Name: Ganesh Gupta Div:D15C Roll No.13

Adv DevOps Practical 7

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

Integrating Jenkins with SonarQube:

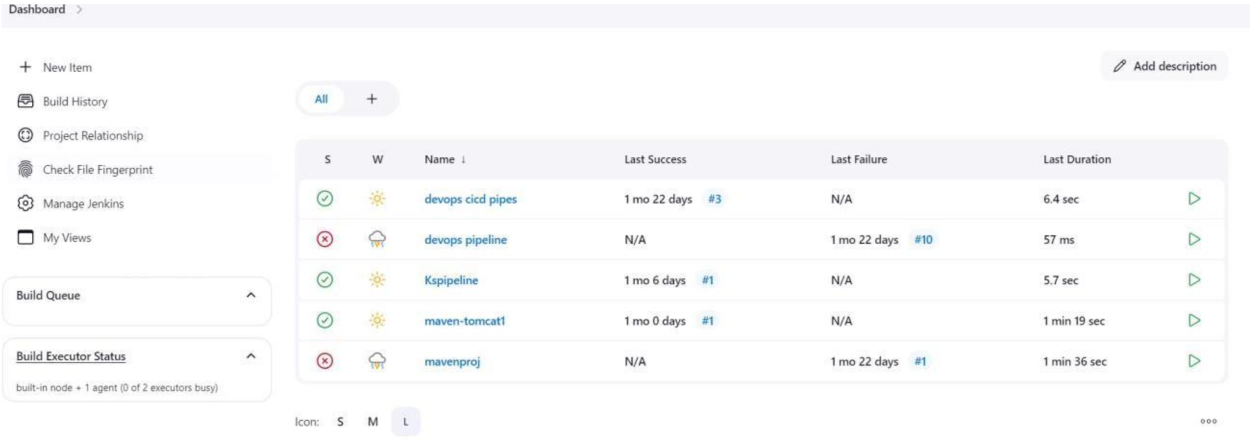
● Jenkins installed

● Docker Installed (for SonarQube)

● SonarQube Docker Image

Steps to integrate Jenkins with SonarQube

1. Open up Jenkins Dashboard on localhost, port 8090 or whichever port it is at for you.

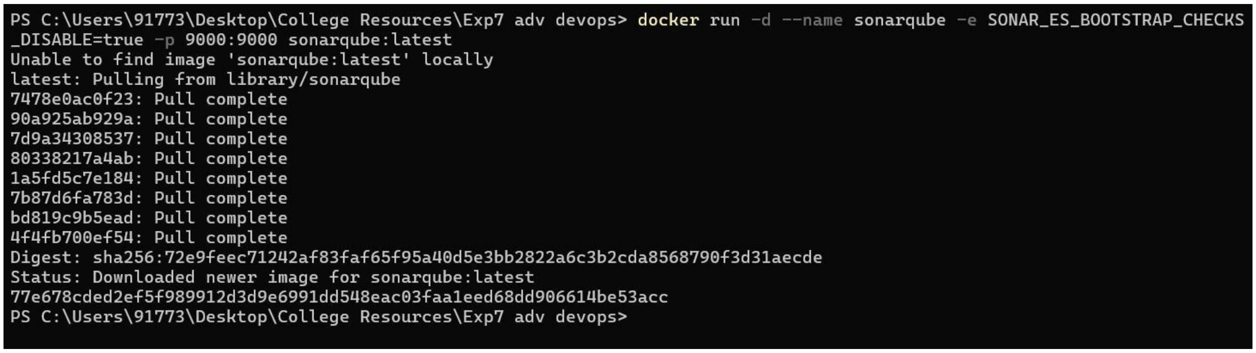


S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☀	devops cicd pipes	1 mo 22 days #3	N/A	6.4 sec
✗	🌧	devops pipeline	N/A	1 mo 22 days #10	57 ms
✓	☀	Kspipeline	1 mo 6 days #1	N/A	5.7 sec
✓	☀	maven-tomcat1	1 mo 0 days #1	N/A	1 min 19 sec
✗	🌧	mavenproj	N/A	1 mo 22 days #1	1 min 36 sec

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

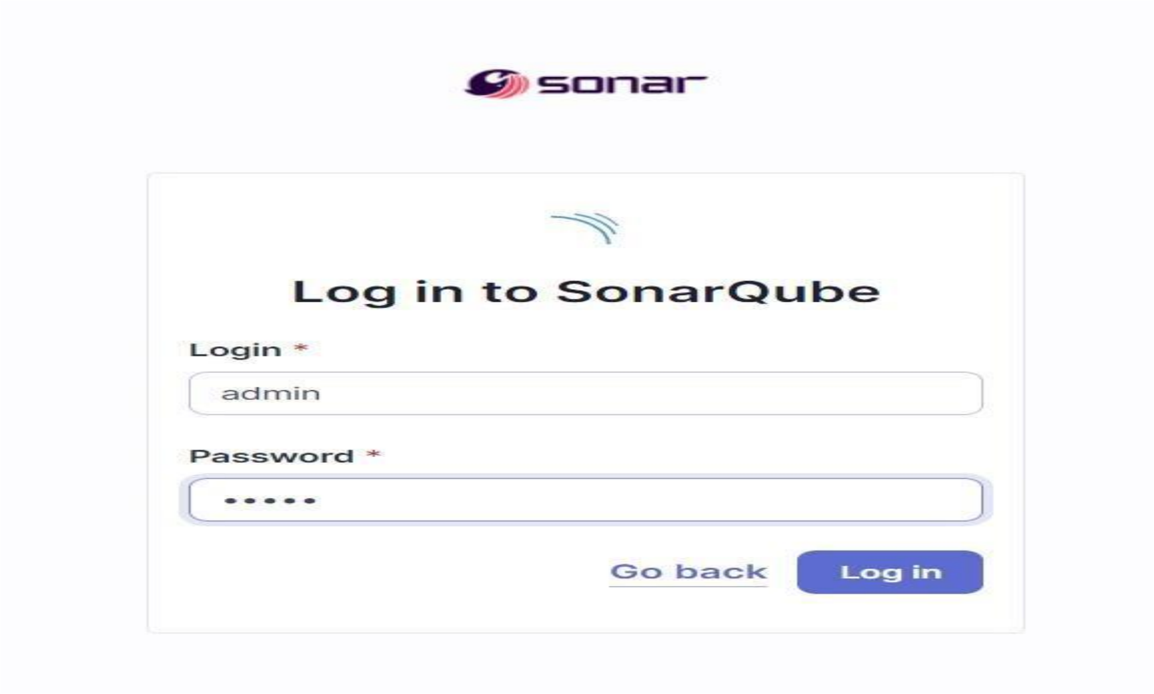
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

------------------Warning: run below command only once



```
PS C:\Users\91773\Desktop\College Resources\Exp7 adv devops> 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
77e678cded2ef5f989912d3d9e6991dd548eac03faa1eed68dd906614be53acc
PS C:\Users\91773\Desktop\College Resources\Exp7 adv devops>
```

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

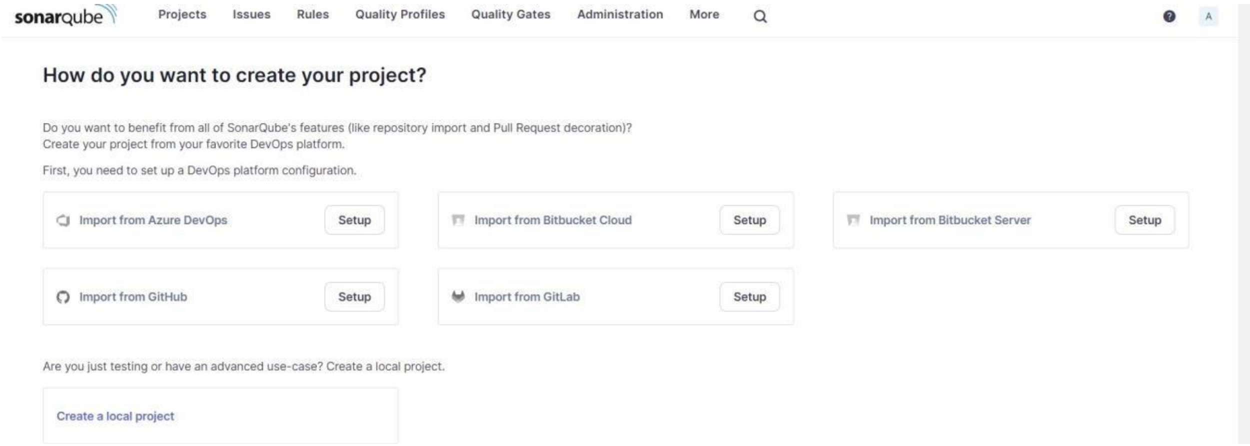


Log in to SonarQube

Login * admin

Password * •••••

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



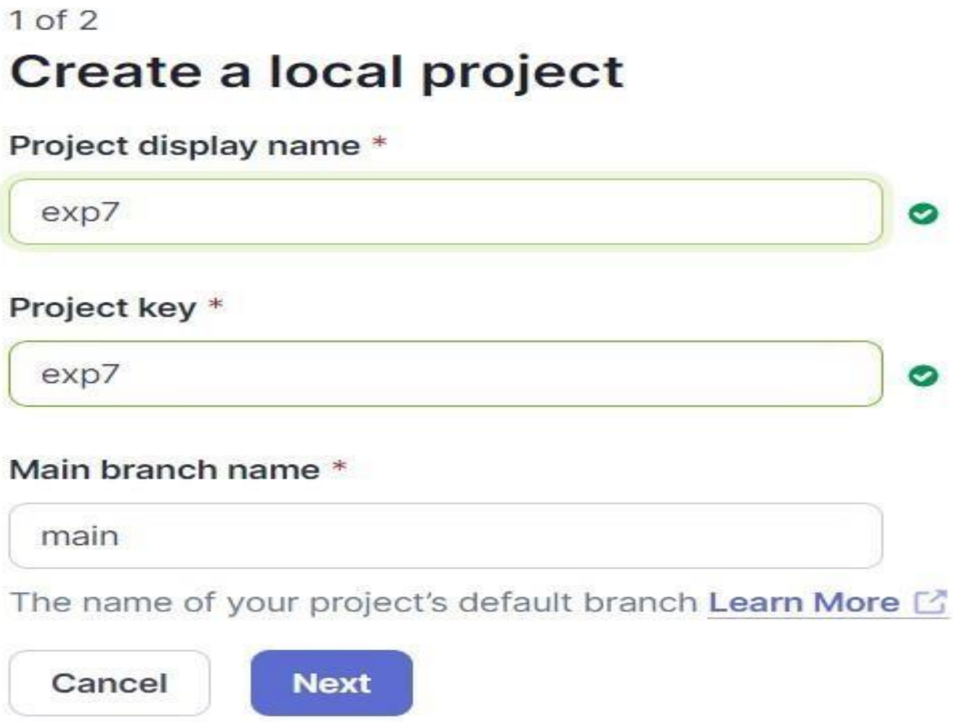
How do you want to create your project?

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.

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

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


1 of 2

Create a local project

Project display name *

exp7

Project key *

exp7

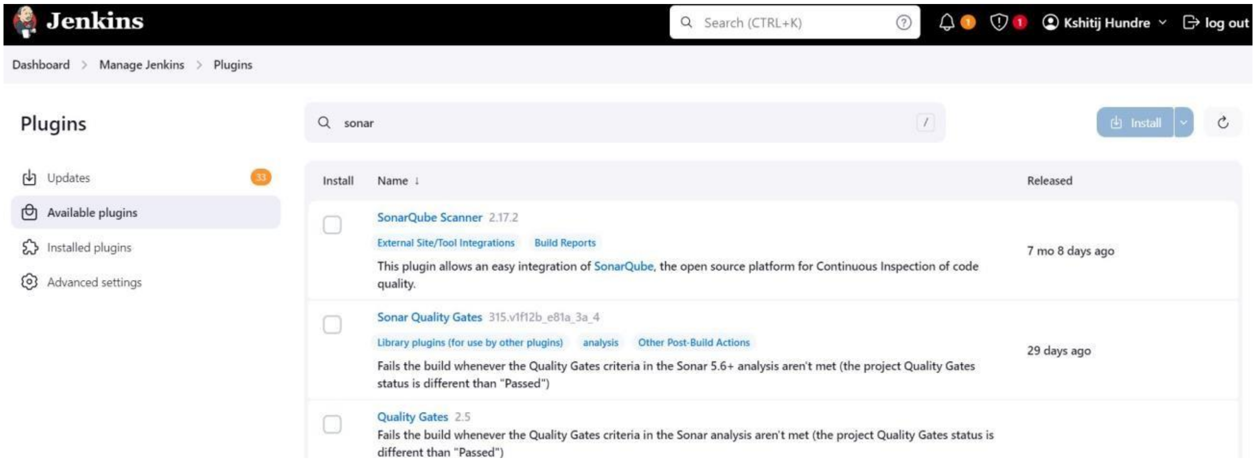
Main branch name *

main

The name of your project's default branch Learn More

Setup the project and come back to Jenkins Dashboard.

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



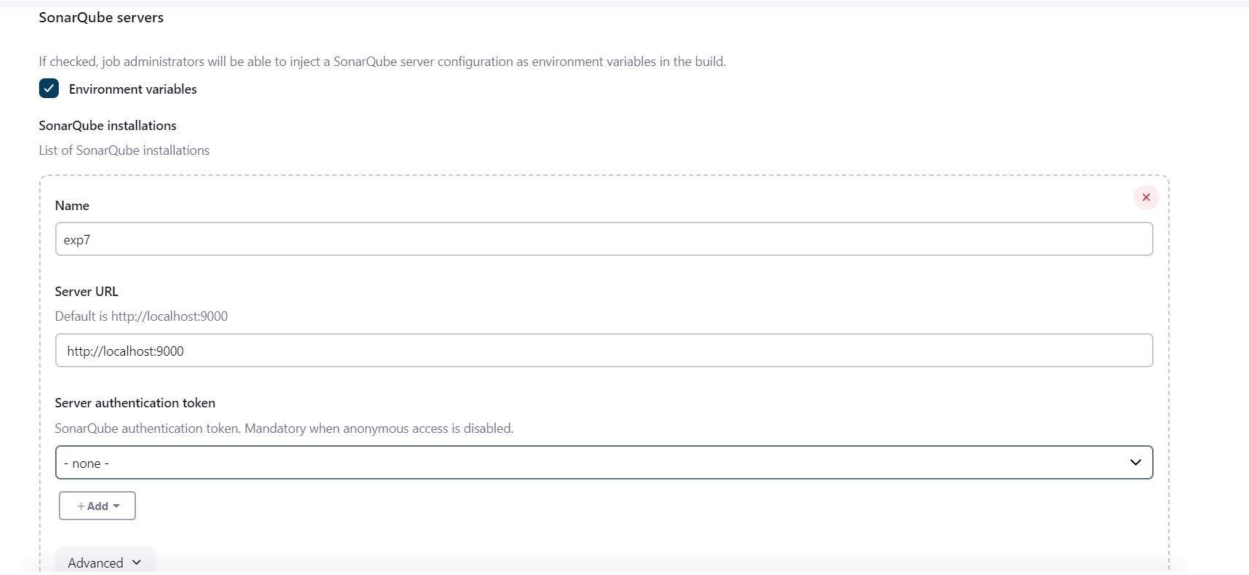
Install	Name ↓	Released
□	SonarQube Scanner 2.17.2 External Site/Tool Integrations Build Reports This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.	7 mo 8 days ago
□	Sonar Quality Gates 315.v1f12b_e81a_3a_4 Library plugins (for use by other plugins) analysis Other Post-Build Actions Fails the build whenever the Quality Gates criteria in the Sonar 5.6+ analysis aren't met (the project Quality Gates status is different than "Passed")	29 days ago
□	Quality Gates 2.5 Fails the build whenever the Quality Gates criteria in the Sonar analysis aren't met (the project Quality Gates status is different than "Passed")	

6. Under Jenkins 'Manage Jenkins' then go to 'system', scroll and look for **SonarQube Servers** and enter the details.

Enter the Server Authentication token if needed.

In SonarQube installations: Under **Name** add <project name of sonarqube> for me **sahilexp7** In

Server URL Default is http://localhost:9000



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: exp7

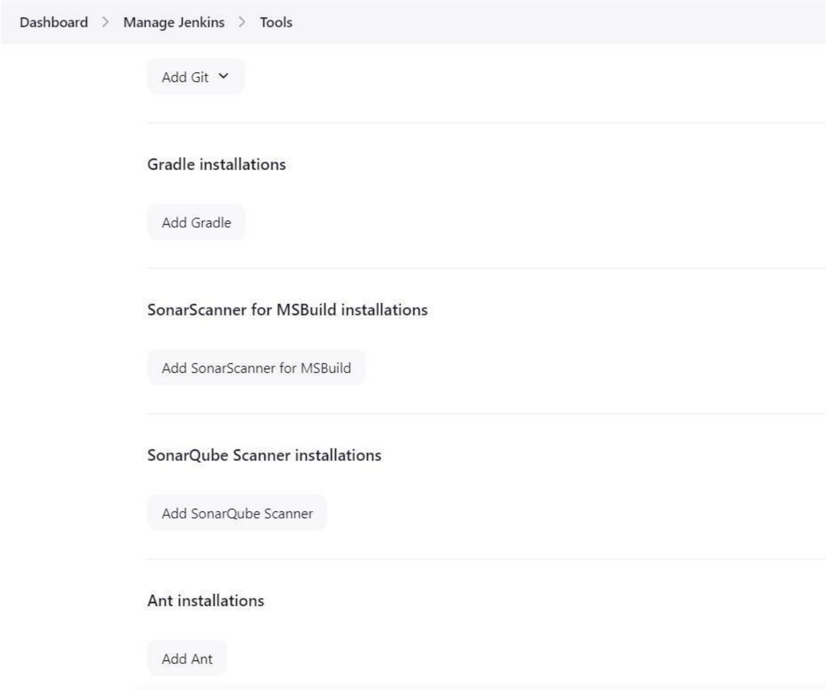
Server URL
Default is http://localhost:9000
http://localhost:9000

Server authentication token
SonarQube authentication token. Mandatory when anonymous access is disabled.
- none -

7. Search for SonarQube Scanner under Global Tool Configuration. Choose the latest

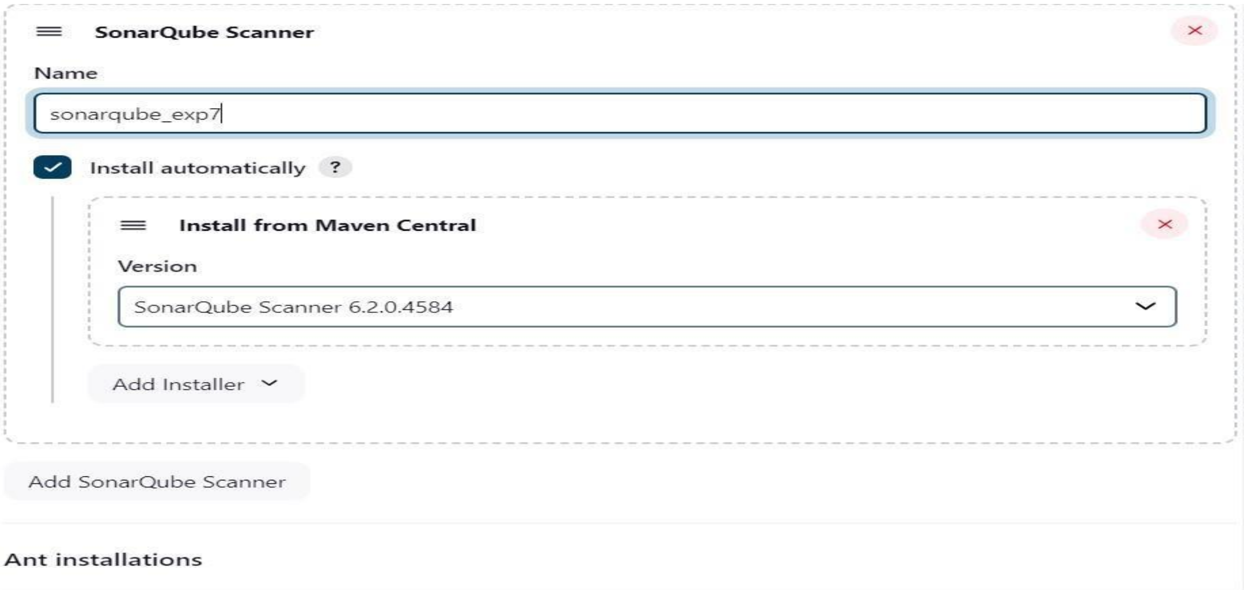
configuration and choose Install automatically.

Dashboard > Manage Jenkins > Tools



Dashboard > Manage Jenkins > Tools
Add Git
Gradle installations
Add Gradle
SonarScanner for MSBuild installations
Add SonarScanner for MSBuild
SonarQube Scanner installations
Add SonarQube Scanner
Ant installations
Add Ant

Check the “Install automatically” option. → Under name any name as identifier → Check the “Install automatically” option.



SonarQube Scanner

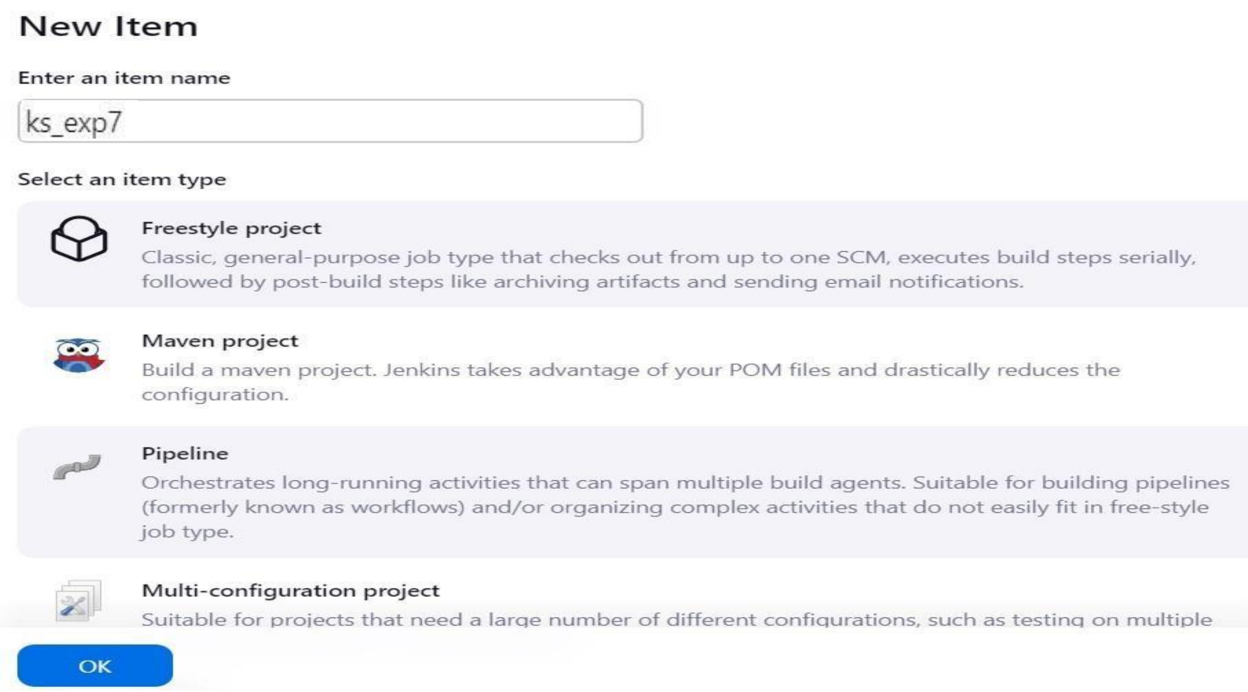
Name: sonarqube_exp7

☑ Install automatically

Install from Maven Central

Version: SonarQube Scanner 6.2.0.4584

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



New Item

Enter an item name

ks_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.

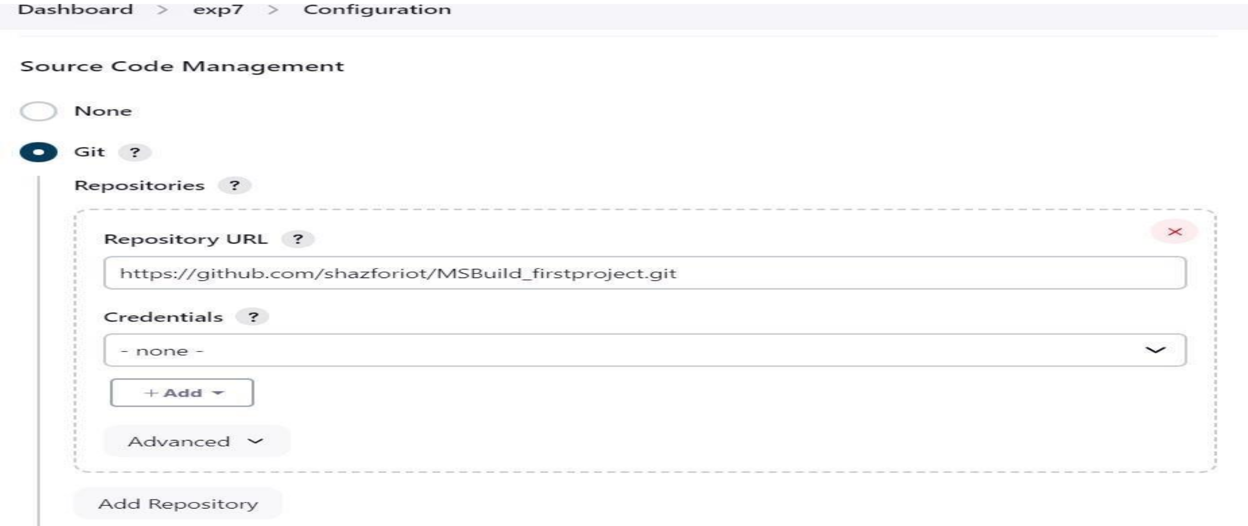
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple

9. 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.



Dashboard > exp7 > Configuration

Source Code Management

○ None

● Git ?

Repositories ?

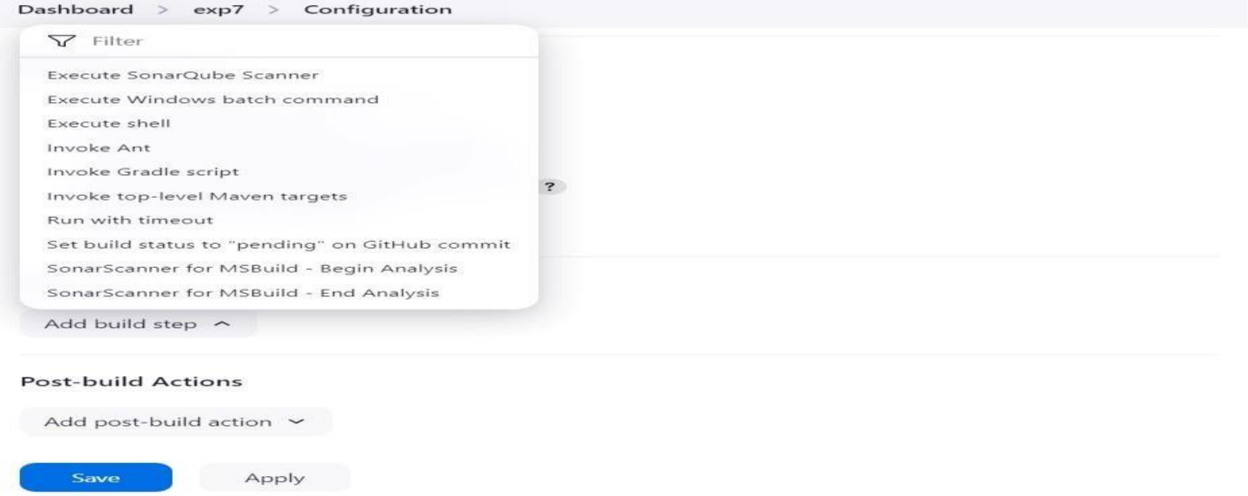
Repository URL ?

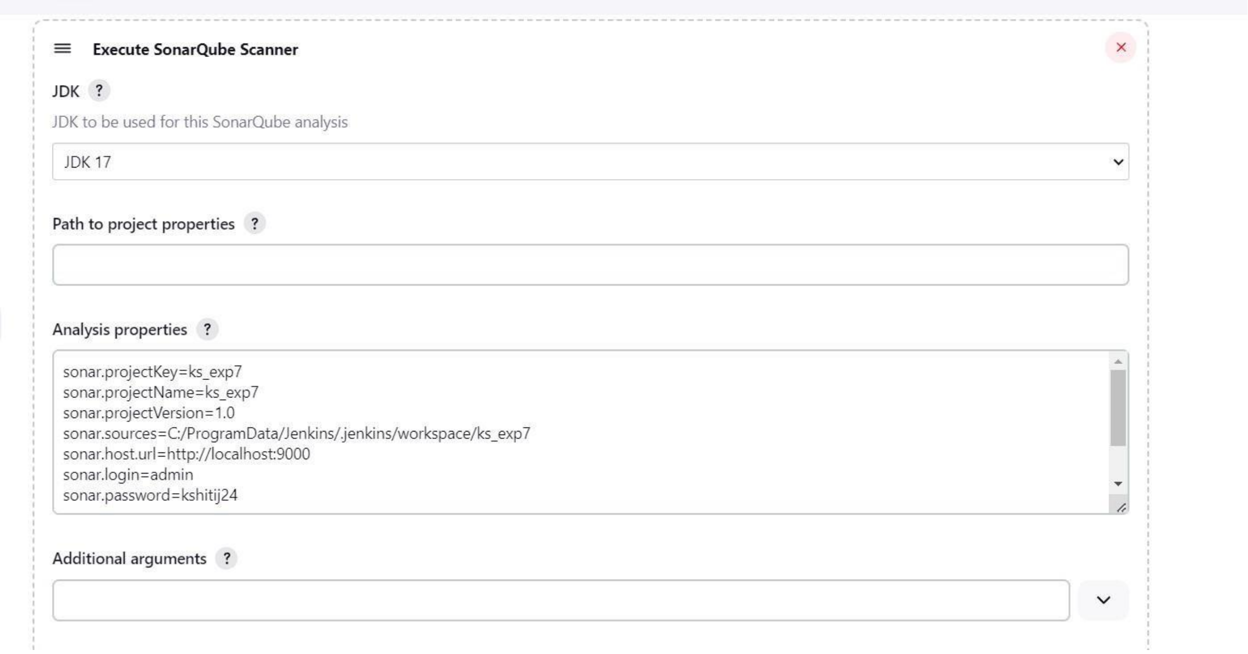
https://github.com/shazforiot/MSBuild_firstproject.git

Credentials ?

- none -

10. Under **Select project → Configuration → Build steps → Execute SonarQube Scanner**, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.


Dashboard > exp7 > Configuration
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


Execute SonarQube Scanner

JDK ?

JDK to be used for this SonarQube analysis

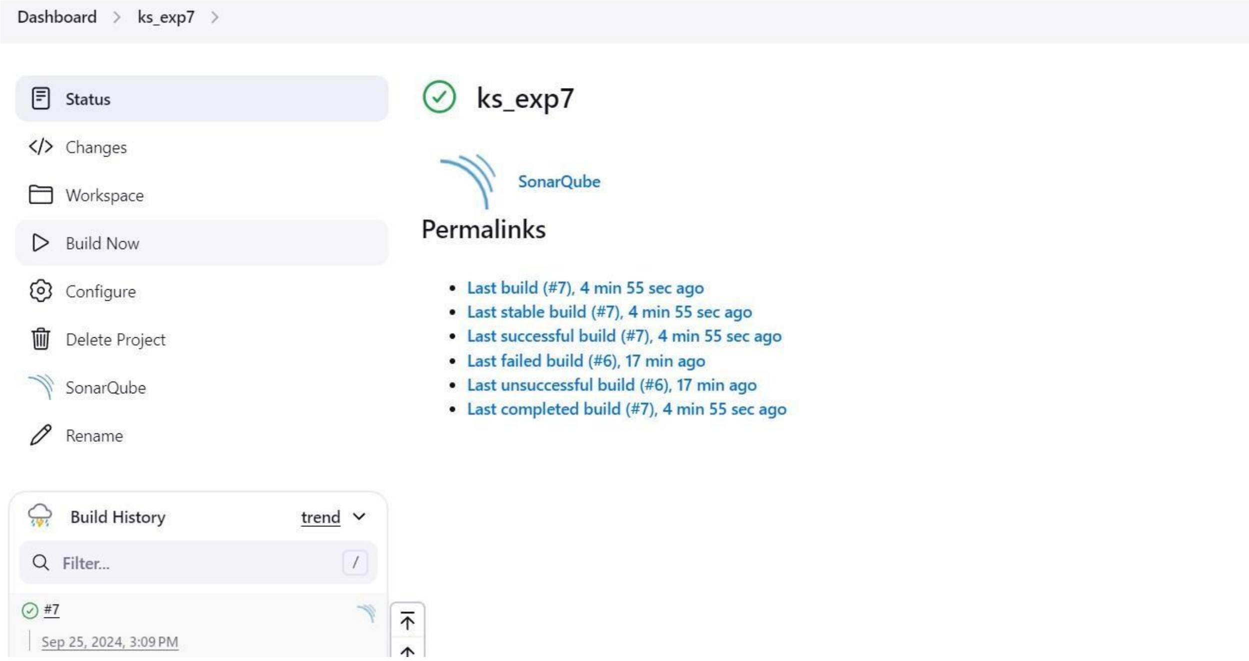
JDK 17

Path to project properties ?

Analysis properties ?

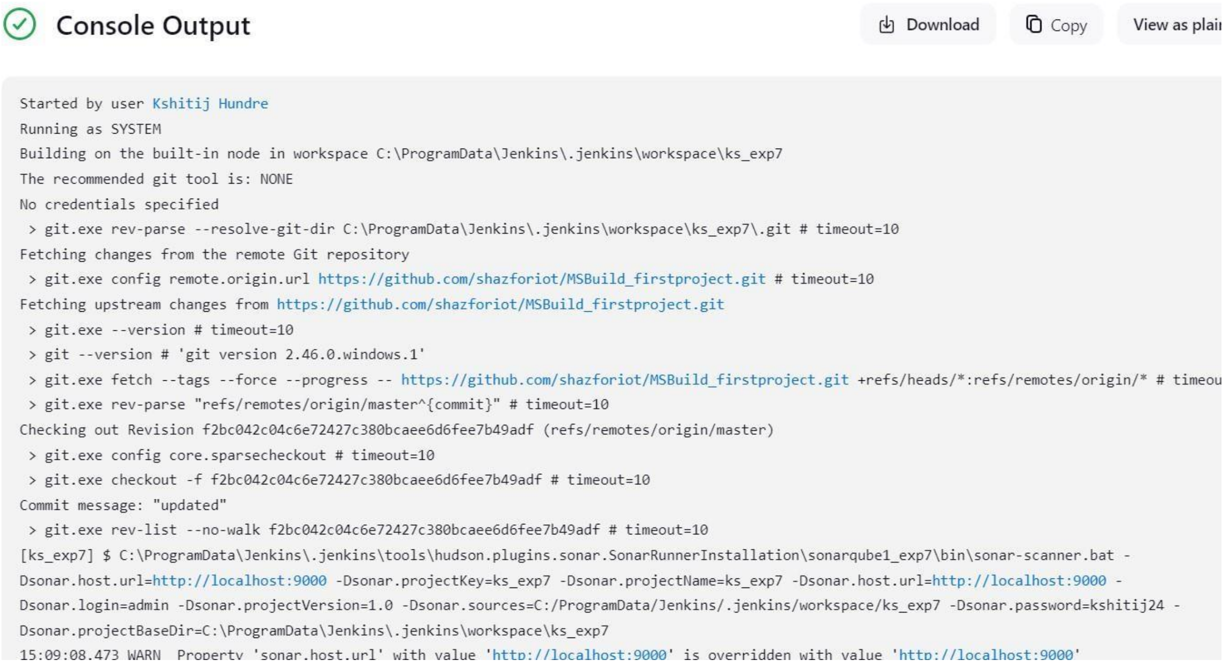
```
sonar.projectKey=ks_exp7
sonar.projectName=ks_exp7
sonar.projectVersion=1.0
sonar.sources=C:/ProgramData/Jenkins/.jenkins/workspace/ks_exp7
sonar.host.url=http://localhost:9000
sonar.login=admin
sonar.password=kshitij24
```

Additional arguments ?



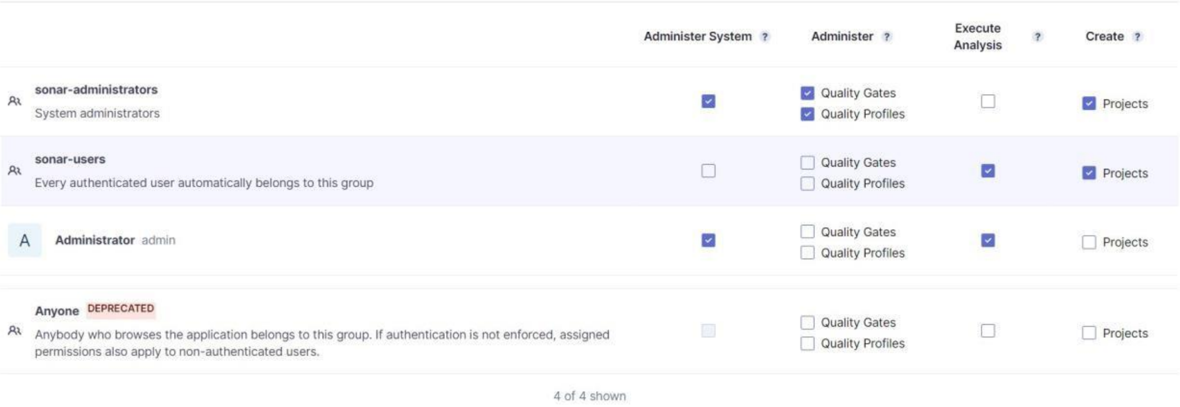
Permalinks

- Last build (#7), 4 min 55 sec ago
- Last stable build (#7), 4 min 55 sec ago
- Last successful build (#7), 4 min 55 sec ago
- Last failed build (#6), 17 min ago
- Last unsuccessful build (#6), 17 min ago
- Last completed build (#7), 4 min 55 sec ago



```
Started by user Kshitij Hundre
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\.jenkins\workspace\ks_exp7
The recommended git tool is: NONE
No credentials specified
 > git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\ks_exp7\.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 2.46.0.windows.1'
 > git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject.git +refs/heads/*:refs/remotes/origin/* # timeou
 > 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
[ks_exp7] $ C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube1_exp7\bin\sonar-scanner.bat -
Dsonar.host.url=http://localhost:9000 -Dsonar.projectKey=ks_exp7 -Dsonar.projectName=ks_exp7 -Dsonar.host.url=http://localhost:9000 -
Dsonar.login=admin -Dsonar.projectVersion=1.0 -Dsonar.sources=C:/ProgramData/Jenkins/.jenkins/workspace/ks_exp7 -Dsonar.password=kshitij24 -
Dsonar.projectBaseDir=C:\ProgramData\Jenkins\.jenkins\workspace\ks_exp7
15:09:08.473 WARN  Property 'sonar.host.url' with value 'http://localhost:9000' is overridden with value 'http://localhost:9000'
```

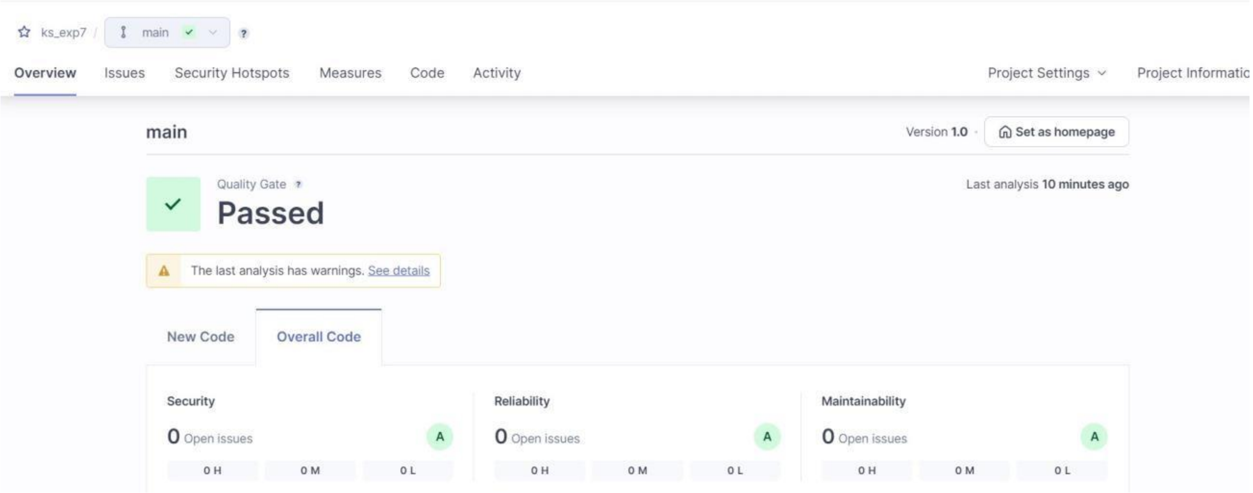
11. Go to http://localhost:9000/<user_name>/permissions and allow Execute Permissions to the Admin user.



	Administer System	Administer	Execute Analysis	Create
sonar-administrators System administrators	☑	☑ Quality Gates ☑ Quality Profiles	□	☑ Projects
sonar-users Every authenticated user automatically belongs to this group	□	□ Quality Gates □ Quality Profiles	☑	☑ Projects
A **Administrator** admin	☑	□ Quality Gates □ Quality Profiles	☑	□ Projects
Anyone DEPRECATED Anybody who browses the application belongs to this group. If authentication is not enforced, assigned permissions also apply to non-authenticated users.	□	□ Quality Gates □ Quality Profiles	□	□ Projects

4 of 4 shown

13. Once the build is complete, check project on SonarQube



main

Version **1.0** · Set as homepage

Quality Gate

Passed

Last analysis **10 minutes ago**

The last analysis has warnings. See details

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

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

Conclusion:

In this project, we integrated Jenkins with SonarQube for automated static application security testing (SAST). We set up SonarQube using Docker, configured Jenkins with the necessary plugins and authentication, and linked it to a GitHub repository. The SonarQube scanner was added as a build step, enabling continuous code analysis for vulnerabilities, code smells, and quality issues, ensuring automated reporting and continuous code quality improvement.