Snehal A. Patil D15A 39

EXPERIMENT NO:7

Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

Theory:

Static Application Security Testing (SAST) is a method of debugging by examining source code before a program is run. It involves analyzing the application's source code, bytecode, or binary code to identify vulnerabilities and security flaws. SAST tools scan code for common
security vulnerabilities such as SQL injection, cross-site scripting (XSS), and buffer overflows, among others.

Problems SAST Solves:

1. **Early Detection of Vulnerabilities**: SAST enables developers to find security flaws early in the development lifecycle, reducing the cost and effort required to fix them later.
2. **Compliance with Security Standards**: It helps organizations comply with various security regulations and standards, such as PCI DSS, OWASP Top Ten, and ISO 27001, by identifying security weaknesses that need to be addressed.
3. **Integration into CI/CD Pipelines**: SAST tools can be integrated into Continuous Integration/Continuous Deployment (CI/CD) pipelines, allowing for automated security checks during the development process.
4. **Comprehensive Coverage**: It scans all code paths and identifies vulnerabilities that may not be detected during dynamic testing (which tests the application while it runs).
5. **Reduction of Technical Debt**: By catching vulnerabilities early, SAST helps prevent the accumulation of technical debt related to security issues, making the codebase more maintainable.
6. **Improved Code Quality**: Besides security, SAST tools often identify coding best practices and help improve overall code quality.
7. **Enhanced Collaboration**: By providing clear reports and insights, SAST tools foster better communication between development and security teams.
8. **Risk Mitigation**: It helps organizations manage risks associated with software vulnerabilities, thereby protecting against data breaches and cyberattacks.

Prerequisites:

● Jenkins installed

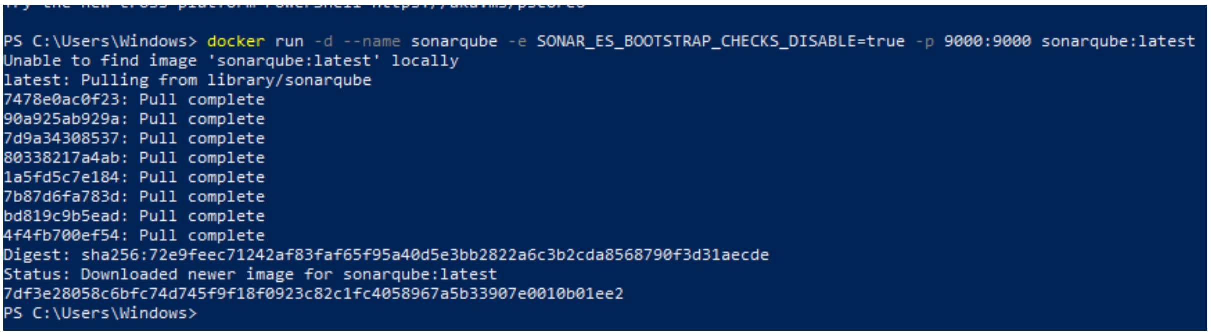
● Docker Installed (for SonarQube)

Steps to integrate Jenkins with SonarQube:

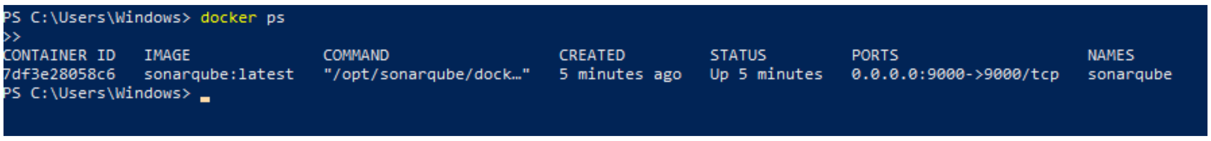
1.Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.

2. Run SonarQube in a Docker container using this command -

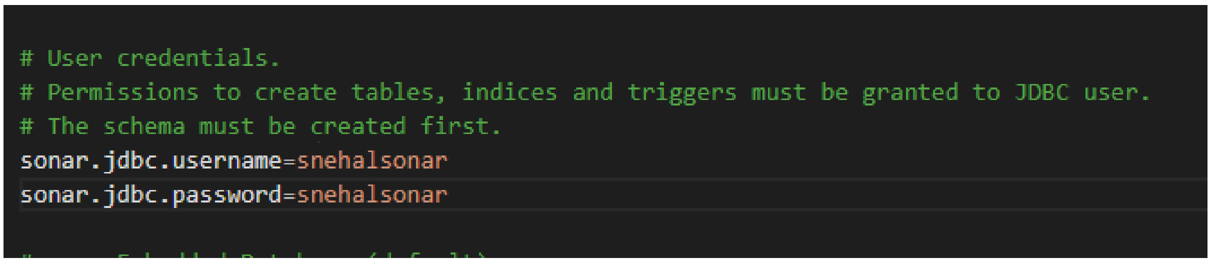
Command: docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest



```
PS C:\Users\Windows> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
7df3e28058c6bfc74d745f9f18f0923c82c1fc4058967a5b33907e0010b01ee2
PS C:\Users\Windows>
```



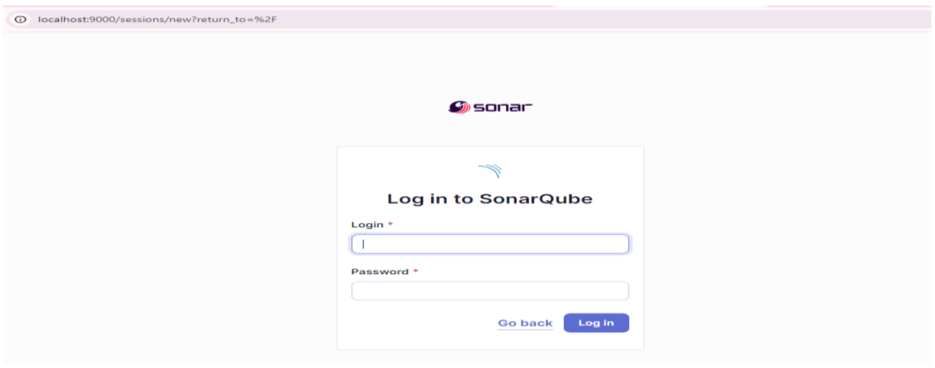
```
PS C:\Users\Windows> docker ps
>>
CONTAINER ID   IMAGE              COMMAND                  CREATED         STATUS         PORTS                    NAMES
7df3e28058c6   sonarqube:latest   "/opt/sonarqube/dock…"   5 minutes ago   Up 5 minutes   0.0.0.0:9000->9000/tcp   sonarqube
PS C:\Users\Windows> _
```



```
# User credentials.
# Permissions to create tables, indices and triggers must be granted to JDBC user.
# The schema must be created first.
sonar.jdbc.username=snehalsonar
sonar.jdbc.password=snehalsonar
```

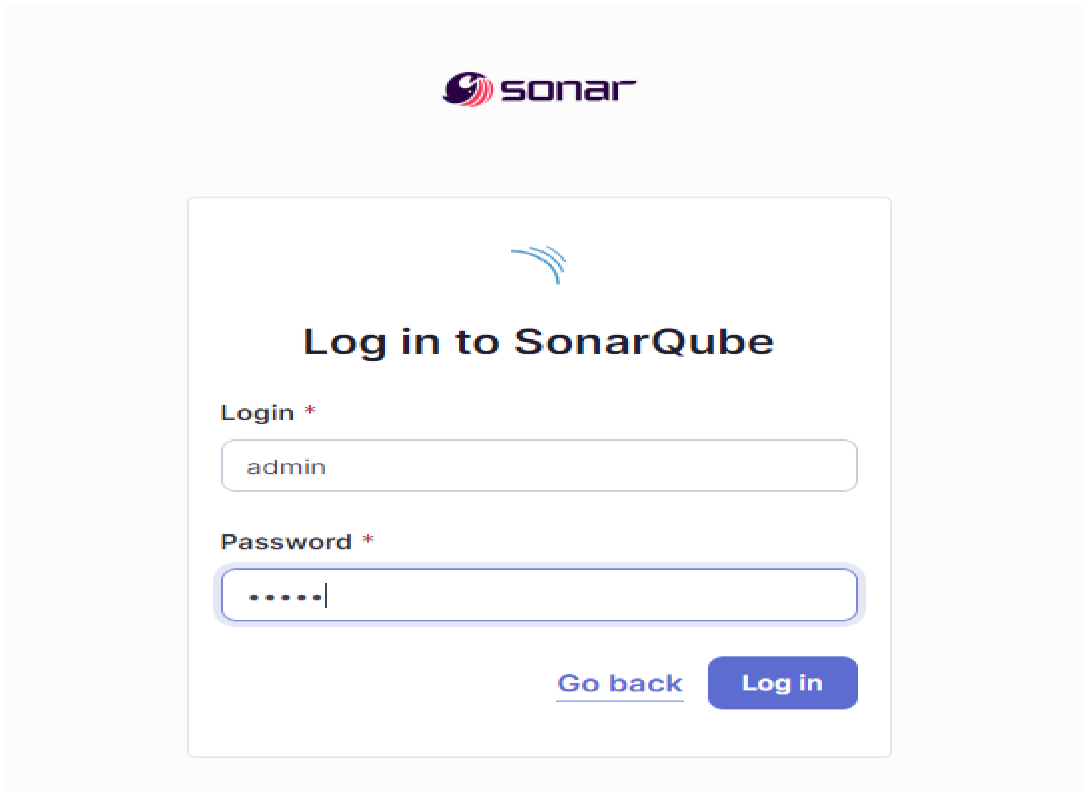
3. Once the container is up and running, you can check the status of SonarQube at

localhost port 9000.



localhost:9000/sessions/new?return_to=%2F

4. Login to SonarQube using username admin and password admin.



Log in to SonarQube

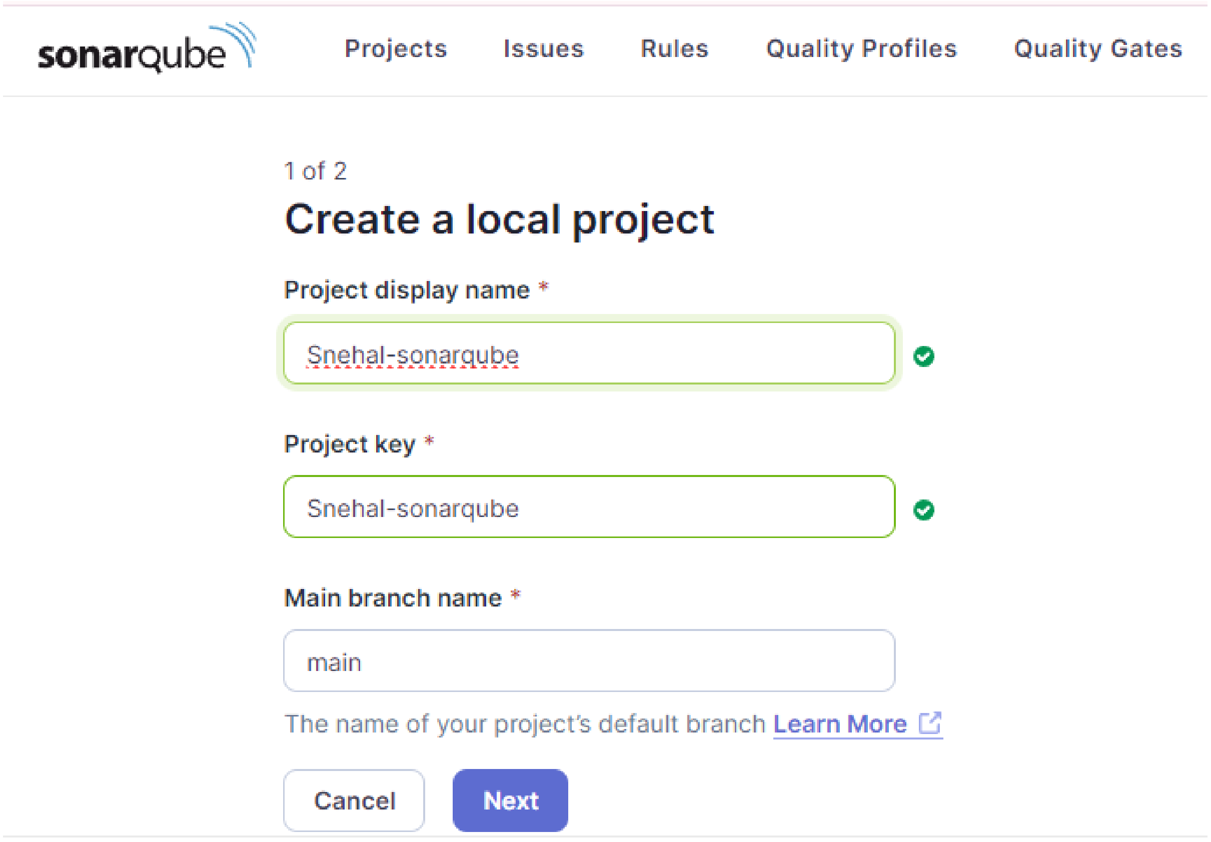
Login *

admin

Password *

•••••

5.Create a manual project in SonarQube with the name sonarqube



1 of 2

Create a local project

Project display name *

Snehal-sonarqube

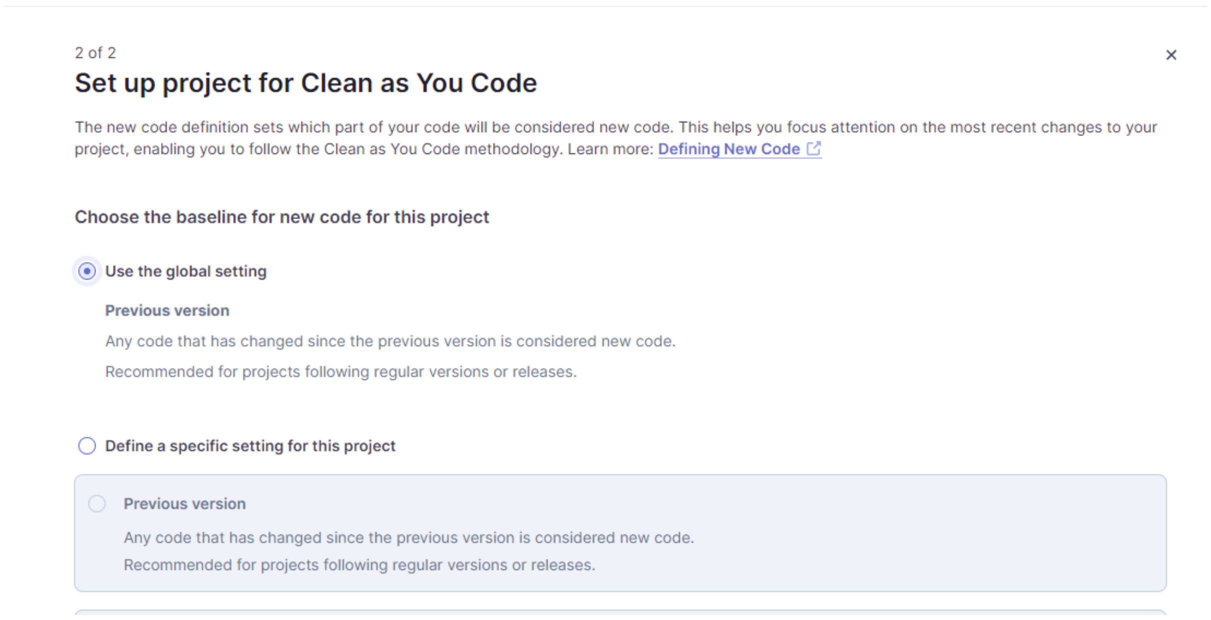
Project key *

Snehal-sonarqube

Main branch name *

main

The name of your project's default branch Learn More



2 of 2

Set up project for Clean as You Code

The new code definition sets which part of your code will be considered new code. This helps you focus attention on the most recent changes to your project, enabling you to follow the Clean as You Code methodology. Learn more: Defining New Code

Choose the baseline for new code for this project

[✓ Use the global setting]

Previous version

Any code that has changed since the previous version is considered new code.

Recommended for projects following regular versions or releases.

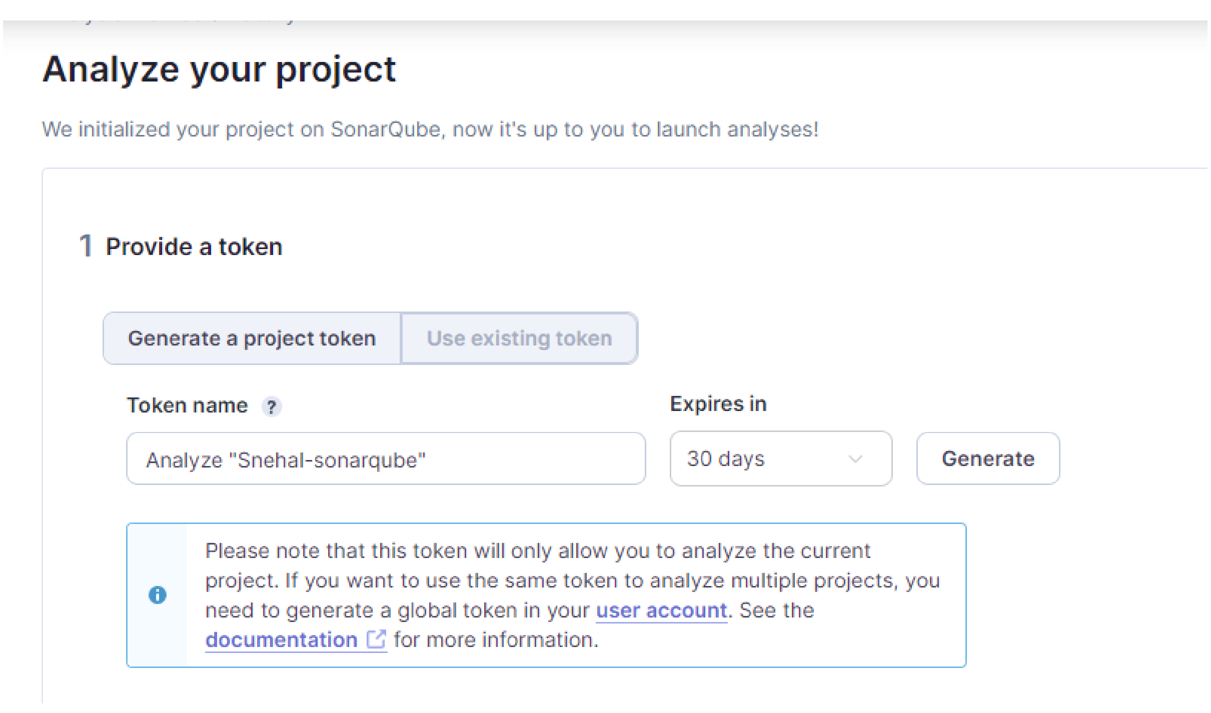
Define a specific setting for this project

Previous version

Any code that has changed since the previous version is considered new code.

Recommended for projects following regular versions or releases.

6. Generate a token



Analyze your project

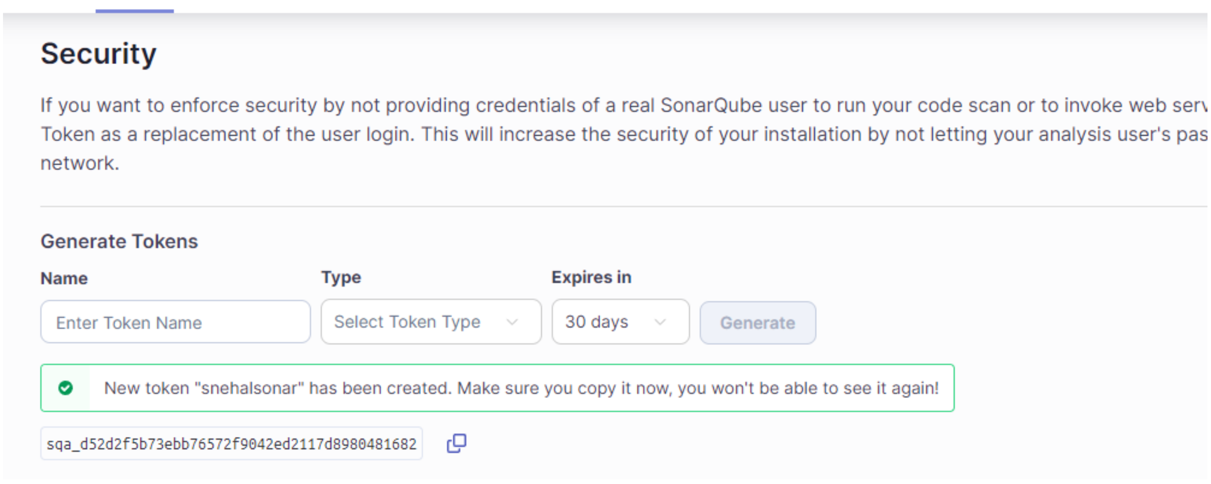
We initialized your project on SonarQube, now it's up to you to launch analyses!

1 Provide a token

Token name: Analyze "Snehal-sonarqube"

Expires in: 30 days

Please note that this token will only allow you to analyze the current project. If you want to use the same token to analyze multiple projects, you need to generate a global token in your user account. See the documentation for more information.



Security

If you want to enforce security by not providing credentials of a real SonarQube user to run your code scan or to invoke web serv
Token as a replacement of the user login. This will increase the security of your installation by not letting your analysis user's pas
network.

Generate Tokens

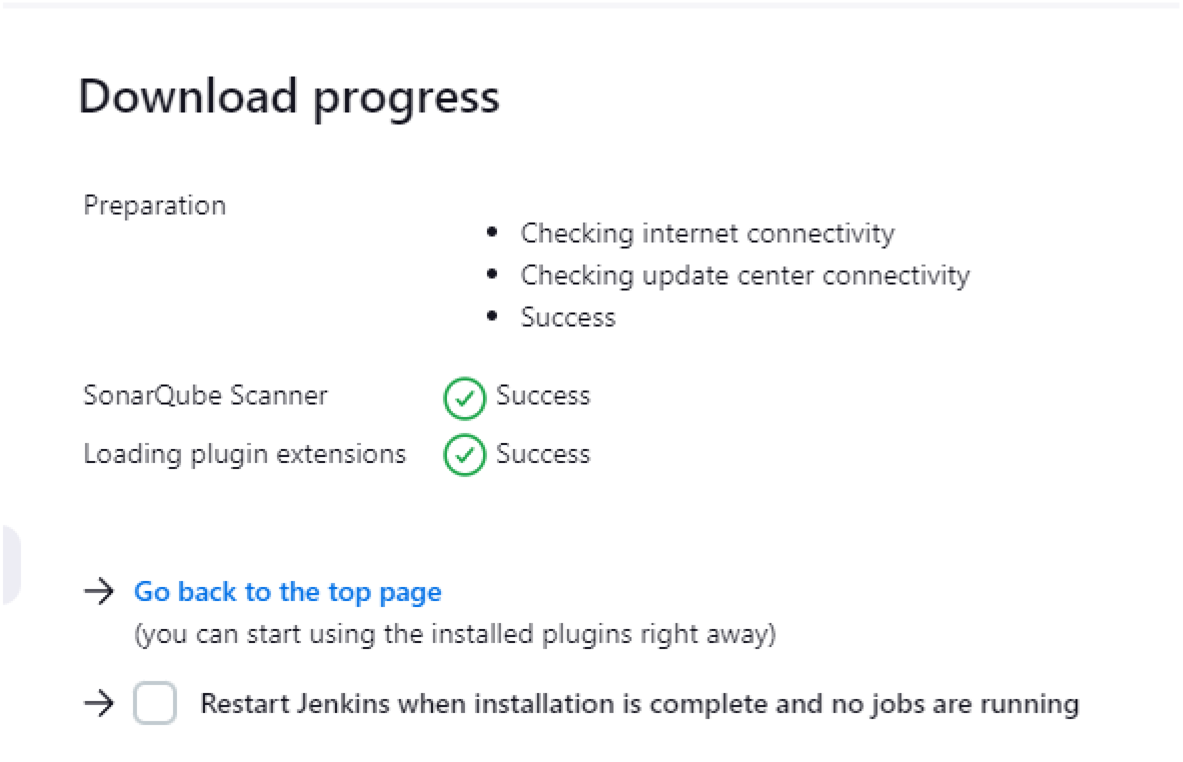
Name	Type	Expires in	
Enter Token Name	Select Token Type	30 days	Generate

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

```
sqa_d52d2f5b73ebb76572f9042ed2117d8980481682
```

7. Setup the project and come back to Jenkins Dashboard.

Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.



Download progress

Preparation
- Checking internet connectivity
- Checking update center connectivity
- Success

SonarQube Scanner ✓ Success

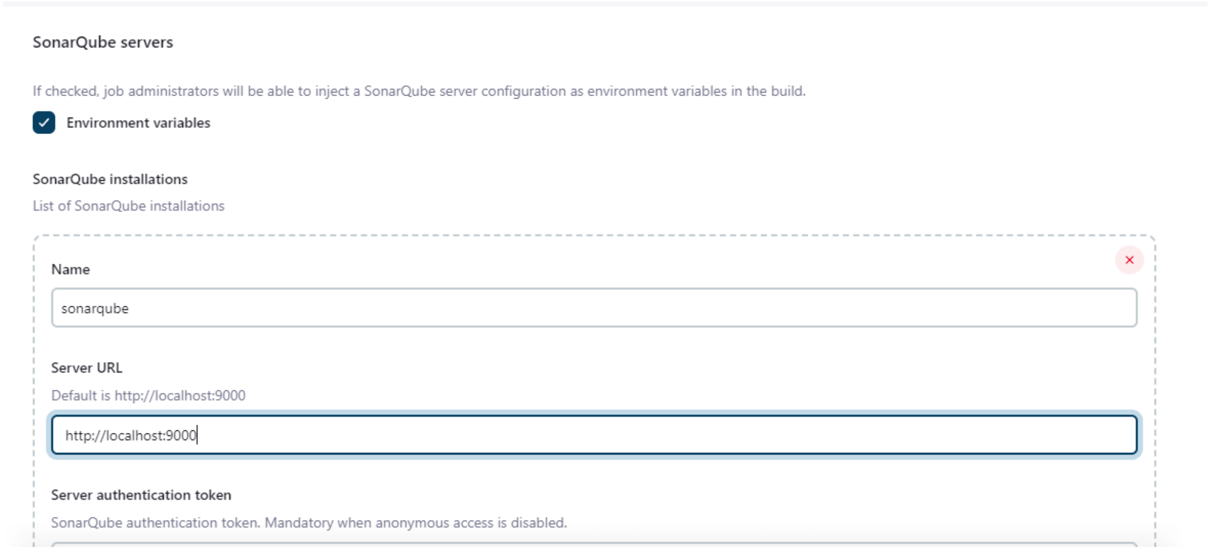
Loading plugin extensions ✓ Success

→ Go back to the top page
(you can start using the installed plugins right away)

→ ☐ Restart Jenkins when installation is complete and no jobs are running

8. Under Jenkins 'Configure System', look for SonarQube Servers and enter the details.

Enter the Server Authentication token if needed.



SonarQube servers

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

☑ Environment variables

SonarQube installations

List of SonarQube installations

Name

sonarqube

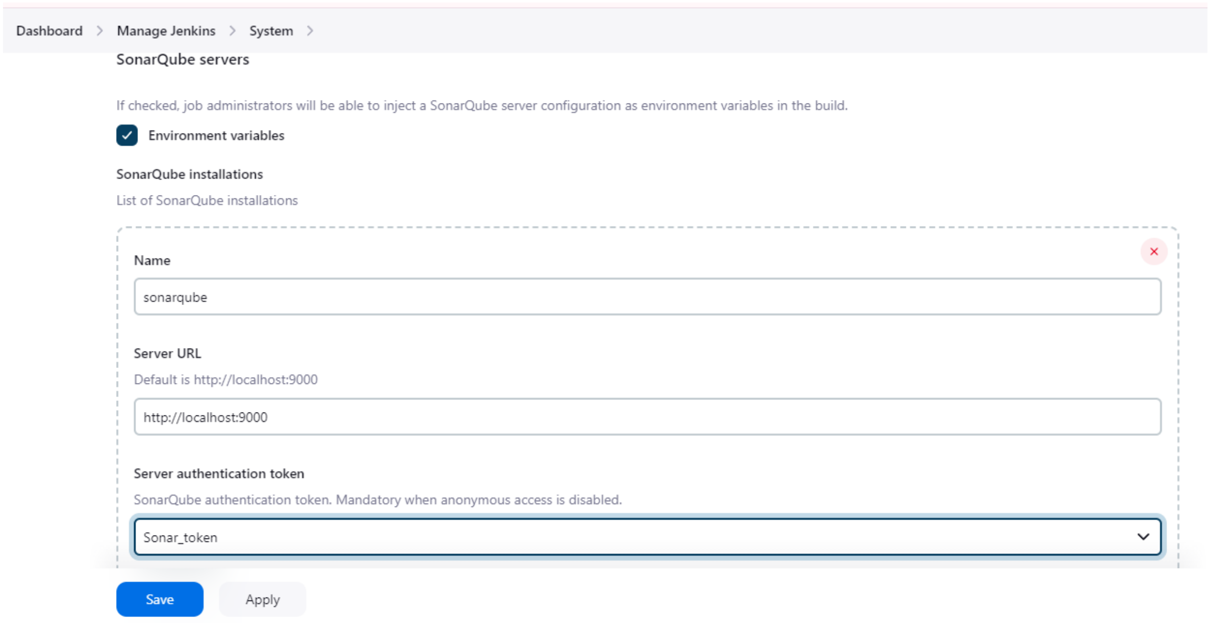
Server URL

Default is http://localhost:9000

http://localhost:9000

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.


Dashboard > Manage Jenkins > System >

SonarQube servers

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

☑ Environment variables

SonarQube installations

List of SonarQube installations

Name: sonarqube

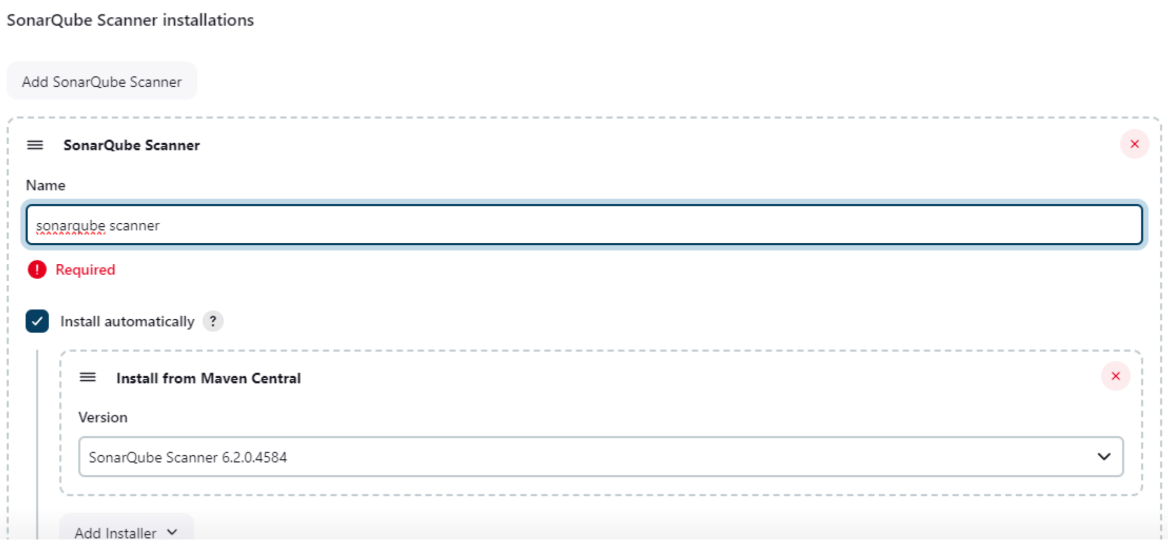
Server URL: Default is http://localhost:9000

http://localhost:9000

Server authentication token: SonarQube authentication token. Mandatory when anonymous access is disabled.

Sonar_token

9.Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.


SonarQube Scanner installations

Add SonarQube Scanner

SonarQube Scanner

Name: sonarqube scanner

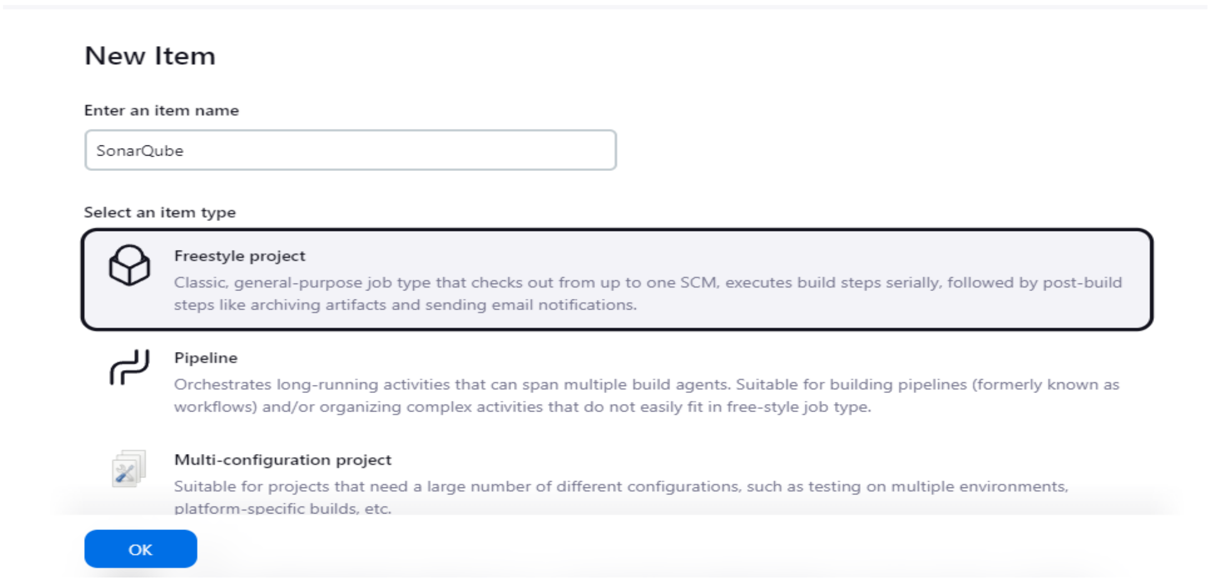
Required

☑ Install automatically

Install from Maven Central

Version: SonarQube Scanner 6.2.0.4584

10.After the configuration, create a New Item in Jenkins, choose a freestyle project.


New Item

Enter an item name: SonarQube

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

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.

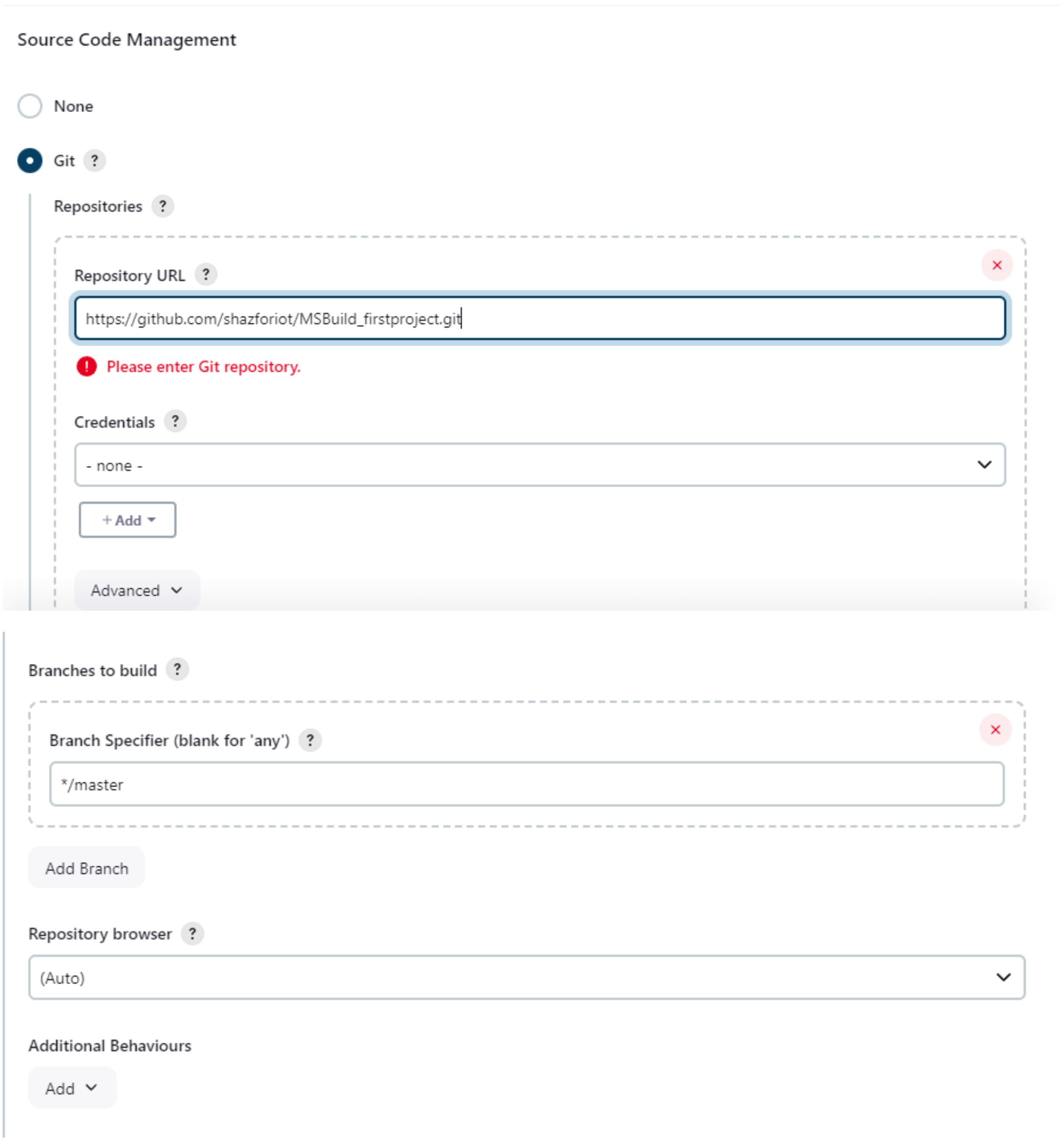
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

11.Choose this GitHub repository in Source Code

Management.

https://github.com/shazforiot/MSBuild_firstproject.git

It is a sample hello-world project with no vulnerabilities and issues, just to test



Source Code Management

None

[✓ Git]

Repositories

Repository URL

https://github.com/shazforiot/MSBuild_firstproject.git

Please enter Git repository.

Credentials

- none -

Branches to build

Branch Specifier (blank for 'any')

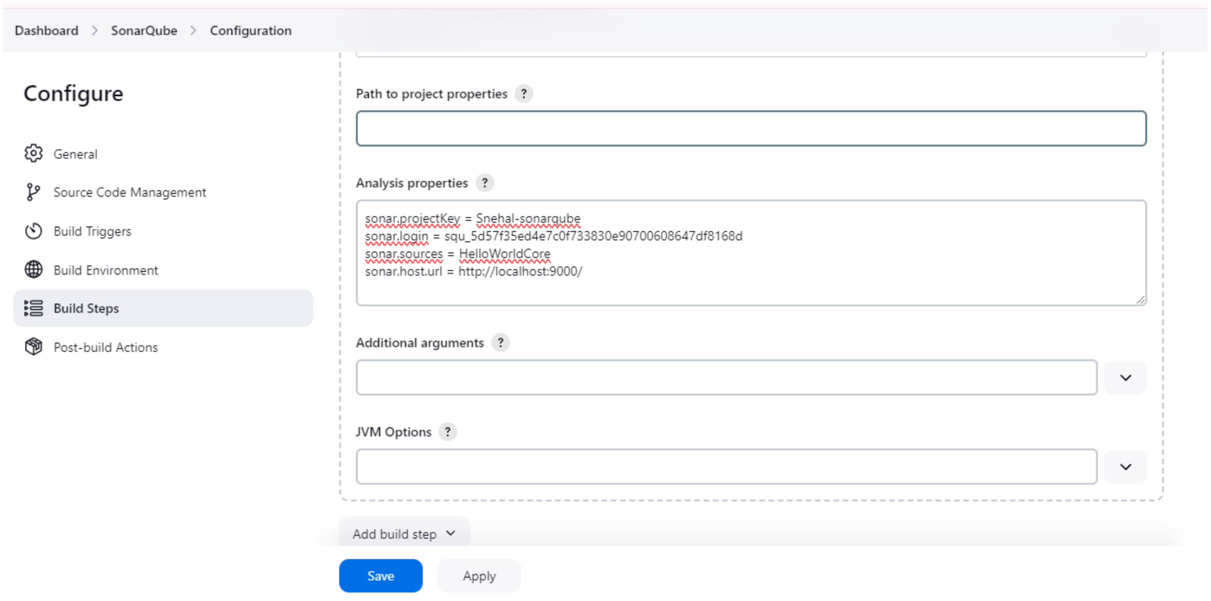
*/master

Repository browser

(Auto)

Additional Behaviours

12. Under Build-> Execute SonarQube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.



Path to project properties

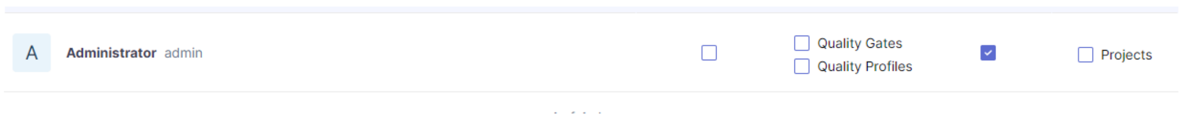
Analysis properties

```
sonar.projectKey = Snehal-sonarqube
sonar.login = squ_5d57f35ed4e7c0f733830e90700608647df8168d
sonar.sources = HelloWorldCore
sonar.host.url = http://localhost:9000/
```

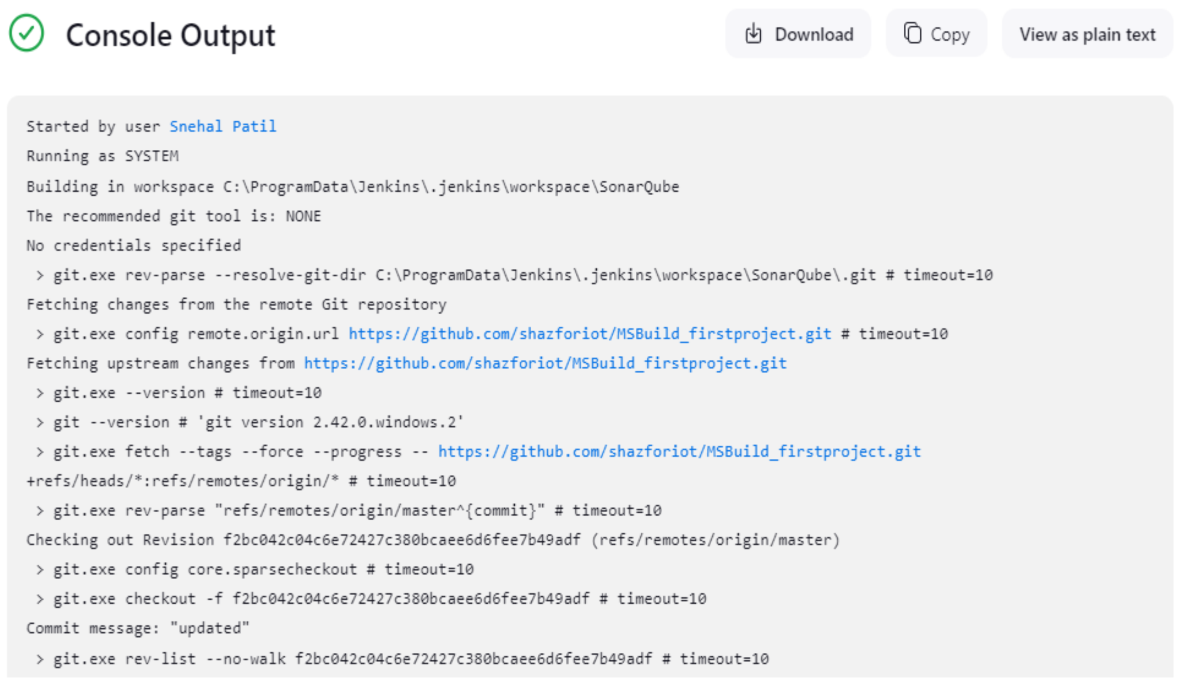
Additional arguments

JVM Options

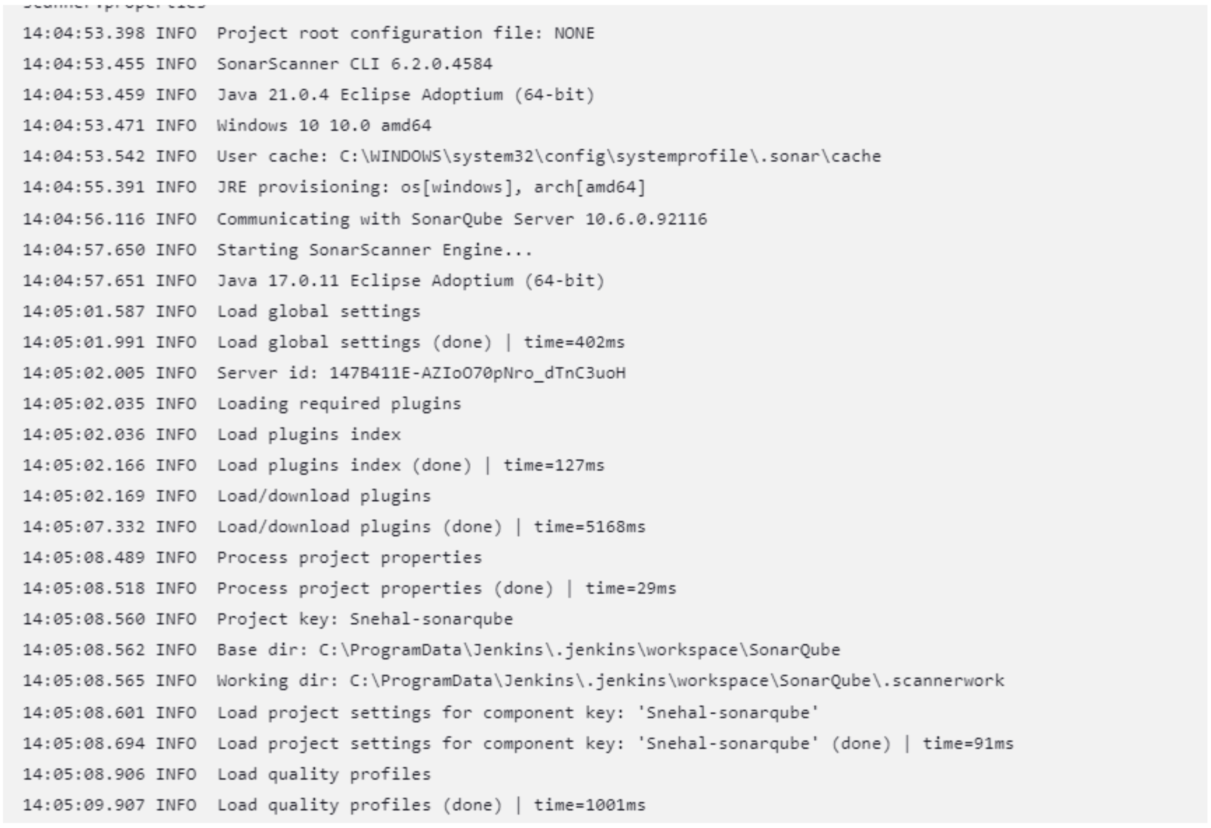
13.Go to http://localhost:9000/<user_name>/permissions and allow Execute Permissions to the Admin user.



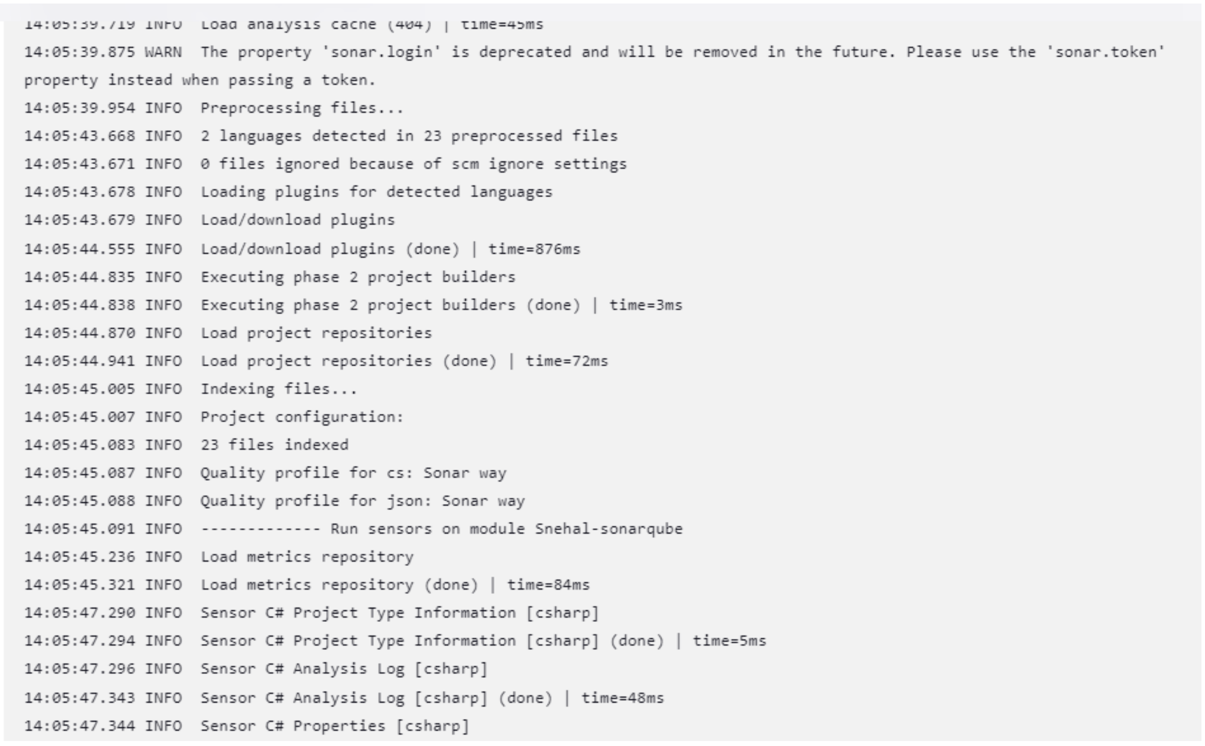
Check the console output.



```
Started by user Snehal Patil
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\SonarQube
The recommended git tool is: NONE
No credentials specified
 > git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\SonarQube\.git # timeout=10
Fetching changes from the remote Git repository
 > git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject.git # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject.git
 > git.exe --version # timeout=10
 > git --version # 'git version 2.42.0.windows.2'
 > git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
 > git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcaee6d6fee7b49adf (refs/remotes/origin/master)
 > git.exe config core.sparsecheckout # timeout=10
 > git.exe checkout -f f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
Commit message: "updated"
 > git.exe rev-list --no-walk f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
```



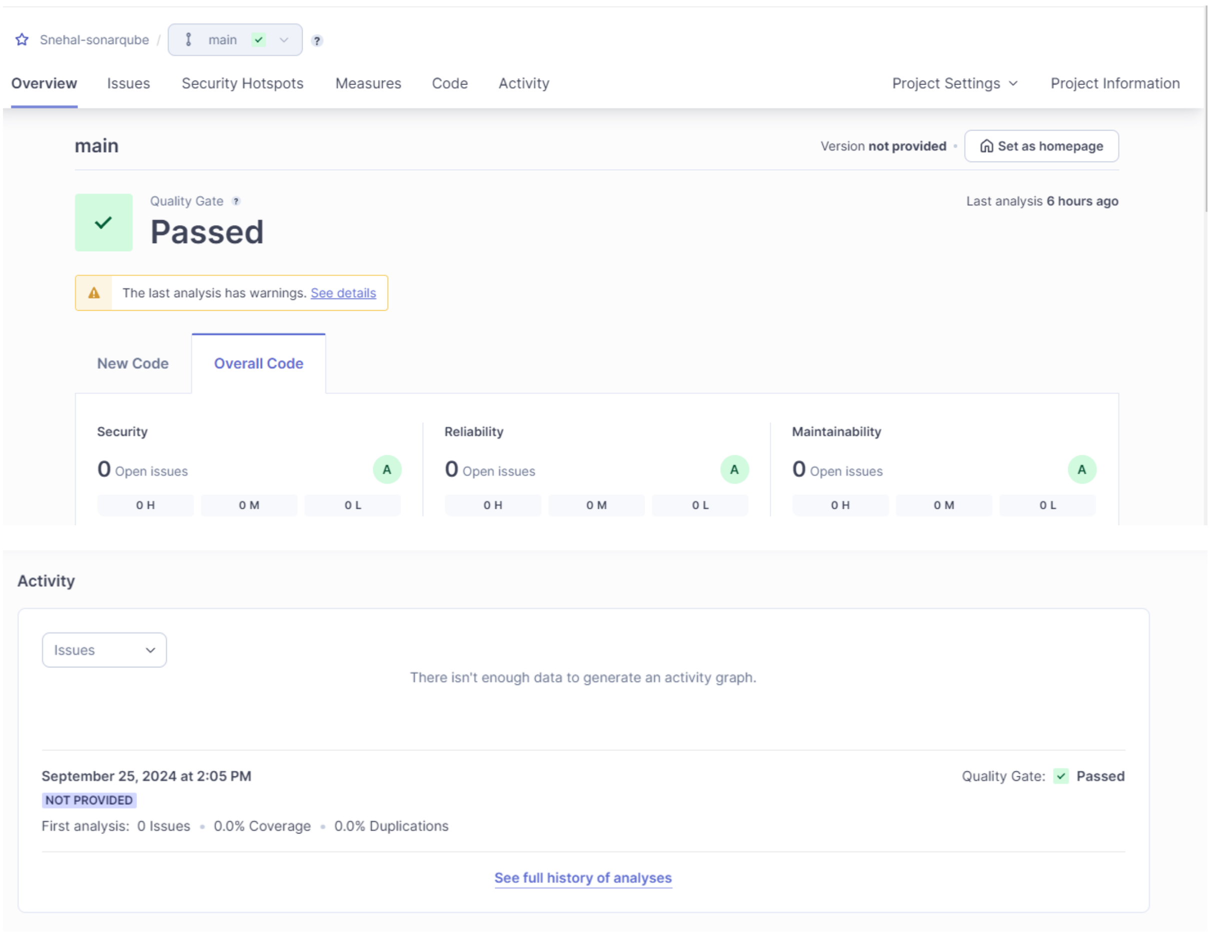
```
scanner.properties
14:04:53.398 INFO  Project root configuration file: NONE
14:04:53.455 INFO  SonarScanner CLI 6.2.0.4584
14:04:53.459 INFO  Java 21.0.4 Eclipse Adoptium (64-bit)
14:04:53.471 INFO  Windows 10 10.0 amd64
14:04:53.542 INFO  User cache: C:\WINDOWS\system32\config\systemprofile\.sonar\cache
14:04:55.391 INFO  JRE provisioning: os[windows], arch[amd64]
14:04:56.116 INFO  Communicating with SonarQube Server 10.6.0.92116
14:04:57.650 INFO  Starting SonarScanner Engine...
14:04:57.651 INFO  Java 17.0.11 Eclipse Adoptium (64-bit)
14:05:01.587 INFO  Load global settings
14:05:01.991 INFO  Load global settings (done) | time=402ms
14:05:02.005 INFO  Server id: 147B411E-AZIoO70pNro_dTnC3uoH
14:05:02.035 INFO  Loading required plugins
14:05:02.036 INFO  Load plugins index
14:05:02.166 INFO  Load plugins index (done) | time=127ms
14:05:02.169 INFO  Load/download plugins
14:05:07.332 INFO  Load/download plugins (done) | time=5168ms
14:05:08.489 INFO  Process project properties
14:05:08.518 INFO  Process project properties (done) | time=29ms
14:05:08.560 INFO  Project key: Snehal-sonarqube
14:05:08.562 INFO  Base dir: C:\ProgramData\Jenkins\.jenkins\workspace\SonarQube
14:05:08.565 INFO  Working dir: C:\ProgramData\Jenkins\.jenkins\workspace\SonarQube\.scannerwork
14:05:08.601 INFO  Load project settings for component key: 'Snehal-sonarqube'
14:05:08.694 INFO  Load project settings for component key: 'Snehal-sonarqube' (done) | time=91ms
14:05:08.906 INFO  Load quality profiles
14:05:09.907 INFO  Load quality profiles (done) | time=1001ms
```



```
14:05:39.719 INFO  Load analysis cache (404) | time=45ms
14:05:39.875 WARN  The property 'sonar.login' is deprecated and will be removed in the future. Please use the 'sonar.token'
property instead when passing a token.
14:05:39.954 INFO  Preprocessing files...
14:05:43.668 INFO  2 languages detected in 23 preprocessed files
14:05:43.671 INFO  0 files ignored because of scm ignore settings
14:05:43.678 INFO  Loading plugins for detected languages
14:05:43.679 INFO  Load/download plugins
14:05:44.555 INFO  Load/download plugins (done) | time=876ms
14:05:44.835 INFO  Executing phase 2 project builders
14:05:44.838 INFO  Executing phase 2 project builders (done) | time=3ms
14:05:44.870 INFO  Load project repositories
14:05:44.941 INFO  Load project repositories (done) | time=72ms
14:05:45.005 INFO  Indexing files...
14:05:45.007 INFO  Project configuration:
14:05:45.083 INFO  23 files indexed
14:05:45.087 INFO  Quality profile for cs: Sonar way
14:05:45.088 INFO  Quality profile for json: Sonar way
14:05:45.091 INFO  ------------- Run sensors on module Snehal-sonarqube
14:05:45.236 INFO  Load metrics repository
14:05:45.321 INFO  Load metrics repository (done) | time=84ms
14:05:47.290 INFO  Sensor C# Project Type Information [csharp]
14:05:47.294 INFO  Sensor C# Project Type Information [csharp] (done) | time=5ms
14:05:47.296 INFO  Sensor C# Analysis Log [csharp]
14:05:47.343 INFO  Sensor C# Analysis Log [csharp] (done) | time=48ms
14:05:47.344 INFO  Sensor C# Properties [csharp]
```

```
14:05:51.146 WARN  Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider
setting 'sonar.projectBaseDir' property.
14:05:51.147 INFO  Sensor C# File Caching Sensor [csharp] (done) | time=1ms
14:05:51.147 INFO  Sensor Zero Coverage Sensor
14:05:51.162 INFO  Sensor Zero Coverage Sensor (done) | time=18ms
14:05:51.170 INFO  SCM Publisher SCM provider for this project is: git
14:05:51.174 INFO  SCM Publisher 2 source files to be analyzed
14:05:52.968 INFO  SCM Publisher 2/2 source files have been analyzed (done) | time=1792ms
14:05:52.973 INFO  CPD Executor Calculating CPD for 0 files
14:05:52.984 INFO  CPD Executor CPD calculation finished (done) | time=0ms
14:05:52.998 INFO  SCM revision ID 'f2bc042c04c6e72427c380bcaee6d6fee7b49adf'
14:05:53.651 INFO  Analysis report generated in 374ms, dir size=199.1 kB
14:05:53.845 INFO  Analysis report compressed in 123ms, zip size=20.5 kB
14:05:54.150 INFO  Analysis report uploaded in 299ms
14:05:54.156 INFO  ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=Snehal-sonarqube
14:05:54.160 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted
analysis report
14:05:54.162 INFO  More about the report processing at http://localhost:9000/api/ce/task?id=1b0e8114-e5a0-4256-ba7f-
e9eb666d5810
14:05:54.202 INFO  Analysis total time: 46.510 s
14:05:54.211 INFO  SonarScanner Engine completed successfully
14:05:54.357 INFO  EXECUTION SUCCESS
14:05:54.360 INFO  Total time: 1:00.973s
Finished: SUCCESS
```

14. Once the build is complete, check the project in SonarQube.


Snehal-sonarqube / main ✓ ?

Overview | Issues | Security Hotspots | Measures | Code | Activity | Project Settings | Project Information

main

Version **not provided** · Set as homepage

Quality Gate ?

Passed

Last analysis **6 hours ago**

The last analysis has warnings. See details

New Code | Overall Code

Security	Reliability	Maintainability
0 Open issues A	0 Open issues A	0 Open issues A
0 H · 0 M · 0 L	0 H · 0 M · 0 L	0 H · 0 M · 0 L

Activity

Issues

There isn't enough data to generate an activity graph.

September 25, 2024 at 2:05 PM — Quality Gate: ✓ Passed

NOT PROVIDED

First analysis: 0 Issues · 0.0% Coverage · 0.0% Duplications

See full history of analyses