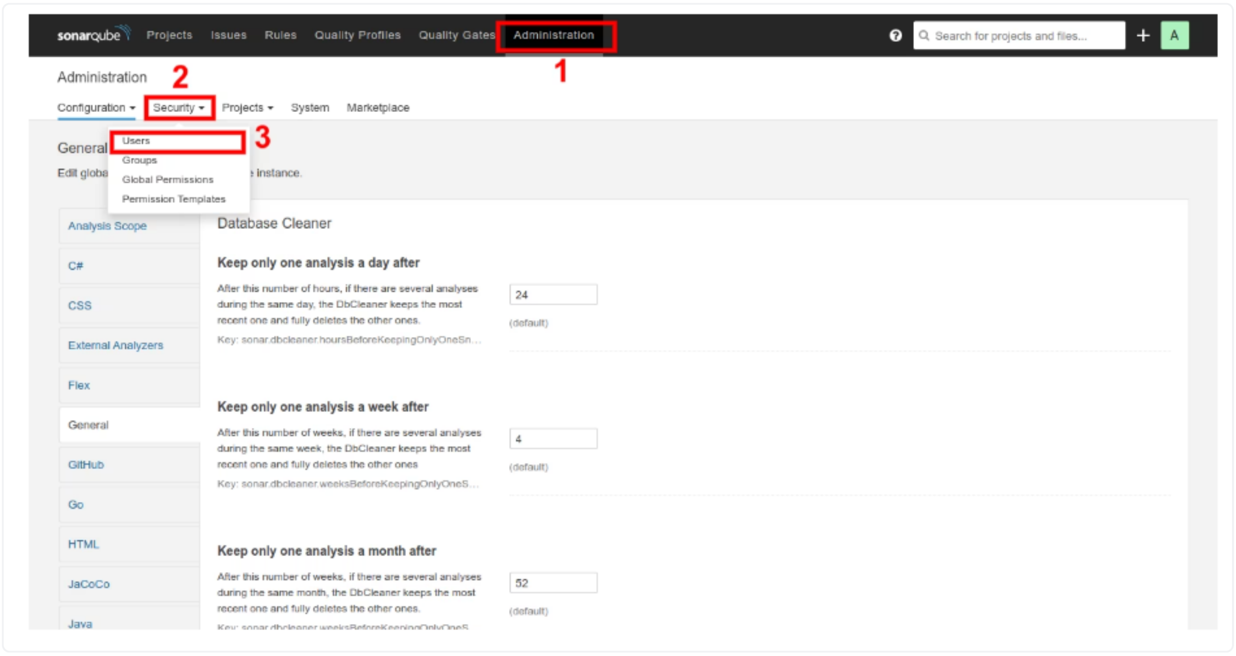
1. SonarqubeLogin credentials.

Jenkins-SonarQube Integration:

- **Assume a Scenario:** After I committed code to *GitHub*. I want to ensure my code quality and know bugs, vulnerabilities, code smells, etc. (static code analysis) for my code before I build my code automatically with *Jenkins* and I want this activity to perform every time I commit code.
- In this scenario for **Continuous Integration** of the code. We will follow the best practice using ***GitHub-Jenkins-SonarQube*** Integration for this scenario.
- **Flow:** As soon as a developer commits the code to *GitHub*, *Jenkins* will fetch/pull the code from the repository and will perform static code analysis with help of *Sonar Scanner* and send the analysis report to *SonarQube Server* then it will automatically build the project code.

SonarQube Configuration:

  - We will begin with SonarQube. To connect it with Jenkins, you need to generate the token to access the SonarQube instance.
  - Login into the SonarQube dashboard and go to the Administrator tab. In the Security drop-down menu, select the User tab as shown in the below image:
  - Only Admin can create the Token this Administration icon will be shown only for admins.



Edit globa… instance.

Database Cleaner

Keep only one analysis a day after

After this number of hours, if there are several analyses during the same day, the DbCleaner keeps the most recent one and fully deletes the other ones.

Key: sonar.dbcleaner.hoursBeforeKeepingOnlyOneSn…

24 (default)

Keep only one analysis a week after

After this number of weeks, if there are several analyses during the same week, the DbCleaner keeps the most recent one and fully deletes the other ones

Key: sonar.dbcleaner.weeksBeforeKeepingOnlyOneS…

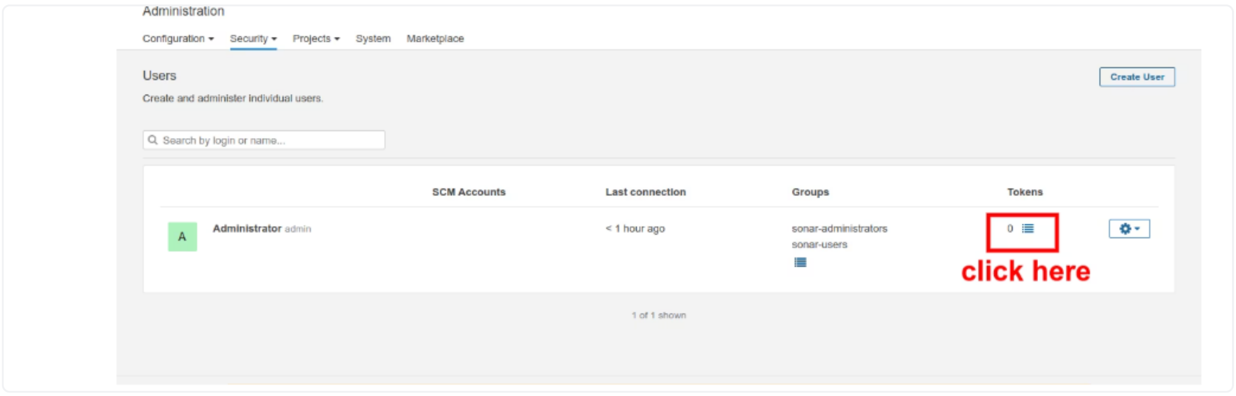
4 (default)

Keep only one analysis a month after

After this number of weeks, if there are several analyses during the same month, the DbCleaner keeps the most recent one and fully deletes the other ones.

52 (default)

- Here you will find the Administrator user, for which you are going to generate the access token:

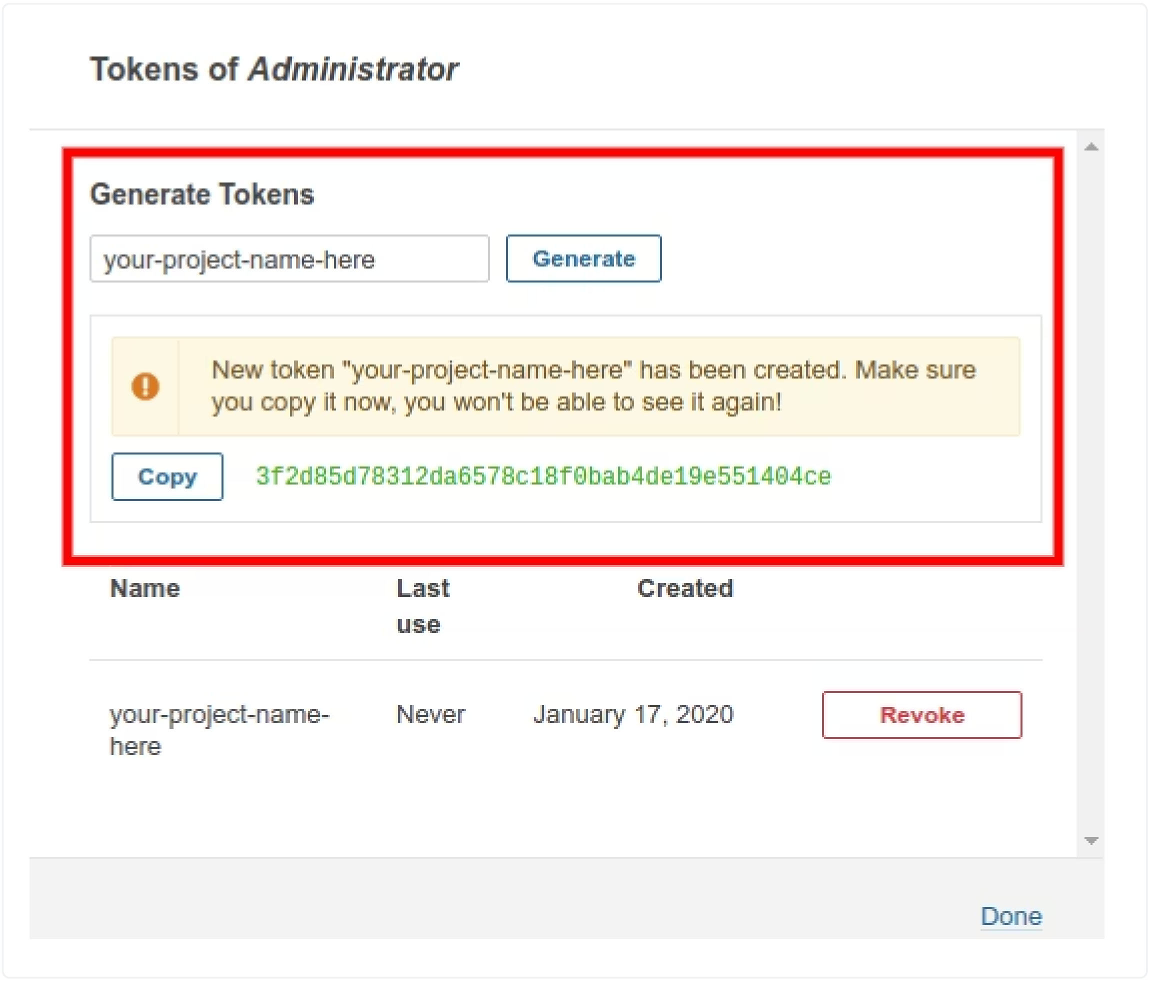


Create and administer individual users.

	SCM Accounts	Last connection	Groups	Tokens
A Administrator admin		< 1 hour ago	sonar-administrators sonar-users	0

1 of 1 shown

- In the pop-up that displays on your screen, enter the desired name and click on the *Generate* button. Copy the newly generated token and save it somewhere safe, as you won't be able to view/copy the generated token again.



Tokens of *Administrator*

Generate Tokens

your-project-name-here | Generate

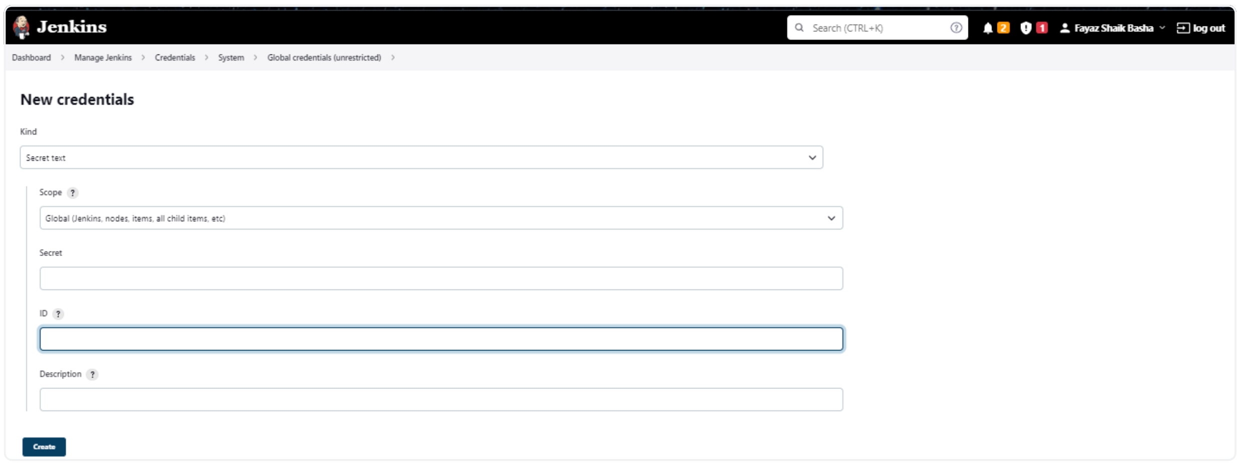
New token "your-project-name-here" has been created. Make sure you copy it now, you won't be able to see it again!

Copy `3f2d85d78312da6578c18f0bab4de19e551404ce`

Name	Last use	Created	
your-project-name-here	Never	January 17, 2020	Revoke

Done

- ◦ Lastly, you have to add the access token you generated on your SonarQube server in Jenkins. go to `Dashboard >manage jenkins >manage Credentials > System`, as shown below:



Jenkins | Search (CTRL+K) | Fayaz Shaik Basha | log out

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

New credentials

Kind: Secret text

Scope: Global (Jenkins, nodes, items, all child items, etc)

Secret

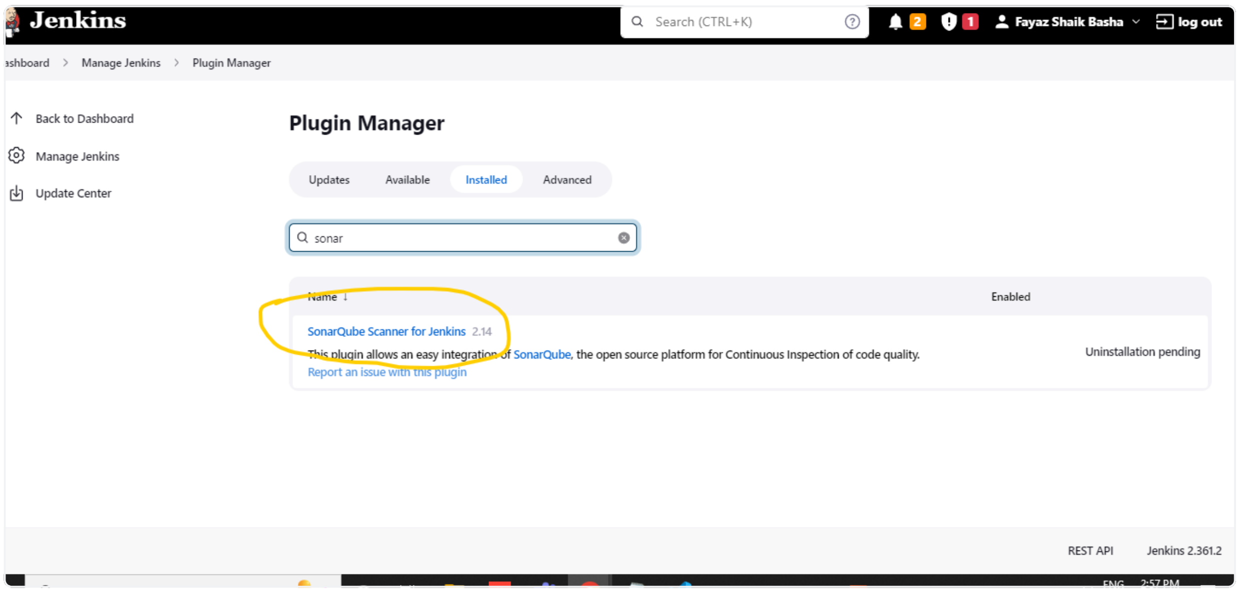
ID

Description

Create

- ◦ select kind as a secret test then pass your token to the secret field and in ID give some name like Ex:sonar_token and add some description and click create.

- Install ***SonarQube plugin*** to Jenkins. Go to *Manage Jenkins* > *Manage Plugins* > *Available* > Search for *SonarQube Scanner> Install.*



Jenkins | Search (CTRL+K) | Fayaz Shaik Basha | log out

ashboard > Manage Jenkins > Plugin Manager

Back to Dashboard
Manage Jenkins
Update Center

Plugin Manager

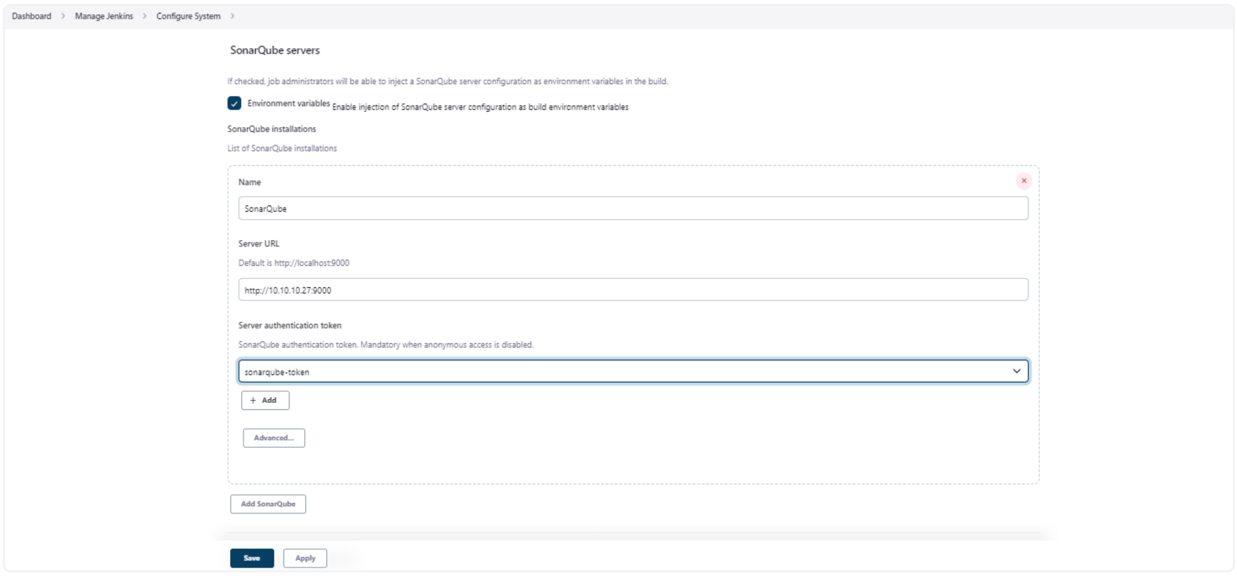
Updates | Available | Installed | Advanced

sonar

Name ↓	Enabled
SonarQube Scanner for Jenkins 2.14 This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality. Report an issue with this plugin	Uninstallation pending

REST API | Jenkins 2.361.2

- Here, you are going to add the access token you previously created to your Jenkins server.



Dashboard > Manage Jenkins > Configure System >

SonarQube servers

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

[✓] Environment variables Enable injection of SonarQube server configuration as build environment variables

SonarQube installations
List of SonarQube installations

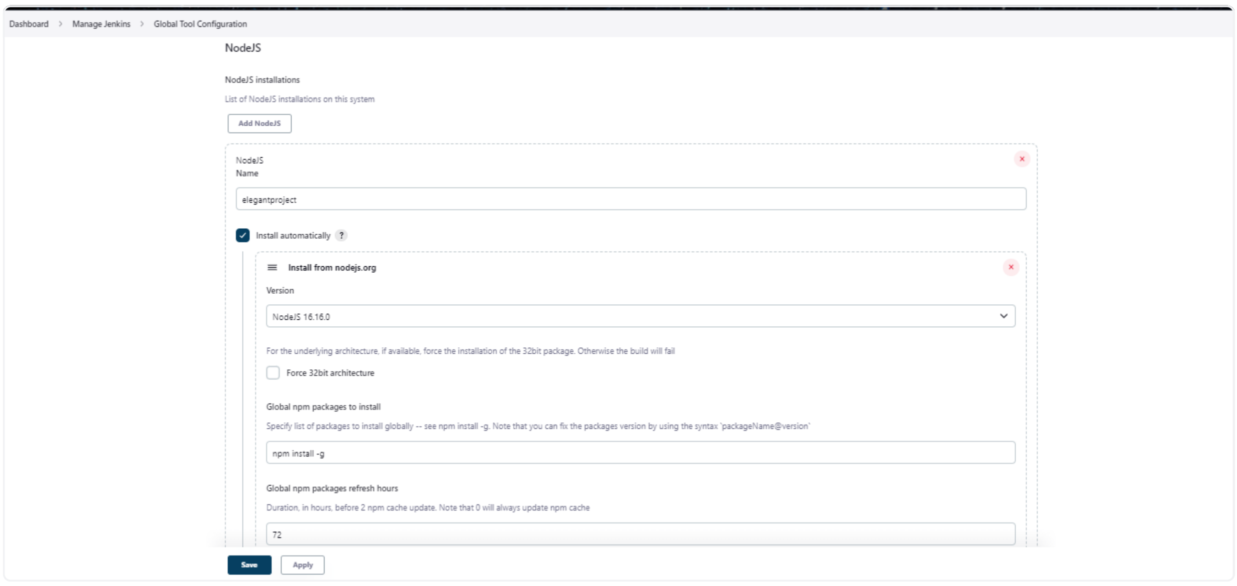
Name: SonarQube

Server URL
Default is http://localhost:9000
http://10.10.10.27:9000

Server authentication token
SonarQube authentication token. Mandatory when anonymous access is disabled.
sonarqube-token

- click apply and save.
- In this guide, we are going to use a simple NodeJs application; hence it's time to install NodeJS on your Jenkins Server.
- To install the NodeJS plugin, go to `Dashboard > Manage Jenkins > Manage Plugins > Available` and search and select NodeJS. Click on the *Install without restart* button, as shown in the figure below.
- Upon successful installation of the NodeJS plugin in Jenkins, make sure that you restart Jenkins.

To configure the NodeJS plugin, go to `Dashboard > Manage Jenkins > Global Tool Configuration` and find "NodeJS". Click on the NodeJS installation button and add the necessary details, as shown in the figure:


Dashboard > Manage Jenkins > Global Tool Configuration

NodeJS

NodeJS installations

List of NodeJS installations on this system

Add NodeJS

NodeJS
Name

elegantproject

☑ Install automatically

Install from nodejs.org

Version

NodeJS 16.16.0

For the underlying architecture, if available, force the installation of the 32bit package. Otherwise the build will fail

☐ Force 32bit architecture

Global npm packages to install

Specify list of packages to install globally -- see npm install -g. Note that you can fix the packages version by using the syntax `packageName@version`

npm install -g

Global npm packages refresh hours

Duration, in hours, before 2 npm cache update. Note that 0 will always update npm cache

72

- Give a name and select the NodeJS version as per your requirement; you can also install Global NPM packages and set the refresh rate depending upon the project requirements.
- Once done, you need to save the configuration. As we are through with the gruesome part of installing and configuring the environment, it's time to create the project pipeline.

Creating Jenkins Pipeline:

- Before creating a pipeline you should add the sonar-properties.js file in your project and add the below.

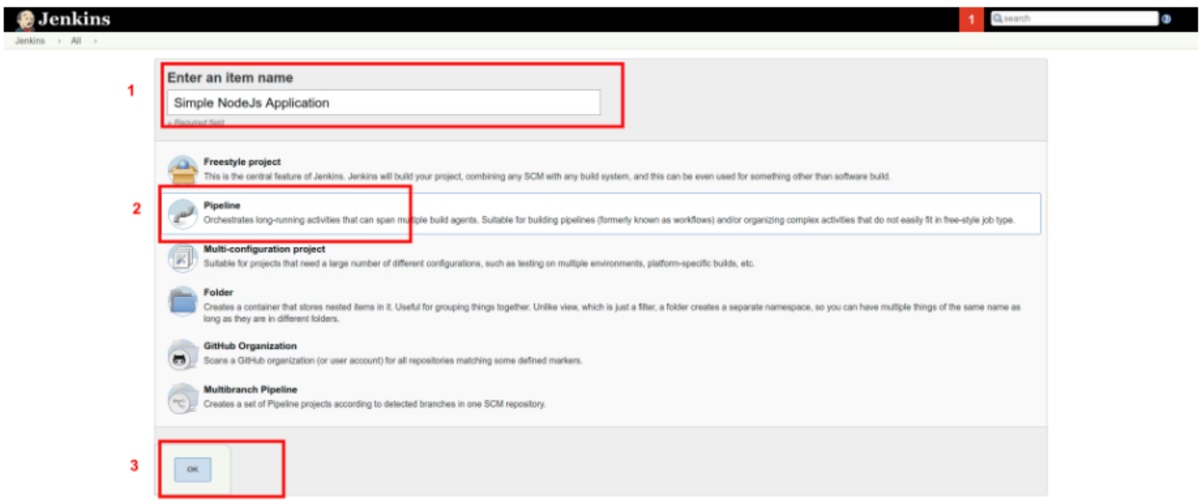
```
const sonarqubeScanner = require('sonarqube-scanner');
sonarqubeScanner({
    serverUrl:'http://10.10.10.27:9000', //sonarqube server url
    options : {
        'sonar.projectDescription': 'This is a node JS project',
        'sonar.projectName':'elegant-latestNew', //project name
        'sonar.projectKey':'elegant-latest', //project-key
        'sonar.login':'f0895645c9146816499912*********', //paste
your sonar token here
        'sonar.projectVersion':'1.0',
        'sonar.language':'js',
        ' sonar.sourceEncoding':'UTF-8',
        'sonar-sources':'.'
    },
}, () => {});
```

- Install the package "sonarqube-scanner" : "^2.7.0"

```
//in package.json
"scripts": {
 "sonar":"node sonar-project.js"
  }, //add this into scripts
   "dependencies": {
   "sonarqube-scanner" : "^2.7.0"
  }
```

Step 1 - Create a New Job

Go to Jenkins' Dashboard and click on the "New Item" link. Now, enter the item name and select *the Pipeline* option as shown in the figure:



Enter an item name

Simple NodeJs Application

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

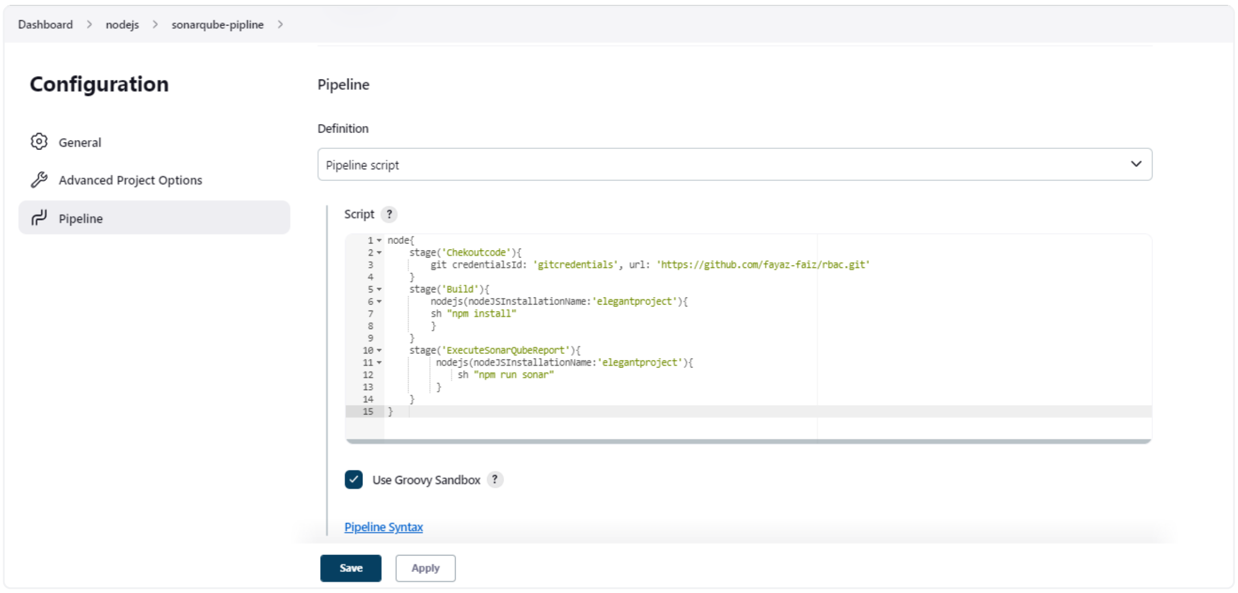
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

GitHub Organization
Scans a GitHub organization (or user account) for all repositories matching some defined markers.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

Step 2 - Pipeline



```
node{
    stage('Chekoutcode'){
        git credentialsId: 'gitcredentials', url: 'https://github.com/fayaz-faiz/rbac.git'
    }
    stage('Build'){
        nodejs(nodeJSInstallationName:'elegantproject'){
        sh "npm install"
        }
    }
    stage('ExecuteSonarQubeReport'){
         nodejs(nodeJSInstallationName:'elegantproject'){
             sh "npm run sonar"
         }
    }
}
```

[✓] Use Groovy Sandbox

script below:-

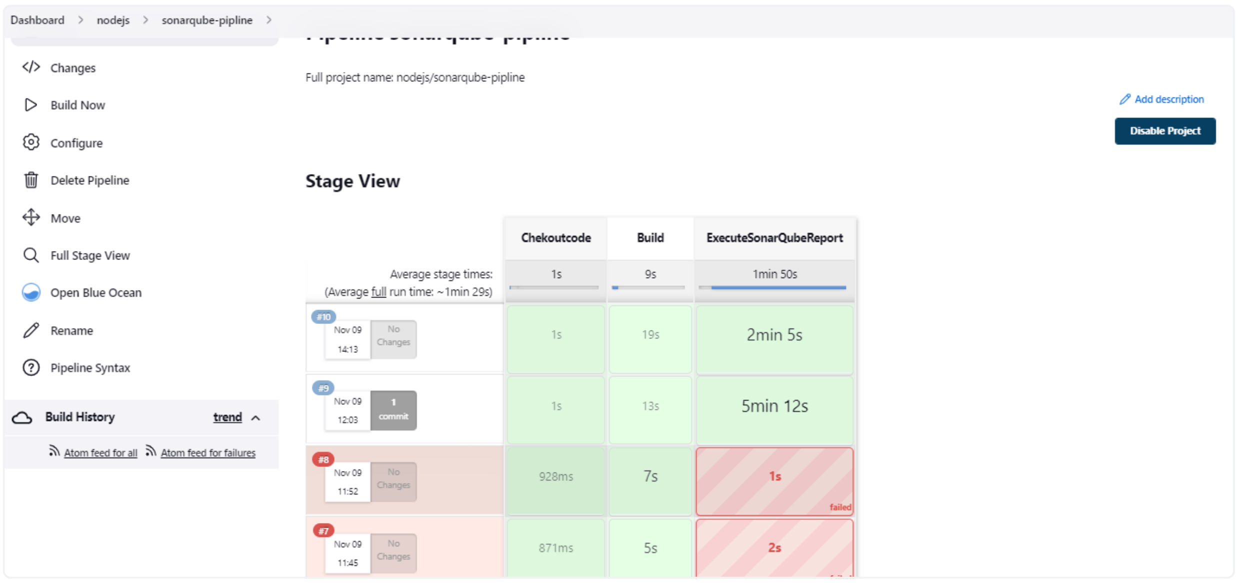
- Select **Pipeline Script** and add the following script:

```

node{
    stage('Chekoutcode'){
        git credentialsId: 'gitcredentials', url:
'https://github.com/fayaz-faiz/rbac.git'
    }
    stage('Build'){
        nodejs(nodeJSInstallationName:'elegantproject'){
        sh "npm install"
        }
    }
    stage('ExecuteSonarQubeReport'){
         nodejs(nodeJSInstallationName:'elegantproject'){
             sh "npm run sonar"
         }
    }
}
```

✓ Click on **Save** and **Apply**

- As soon as you click the *Build Now* link, Jenkins will start building project as per pipeline script. In Build History, you will see the progress bar for the current build along with the Stage View:
- If your build runs successfully, you will be able to see the time taken by each stage, in *Stage View*.

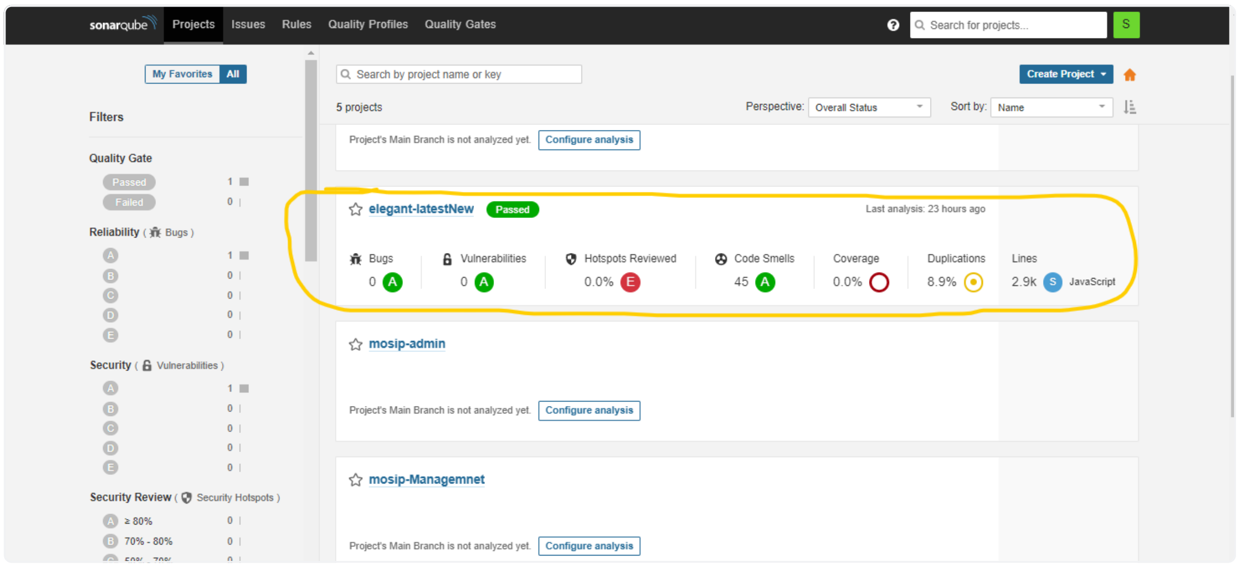


Full project name: nodejs/sonarqube-pipline

Stage View

	Chekoutcode	Build	ExecuteSonarQubeReport
Average stage times: (Average full run time: ~1min 29s)	1s	9s	1min 50s
#10 Nov 09 14:13 No Changes	1s	19s	2min 5s
#9 Nov 09 12:03 1 commit	1s	13s	5min 12s
#8 Nov 09 11:52 No Changes	928ms	7s	1s failed
#7 Nov 09 11:45 No Changes	871ms	5s	2s

- Also, you can visit the SonarQube dashboard to see the project code report, by visiting the link named as "SonarQube" on project pipeline page.
-



Filters

Quality Gate: Passed 1; Failed 0

Reliability (Bugs): A 1; B 0; C 0; D 0; E 0

Security (Vulnerabilities): A 1; B 0; C 0; D 0; E 0

Security Review (Security Hotspots): A ≥ 80% 0; B 70% - 80% 0

5 projects

Perspective: Overall Status | Sort by: Name

Project's Main Branch is not analyzed yet. Configure analysis

elegant-latestNew Passed — Last analysis: 23 hours ago

Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications	Lines
0 A	0 A	0.0% E	45 A	0.0%	8.9%	2.9k S JavaScript

mosip-admin — Project's Main Branch is not analyzed yet. Configure analysis

mosip-Managemnet — Project's Main Branch is not analyzed yet. Configure analysis

- That's it! You have successfully created a Jenkins Pipeline while using SonarQube and Github. Now, every time you push the code to the repo, you can build the project, which will show the code quality.