Name : Advik Ashok Hegde **Div/Roll No. : D15C/15**

Experiment - 7 :

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

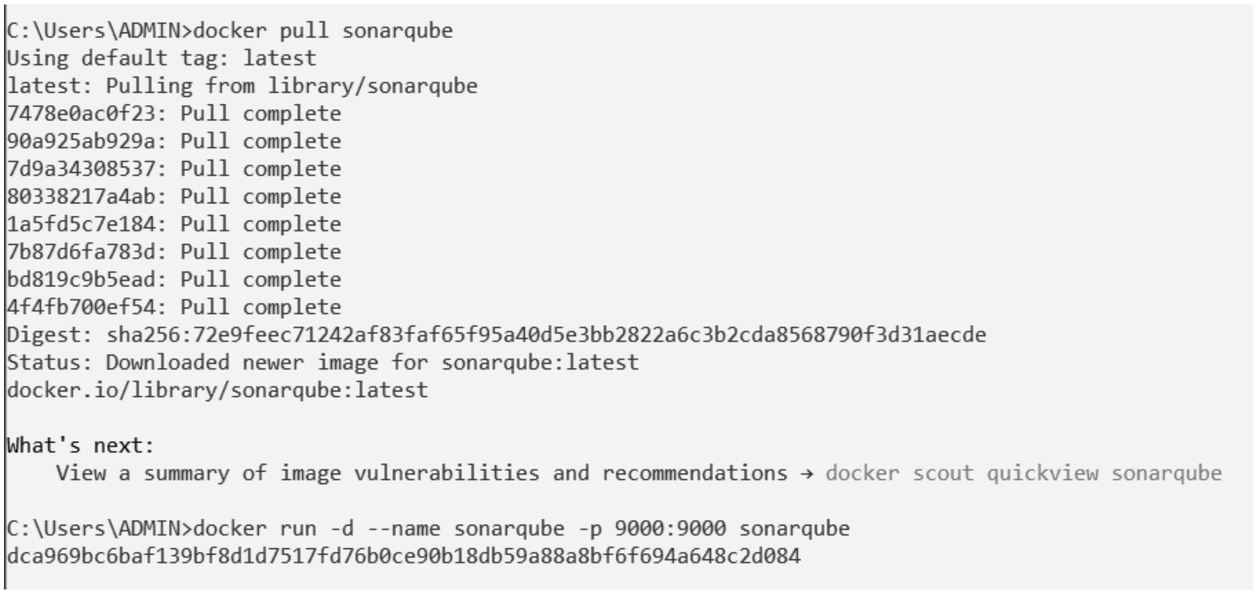
Integrating Jenkins with SonarQube (Prerequisites):

- **Jenkins installed :** To perform this experiment,it is necessary to have jenkins pre installed and up and running at some port like 8080.
- **Docker Installed (**for SonarQube) : It is also a requirement to have docker installed.

Steps to integrate Jenkins with SonarQube

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

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



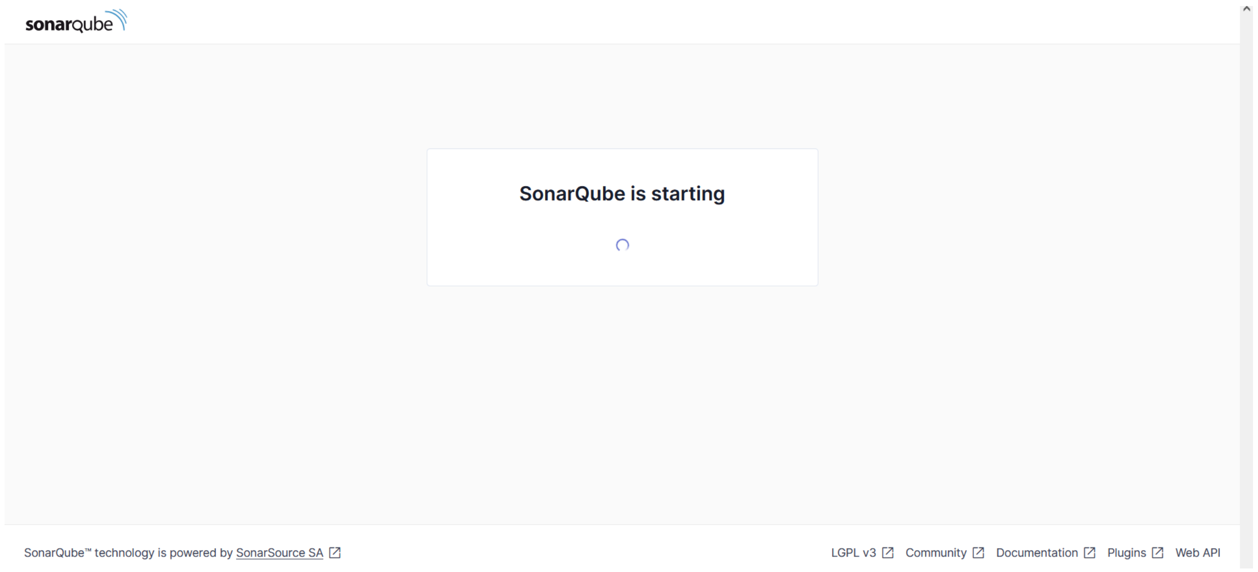
```
C:\Users\ADMIN>docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview sonarqube

C:\Users\ADMIN>docker run -d --name sonarqube -p 9000:9000 sonarqube
dca969bc6baf139bf8d1d7517fd76b0ce90b18db59a88a8bf6f694a648c2d084
```

Run the given command only once and replace the name ‘sonarqube’ with any name that you prefer.In my case I have not changed the name.

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

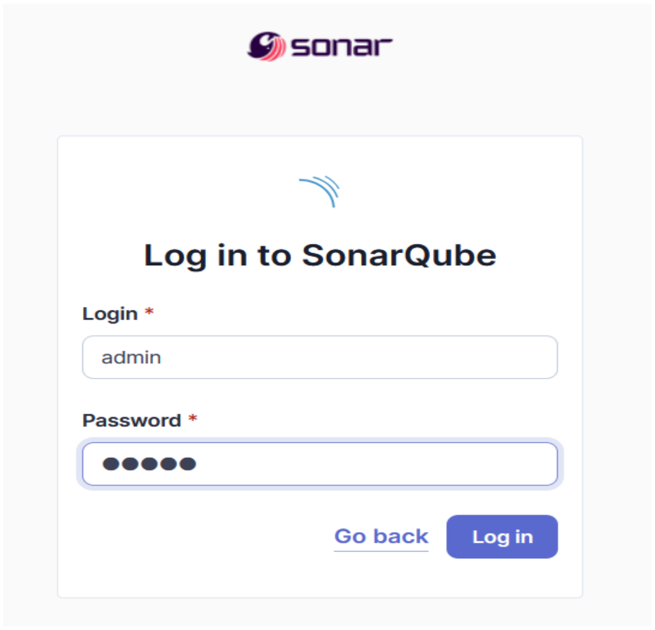
Note: Run the above command only once as once it is run,the sonarqube container is created.Running the same command again will result in an error as the container with that name will already have been created.

STEP: 3. Once the container is up and running,you can check the status ofSonarQube at localhost port 9000.

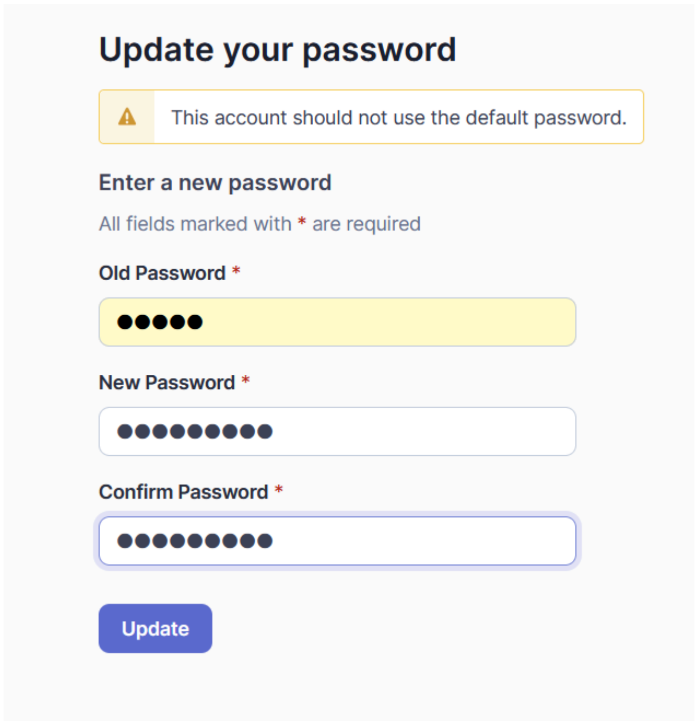


If you want to access this sonarqube dashboard again once you have closed this tab,you do not have to run the command again,rather,just open your docker and within it run the container that was created and then access the port 9000.

STEP: 4. Login to SonarQube using username *admin* and password *admin*.This is the default username and password for the first log-in into sonarqube.



After Logging in,you will be asked to reset the password,create a new personal password and store it for future use.



Update your password

This account should not use the default password.

Enter a new password

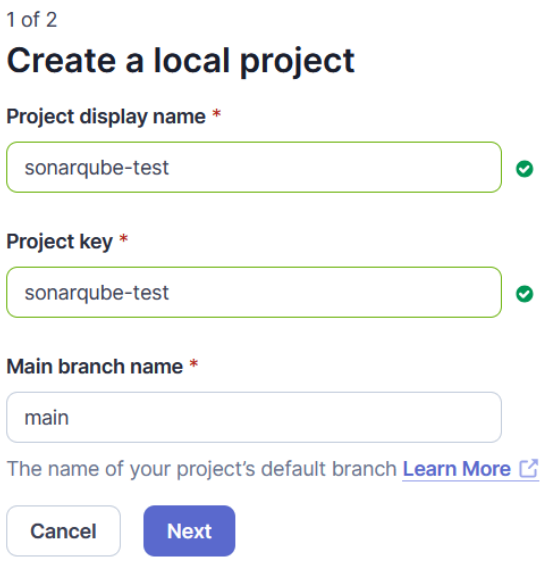
All fields marked with * are required

Old Password * ●●●●●

New Password * ●●●●●●●●●

Confirm Password * ●●●●●●●●●

STEP:5**.** Create a manual project in SonarQube with the name **sonarqube-test.** Again,you can set the name as per your wish,I have decided to name the project as sonarqube-test.



1 of 2

Create a local project

Project display name * sonarqube-test

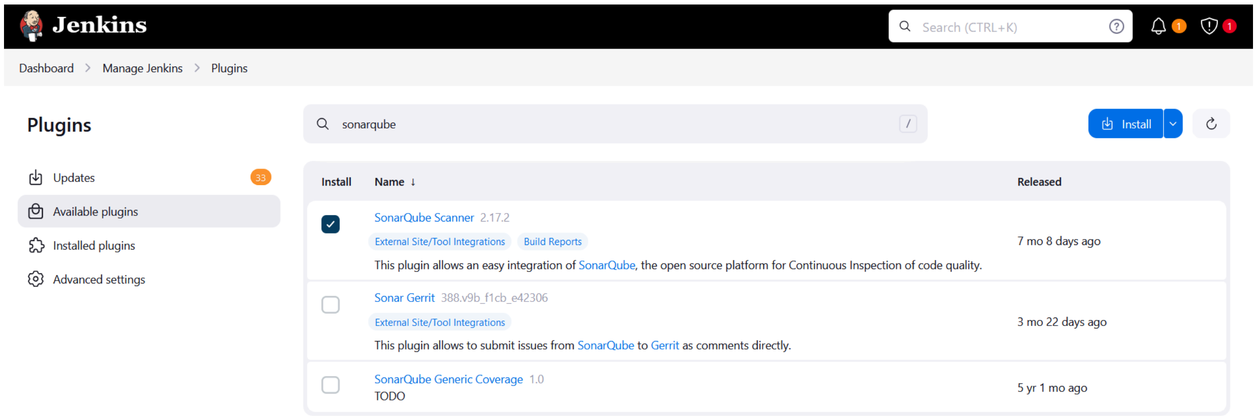
Project key * sonarqube-test

Main branch name * main

The name of your project's default branch Learn More

After completing the process of setting up the project,come back to the Jenkins Dashboard.

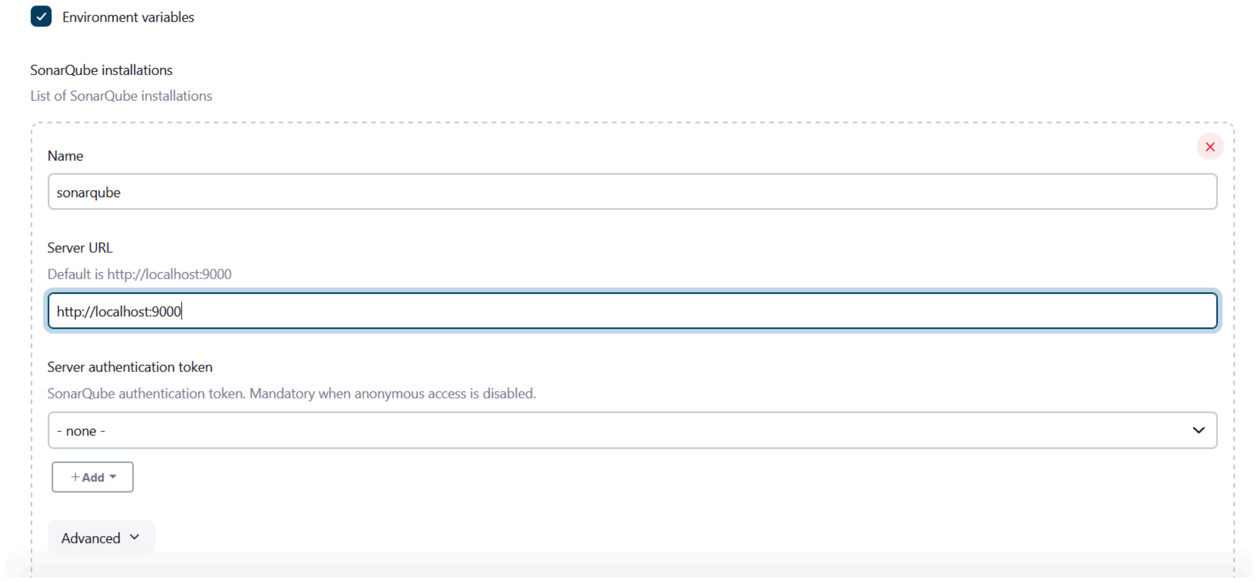
Go to Manage Jenkins and search for SonarQube Scanner for Jenkins.It is an available plugin that you can search for and then install.



sonarqube

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 Gerrit 388.v9b_f1cb_e42306 External Site/Tool Integrations This plugin allows to submit issues from SonarQube to Gerrit as comments directly.	3 mo 22 days ago
☐	SonarQube Generic Coverage 1.0 TODO	5 yr 1 mo ago

STEP : 6 . Under Jenkins ‘Configure System’, look for SonarQube Installations and enter the details. Provide the name for the project and enter the url based on the port that is used to run sonarqube which in my case is port **9000**.



✓ Environment variables

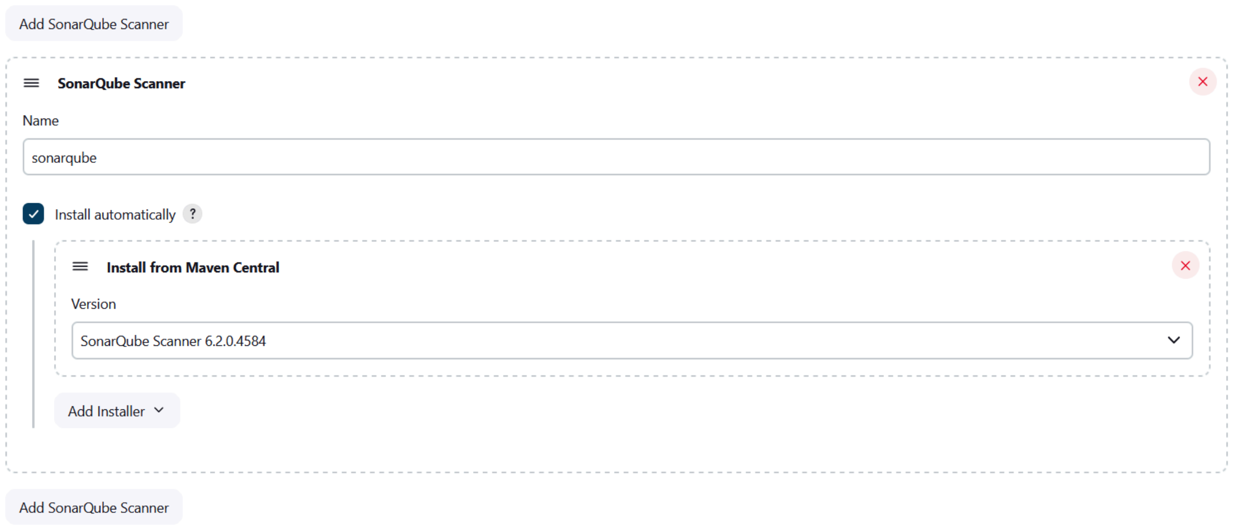
SonarQube installations
List of SonarQube installations

Name
sonarqube

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

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

STEP : 7 .Search for SonarQube Scanner under Global Tool Configuration.Choose the latest configuration and choose Install automatically.


Add SonarQube Scanner

SonarQube Scanner

Name

sonarqube

☑ Install automatically ?

Install from Maven Central

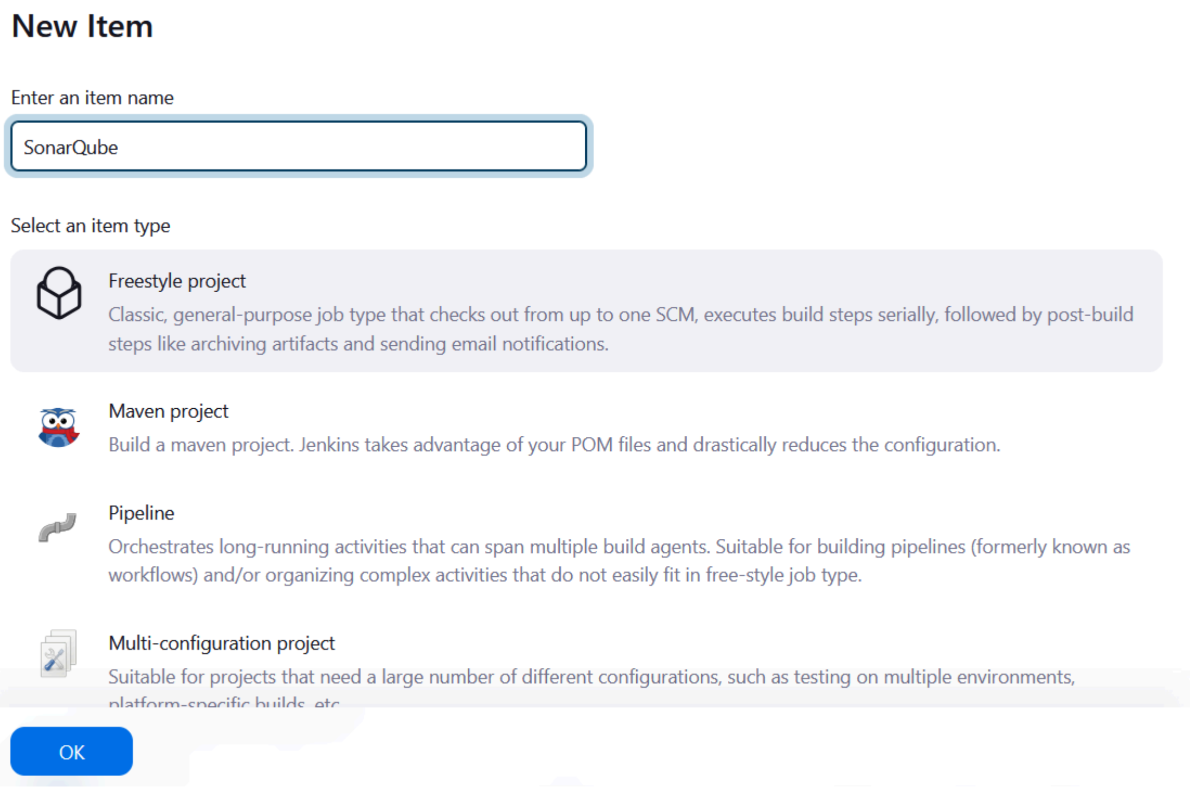
Version

SonarQube Scanner 6.2.0.4584

Add Installer

Add SonarQube Scanner

STEP : 8. 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.

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

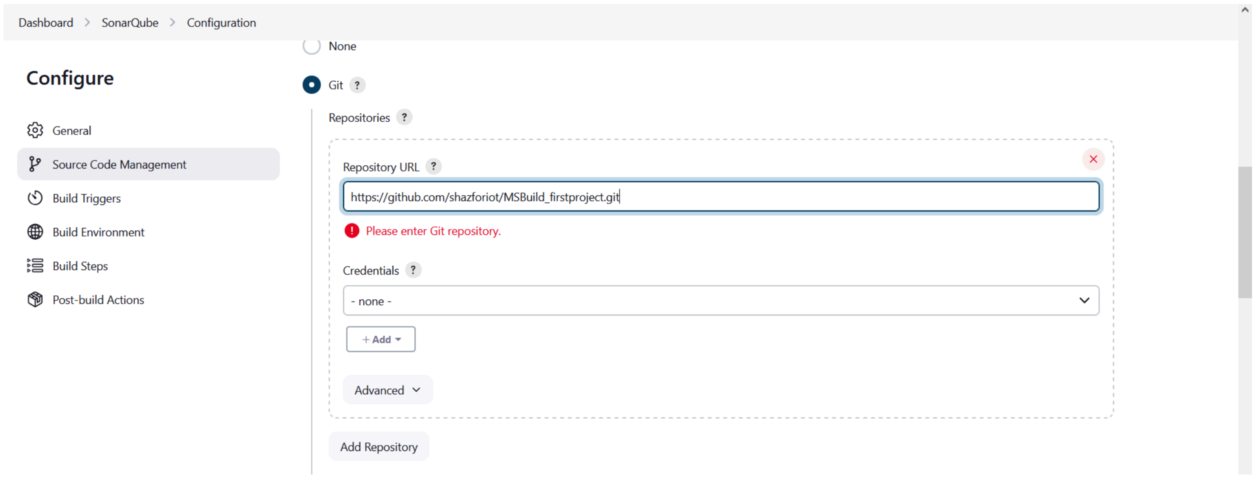
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

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

OK

STEP : 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.

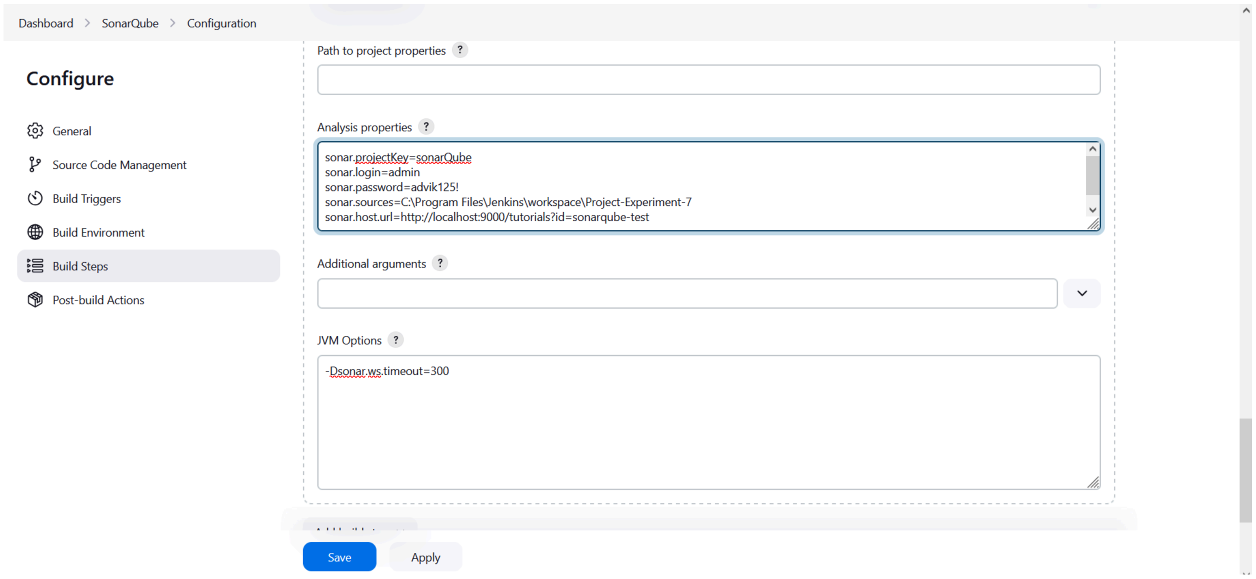


https://github.com/shazforiot/MSBuild_firstproject.git

Please enter Git repository.

STEP : 10. Under Build-> Execute SonarQube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.

- Project-Key:sonarQube
- Login:admin
- Password:advik125!
- Host URL : http://localhost9000/

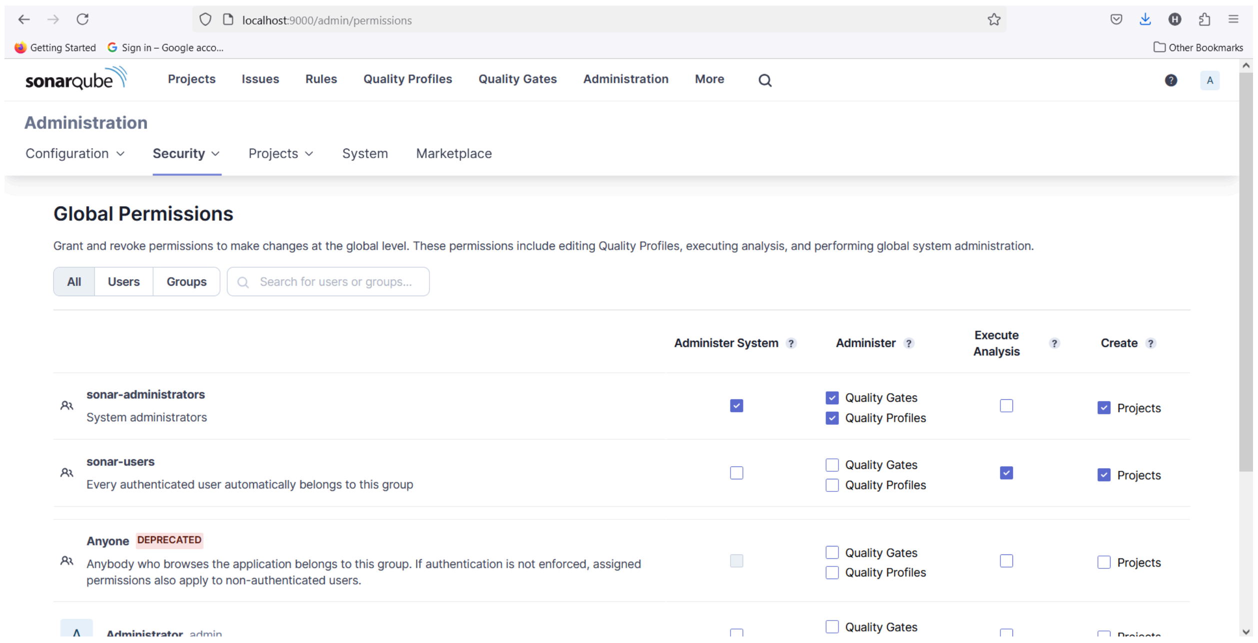


```
sonar.projectKey=sonarQube
sonar.login=admin
sonar.password=advik125!
sonar.sources=C:\Program Files\Jenkins\workspace\Project-Experiment-7
sonar.host.url=http://localhost:9000/tutorials?id=sonarqube-test
```

```
-Dsonar.ws.timeout=300
```

Note : Carefully enter the values for the Project Key,Login,Password,Source path and Host URL as any error will result in a **Failed Build Attempt.**

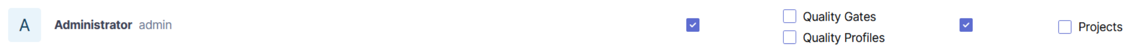
STEP : 11. Go to http://localhost:9000/<user_name>/permissions and allow Execute Permissions to the Admin user. In my case ,the user_name will be replaced with ‘admin’.



Global Permissions

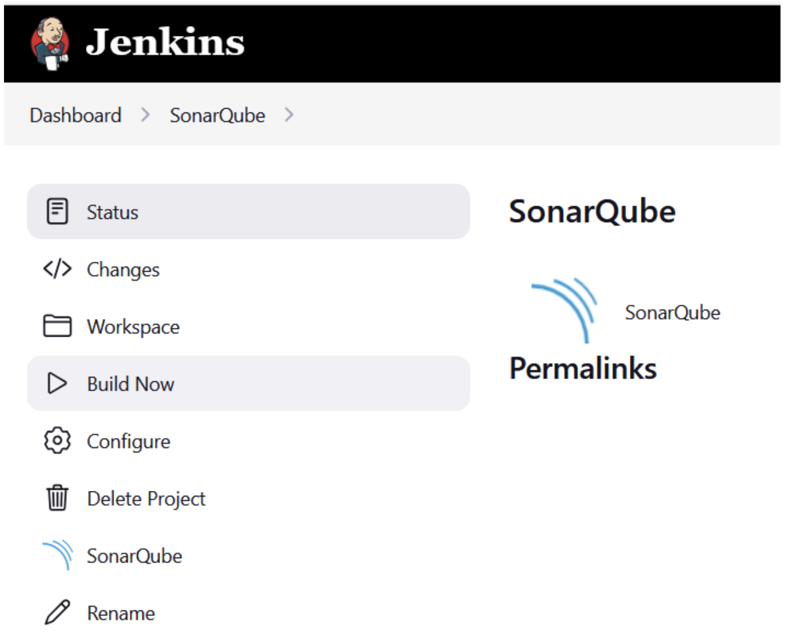
Grant and revoke permissions to make changes at the global level. These permissions include editing Quality Profiles, executing analysis, and performing global system administration.

	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
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
Administrator admin	☐	☐ Quality Gates	☐	☐

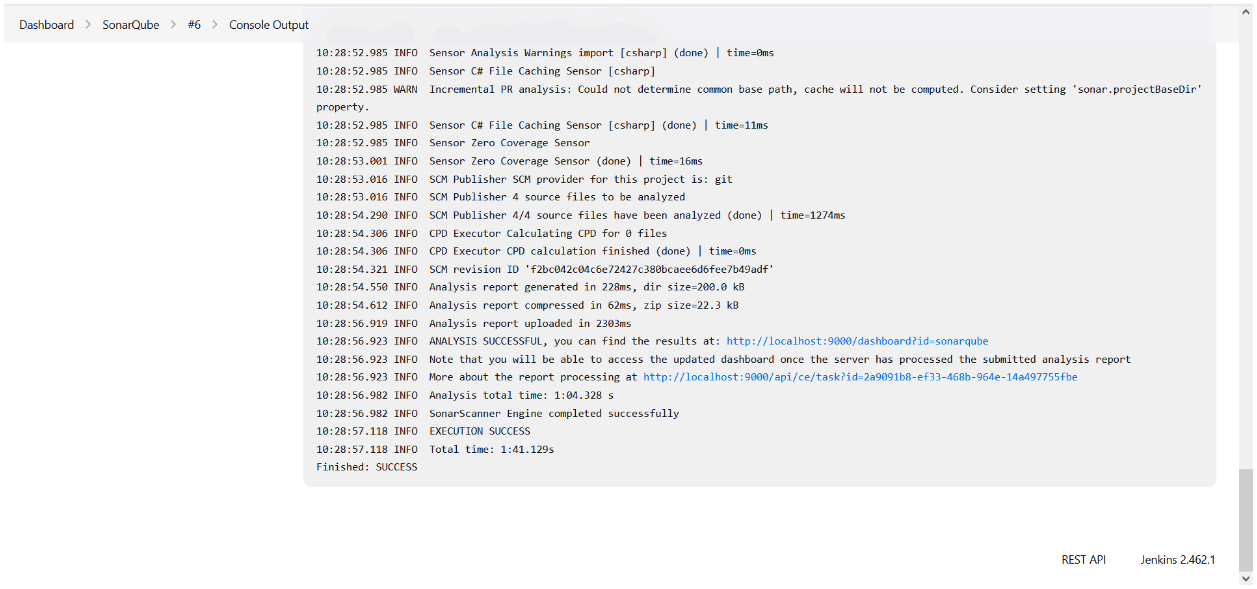


	Administer System	Administer	Execute Analysis	Create
A **Administrator** admin	✓	☐ Quality Gates ☐ Quality Profiles	✓	☐ Projects

STEP : 12. Run The Build.

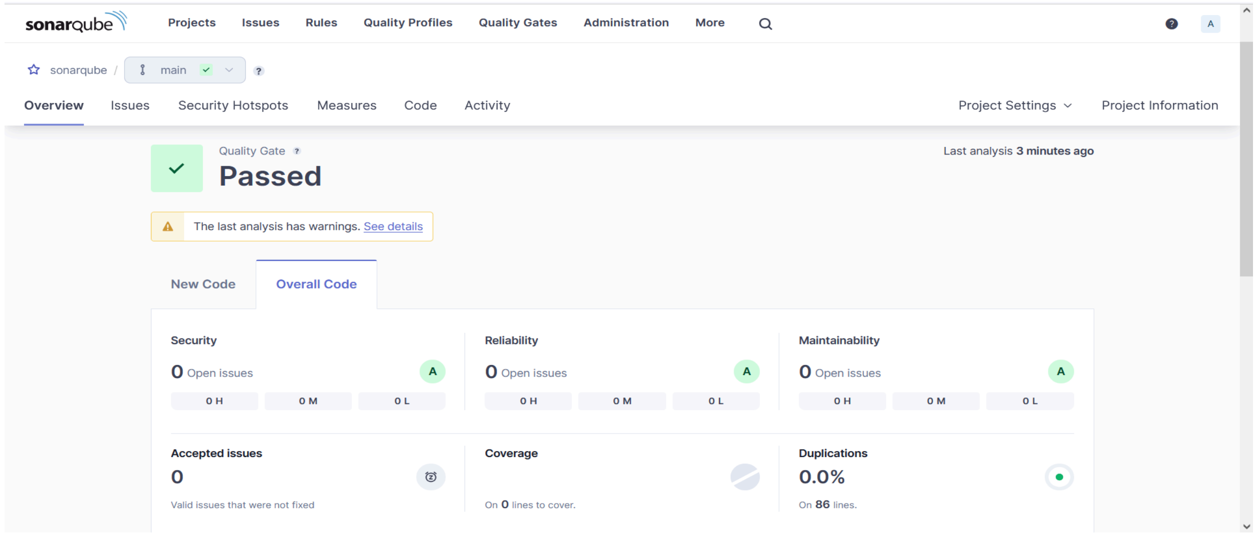


Once the build is complete,check the console output to determine whether the build was a success or a failure and identify the error in case of any errors.



```
10:28:52.985 INFO  Sensor Analysis Warnings import [csharp] (done) | time=0ms
10:28:52.985 INFO  Sensor C# File Caching Sensor [csharp]
10:28:52.985 WARN  Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir'
property.
10:28:52.985 INFO  Sensor C# File Caching Sensor [csharp] (done) | time=11ms
10:28:52.985 INFO  Sensor Zero Coverage Sensor
10:28:53.001 INFO  Sensor Zero Coverage Sensor (done) | time=16ms
10:28:53.016 INFO  SCM Publisher SCM provider for this project is: git
10:28:53.016 INFO  SCM Publisher 4 source files to be analyzed
10:28:54.290 INFO  SCM Publisher 4/4 source files have been analyzed (done) | time=1274ms
10:28:54.306 INFO  CPD Executor Calculating CPD for 0 files
10:28:54.306 INFO  CPD Executor CPD calculation finished (done) | time=0ms
10:28:54.321 INFO  SCM revision ID 'f2bc042c04c6e72427c380bcaee6d6fee7b49adf'
10:28:54.550 INFO  Analysis report generated in 228ms, dir size=200.0 kB
10:28:54.612 INFO  Analysis report compressed in 62ms, zip size=22.3 kB
10:28:56.919 INFO  Analysis report uploaded in 2303ms
10:28:56.923 INFO  ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube
10:28:56.923 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
10:28:56.923 INFO  More about the report processing at http://localhost:9000/api/ce/task?id=2a9091b8-ef33-468b-964e-14a497755fbe
10:28:56.982 INFO  Analysis total time: 1:04.328 s
10:28:56.982 INFO  SonarScanner Engine completed successfully
10:28:57.118 INFO  EXECUTION SUCCESS
10:28:57.118 INFO  Total time: 1:41.129s
Finished: SUCCESS
```

STEP : 13. Once the build is complete and successful, check the project in SonarQube.The overview of the project should display Passed.



Quality Gate
Passed
Last analysis 3 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
Accepted issues 0	**Coverage**	**Duplications** 0.0%
Valid issues that were not fixed	On 0 lines to cover.	On 86 lines.

Conclusion : In this experiment, we integrated Jenkins with SonarQube using Docker to automate code quality analysis. SonarQube, deployed via Docker, efficiently performed static code checks, while Jenkins orchestrated the process through a pipeline that triggered analysis after every code update. This setup ensured continuous monitoring of code quality, providing immediate feedback on potential issues such as bugs and code smells. Docker simplified the management of SonarQube, making the entire process more efficient. Overall, this integration streamlined the workflow and enhanced code quality through automated analysis.