EXPERIMENT NO. 08 DATE –

Aim -

Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web / Java / Python application

Theory -

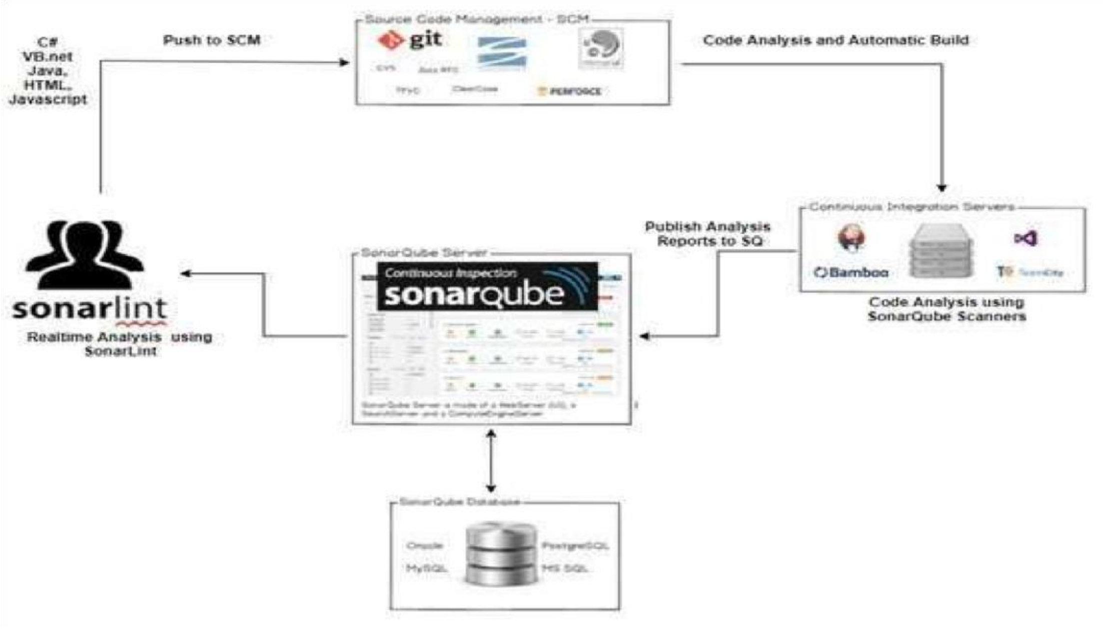
SonarQube

SonarQube is an open-source platform, which is used for continuous analysis of source code quality by performing analysis on your code to detect duplications, bugs, security vulnerabilities and code smells on programming languages.

Jenkins

Jenkins- an open-source automation tool is created using Java programming language. For the initial setup, it facilitates users with CI/CD(continuous integration (CI) or continuous delivery) technique that simplifies the use and management of processes. It is fundamentally focused on continuously building and testing software projects for developers and to implement changes in real-time. In addition, it also allows users to plan a new build whenever the need arises.

Below image shows the architecture diagram which shows an overview of how the Sonarqube will be integrated into the Jenkins build pipeline.


C#
VB.net
Java,
HTML,
Javascript
Push to SCM
Source Code Management - SCM
git
CVS
Jazz RTC
TFVC
ClearCase
PERFORCE
Code Analysis and Automatic Build
Continuous Integration Servers
Bamboo
TeamCity
Code Analysis using SonarQube Scanners
Publish Analysis Reports to SQ
SonarQube Server
Continuous Inspection
sonarqube
sonarlint
Realtime Analysis using SonarLint
SonarQube Database
Oracle
MySQL
PostgreSQL
MS SQL

Architecture of Sonarqube-Jenkins integration as a Continuous code inspection tool.

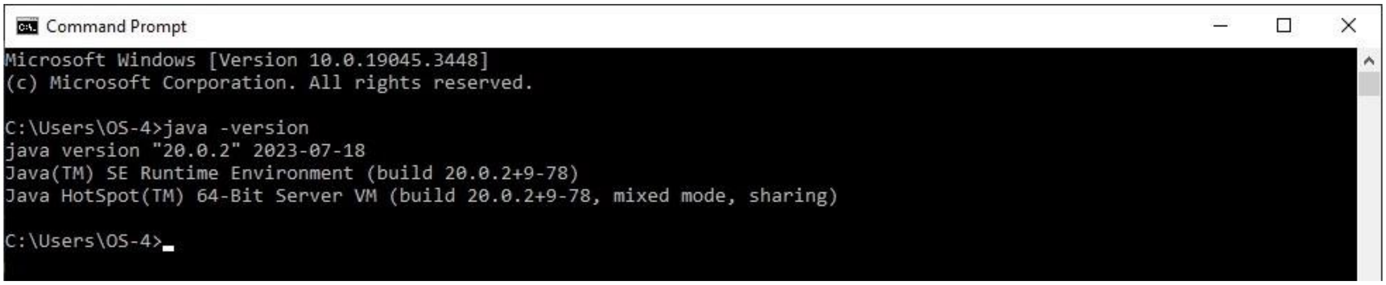
Steps to install SonarQube and Jenkins in your system or machine

Step I:

It is mandatory to install Java SDK on your machine before you decide to install SonarQube. And Java SDK is also needed for the Jenkins automation server running on your machine. If you don't have java SDK on your machine then you install it by clicking the link below.

https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

After installing the Java SDK in your machine or server then you can verify it by opening the command prompt and type java –version as shown in the below image.



```
Microsoft Windows [Version 10.0.19045.3448]
(c) Microsoft Corporation. All rights reserved.

C:\Users\OS-4>java -version
java version "20.0.2" 2023-07-18
Java(TM) SE Runtime Environment (build 20.0.2+9-78)
Java HotSpot(TM) 64-Bit Server VM (build 20.0.2+9-78, mixed mode, sharing)

C:\Users\OS-4>
```

Step II:

Once the installation of Java SDK is completed, there is a need to set up variables in the Java environment.

You can refer to the link below, which explains the steps in order to set up the environmental variables in your machine.

https://confluence.atlassian.com/doc/setting-the-java_home-variable-in-windows-8895.html

Step III:

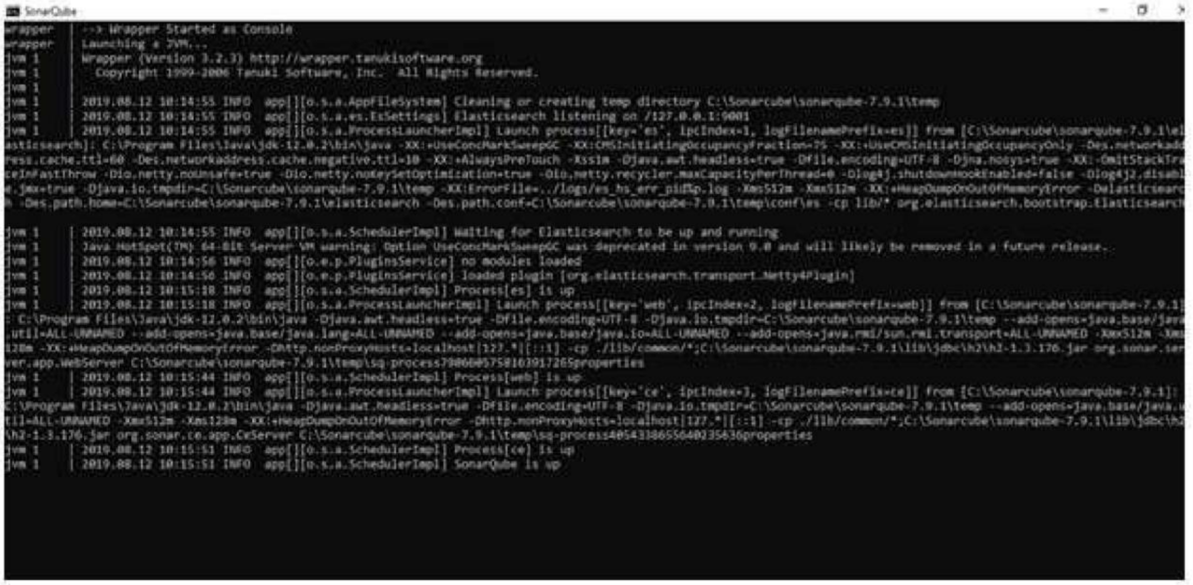
Now the next step is to install the Sonarqube community edition which is free by using the link below As per our expertise, we would recommend to opt for developer’s edition because of its advanced functionalities and features. But if you are a beginner, go for a community edition. https://www.sonarqube.org/downloads/

Unzip the downloaded package and then place the extracted files in your preferred folder. Let say C:\sonarqub

Step IV:

Navigate to the folder sonarqube/conf folder in that you there is a file called wrapper.conf, there you have to the path for the java else you’ll receive an error message show – unable to start JVM Set %JAVA_HOME%/bin/java to the property wrapper.java.command.

After this, go to the folder named windows under the bin folder and run a bat file called startsonar in an administrator user mode. Wait for a few minutes, and then if everything works perfectly.After this step, you will receive a console message as shown in the screenshot with an output message that says “SonarQube is Up”



```
wrapper  | --> Wrapper Started as Console
wrapper  | Launching a JVM...
jvm 1    | Wrapper (Version 3.2.3) http://wrapper.tanukisoftware.org
jvm 1    |   Copyright 1999-2006 Tanuki Software, Inc.  All Rights Reserved.
jvm 1    |
jvm 1    | 2019.08.12 10:14:55 INFO  app[][o.s.a.AppFileSystem] Cleaning or creating temp directory C:\Sonarcube\sonarqube-7.9.1\temp
jvm 1    | 2019.08.12 10:14:55 INFO  app[][o.s.a.es.EsSettings] Elasticsearch listening on /127.0.0.1:9001
jvm 1    | 2019.08.12 10:14:55 INFO  app[][o.s.a.ProcessLauncherImpl] Launch process[[key='es', ipcIndex=1, logFilenamePrefix=es]] from [C:\Sonarcube\sonarqube-7.9.1\el
asticsearch]: C:\Program Files\Java\jdk-12.0.2\bin\java -XX:+UseConcMarkSweepGC -XX:CMSInitiatingOccupancyFraction=75 -XX:+UseCMSInitiatingOccupancyOnly -Des.networkadd
ress.cache.ttl=60 -Des.networkaddress.cache.negative.ttl=10 -XX:+AlwaysPreTouch -Xss1m -Djava.awt.headless=true -Dfile.encoding=UTF-8 -Djna.nosys=true -XX:-OmitStackTra
ceInFastThrow -Dio.netty.noUnsafe=true -Dio.netty.noKeySetOptimization=true -Dio.netty.recycler.maxCapacityPerThread=0 -Dlog4j.shutdownHookEnabled=false -Dlog4j2.disabl
e.jmx=true -Djava.io.tmpdir=C:\Sonarcube\sonarqube-7.9.1\temp -XX:ErrorFile=../logs/es_hs_err_pid%p.log -Xms512m -Xmx512m -XX:+HeapDumpOnOutOfMemoryError -Delasticsearc
h -Des.path.home=C:\Sonarcube\sonarqube-7.9.1\elasticsearch -Des.path.conf=C:\Sonarcube\sonarqube-7.9.1\temp\conf\es -cp lib/* org.elasticsearch.bootstrap.Elasticsearch

jvm 1    | 2019.08.12 10:14:55 INFO  app[][o.s.a.SchedulerImpl] Waiting for Elasticsearch to be up and running
jvm 1    | Java HotSpot(TM) 64-Bit Server VM warning: Option UseConcMarkSweepGC was deprecated in version 9.0 and will likely be removed in a future release.
jvm 1    | 2019.08.12 10:14:56 INFO  app[][o.e.p.PluginsService] no modules loaded
jvm 1    | 2019.08.12 10:14:56 INFO  app[][o.e.p.PluginsService] loaded plugin [org.elasticsearch.transport.Netty4Plugin]
jvm 1    | 2019.08.12 10:15:18 INFO  app[][o.s.a.SchedulerImpl] Process[es] is up
jvm 1    | 2019.08.12 10:15:18 INFO  app[][o.s.a.ProcessLauncherImpl] Launch process[[key='web', ipcIndex=2, logFilenamePrefix=web]] from [C:\Sonarcube\sonarqube-7.9.1]
: C:\Program Files\Java\jdk-12.0.2\bin\java -Djava.awt.headless=true -Dfile.encoding=UTF-8 -Djava.io.tmpdir=C:\Sonarcube\sonarqube-7.9.1\temp --add-opens=java.base/java
.util=ALL-UNNAMED --add-opens=java.base/java.lang=ALL-UNNAMED --add-opens=java.base/java.io=ALL-UNNAMED --add-opens=java.rmi/sun.rmi.transport=ALL-UNNAMED -Xmx512m -Xms
128m -XX:+HeapDumpOnOutOfMemoryError -Dhttp.nonProxyHosts=localhost|127.*|[::1] -cp ./lib/common/*;C:\Sonarcube\sonarqube-7.9.1\lib\jdbc\h2\h2-1.3.176.jar org.sonar.ser
ver.app.WebServer C:\Sonarcube\sonarqube-7.9.1\temp\sq-process7986085758163917265properties
jvm 1    | 2019.08.12 10:15:44 INFO  app[][o.s.a.SchedulerImpl] Process[web] is up
jvm 1    | 2019.08.12 10:15:44 INFO  app[][o.s.a.ProcessLauncherImpl] Launch process[[key='ce', ipcIndex=3, logFilenamePrefix=ce]] from [C:\Sonarcube\sonarqube-7.9.1]:
C:\Program Files\Java\jdk-12.0.2\bin\java -Djava.awt.headless=true -Dfile.encoding=UTF-8 -Djava.io.tmpdir=C:\Sonarcube\sonarqube-7.9.1\temp --add-opens=java.base/java.u
til=ALL-UNNAMED -Xmx512m -Xms128m -XX:+HeapDumpOnOutOfMemoryError -Dhttp.nonProxyHosts=localhost|127.*|[::1] -cp ./lib/common/*;C:\Sonarcube\sonarqube-7.9.1\lib\jdbc\h2
\h2-1.3.176.jar org.sonar.ce.app.CeServer C:\Sonarcube\sonarqube-7.9.1\temp\sq-process4054338655640235636properties
jvm 1    | 2019.08.12 10:15:51 INFO  app[][o.s.a.SchedulerImpl] Process[ce] is up
jvm 1    | 2019.08.12 10:15:51 INFO  app[][o.s.a.SchedulerImpl] SonarQube is up
```

To install Sonarqube for software testing service then you have to execute the bat file called Installntservice and then execute the other bat file called Startntserver in that order to start the service.

There is a bat file called Stopntservice, you can click on this file when you want to stop the service and then there is another file called UninstallNTService.bat to uninstall the service in your server.

Step V:

When the sonarqube service is up, then you should be able to access the sonarqube dashboard in your browser by accessing localhost:9000

There will be a pop-up window asking for username and password to login.

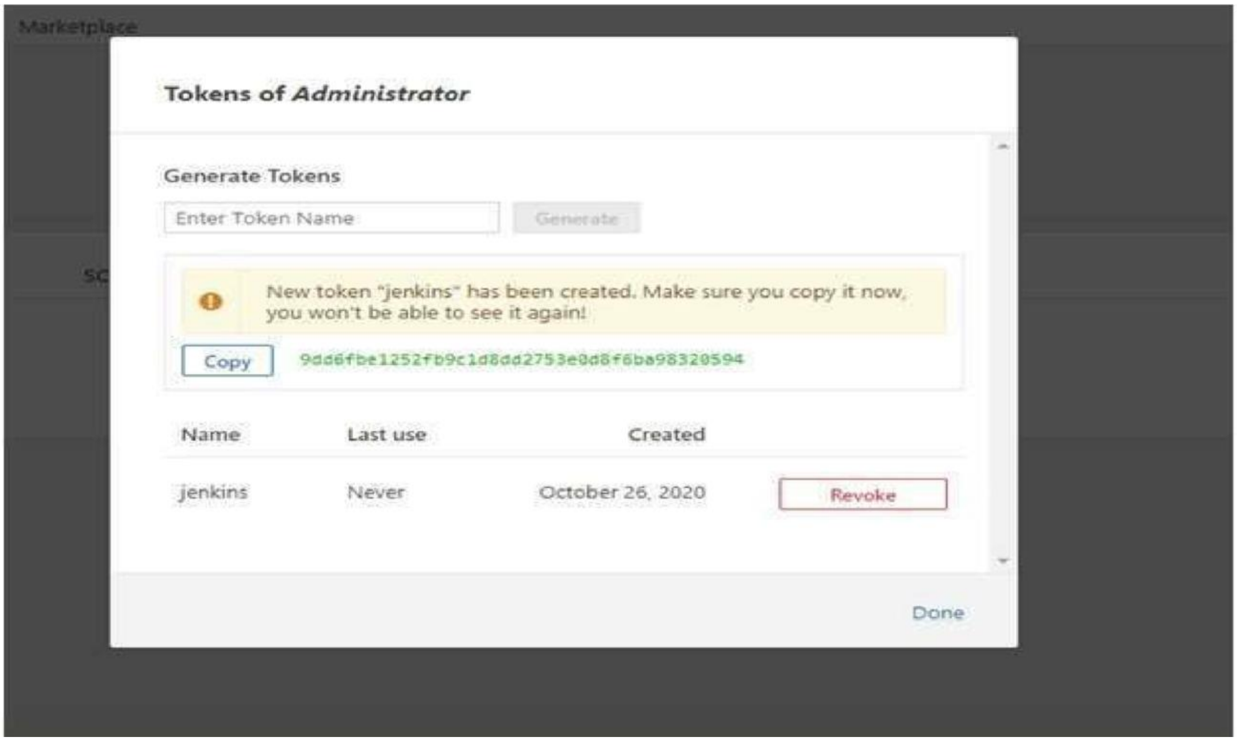
Put username- admin, password – admin

Now, select the administration section where there are options available to configure access and make changes in user responsibilities.

Step VI:

To Integrate the Jenkins DevOps environment with sonarqube installation is to generate an access token.

Now generate token with an appropriate name, which is under Administration/Security/Users/Tokens This generated token will be later used in Jenkins for Sonar authentication.



Tokens of *Administrator*

Generate Tokens

New token "jenkins" has been created. Make sure you copy it now, you won't be able to see it again!

```
9dd6fbe1252fb9c1d8dd2753e0d8f6ba98320594
```

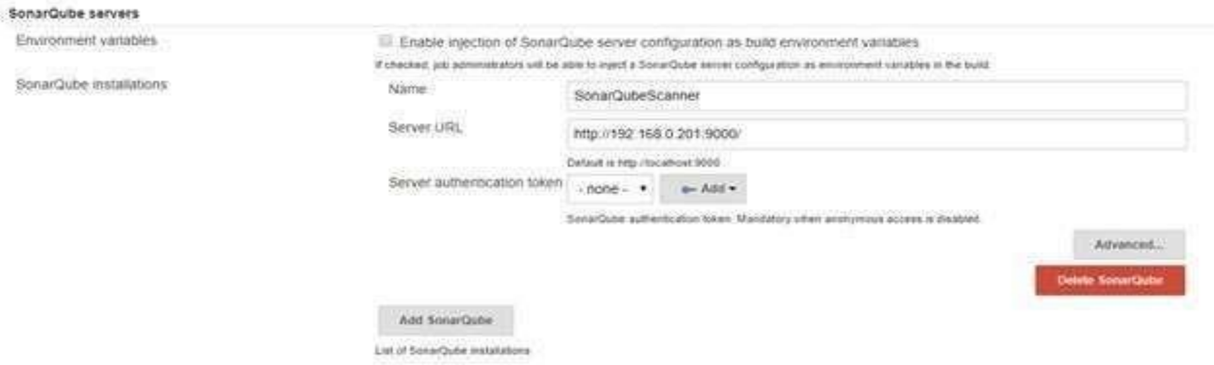
Name	Last use	Created	
jenkins	Never	October 26, 2020	Revoke

Step VII:

Then we have to configure the Jenkins with Sonarqube installations by using the generated access tokens in Step VI.

Put the server URL as http://localhost:9000 under then manage Jenkins / go to sonarqube servers section / add sonarqube.

For special cases, where if the server is installed on the same machine, or if the server is installed separately or in third scenario if the entire port is different for the server, then in all the three cases, the user needs to put the accurate server URL/ click on add option to create a secret text link. Then add the generated token as a secret key and save the whole operation.



SonarQube servers

Environment variables: □ Enable injection of SonarQube server configuration as build environment variables

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

SonarQube installations

Name: SonarQubeScanner

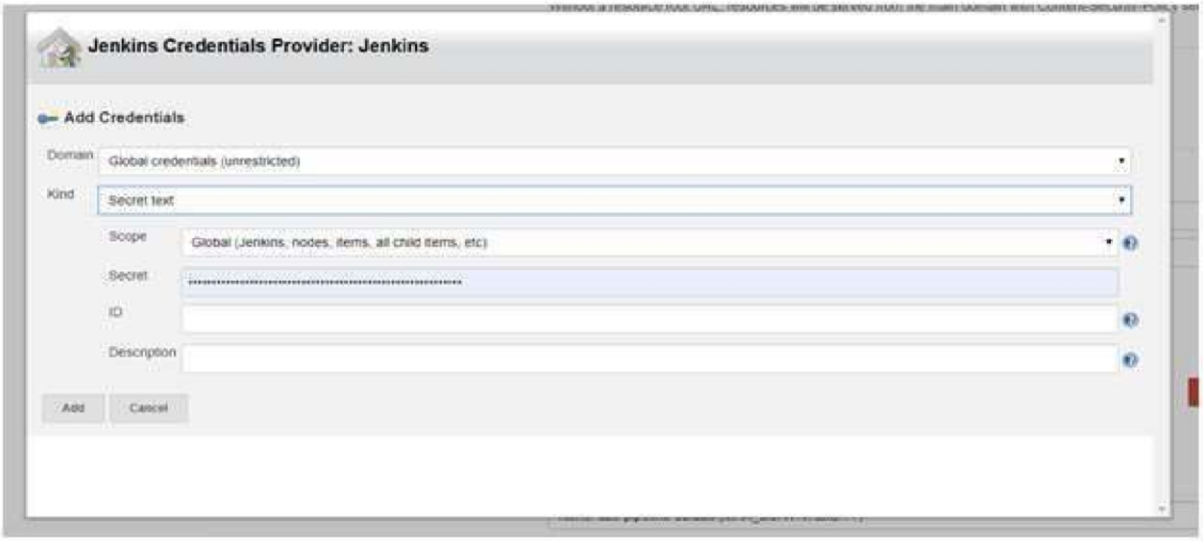
Server URL: http://192.168.0.201:9000/

Default is http://localhost:9000

Server authentication token: - none -

SonarQube authentication token. Mandatory when anonymous access is disabled.

List of SonarQube installations



Jenkins Credentials Provider: Jenkins

Add Credentials

Domain: Global credentials (unrestricted)

Kind: Secret text

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

Secret:

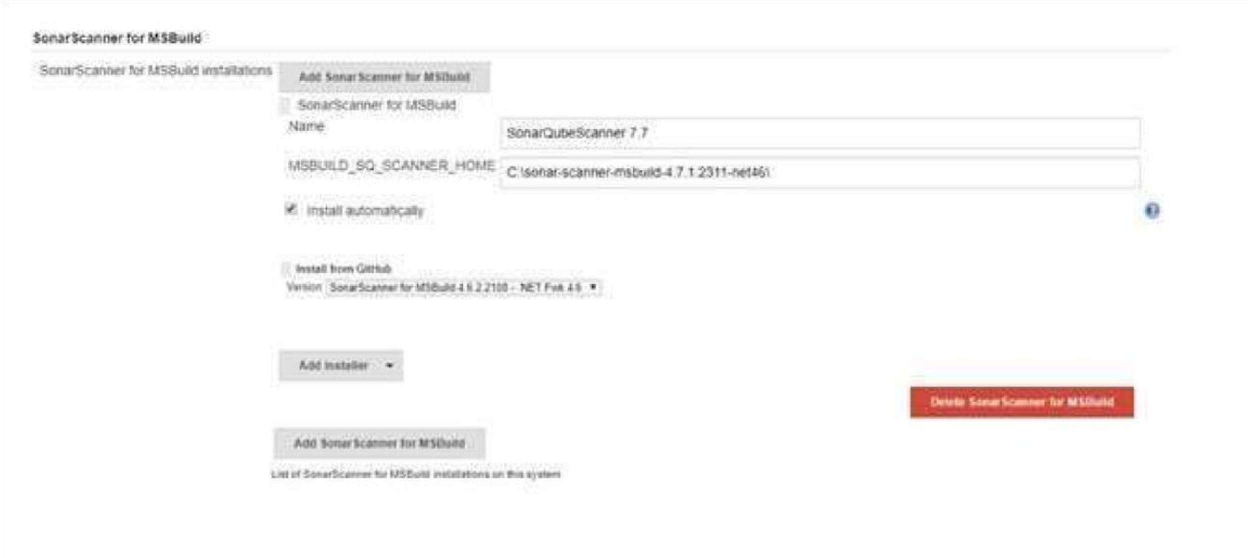
ID:

Description:

Step VIII:
After successfully completing integration proceIntegration.ss we need to install the sonar scanner which goes through the code and analysis.

Click on download now and install then click on the after restart link which is available under the manage Jenkins/manage plugins/go to the available section/ search for sonarqube scanner and install. Now go to Global Tool configuration under Manage Jenkins, then follow the below steps,

Finally, click on the Global Tool configuration under section Manage Jenkins, then keep following the below-mentioned steps. SonarScanner by MSBuild must be installed for all .NET framework related projects. Set name for the scanner and select the matching SonarScanner for MSBuild version.



SonarScanner for MSBuild

SonarScanner for MSBuild installations

□ SonarScanner for MSBuild

Name: SonarQubeScanner 7.7

MSBUILD_SQ_SCANNER_HOME: C:\sonar-scanner-msbuild-4.7.1.2311-net46\

☑ Install automatically

□ Install from GitHub

Version: SonarScanner for MSBuild 4.6.2.2108 - .NET Fwk 4.6

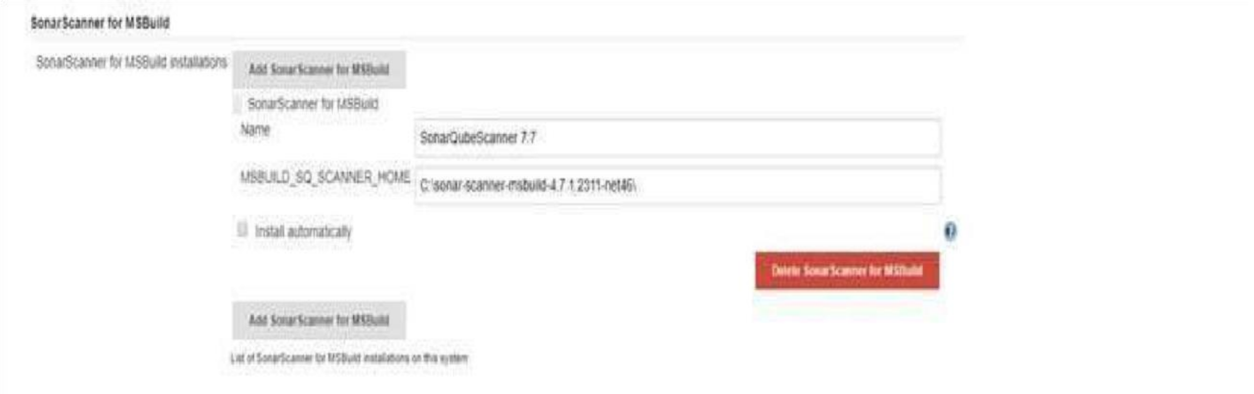
List of SonarScanner for MSBuild installations on this system

Add SonarScanner for MSBuild
Add Installer
Delete SonarScanner for MSBuild
Add SonarScanner for MSBuild

Steps to follow in .NET framework based projects

1. We have to first configure the sonarqube scanner.

2. Now go to the global to configuration option in manage Jenkins, in the sonarscanner for MSBuild installation -> give a name for the scanner -> and select the correct MSBuild version of your project.



SonarScanner for MSBuild

SonarScanner for MSBuild installations

□ SonarScanner for MSBuild

Name: SonarQubeScanner 7.7

MSBUILD_SQ_SCANNER_HOME: C:\sonar-scanner-msbuild-4.7.1.2311-net46\

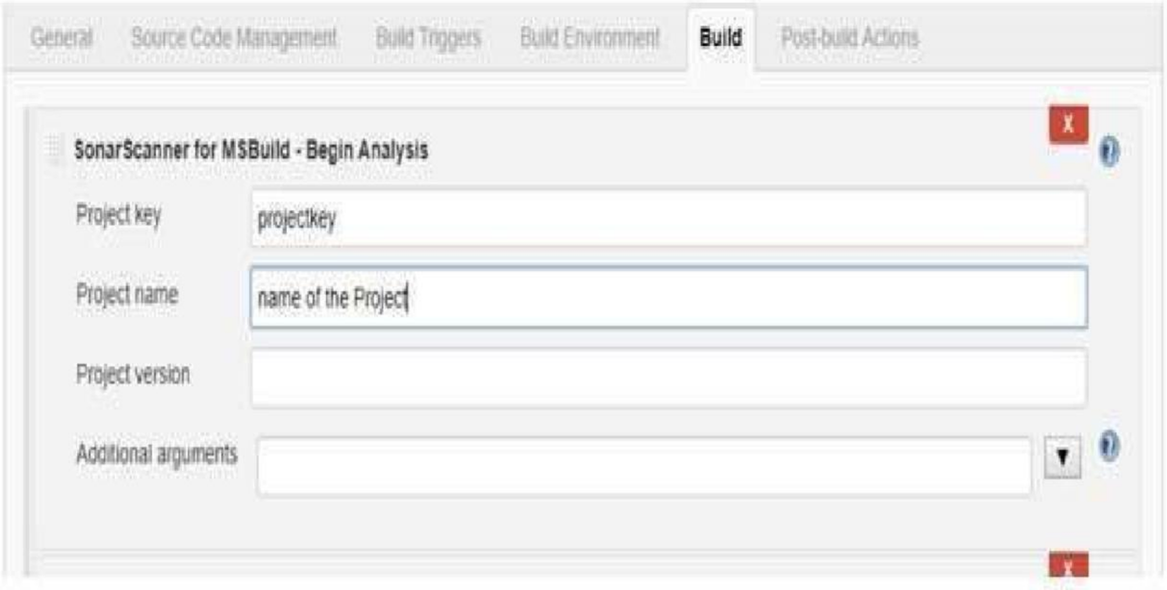
□ Install automatically

List of SonarScanner for MSBuild installations on this system

Add SonarScanner for MSBuild
Delete SonarScanner for MSBuild
Add SonarScanner for MSBuild

3. You have to select the right sonarscanner for the MSBuild version that matches your project. Click apply & save. Click on the relevant projects and then into the build pipeline for further configuration

4. Select the option to add a build step for sonarscanner for MSBuild begin analysis and put it at the starting position of the build pipeline as shown in the below screenshot.



SonarScanner for MSBuild - Begin Analysis

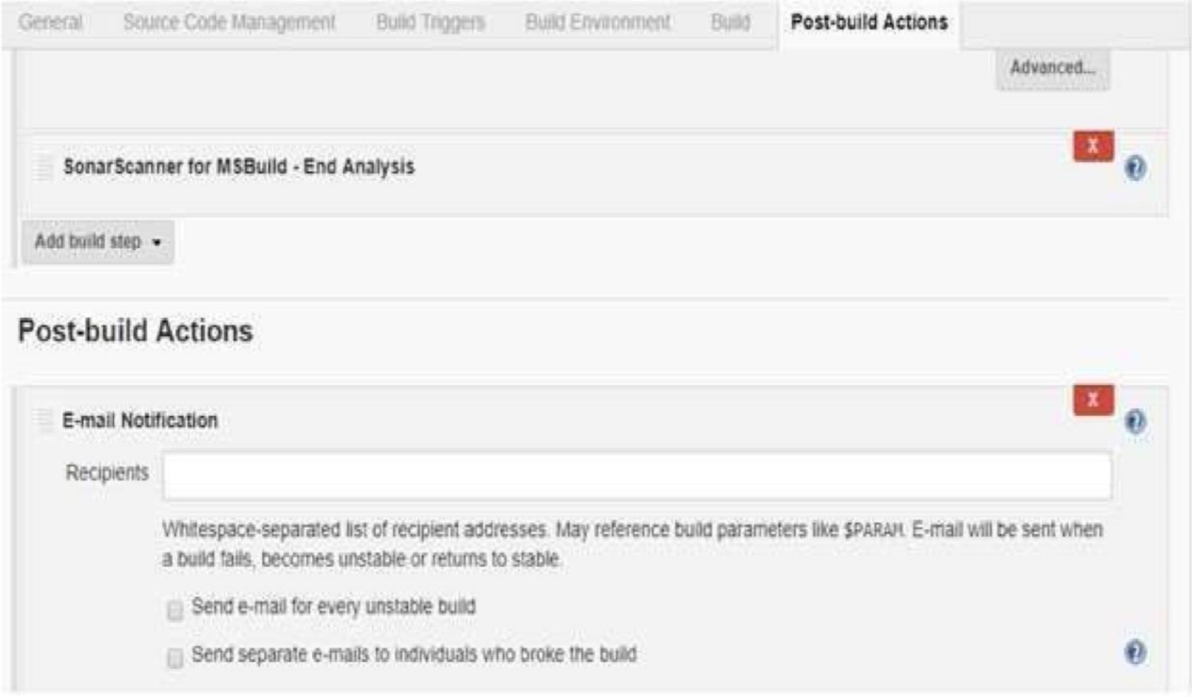
Project key: projectkey

Project name: name of the Project

Project version:

Additional arguments:

5. Put the sonarscanner for MSBuild and then end analysis as shown in the below screenshot



SonarScanner for MSBuild - End Analysis

Post-build Actions

E-mail Notification

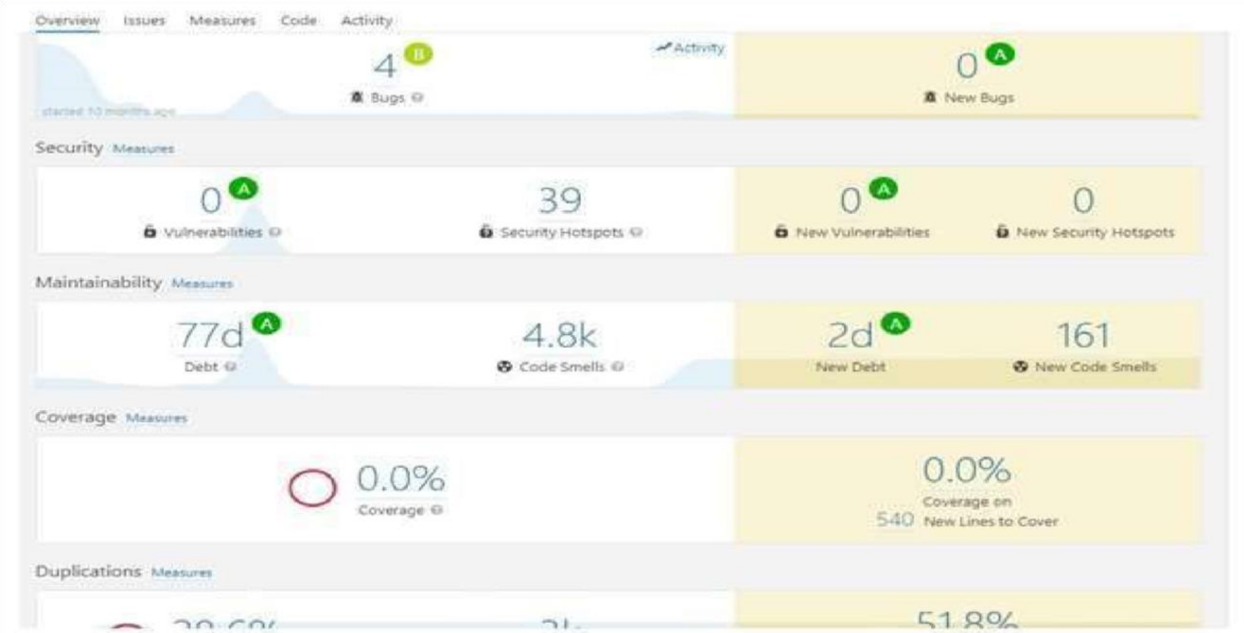
Recipients:

Whitespace-separated list of recipient addresses. May reference build parameters like $PARAM. E-mail will be sent when a build fails, becomes unstable or returns to stable.

□ Send e-mail for every unstable build

□ Send separate e-mails to individuals who broke the build

5. Now run the build and check whether it gets successfully built or not. Lastly, do a final visit on SonarQube portal to check if the selected project is visible on the dashboard or not. When the build & integration gets successful, you’ll see the dashboard of the analysis overview as shown :



Conclusion -

With the help of Jenkins, it simply means to integrate coding done by multiple developers and to achieve frequent builds. As soon as developer commits the code to GitHub, Jenkins will fetch/pull the code from repository and will perform static code analysis with help of Sonar Scanner and send analysis report to SonarQube Server then it will automatically build the project code. This all above process helps in understanding software development in better way.