

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

Prerequisites:

1) Docker

Run docker -v command.

Use this command to check if docker is installed and running on your system.

```
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab7> docker -v
Docker version 27.0.3, build 7d4bcd8
```

2) Install SonarQube image

Command: docker pull sonarqube

This command helps you to install an image of SonarQube that can be used on the local system without actually installing the SonarQube installer.

```
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab7> docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
762bedf4b1b7: Pull complete
95f9bd9906fa: Pull complete
a32d681e6b99: Pull complete
aabdd0a18314: Pull complete
5161e45ecd8d: Pull complete
aeb0020dfa06: Pull complete
01548d361aea: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:bb44c58c1e04d8a147a3bb12af941c57e0100a5b21d10e599384d59bed36c86
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
```

3) Keep jenkins installed on your system.

Experiment Steps:

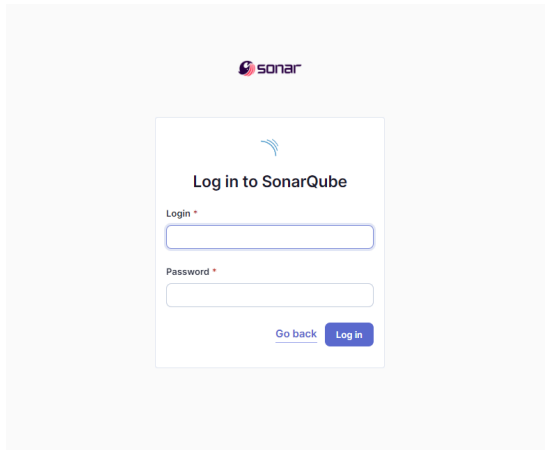
Step 1: Run SonarQube image

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

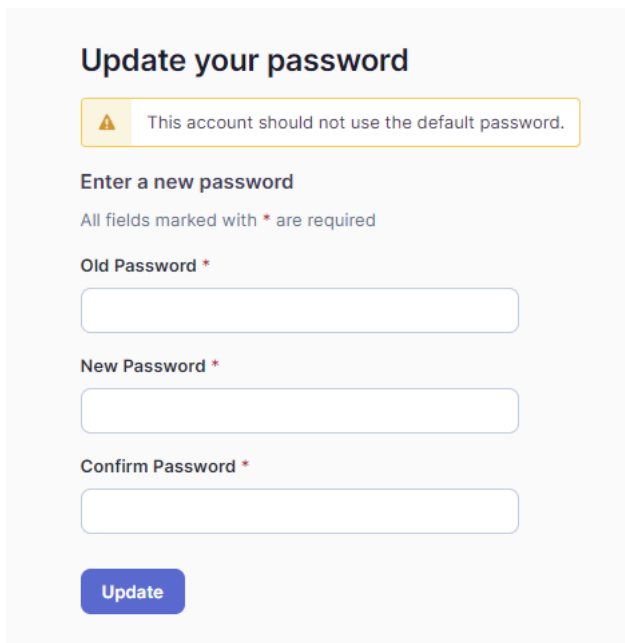
This command will run the SonarQube image that was just installed using docker.

```
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab7> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
36ff8a656bd28857ba9a28bf2bb0174099ae3232a9fc9ba2766d46f0c14d08a6
```

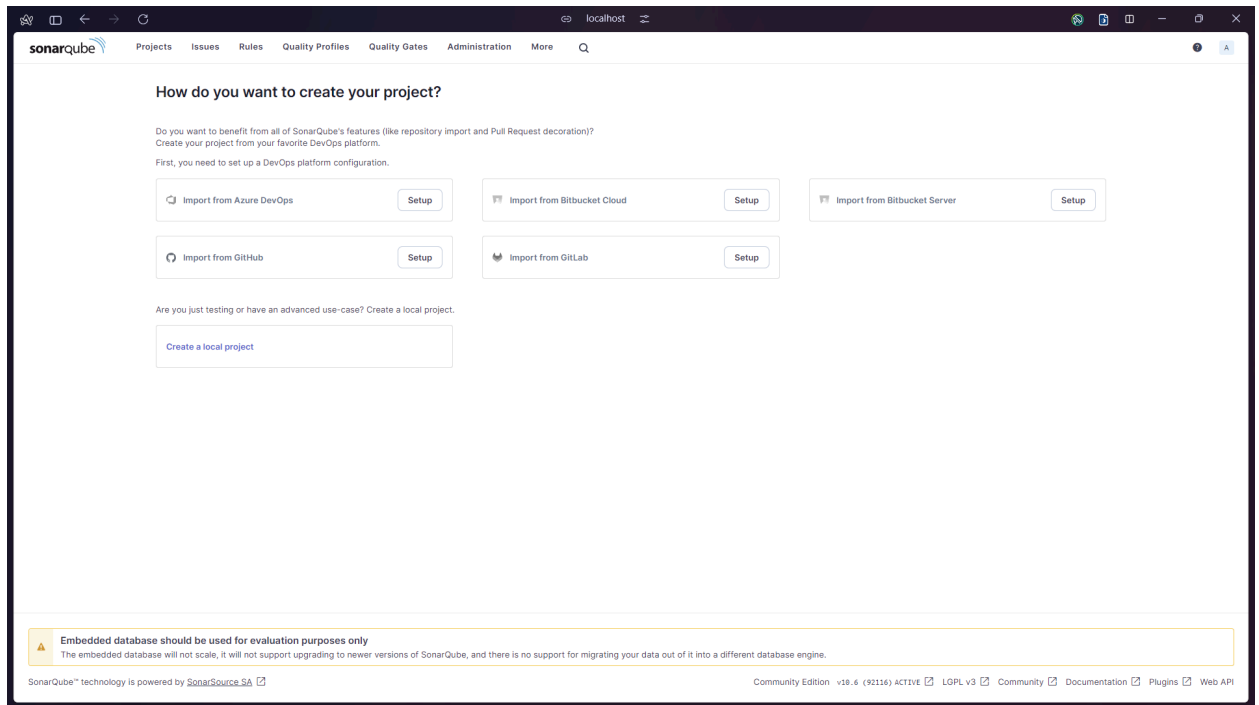
Step 2: Once the SonarQube image is started, you can go to <http://localhost:9000> to find the SonarQube that has started



Step 3: On this interface, login with username = 'admin' and password = 'admin'. Once logged in successfully, SonarQube will ask you to reset this password. Reset it and remember this password.

The image shows the "Update your password" form. At the top, there is a warning message: "This account should not use the default password." Below this, the text "Enter a new password" is displayed, followed by a note: "All fields marked with \* are required". There are three input fields: "Old Password \*", "New Password \*", and "Confirm Password \*". At the bottom, there is an "Update" button.

Step 4: After changing the password, you will be directed to this screen. Click on Create a Local Project.



Give the project a display name and project key

1 of 2

## Create a local project

Project display name \*



Project key \*



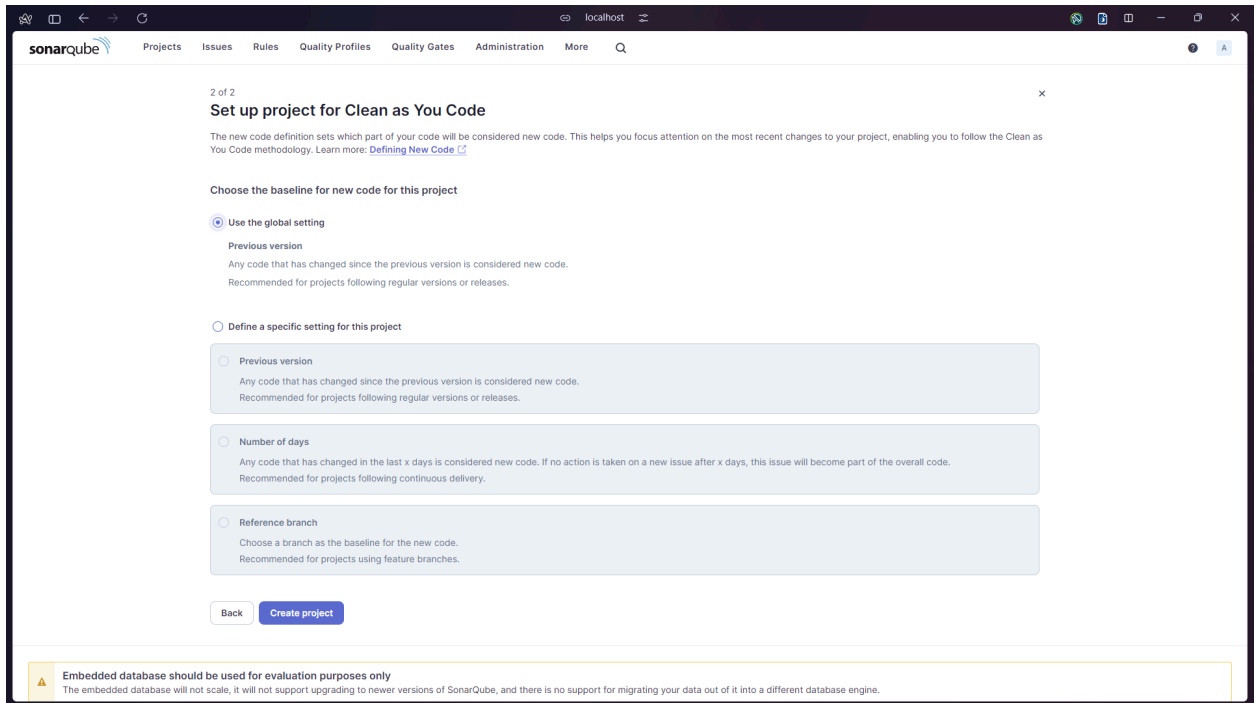
Main branch name \*

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

Cancel

Next

Set up the project as required and click on create.



The image shows the SonarQube web interface for setting up a project. The page title is "Set up project for Clean as You Code". It explains that the new code definition sets which part of the code will be considered new code. The user is prompted to "Choose the baseline for new code for this project". There are three radio button options: "Use the global setting", "Previous version", and "Define a specific setting for this project". The "Use the global setting" option is selected. Below it, there are three sub-options: "Previous version", "Number of days", and "Reference branch". The "Previous version" option is selected. At the bottom, there are "Back" and "Create project" buttons. A warning message at the bottom states: "Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine."

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.

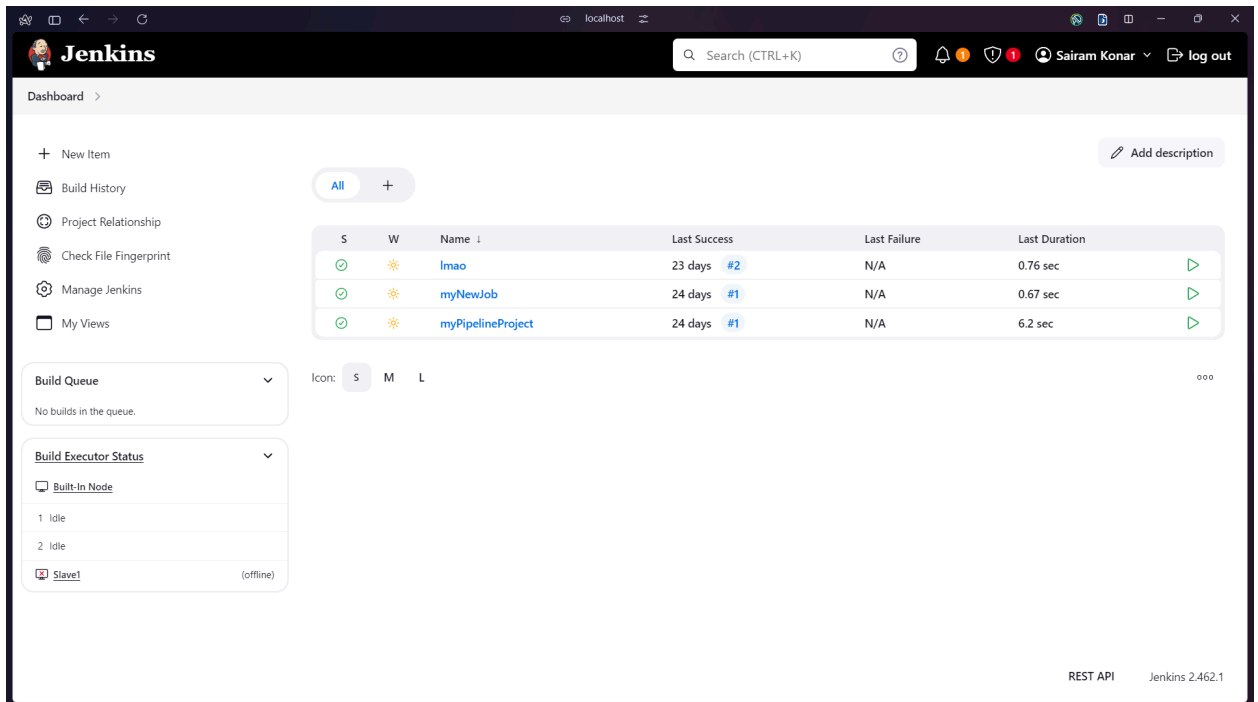
☐ Number of days  
Any code that has changed in the last x days is considered new code. If no action is taken on a new issue after x days, this issue will become part of the overall code.  
Recommended for projects following continuous delivery.

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

[Back](#) [Create project](#)

**Embedded database should be used for evaluation purposes only**  
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

Step 5: Open Jenkins on whichever port it is installed. ([http://localhost:<port\\_number>](http://localhost:<port_number>)).



The image shows the Jenkins Dashboard. The top navigation bar includes the Jenkins logo, a search bar, and user information (Sairam Konar). The main content area displays a table of builds. The table has columns for status (S), warning (W), name, last success, last failure, and last duration. There are three builds listed: "Imao", "myNewJob", and "myPipelineProject". The "Build Queue" section shows "No builds in the queue." The "Build Executor Status" section shows "1 idle", "2 idle", and "Slave1 (offline)".

**Jenkins**

Search (CTRL+K)

Sairam Konar log out

Dashboard

+ New Item

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Add description

S	W	Name	Last Success	Last Failure	Last Duration
✓	⚠	Imao	23 days #2	N/A	0.76 sec
✓	⚠	myNewJob	24 days #1	N/A	0.67 sec
✓	⚠	myPipelineProject	24 days #1	N/A	6.2 sec

Icon: S M L

**Build Queue**

No builds in the queue.

**Build Executor Status**

Built-In Node

1 idle

2 idle

Slave1 (offline)

REST API Jenkins 2.462.1

Step 6: Go to manage jenkins → Search for Sonarqube Scanner for Jenkins and install it.

The screenshot shows the Jenkins web interface. The top navigation bar includes the Jenkins logo, a search bar, and user information. The main content area is titled 'Plugins' and shows a search for 'sonarqube'. A table lists available plugins:

Install	Name	Released
<input type="checkbox"/>	<b>SonarQube Scanner</b> 2.17.2 <a href="#">External Site/Tool Integrations</a> <a href="#">Build Reports</a> This plugin allows an easy integration of <a href="#">SonarQube</a> , the open source platform for Continuous Inspection of code quality.	6 mo 29 days ago
<input type="checkbox"/>	<b>Sonar Gerrit</b> 388.v9b_f1cb_e42306 <a href="#">External Site/Tool Integrations</a> This plugin allows to submit issues from <a href="#">SonarQube</a> to <a href="#">Gerrit</a> as comments directly.	3 mo 13 days ago
<input type="checkbox"/>	<b>SonarQube Generic Coverage</b> 1.0 TODO	5 yr 1 mo ago

A red error banner at the bottom states: 'There were errors checking the update sites: UnknownHostException: updates.jenkins.io'. The bottom right corner shows 'REST API' and 'Jenkins 2.462.1'.

Step 7: Now, go to Manage Jenkins → System. Under Sonarqube servers, add a server. Add server authentication token if needed.

The screenshot shows the 'SonarQube servers' configuration page in Jenkins. It includes a checkbox for 'Environment variables' and a section for 'SonarQube installations'. The 'Add SonarQube' form contains the following fields:

- Name:** A text input field with a red error message: 'This property is mandatory.'
- Server URL:** A text input field with a default value of 'http://localhost:9000'.
- Server authentication token:** A dropdown menu currently set to 'none', with a note: 'SonarQube authentication token. Mandatory when anonymous access is disabled.'

At the bottom, there are 'Add SonarQube', 'Save', and 'Apply' buttons.

Step 8: Go to Manage Jenkins → Tools. Go to SonarQube scanner, choose the latest configuration and choose install automatically.

SonarQube Scanner installations

SonarQube Scanner installations ^ Edited

Add SonarQube Scanner

SonarQube Scanner

Name

sonarqube\_lab

☒ Install automatically ?

Install from Maven Central

Version

SonarQube Scanner 6.1.0.4477

Add Installer

Add SonarQube Scanner

Step 9: After configuration, create a New Item → choose a freestyle project.

Jenkins

Search (CTRL+K)

Sairam Konar log out

Dashboard > All > New Item

### New Item

Enter an item name

sonarqube-27

Select an item type

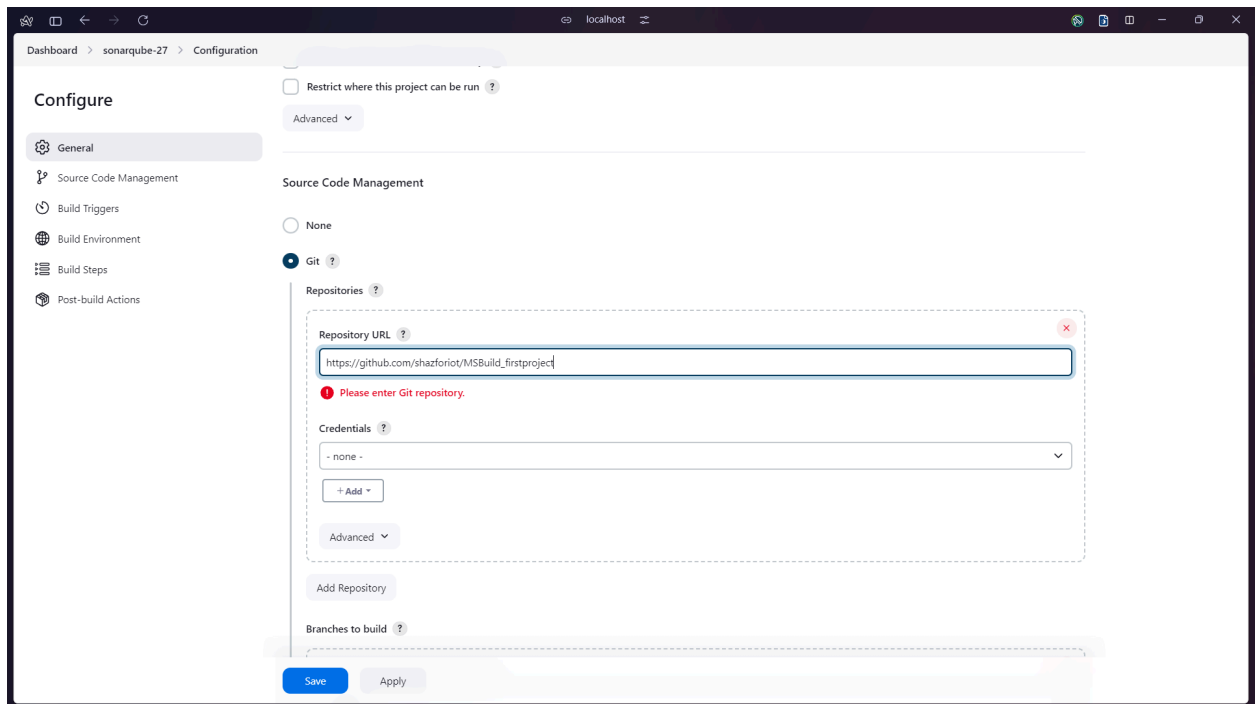
- Freestyle project**  
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
- Maven project**  
Build a maven project. Jenkins takes advantage of your POM files and 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

Step 10: Use this github repository in Source Code Management.

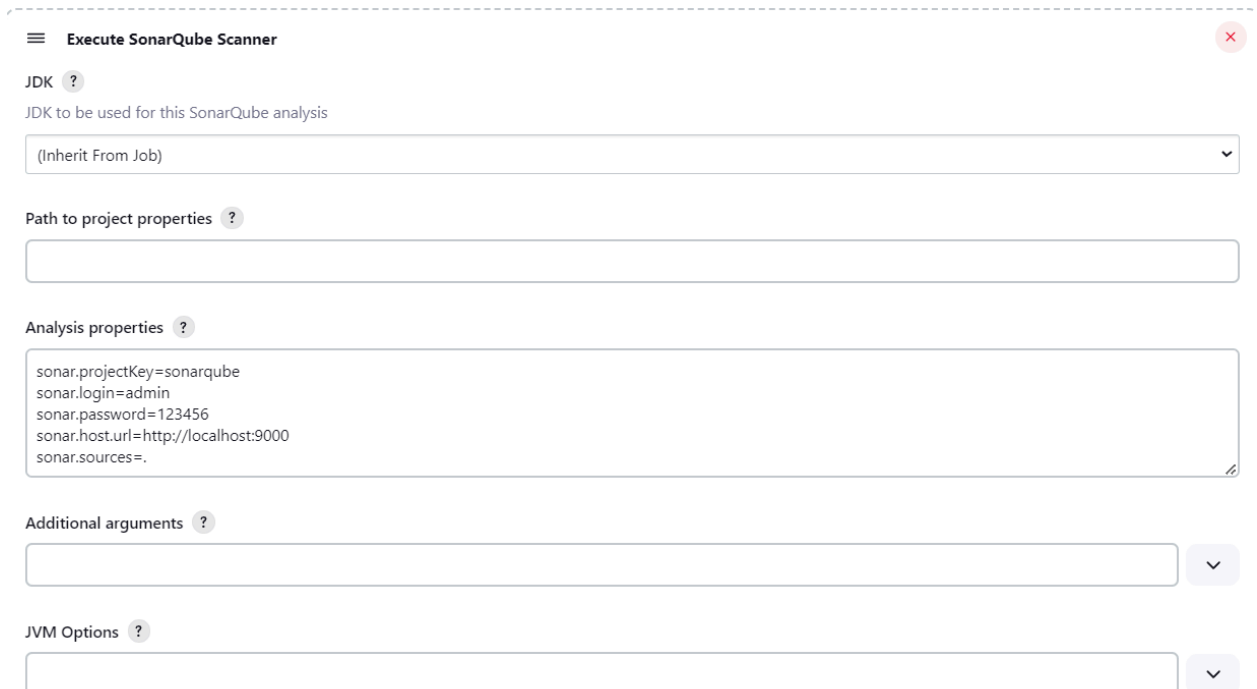
[https://github.com/shazforiot/MSBuild\\_firstproject](https://github.com/shazforiot/MSBuild_firstproject)

It is a sample hello-world project with no vulnerabilities.



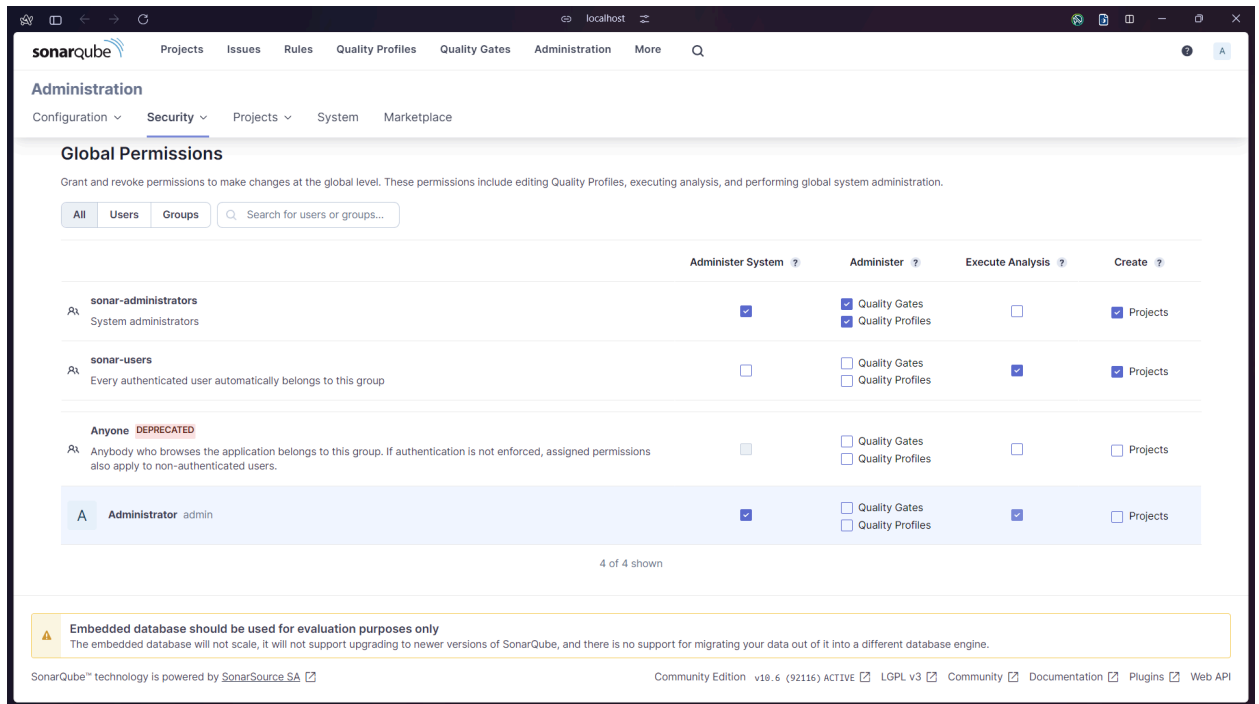
The screenshot shows the SonarQube Configuration page for a project named 'sonarqube-27'. The 'Source Code Management' section is active, showing the 'Git' option selected. The 'Repository URL' field contains 'https://github.com/shazforiot/MSBuild\_firstproject'. A red error message 'Please enter Git repository.' is displayed below the URL field. The 'Credentials' dropdown is set to 'none'. The 'Add Repository' button is visible. The left sidebar shows the 'Configure' menu with options: General, Source Code Management, Build Triggers, Build Environment, Build Steps, and Post-build Actions. The top navigation bar shows 'Dashboard > sonarqube-27 > Configuration'.

Step 11: Under Build Steps, enter Sonarqube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.



The screenshot shows the 'Execute SonarQube Scanner' build step configuration. The 'JDK' dropdown is set to '(Inherit From Job)'. The 'Path to project properties' field is empty. The 'Analysis properties' text area contains the following properties: sonar.projectKey=sonarqube, sonar.login=admin, sonar.password=123456, sonar.host.url=http://localhost:9000, and sonar.sources=. The 'Additional arguments' and 'JVM Options' fields are also empty. The top right corner has a red close button. The left sidebar shows the 'Execute SonarQube Scanner' step selected.

Step 12: Now, you need to grant the local user (here admin user) permissions to Execute the Analysis stage on SoanrQube. For this, go to [http://localhost:<port\\_number>/admin/permissions](http://localhost:<port_number>/admin/permissions) and check the 'Execute Analysis' checkbox under Administrator.









The screenshot shows the SonarQube Administration interface. The 'Global Permissions' section is active, displaying a table of users and their permissions. The 'Administrator' user is highlighted, and the 'Execute Analysis' checkbox is checked.

	Administer System ?	Administer ?	Execute Analysis ?	Create ?
<b>sonar-administrators</b> 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
<b>sonar-users</b> 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
<b>Anyone</b> <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
<b>Administrator</b> admin	<input checked="" type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input type="checkbox"/> Projects

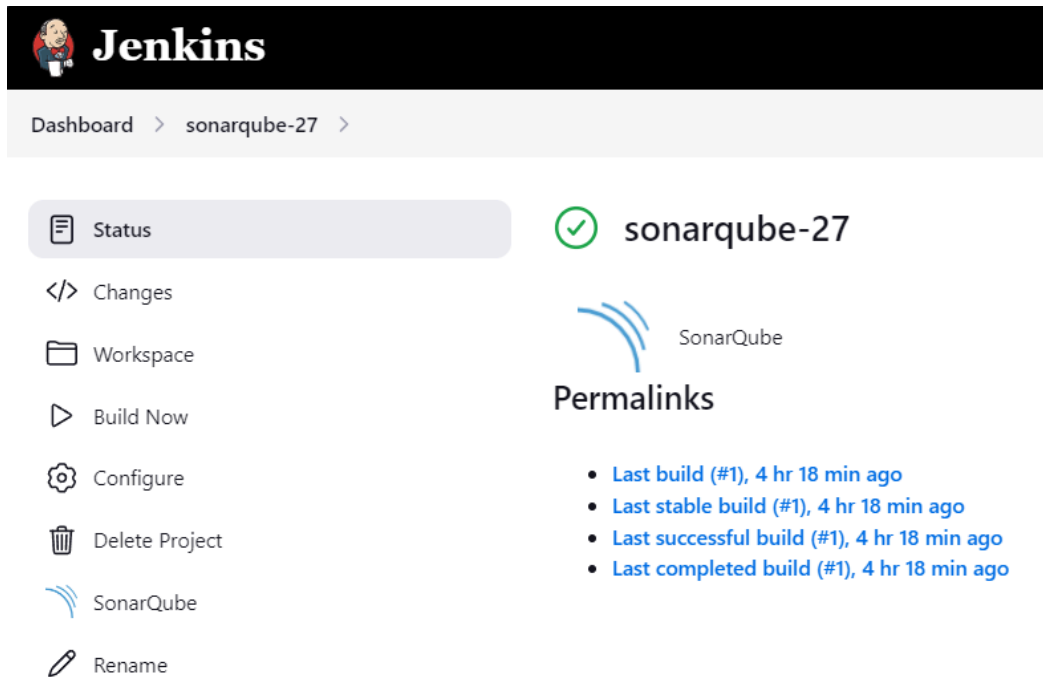
4 of 4 shown

Embedded database should be used for evaluation purposes only  
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

SonarQube™ technology is powered by [SonarSource SA](#) 

Community Edition v10.6 (92116) ACTIVE  LGPL v3  Community  Documentation  Plugins  Web API

Step 13: Go back to Jenkins. Go to the job you had just built and click on Build Now.



The screenshot shows the Jenkins dashboard for the 'sonarqube-27' job. The 'Build Now' button is highlighted in the left sidebar. The main area shows the job status as 'sonarqube-27' with a green checkmark icon. Below the job name, there is a 'Permalinks' section with a list of build links.

**Jenkins**

Dashboard > sonarqube-27 >

**Status**

- Changes
- Workspace
- Build Now**
- Configure
- Delete Project
- SonarQube
- Rename

**sonarqube-27**

**Permalinks**

- Last build (#1), 4 hr 18 min ago
- Last stable build (#1), 4 hr 18 min ago
- Last successful build (#1), 4 hr 18 min ago
- Last completed build (#1), 4 hr 18 min ago

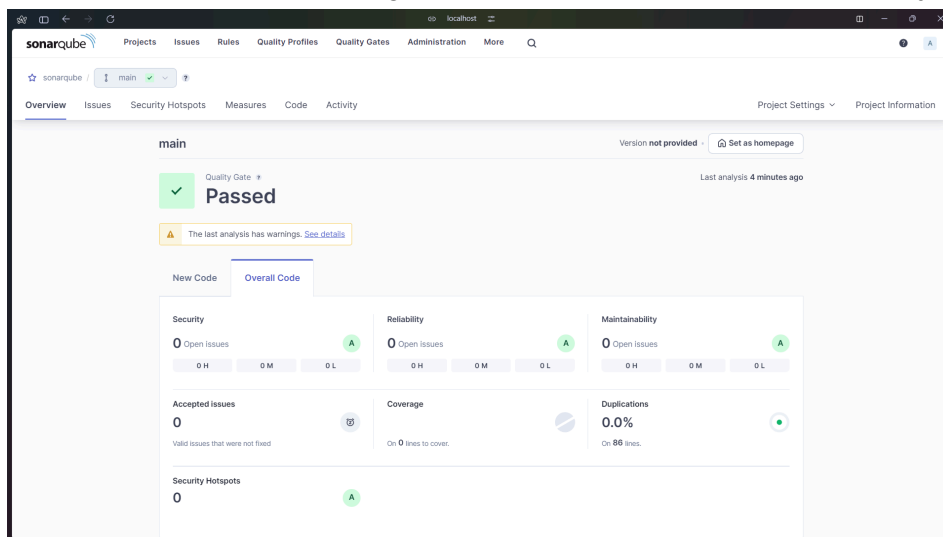


## Check the console Output

```
Started by user Sairam Konar
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube-27
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube-27\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject
> git.exe --version # timeout=10
> git --version # 'git version 2.45.2.windows.1'
> git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/master^(commit)" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcae6d6fee7b49adf (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f f2bc042c04c6e72427c380bcae6d6fee7b49adf # timeout=10
Commit message: "updated"
First time build. Skipping changelog.
[sonarqube-27] $ C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube_lab\bin\sonar-scanner.bat -Dsonar.projectKey=sonarqube -Dsonar.login=admin -Dsonar.host.url=http://localhost:9000 -Dsonar.sources=. -Dsonar.password=123456 -Dsonar.projectBaseDir=C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube-27
09:24:56.227 INFO Scanner configuration file: C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube_lab\bin\..\conf\sonar-scanner.properties
09:24:56.231 INFO Project root configuration file: NONE
09:24:56.240 INFO SonarScanner GIT 6.1.0.4437

09:25:27.830 WARN Your project contains C# files which cannot be analyzed with the scanner you are using. To analyze C# or VB.NET, you must use the SonarScanner for .NET 5.x or higher, see https://redirect.sonarsource.com/doc/install-configure-scanner-msbuild.html
09:25:27.831 INFO Sensor C# [csharp] (done) | time=1ms
09:25:27.831 INFO Sensor Analysis Warnings Import [csharp]
09:25:27.833 INFO Sensor Analysis Warnings Import [csharp] (done) | time=2ms
09:25:27.834 INFO Sensor C# File Caching Sensor [csharp]
09:25:27.835 WARN Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir' property.
09:25:27.835 INFO Sensor C# File Caching Sensor [csharp] (done) | time=1ms
09:25:27.835 INFO Sensor Zero Coverage Sensor
09:25:27.852 INFO Sensor Zero Coverage Sensor (done) | time=18ms
09:25:27.856 INFO SCM Publisher SCM provider for this project is: git
09:25:27.858 INFO SCM Publisher 4 source files to be analyzed
09:25:28.360 INFO SCM Publisher 4/4 source files have been analyzed (done) | time=501ms
09:25:28.363 INFO CPD Executor Calculating CPD for 0 files
09:25:28.363 INFO CPD Executor CPD calculation finished (done) | time=0ms
09:25:28.369 INFO SCM revision ID 'f2bc042c04c6e72427c380bcae6d6fee7b49adf'
09:25:28.697 INFO Analysis report generated in 119ms, dir size=201.0 kB
09:25:28.754 INFO Analysis report compressed in 43ms, zip size=22.4 kB
09:25:28.977 INFO Analysis report uploaded in 221ms
09:25:28.981 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube
09:25:28.981 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
09:25:28.981 INFO More about the report processing at http://localhost:9000/api/ce/task?id=288a1357-213b-46ce-aeab-eb463fb79292
09:25:28.996 INFO Analysis total time: 22.822 s
09:25:28.997 INFO SonarScanner Engine completed successfully
09:25:29.032 INFO EXECUTION SUCCESS
09:25:29.034 INFO Total time: 32.809s
Finished: SUCCESS
```

Step 14: Once the build is complete, go back to SonarQube and check the project linked.



**Conclusion:**

In this experiment, we have learned how to perform Jenkins SAST using SonarQube. For this, we used a docker image of SonarQube so as to not install it locally on our system. After installing the required configurations on Jenkins, using a coe from a gihub repository, we analyze its code using SonarQube. Once we build the project, we can see that SonarQube project displays that the code has no errors.