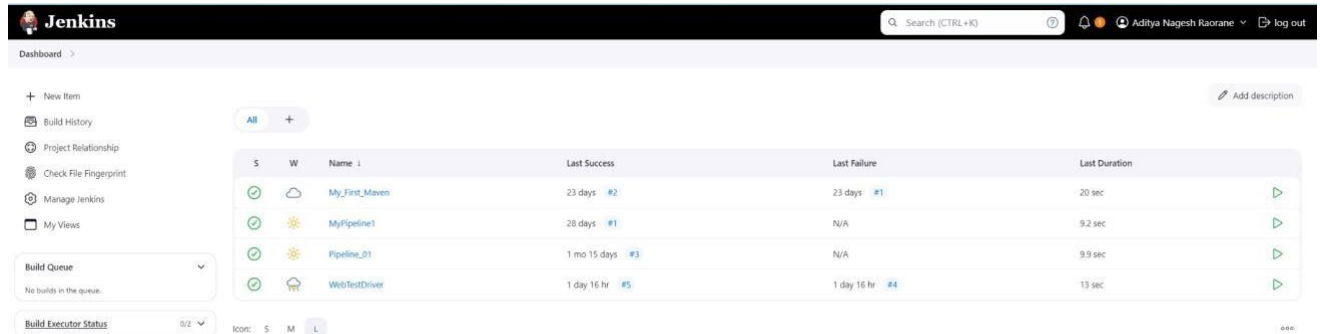


Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

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

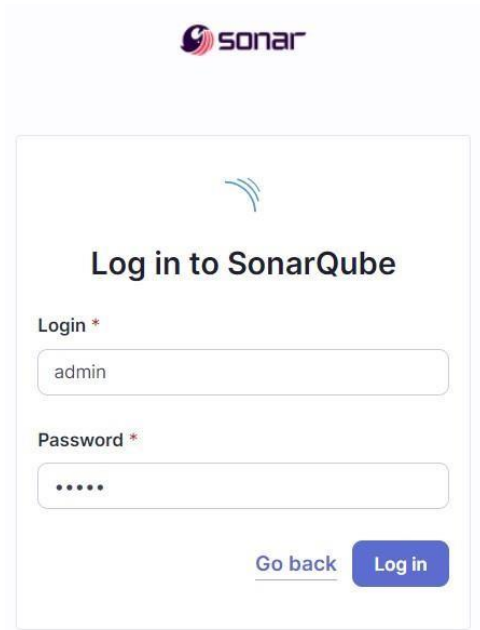


2. Run SonarQube in a Docker container using this command :a] docker -v b] docker pull sonarqube c] docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

```
C:\Users\Muskan>docker -v
Docker version 27.0.3, build 7d4bcd8

C:\Users\adity>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: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecd
Status: Downloaded newer image for sonarqube:latest
4a6e73f4472de892b1ddead1abe77372a85a7b09408cce3a0abd37c5ab6b49a4
```

3. Once the container is up and running, you can check the status of SonarQube at **localhost port 9000**. The login id is **“admin”** and the password is **mus12**



The image shows the SonarQube login interface. At the top is the Sonar logo. Below it is a loading spinner. The main heading is "Log in to SonarQube". There are two input fields: "Login *" with the value "admin" and "Password *" with masked characters. At the bottom are two buttons: "Go back" (a link) and "Log in" (a button).

4. Create a local project in SonarQube with the name sonarqube

1 of 2

Create a local project

Project display name *

sonarqube



Project key *

sonarqube



Main branch name *

main

The name of your project's default branch [Learn More](#)

Cancel

Next

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

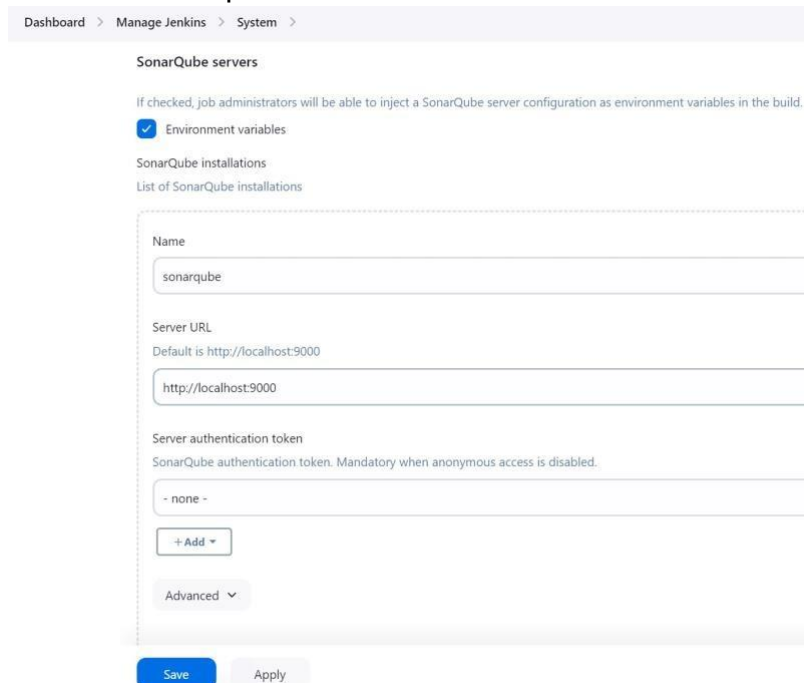
5. Setup the project and come back to Jenkins Dashboard. Go to **Manage Jenkins** → **Plugins** and search for **SonarQube Scanner** in **Available Plugins** and install it.



6. Under '**Manage Jenkins** → **System**', look for **SonarQube Servers** and enter these details.

Name : sonarqube

Server URL : http://localhost:9000



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

Manage Jeknins → Tools → SonarQube Scanner Installation

Dashboard > Manage Jenkins > Tools

SONARQUBE SCANNER FOR MSBUILD

Add SonarScanner for MSBuild

SonarQube Scanner installations

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

Ant installations

Add Ant

Save Apply

8. After the configuration, create a **New Item** in Jenkins, choose a **freestyle project** named **sonarqube**.

Jenkins

Search (CTRL+K)

Aditya Nagesh Raorane log out

Dashboard > All > New Item

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.
- Folder**
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder is a real container.

OK

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.

The screenshot shows the SonarQube Configuration page, specifically the 'Source Code Management' section. The left sidebar lists various configuration categories: General, Source Code Management (selected), Build Triggers, Build Environment, Build Steps, and Post-build Actions. The main content area is titled 'Source Code Management' and includes radio buttons for 'None' and 'Git' (selected). Below this, there is a 'Repositories' section with a text input for 'Repository URL' containing 'https://github.com/shazforiot/MSBuild_firstproject.git', a dropdown for 'Credentials' set to '- none -', and an '+ Add' button. An 'Advanced' dropdown is also present. At the bottom of this section is an 'Add Repository' button. Below the repositories section is a 'Branches to build' section with a text input and a 'Branches to build' button. At the very bottom are 'Save' and 'Apply' buttons.

10. Under **Build-> Execute SonarQube Scanner**, enter these **Analysis Properties**.

Mention the SonarQube Project Key, Login, Password, Source path and Host

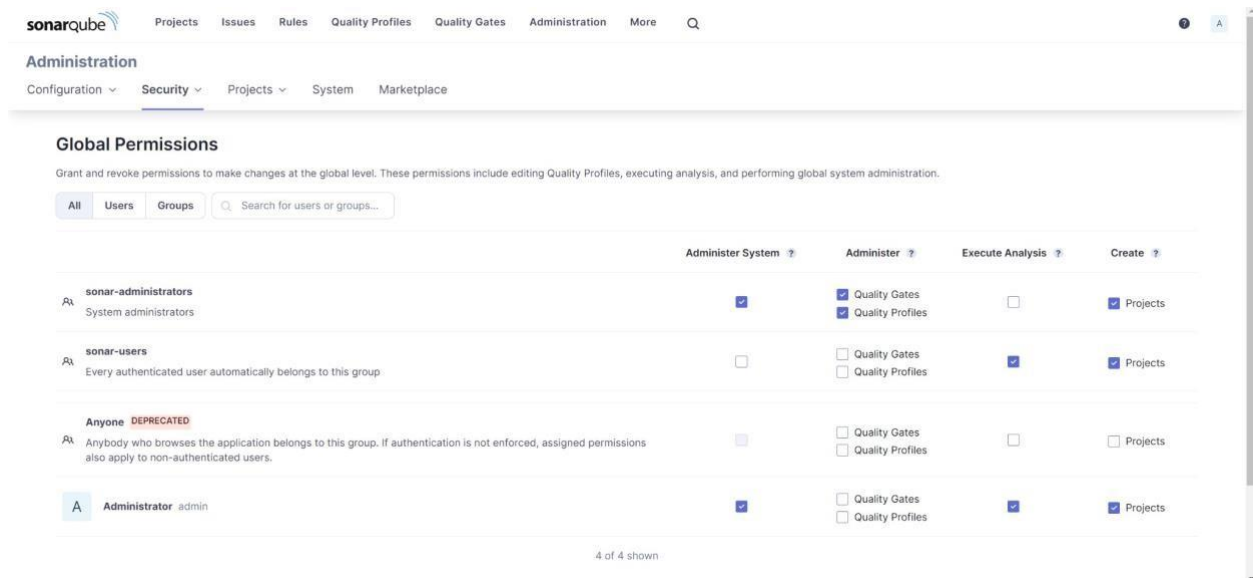
URL. sonar.projectKey=sonarqube sonar.login=admin sonar.password=aditya

sonar.sources=.

sonar.host.url=http://localhost:9000

The screenshot shows the SonarQube Configuration page, specifically the 'Execute SonarQube Scanner' section. The left sidebar is the same as the previous screenshot. The main content area is titled 'Execute SonarQube Scanner' and includes a 'JDK' dropdown set to '(Inherit From Job)', a 'Path to project properties' text input, and an 'Analysis properties' text input containing the following text: 'sonar.projectKey=sonarqube', 'sonar.login=admin', 'sonar.host.url=http://localhost:9000', and 'sonar.sources=.'. Below this is an 'Additional arguments' dropdown and a 'JVM Options' dropdown. At the bottom are 'Add build step', 'Save', and 'Apply' buttons.

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

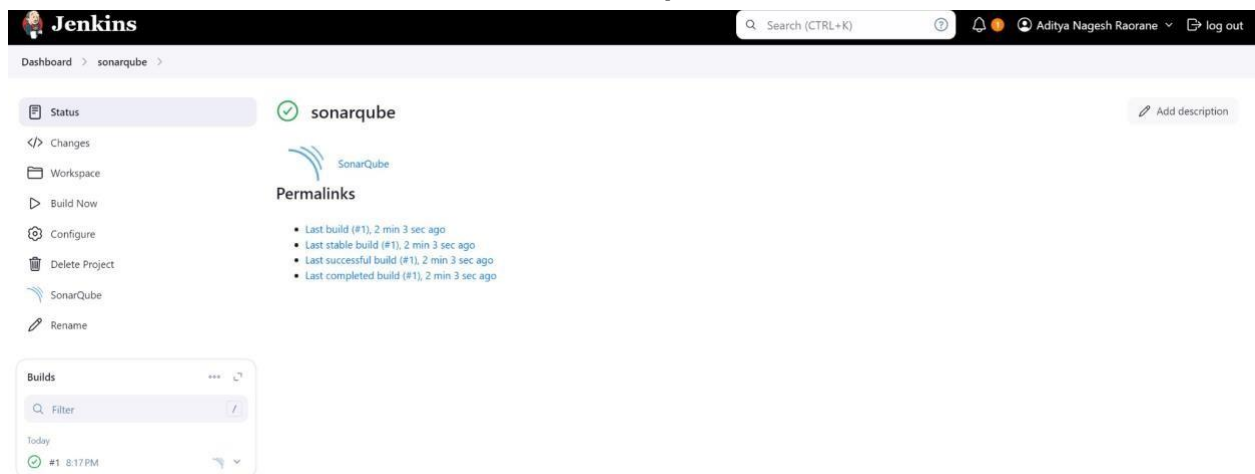


The screenshot shows the SonarQube Administration interface, specifically the 'Global Permissions' section under 'Security'. The page lists four user groups with their respective permissions. The 'Administrator' user is highlighted.

	Administer System ?	Administer ?	Execute Analysis ?	Create ?
sonar-administrators System administrators	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Quality Gates <input checked="" type="checkbox"/> Quality Profiles	<input type="checkbox"/>	<input checked="" type="checkbox"/> Projects
sonar-users Every authenticated user automatically belongs to this group	<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Projects
Anyone <small>DEPRECATED</small> Anybody who browses the application belongs to this group. If authentication is not enforced, assigned permissions also apply to non-authenticated users.	<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input type="checkbox"/>	<input type="checkbox"/> Projects
Administrator admin	<input checked="" type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Projects

4 of 4 shown

12. Run The **Build** and check the **console output**.



The screenshot shows the Jenkins Dashboard for a job named 'sonarqube'. The job is in a 'Status' state. The 'Builds' section shows a list of builds, with the most recent build (#1) completed successfully at 8:17 PM. The 'Permalinks' section provides links to the last build, last stable build, last successful build, and last completed build, all of which are the same build (#1).

Dashboard > sonarqube >

Status

Changes

Workspace

Build Now

Configure

Delete Project

SonarQube

Rename

sonarqube

Permalinks

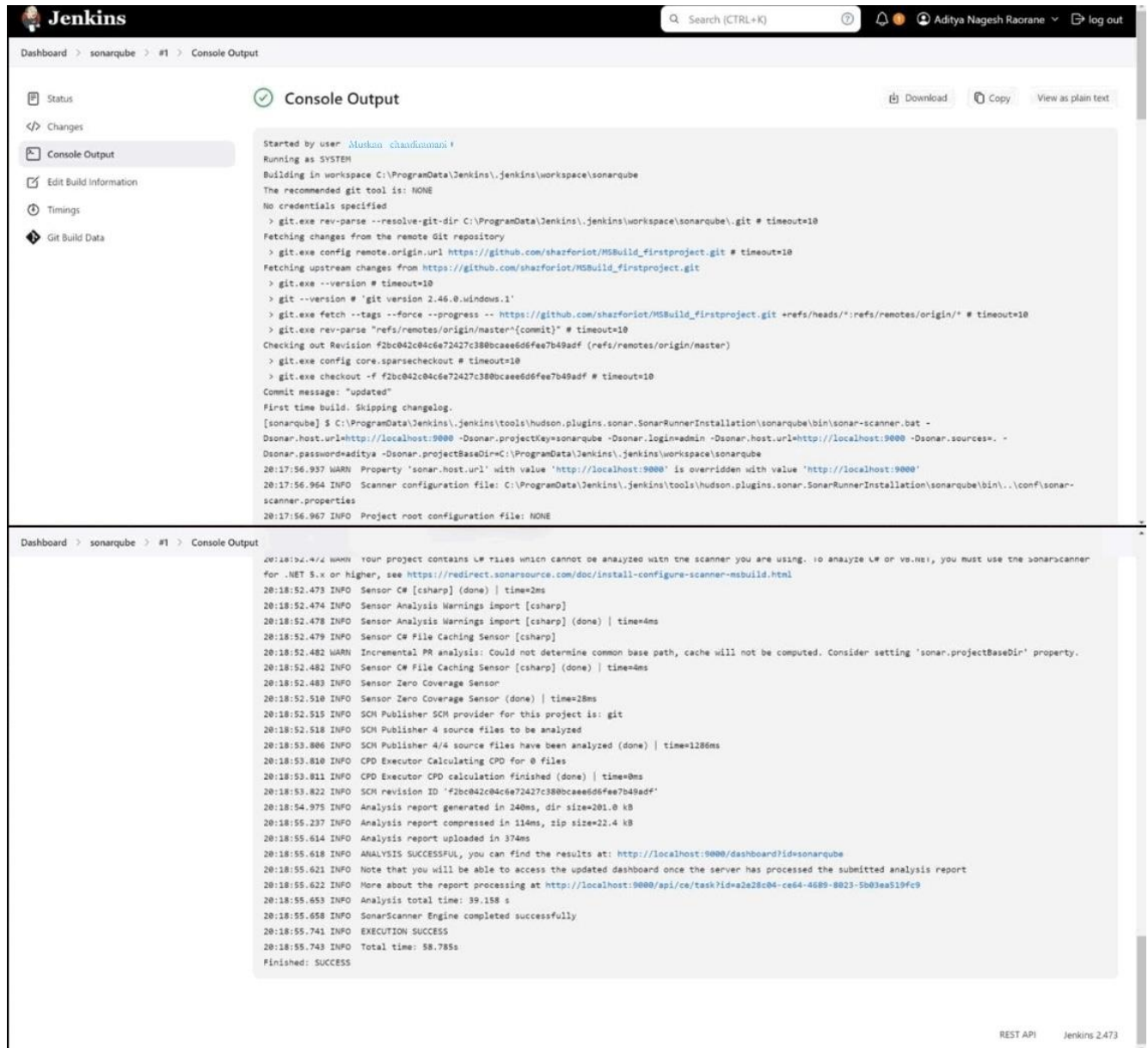
- Last build (#1), 2 min 3 sec ago
- Last stable build (#1), 2 min 3 sec ago
- Last successful build (#1), 2 min 3 sec ago
- Last completed build (#1), 2 min 3 sec ago

Builds

Filter

Today

#1 8:17 PM



The screenshot displays the Jenkins web interface. The top navigation bar includes the Jenkins logo, a search bar, and user information for Aditya Nagesh Raorane. The main content area is titled 'Console Output' and shows the execution of a build job named 'sonarqube'. The build log includes the following details:

- Started by user Muskan chandiramani
- Running as SYSTEM
- Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\sonarqube
- The recommended git tool is: NONE
- No credentials specified
- Git commands for cloning and fetching changes from the remote repository.
- Commit message: "updated"
- First time build. Skipping changelog.
- Execution of SonarScanner.bat with various properties.
- Warnings and information messages from the SonarScanner, including a warning about .NET 5.x and a note about incremental PR analysis.
- Successful completion of the SonarScanner engine.
- Final status: SUCCESS

The bottom of the console output shows a detailed log of the analysis process, including file caching, source file analysis, and the final report generation. The log ends with 'Finished: SUCCESS'.

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

The top screenshot shows the SonarQube 'Projects' page. It features a sidebar with filters for 'Quality Gate' (Passed: 1, Failed: 0) and 'Reliability' (A: 1). The main area displays a list of projects, with 'sonarqube PUBLIC' highlighted as 'Passed' and 'Last analysis: 3 minutes ago'. The bottom screenshot shows the 'main' branch overview for 'sonarqube PUBLIC'. It displays a 'Passed' status with a warning icon and the text 'The last analysis has warnings. See details'. Below this, there are tabs for 'New Code' and 'Overall Code'. The 'Overall Code' tab is active, showing various quality metrics: Security (0 Open issues, A), Reliability (0 Open issues, A), Maintainability (0 Open issues, A), Accepted issues (0, Valid issues that were not fixed), Coverage (0 lines to cover), and Duplications (0.0%, On 86 lines).

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

Conclusion:

In this experiment, we have understood the importance of SAST and have successfully integrated Jenkins with SonarQube for Static Analysis and Code Testing.