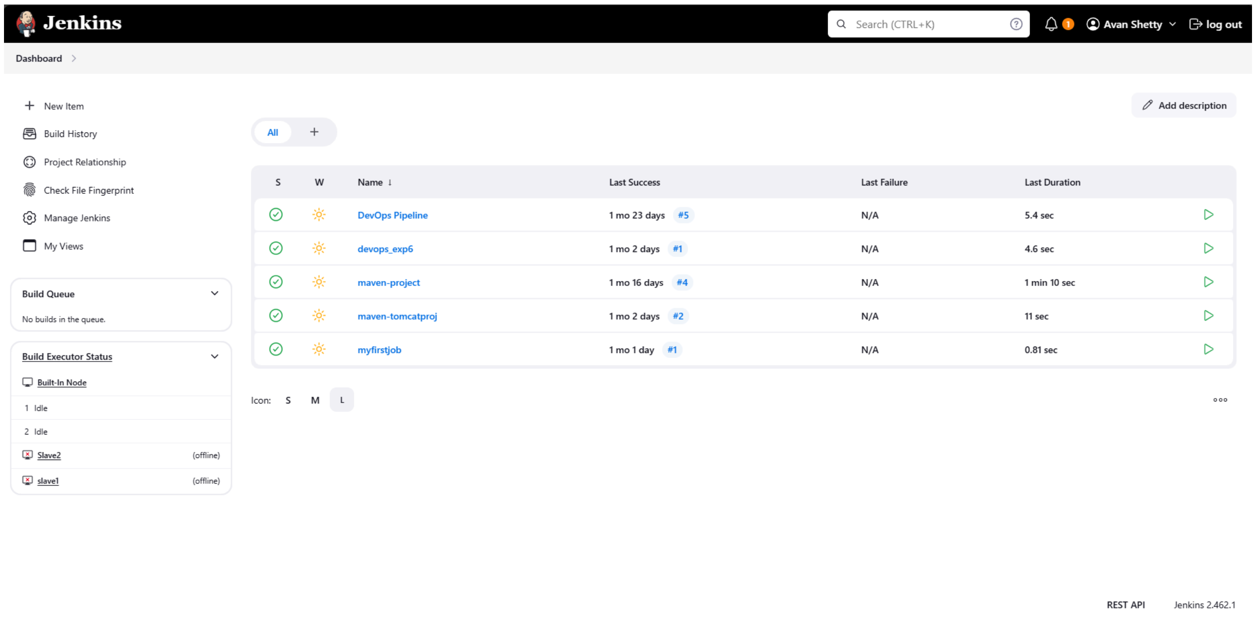
Adv_devops_7 **Avan Shetty** **Roll no:52**

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

Prerequisites:
● Jenkins installed (Java JDK required)
● 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.


Jenkins
Dashboard

New Item
Build History
Project Relationship
Check File Fingerprint
Manage Jenkins
My Views

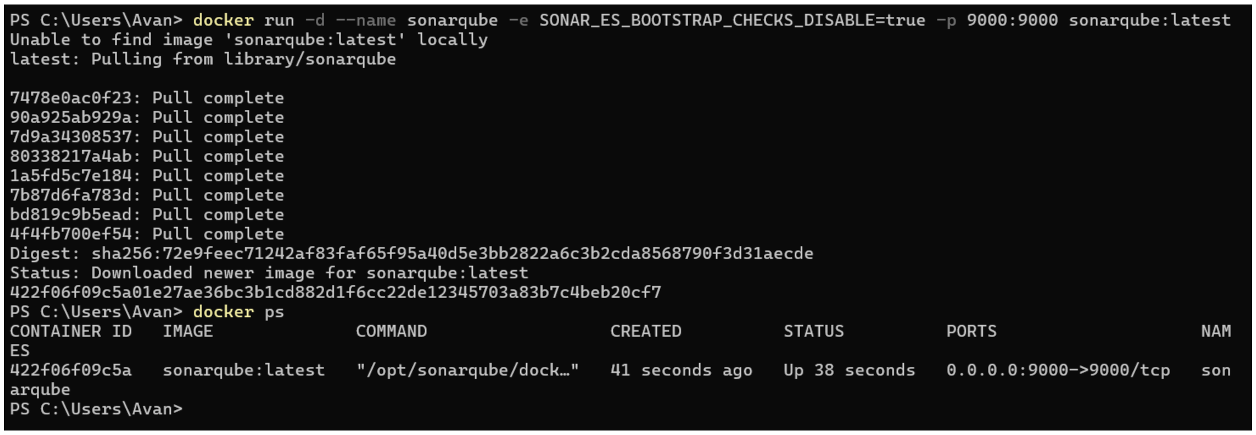
Build Queue
No builds in the queue.

Build Executor Status
Built-In Node
1 Idle
2 Idle
Slave2 (offline)
slave1 (offline)

S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☀	DevOps Pipeline	1 mo 23 days #5	N/A	5.4 sec
✓	☀	devops_exp6	1 mo 2 days #1	N/A	4.6 sec
✓	☀	maven-project	1 mo 16 days #4	N/A	1 min 10 sec
✓	☀	maven-tomcatproj	1 mo 2 days #2	N/A	11 sec
✓	☀	myfirstjob	1 mo 1 day #1	N/A	0.81 sec

SonarQube installed successfully

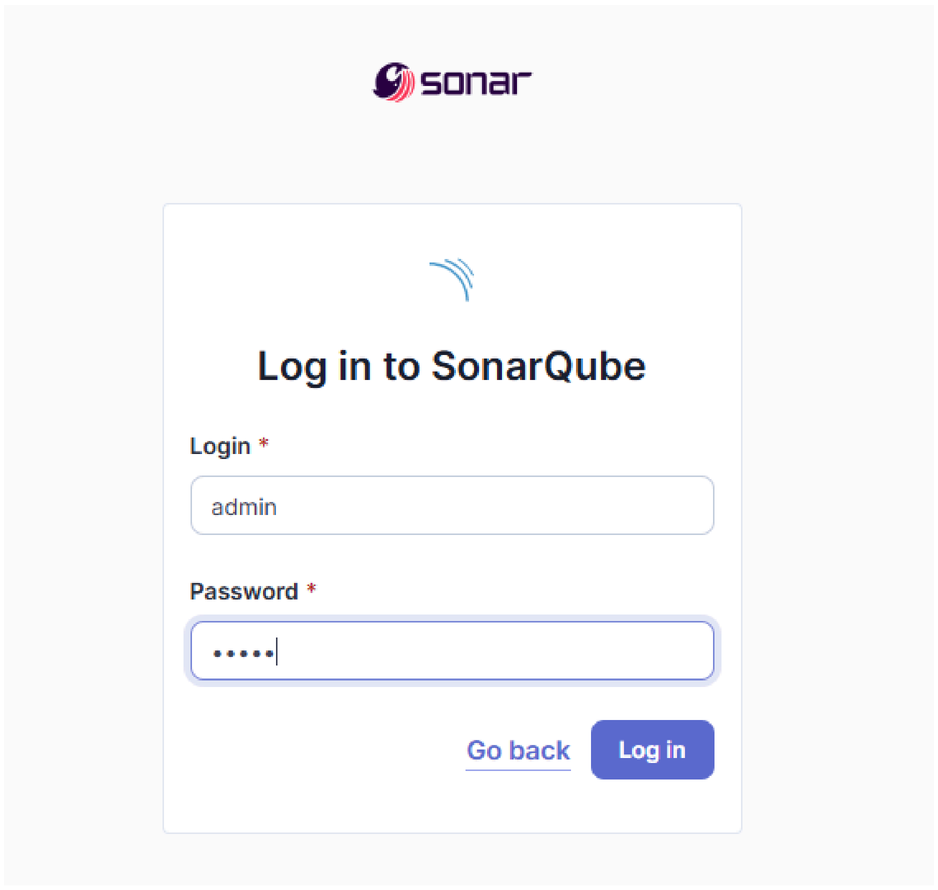
2. Run SonarQube in a Docker container using this command
-docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
Warning: run below command only once



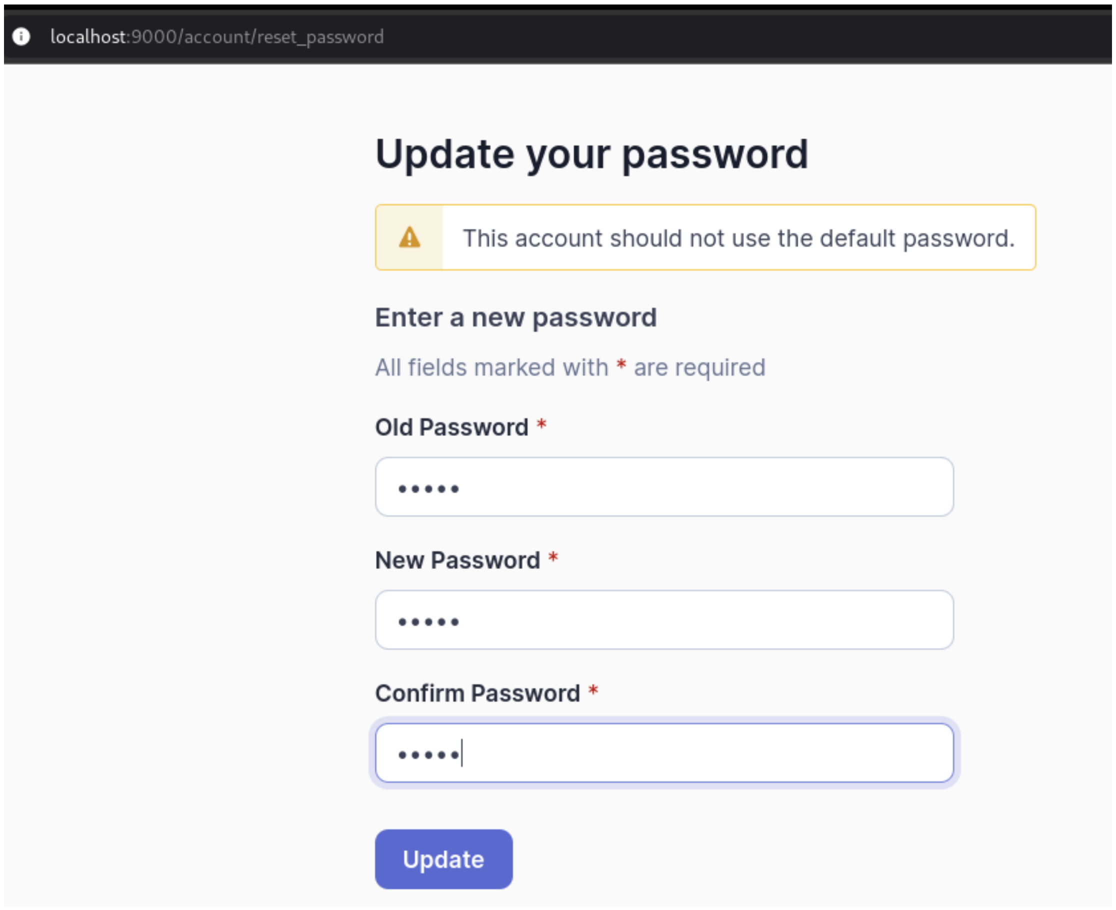
```
PS C:\Users\Avan> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube

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
422f06f09c5a01e27ae36bc3b1cd882d1f6cc22de12345703a83b7c4beb20cf7
PS C:\Users\Avan> docker ps
CONTAINER ID   IMAGE              COMMAND                  CREATED          STATUS          PORTS                    NAM
ES
422f06f09c5a   sonarqube:latest   "/opt/sonarqube/dock…"   41 seconds ago   Up 38 seconds   0.0.0.0:9000->9000/tcp   son
arqube
PS C:\Users\Avan>
```

Our jenkins is running on port 8080

3. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.



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


localhost:9000/account/reset_password

Update your password

This account should not use the default password.

Enter a new password

All fields marked with * are required

Old Password * : •••••

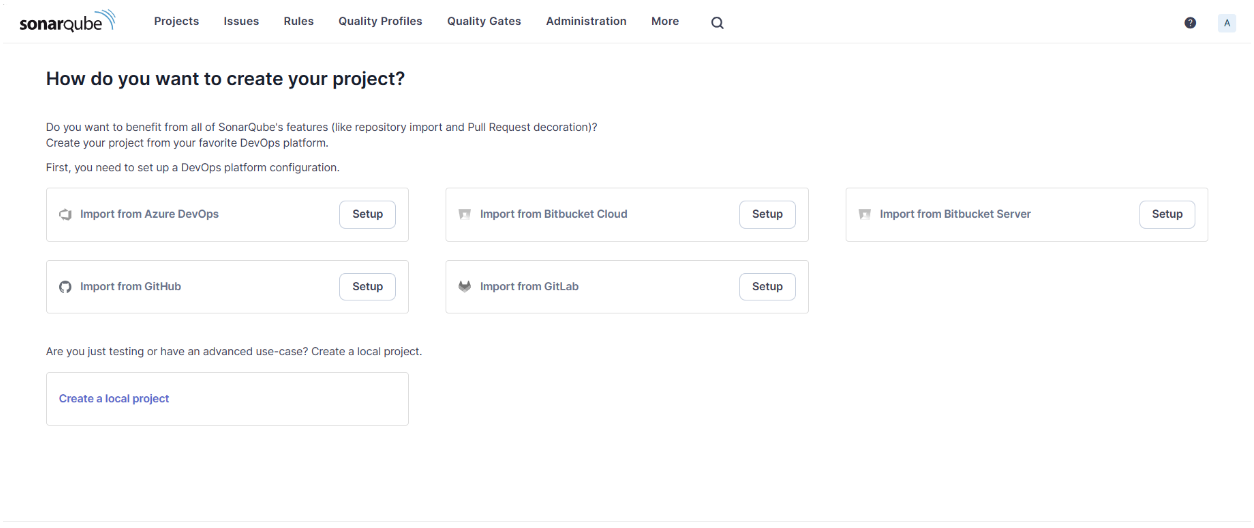
New Password * : •••••

Confirm Password * : •••••

After logging, we have to change default password

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

Click on **create a local project** on dashboard



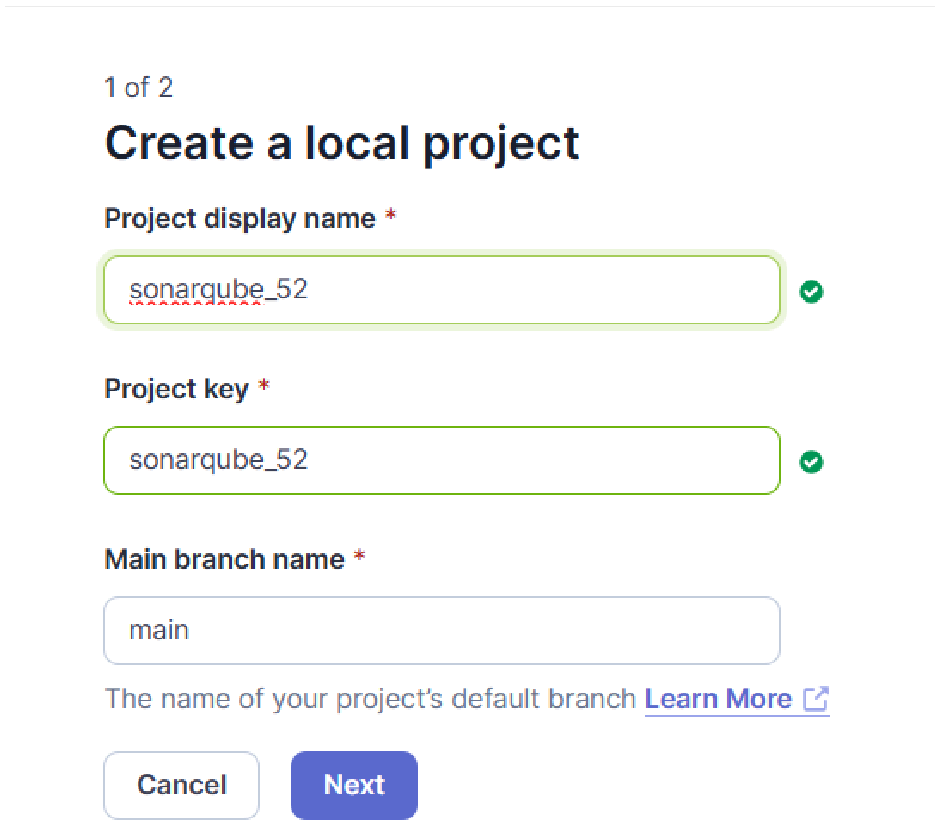
How do you want to create your project?

Do you want to benefit from all of SonarQube's features (like repository import and Pull Request decoration)?
Create your project from your favorite DevOps platform.

First, you need to set up a DevOps platform configuration.

Import from Azure DevOps Setup
Import from Bitbucket Cloud Setup
Import from Bitbucket Server Setup
Import from GitHub Setup
Import from GitLab Setup

Are you just testing or have an advanced use-case? Create a local project.



1 of 2

Create a local project

Project display name *

sonarqube_52

Project key *

sonarqube_52

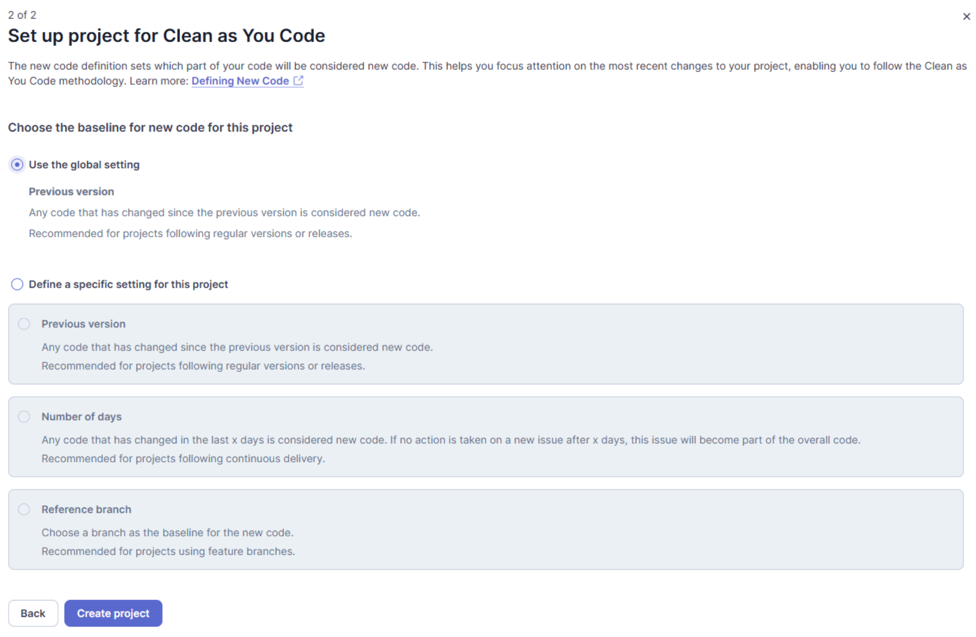
Main branch name *

main

The name of your project's default branch Learn More

Cancel Next

Set the baseline as global setting so that you don't need to make changes continuously every time



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.

Number of days

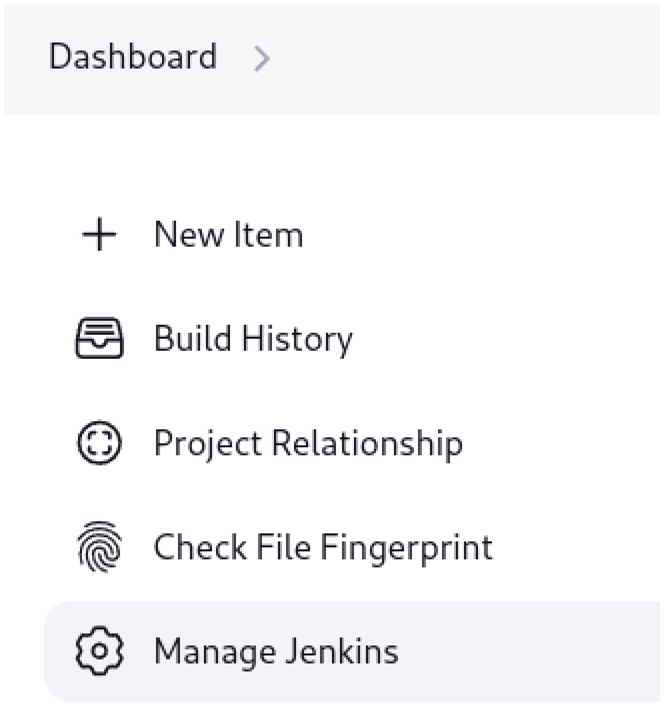
Any code that has changed in the last x days is considered new code. If no action is taken on a new issue after x days, this issue will become part of the overall code.
Recommended for projects following continuous delivery.

Reference branch

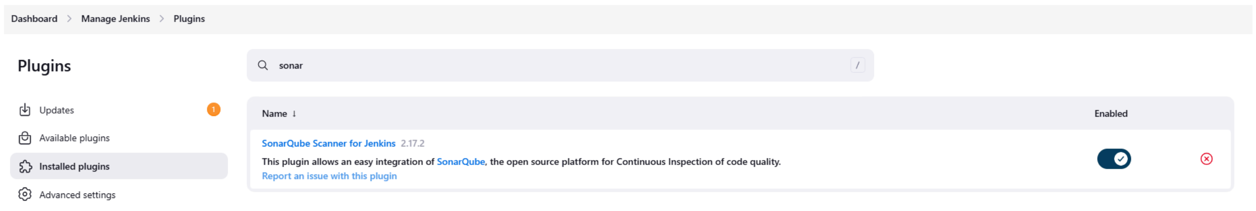
Choose a branch as the baseline for the new code.
Recommended for projects using feature branches.

Back Create project

6. After setting project in sonarqube, go to **Jenkins Dashboard**

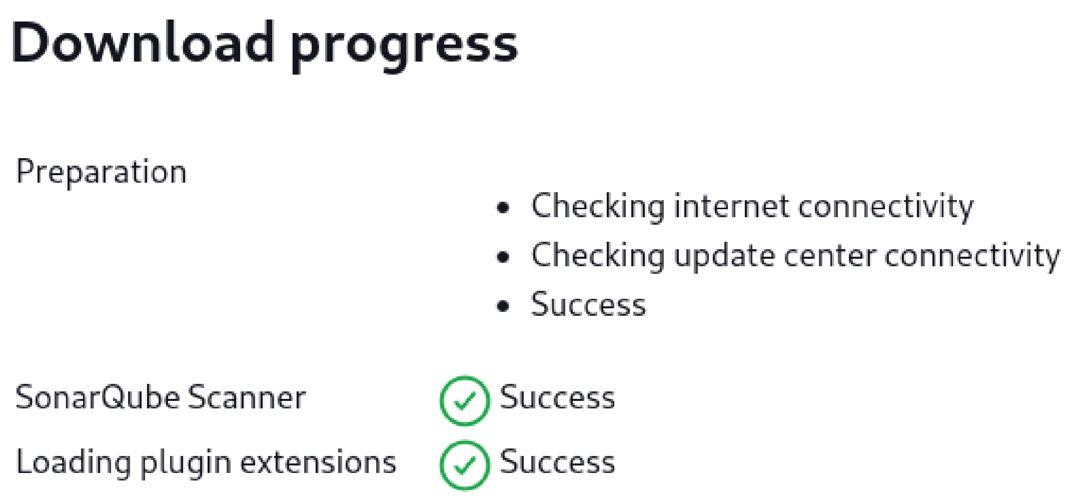


7. Go to Manage Jenkins and search for SonarQube Scanner in Plugins settings and install it.



SonarQube Scanner for Jenkins 2.17.2
This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.
Report an issue with this plugin

Our installation is in progress wait for it to download and install packages



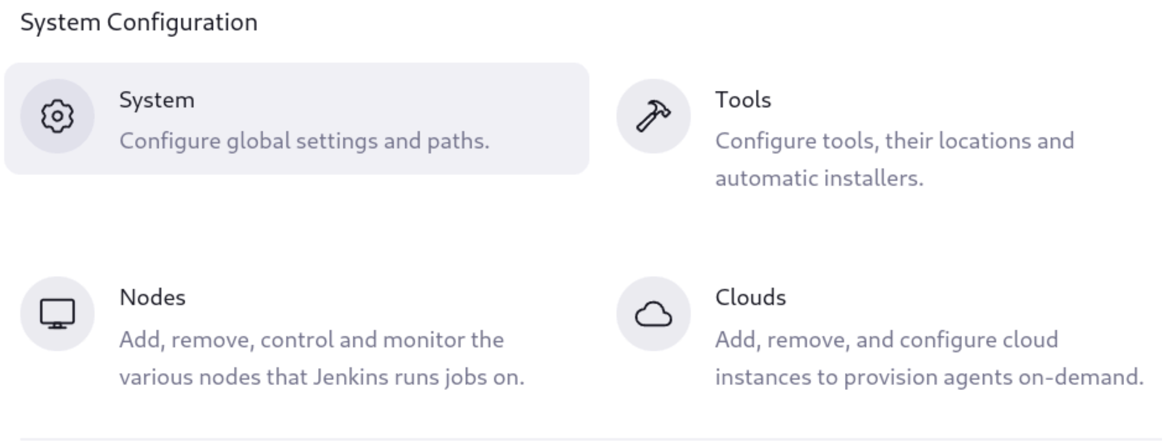
Download progress

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

SonarQube Scanner	✓ Success
Loading plugin extensions	✓ Success

Plugin installed successfully

8. Under Jenkins dashboard ‘Configure System’, look for SonarQube Servers and enter the details.



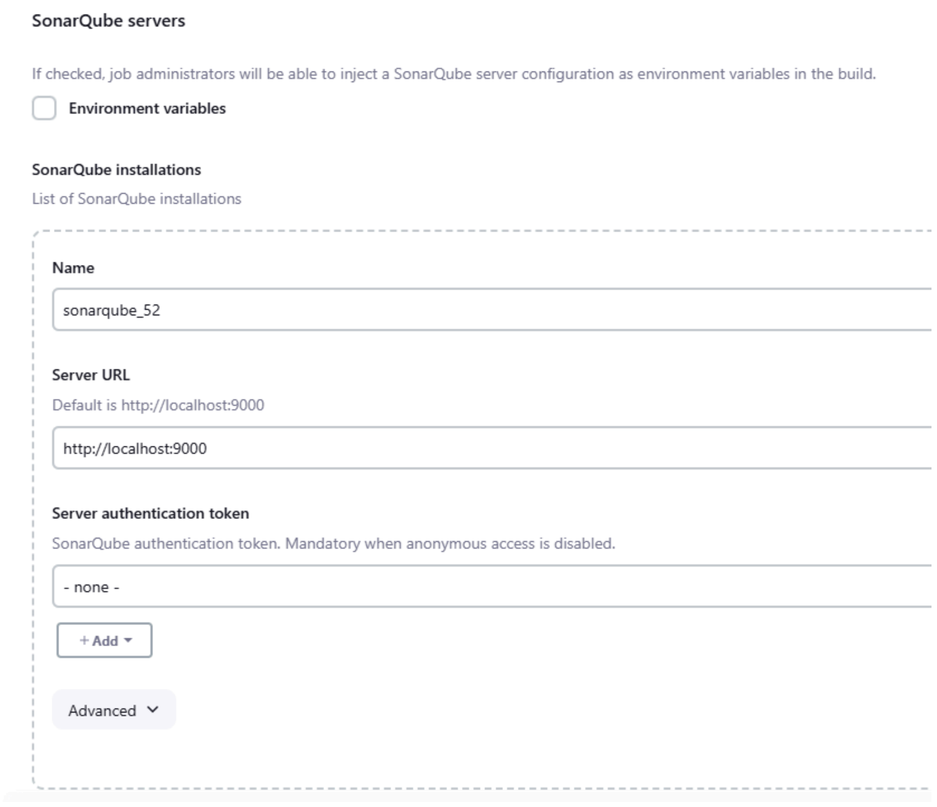
System Configuration

System
Configure global settings and paths.

Tools
Configure tools, their locations and automatic installers.

Nodes
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

Clouds
Add, remove, and configure cloud instances to provision agents on-demand.


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_52

Server URL

Default is http://localhost:9000

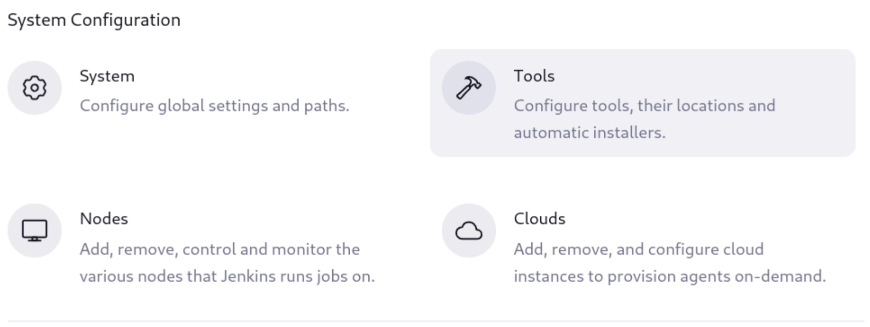
http://localhost:9000

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

- none -

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


System Configuration

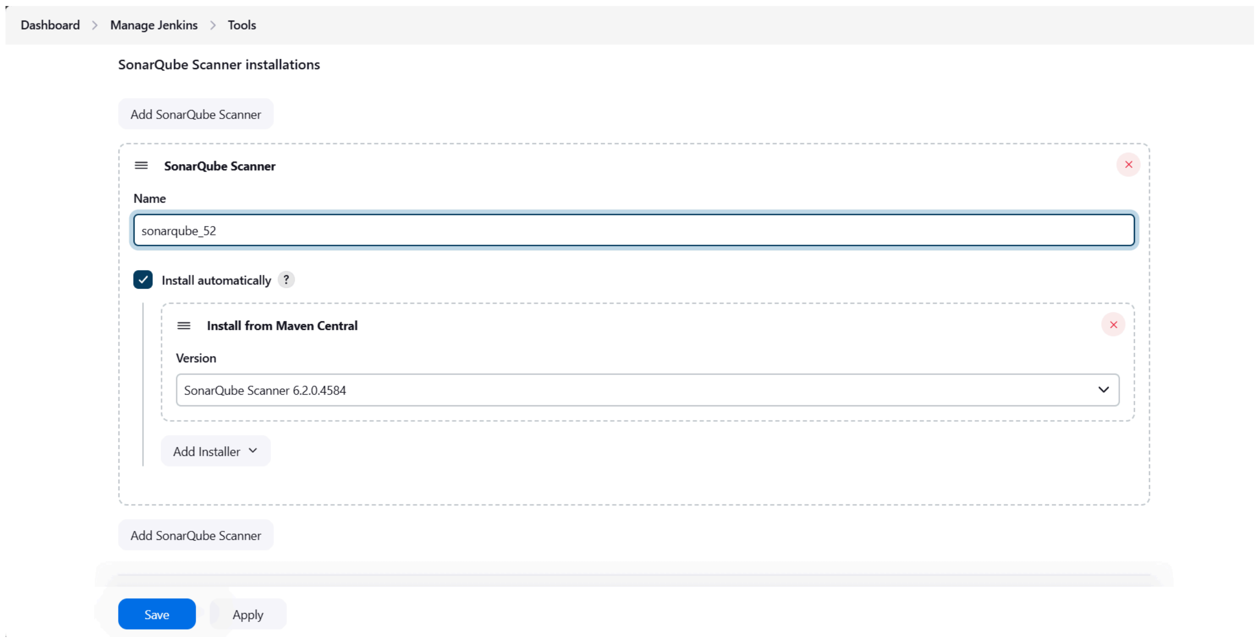
System — Configure global settings and paths.

Tools — Configure tools, their locations and automatic installers.

Nodes — Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

Clouds — Add, remove, and configure cloud instances to provision agents on-demand.

10. Click on **Add SonarQube Scanner**


Dashboard > Manage Jenkins > Tools

SonarQube Scanner installations

Add SonarQube Scanner

SonarQube Scanner

Name: sonarqube_52

☑ Install automatically

Install from Maven Central

Version: SonarQube Scanner 6.2.0.4584

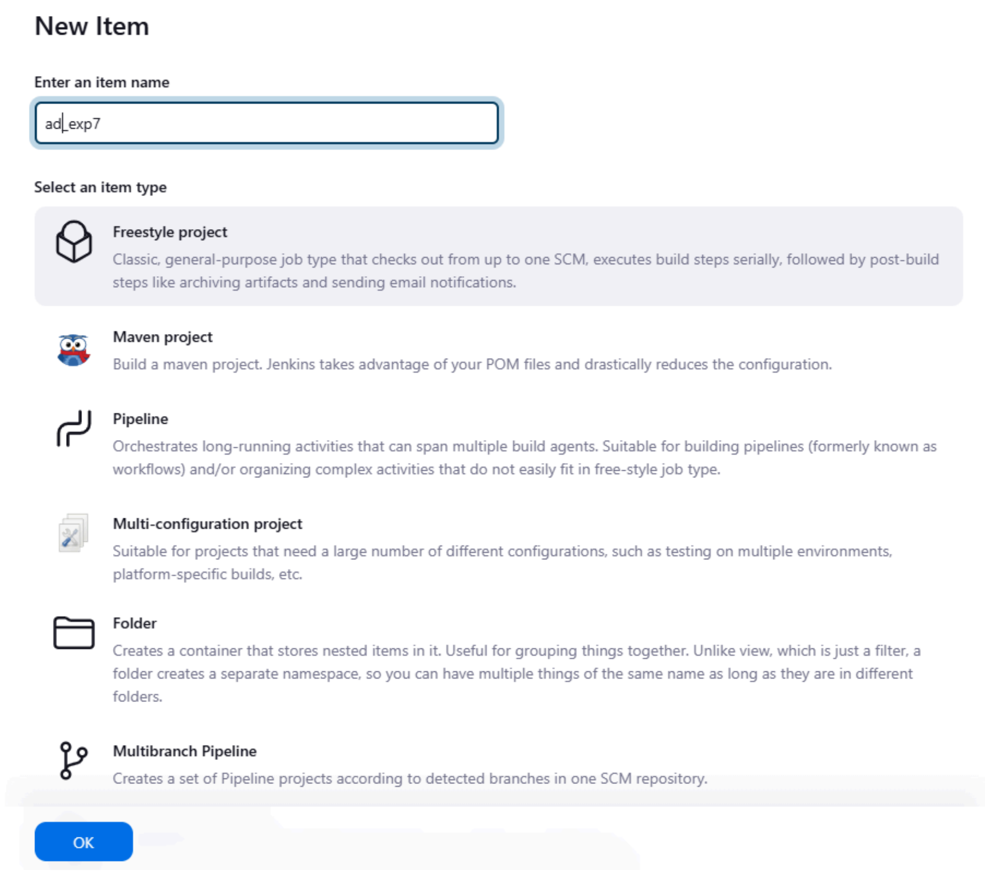
Add Installer

Add SonarQube Scanner

Save Apply

Select Latest version and save configuration

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


New Item

Enter an item name

ad_exp7

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.

Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

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.

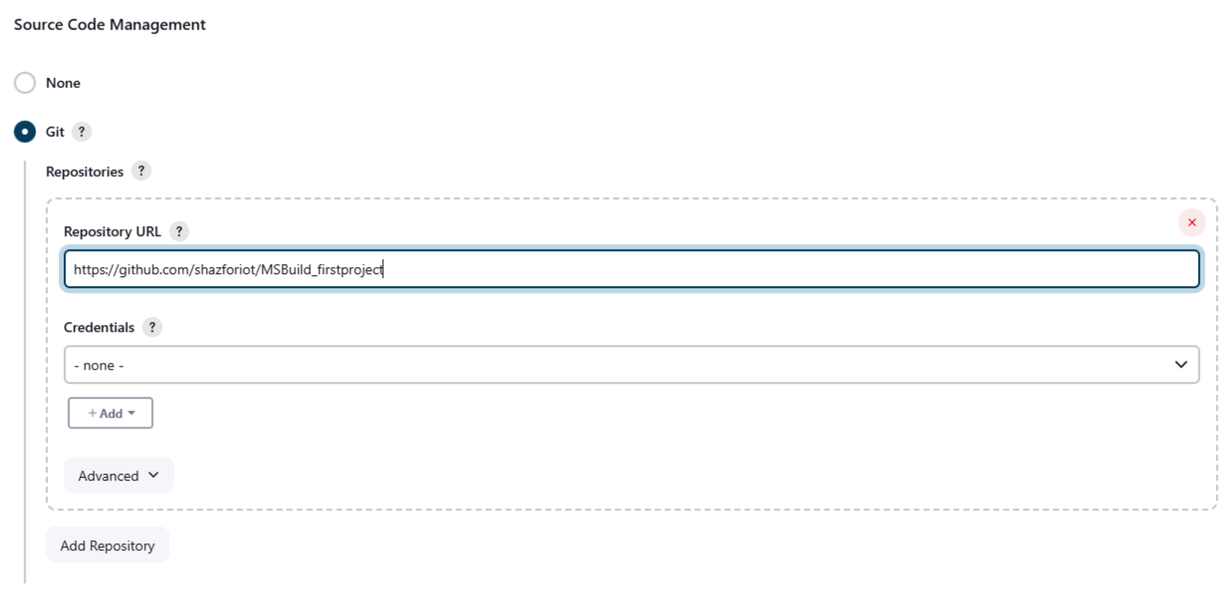
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

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.

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

OK

12. 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 the integration


Source Code Management

None

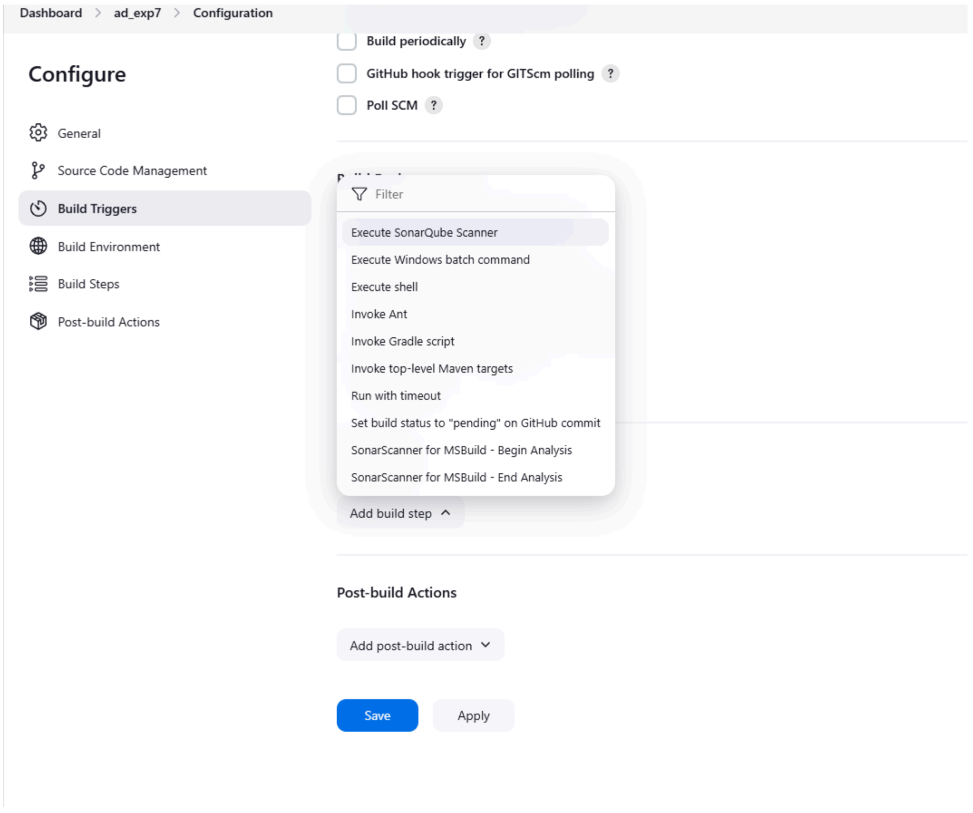
Git

Repositories

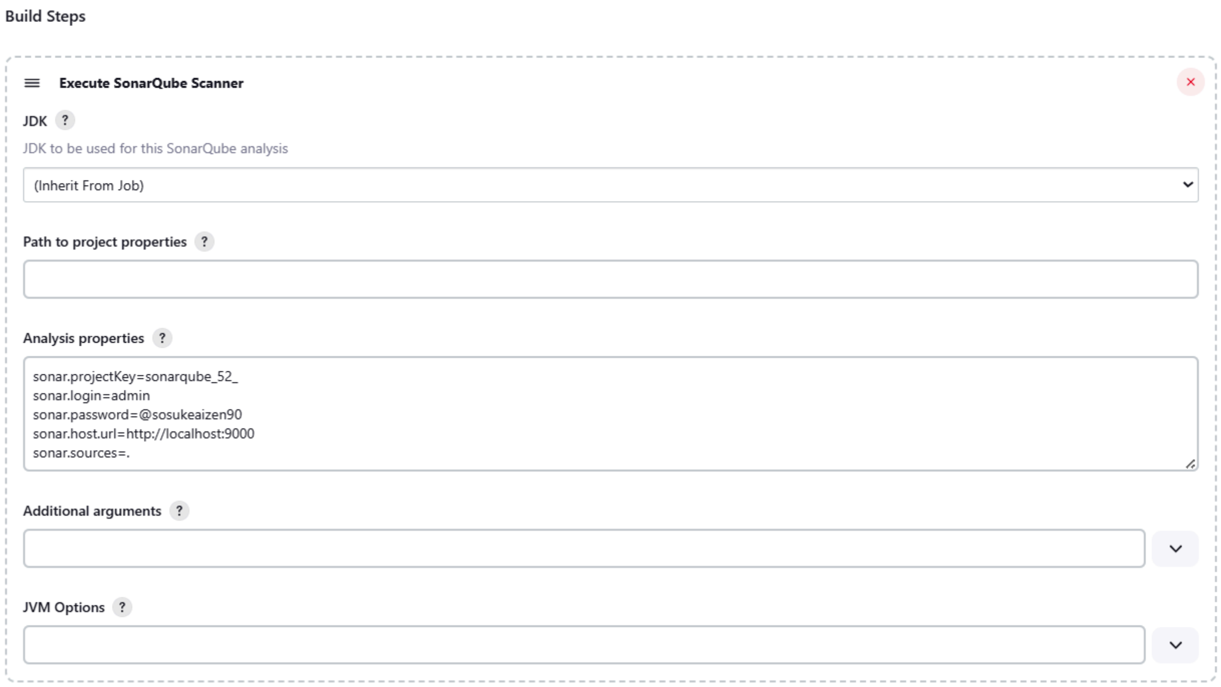
Repository URL: https://github.com/shazforiot/MSBuild_firstproject

Credentials: - none -

13. Under Build-> Execute SonarQube Scanner ->Click on add build steps
Then click on **Execute SonarQube Scanner**


Dashboard > ad_exp7 > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
Build periodically
GitHub hook trigger for GITScm polling
Poll SCM
Filter
Execute SonarQube Scanner
Execute Windows batch command
Execute shell
Invoke Ant
Invoke Gradle script
Invoke top-level Maven targets
Run with timeout
Set build status to "pending" on GitHub commit
SonarScanner for MSBuild - Begin Analysis
SonarScanner for MSBuild - End Analysis
Add build step
Post-build Actions
Add post-build action
Save
Apply

14. Mention the SonarQube Project Key, Login, Password, Source path and Host URL in Analysis properties



Build Steps

Execute SonarQube Scanner

JDK

JDK to be used for this SonarQube analysis

(Inherit From Job)

Path to project properties

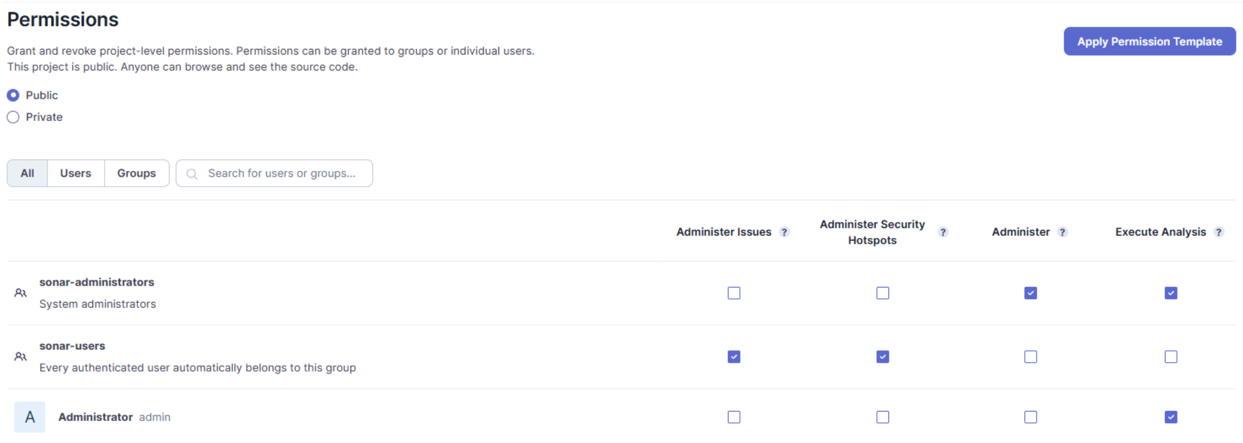
Analysis properties

```
sonar.projectKey=sonarqube_52_
sonar.login=admin
sonar.password=@sosukeaizen90
sonar.host.url=http://localhost:9000
sonar.sources=.
```

Additional arguments

JVM Options

15. Go to http://localhost:9000/project_roles?id=<project_key> and allow Execute Permissions to the Admin user.



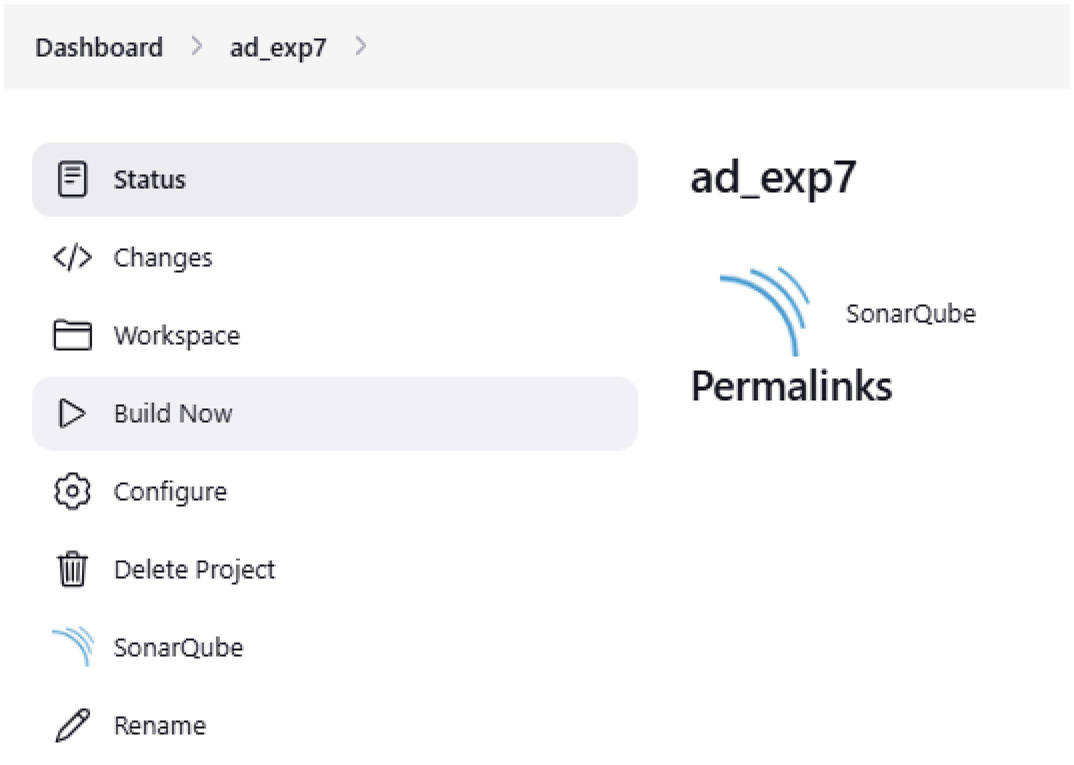
Permissions

Grant and revoke project-level permissions. Permissions can be granted to groups or individual users.
This project is public. Anyone can browse and see the source code.

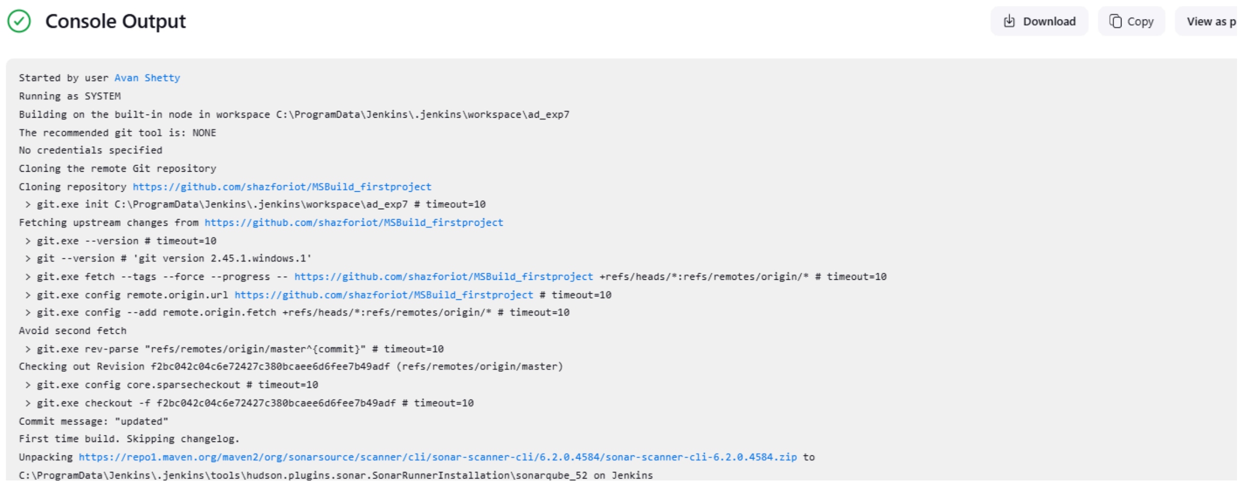
[✓ Public]
Private

	Administer Issues	Administer Security Hotspots	Administer	Execute Analysis
sonar-administrators System administrators	☐	☐	☑	☑
sonar-users Every authenticated user automatically belongs to this group	☑	☑	☐	☐
Administrator admin	☐	☐	☐	☑

Run The Build.

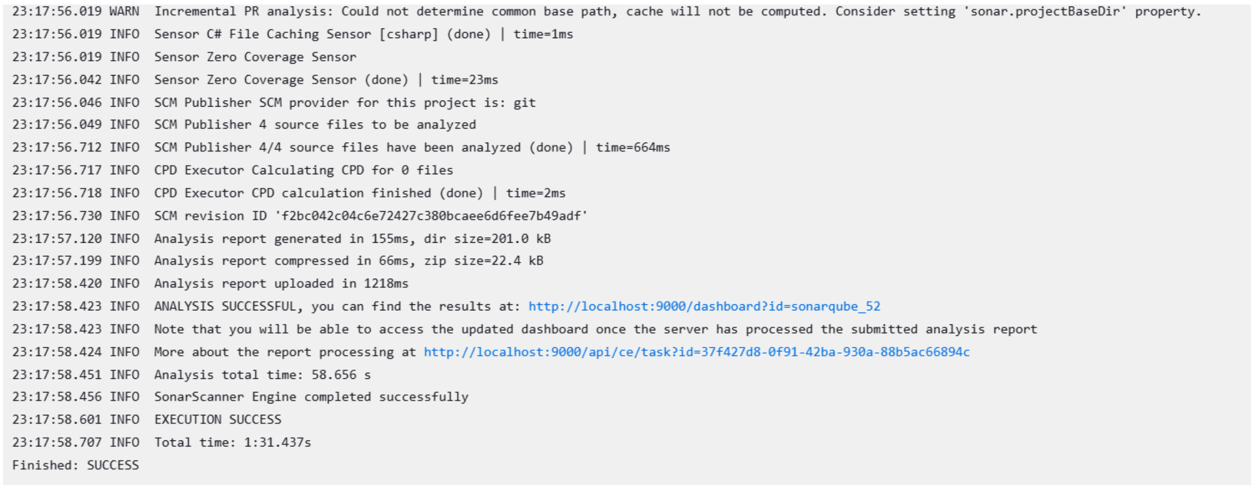


16. Check the console output



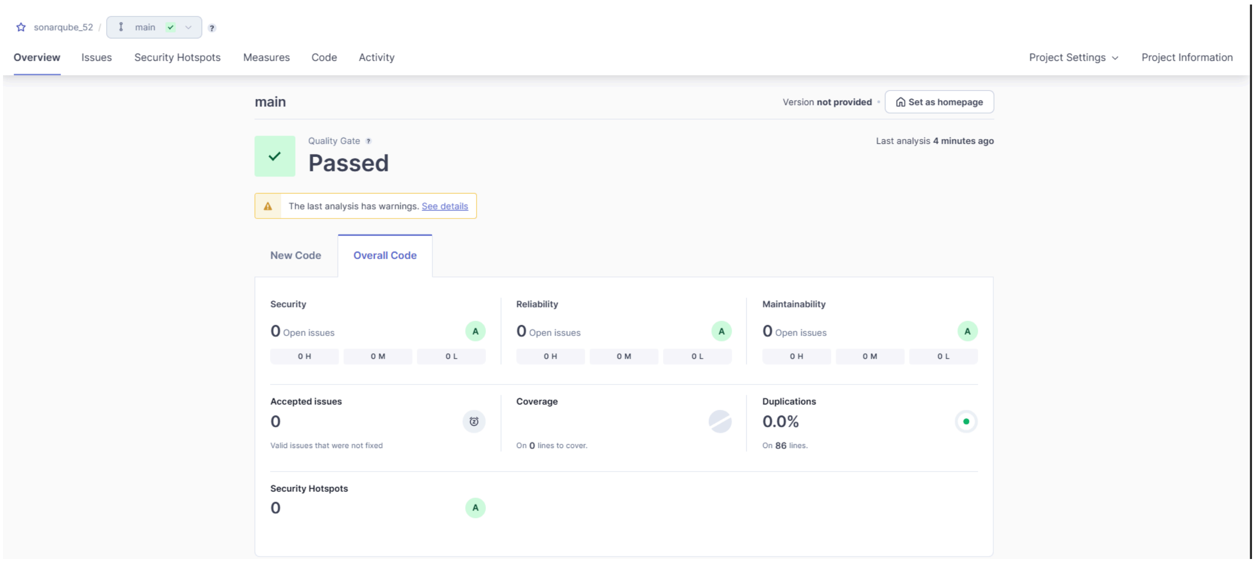
```
Started by user Avan Shetty
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\.jenkins\workspace\ad_exp7
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/shazforiot/MSBuild_firstproject
 > git.exe init C:\ProgramData\Jenkins\.jenkins\workspace\ad_exp7 # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject
 > git.exe --version # timeout=10
 > git --version # 'git version 2.45.1.windows.1'
 > git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject # timeout=10
 > git.exe config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
 > 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"
First time build. Skipping changelog.
Unpacking https://repo1.maven.org/maven2/org/sonarsource/scanner/cli/sonar-scanner-cli/6.2.0.4584/sonar-scanner-cli-6.2.0.4584.zip to
C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube_52 on Jenkins
```

In the end if you view as success then the build is successful



```
23:17:56.019 WARN  Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir' property.
23:17:56.019 INFO  Sensor C# File Caching Sensor [csharp] (done) | time=1ms
23:17:56.019 INFO  Sensor Zero Coverage Sensor
23:17:56.042 INFO  Sensor Zero Coverage Sensor (done) | time=23ms
23:17:56.046 INFO  SCM Publisher SCM provider for this project is: git
23:17:56.049 INFO  SCM Publisher 4 source files to be analyzed
23:17:56.712 INFO  SCM Publisher 4/4 source files have been analyzed (done) | time=664ms
23:17:56.717 INFO  CPD Executor Calculating CPD for 0 files
23:17:56.718 INFO  CPD Executor CPD calculation finished (done) | time=2ms
23:17:56.730 INFO  SCM revision ID 'f2bc042c04c6e72427c380bcaee6d6fee7b49adf'
23:17:57.120 INFO  Analysis report generated in 155ms, dir size=201.0 kB
23:17:57.199 INFO  Analysis report compressed in 66ms, zip size=22.4 kB
23:17:58.420 INFO  Analysis report uploaded in 1218ms
23:17:58.423 INFO  ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube_52
23:17:58.423 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
23:17:58.424 INFO  More about the report processing at http://localhost:9000/api/ce/task?id=37f427d8-0f91-42ba-930a-88b5ac66894c
23:17:58.451 INFO  Analysis total time: 58.656 s
23:17:58.456 INFO  SonarScanner Engine completed successfully
23:17:58.601 INFO  EXECUTION SUCCESS
23:17:58.707 INFO  Total time: 1:31.437s
Finished: SUCCESS
```

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



In this way, we have integrated Jenkins with SonarQube for SAST.

Conclusion:

During the experiment, I successfully set up SonarQube using Docker and created a new project in SonarQube. I integrated it with Jenkins by creating a freestyle project linked to a Git repository for code analysis. After configuring Jenkins with the correct settings and granting necessary permissions, the project ran successfully, with all tests passing in SonarQube.However, there were some challenges, such as ensuring precise SonarQube commands and entering correct credentials. Security access needed to be configured properly, or the build would fail. Additionally, it was crucial to have the JDK installed and the path correctly set in the environment variables.