

## Experiment No :7

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

**PREREQUISITES:**

## 1) Docker :

Run **docker -v command**. We use this command to check if docker is installed and running on your system.

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

## 2) Install SonarQube Image:

The command **docker pull sonarqube** downloads a SonarQube image from Docker's online repository. This image lets you run SonarQube on your system using Docker without needing to install the full SonarQube software manually. It's like getting a ready-to-use version of SonarQube that can be started with Docker.

```
C:\Users\praja>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
```

3) Make sure **Jenkins** is already installed on your system before starting the process. Jenkins will be used to automate tasks, like running SonarQube for code analysis. If Jenkins isn't installed yet, you can download and set it up from the official Jenkins website.

**STEPS:**

**Step1:** The command `docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest` starts SonarQube in the background on port 9000 using Docker, allowing you to access it at <http://localhost:9000>

```
C:\Users\praja>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
650354d0f868ae4ad2d800426080076c604eb09f29b10d4a251aee70f51ce907
C:\Users\praja>
```

Containers

[Give feedback](#)

Container CPU usage ⓘ

282.96% / 800% (8 CPUs available)

Container memory usage ⓘ

1.69GB / 7.41GB

Show charts

Only show running containers

<input type="checkbox"/>	Name	Image	Status	Port(s)	CPU (%)	Last started	Actions
<input type="checkbox"/>	<div><div></div><div><a href="#">welcome-to-docker</a> 0527c31c4bd1 </div></div>	<a href="#">docker/welcome-to-docker:latest</a>	Exited (255)	8088:80	0%	1 month ago	<div><div></div><div></div><div></div></div>
<input type="checkbox"/>	<div><div></div><div><a href="#">sonarqube</a> 650354d0f868 </div></div>	<a href="#">sonarqube:latest</a>	Running	9000:9000	282.96%	52 seconds ago	<div><div><input type="checkbox"/></div><div></div><div></div></div>

**Step2:** After starting the SonarQube image, open your browser and go to <http://localhost:9000> to access SonarQube.

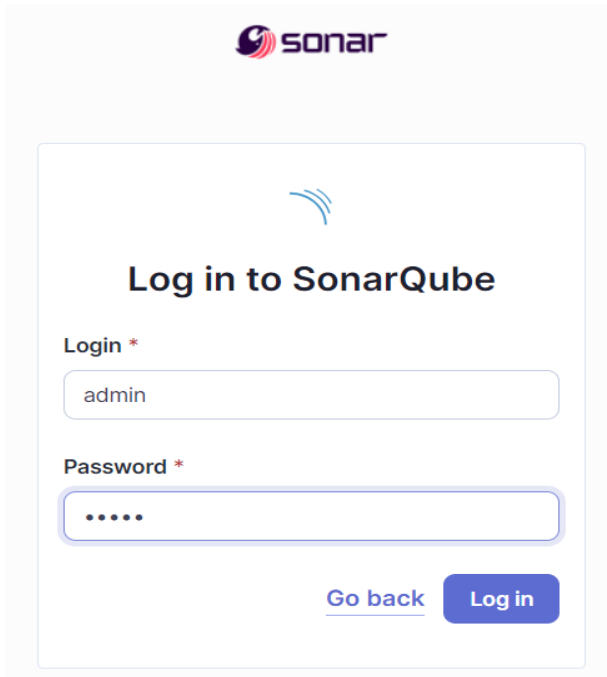
Log in to SonarQube

Login \*

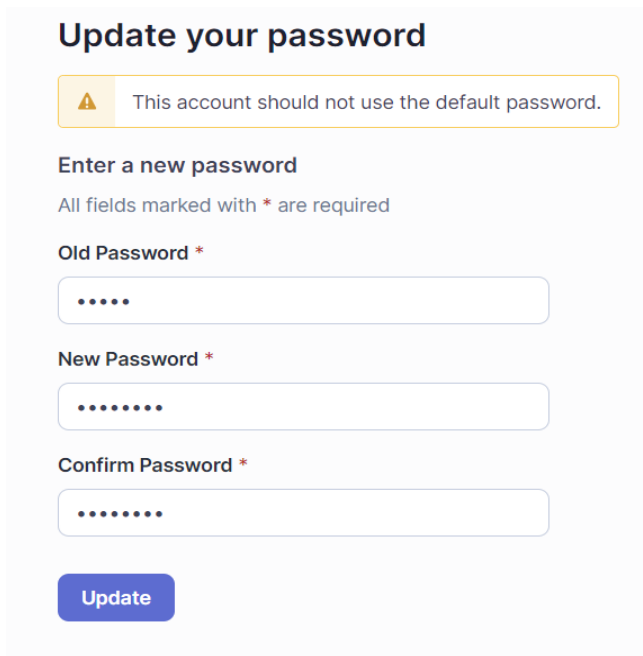
Password \*

[Go back](#)

**Step 3:** On the SonarQube login page, use the default credentials: **Username: admin** , **Password: admin**. After logging in, you'll be prompted to change the password. Set a new password and make sure to remember it.

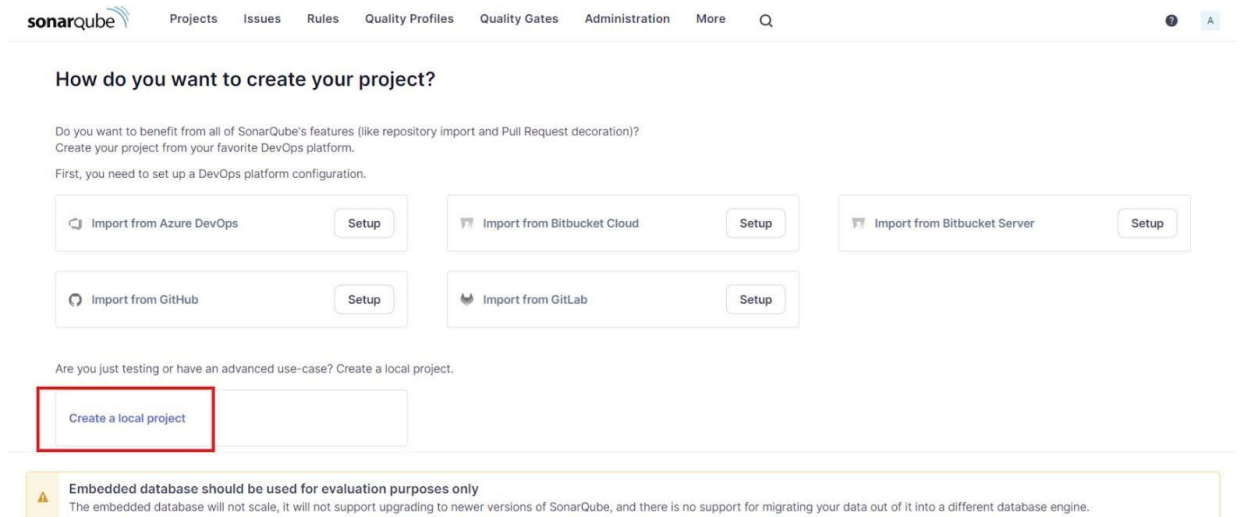
The image shows the SonarQube login page. At the top, there is the Sonar logo. Below it, the text "Log in to SonarQube" is displayed. There are two input fields: "Login \*" with the value "admin" and "Password \*" with five dots. At the bottom right, there are two buttons: "Go back" (a link) and "Log in" (a blue button).

*Click on Log in*

The image shows the SonarQube password update page. At the top, the text "Update your password" is displayed. Below it, there is a yellow warning box with a triangle icon and the text "This account should not use the default password." Underneath, the text "Enter a new password" is displayed, followed by the note "All fields marked with \* are required". There are three input fields: "Old Password \*" with five dots, "New Password \*" with seven dots, and "Confirm Password \*" with seven dots. At the bottom, there is a blue "Update" button.

*Click on Update .*

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



The image shows the SonarQube web interface for creating a new project. At the top, there is a navigation bar with the SonarQube logo and links for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and More. Below the navigation bar, the main heading is "How do you want to create your project?". The text below the heading asks if the user wants to benefit from all of SonarQube's features (like repository import and Pull Request decoration) and suggests creating a project from a favorite DevOps platform. It then lists five options: Import from Azure DevOps, Import from Bitbucket Cloud, Import from Bitbucket Server, Import from GitHub, and Import from GitLab. Each option has a "Setup" button. Below these options, there is a section for "Are you just testing or have an advanced use-case? Create a local project." which contains a "Create a local project" button. At the bottom, there is a warning message about the embedded database.

sonarqube

Projects Issues Rules Quality Profiles Quality Gates Administration More Q

### 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.

Create a local 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 6:** Configure the project by providing the necessary settings like choosing the baseline for the new code for the project , then click **Create** to finalize the setup.

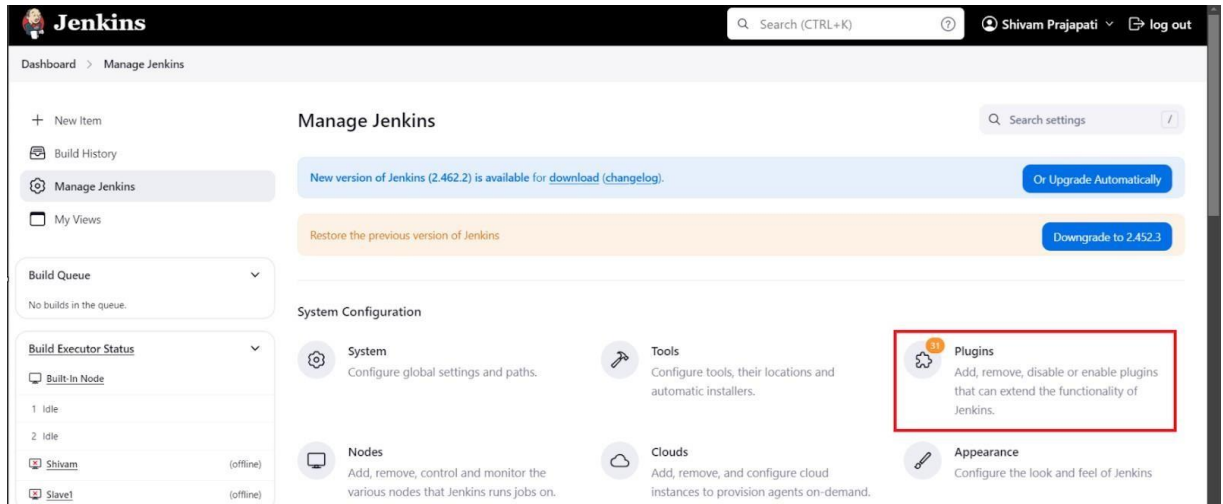
The screenshot shows the SonarQube interface with the 'Set up project for Clean as You Code' configuration page. The page title is 'Set up project for Clean as You Code' and it includes a sub-header 'Choose the baseline for new code for this project'. There are two main radio button options: 'Use the global setting' (selected) and 'Define a specific setting for this project'. Under 'Use the global setting', there are two sub-options: 'Previous version' (selected) and 'Number of days'. The 'Previous version' option is described as 'Any code that has changed since the previous version is considered new code. Recommended for projects following regular versions or releases.' The 'Number of days' option is described as '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.' The 'Define a specific setting for this project' option is also visible but not selected.

*Scroll Down*

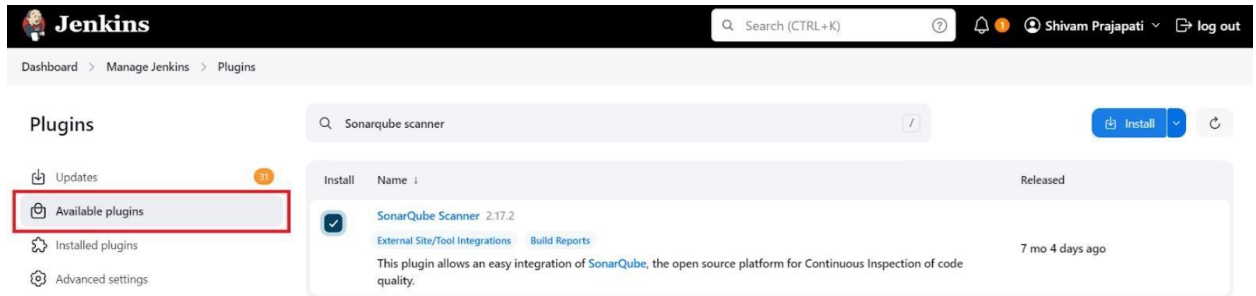
This screenshot shows the same SonarQube configuration page as the previous one, but scrolled down to reveal the 'Create project' button. The 'Create project' button is highlighted with a red rectangle. Below the configuration options, there is a 'Back' button and a 'Create project' button. At the bottom of the page, there is a yellow warning banner that reads: '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.'

*Click on Create project*

**Step 8:** Now go to Manage Jenkin then go for Plugins followed by Available plugins search for **Sonarqube Scanner** where we are going to install it as a plugin.



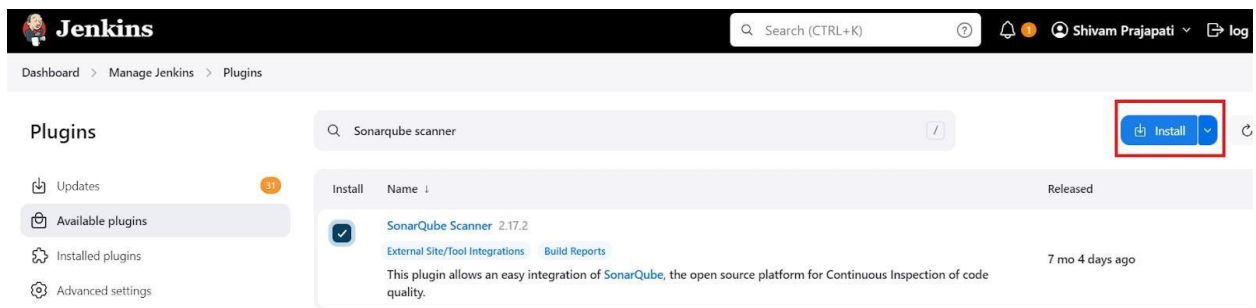
The screenshot shows the Jenkins web interface. At the top, the header includes the Jenkins logo, a search bar, and the user name 'Shivam Prajapati' with a 'log out' button. The main navigation bar on the left contains links for 'New Item', 'Build History', 'Manage Jenkins' (which is selected), and 'My Views'. Below this, there are sections for 'Build Queue' (showing no builds) and 'Build Executor Status' (showing two idle executors: 'Built-In Node' and 'Slave1'). The main content area is titled 'Manage Jenkins' and features a search bar for settings. It includes two notification banners: one for a new version of Jenkins (2.462.2) available for download, and another to restore the previous version. The 'System Configuration' section is visible, containing five sub-sections: 'System' (configure global settings), 'Tools' (configure tool locations), 'Plugins' (highlighted with a red box), 'Nodes' (add and monitor nodes), and 'Clouds' (add and configure cloud instances). The 'Plugins' section description states: 'Add, remove, disable or enable plugins that can extend the functionality of Jenkins.'



The screenshot shows the Jenkins 'Plugins' page. The left sidebar has a red box around 'Available plugins'. The main content area shows a search bar with 'Sonarqube scanner' and a table of installed plugins. The 'SonarQube Scanner' plugin is listed with version 2.17.2, released 7 months and 4 days ago. The 'Install' button in the top right is highlighted with a red box.

Install	Name	Released
<input checked="" type="checkbox"/>	SonarQube Scanner 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.	7 mo 4 days ago

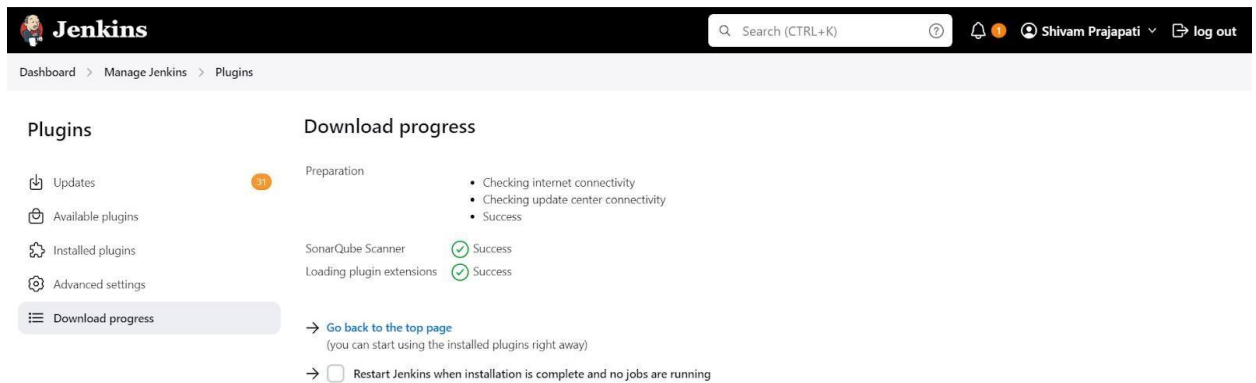
*Click on Available Plugins.*



The screenshot shows the Jenkins 'Plugins' page. The left sidebar has a red box around 'Available plugins'. The main content area shows a search bar with 'Sonarqube scanner' and a table of installed plugins. The 'SonarQube Scanner' plugin is listed with version 2.17.2, released 7 months and 4 days ago. The 'Install' button in the top right is highlighted with a red box.

Install	Name	Released
<input checked="" type="checkbox"/>	SonarQube Scanner 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.	7 mo 4 days ago

*Search in the Search bar the required Plugin Name and click on Install.*



The screenshot shows the Jenkins 'Download progress' page. The left sidebar has a red box around 'Download progress'. The main content area shows the progress of the 'SonarQube Scanner' plugin installation. The progress is 100% complete, with steps: Preparation, SonarQube Scanner, and Loading plugin extensions, all marked as 'Success'. The 'Restart Jenkins when installation is complete and no jobs are running' checkbox is unchecked.

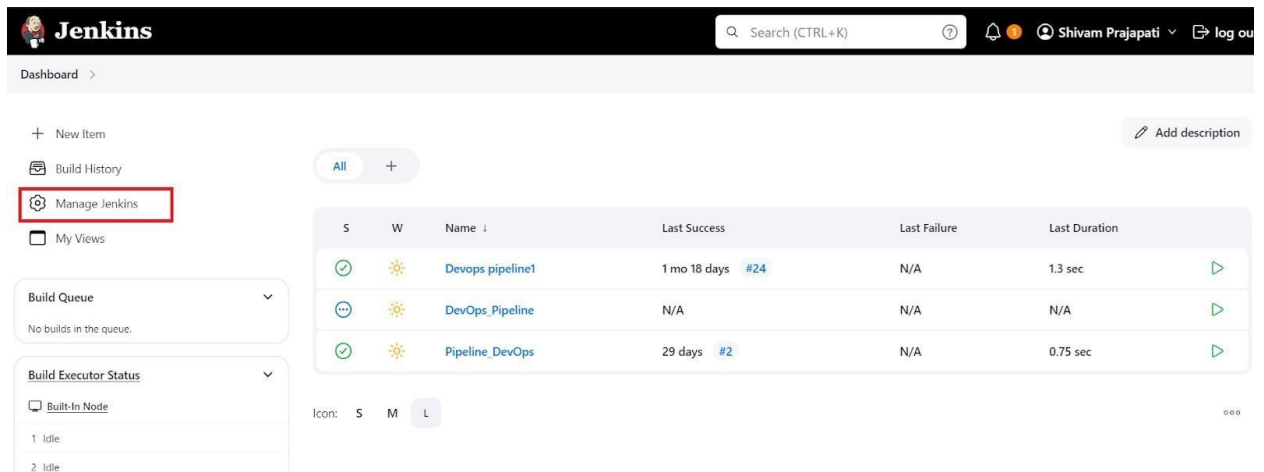
Step	Status
Preparation	Success
SonarQube Scanner	Success
Loading plugin extensions	Success

→ [Go back to the top page](#)  
(you can start using the installed plugins right away)

→ ☐ Restart Jenkins when installation is complete and no jobs are running

*Plugin Installed Successfully.*

**Step 9:** In Jenkins, go to **Manage Jenkins** → **System**, then find **SonarQube** servers. Add a new server, and if required, include the authentication token for secure access

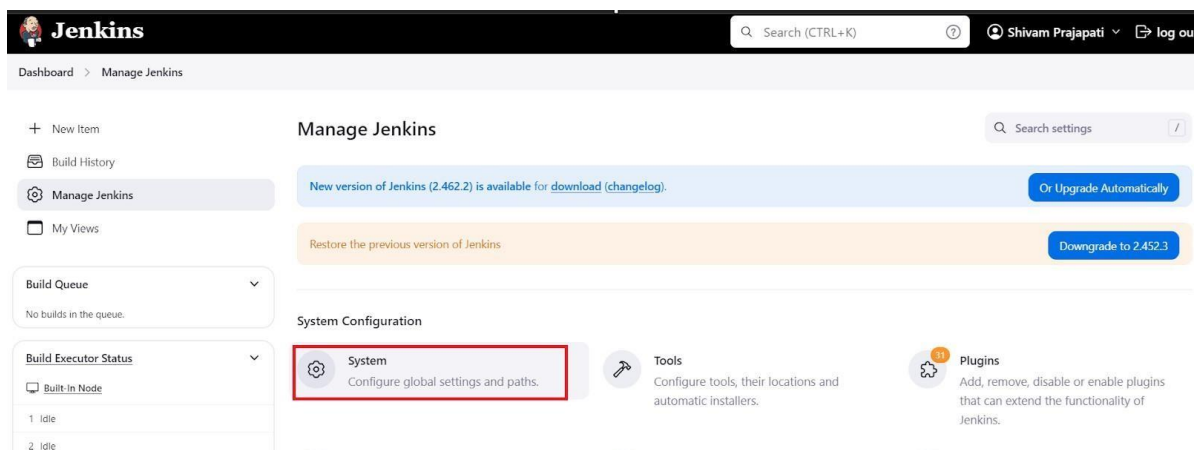


The screenshot shows the Jenkins Dashboard. The left sidebar contains navigation links: New Item, Build History, Manage Jenkins (highlighted with a red box), and My Views. Below these are sections for Build Queue (No builds in the queue) and Build Executor Status (Built-In Node, 1 Idle, 2 Idle). The main content area displays a table of pipelines:

S	W	Name	Last Success	Last Failure	Last Duration
✓	☀	Devops pipeline1	1 mo 18 days #24	N/A	1.3 sec
⋮	☀	DevOps_Pipeline	N/A	N/A	N/A
✓	☀	Pipeline_DevOps	29 days #2	N/A	0.75 sec

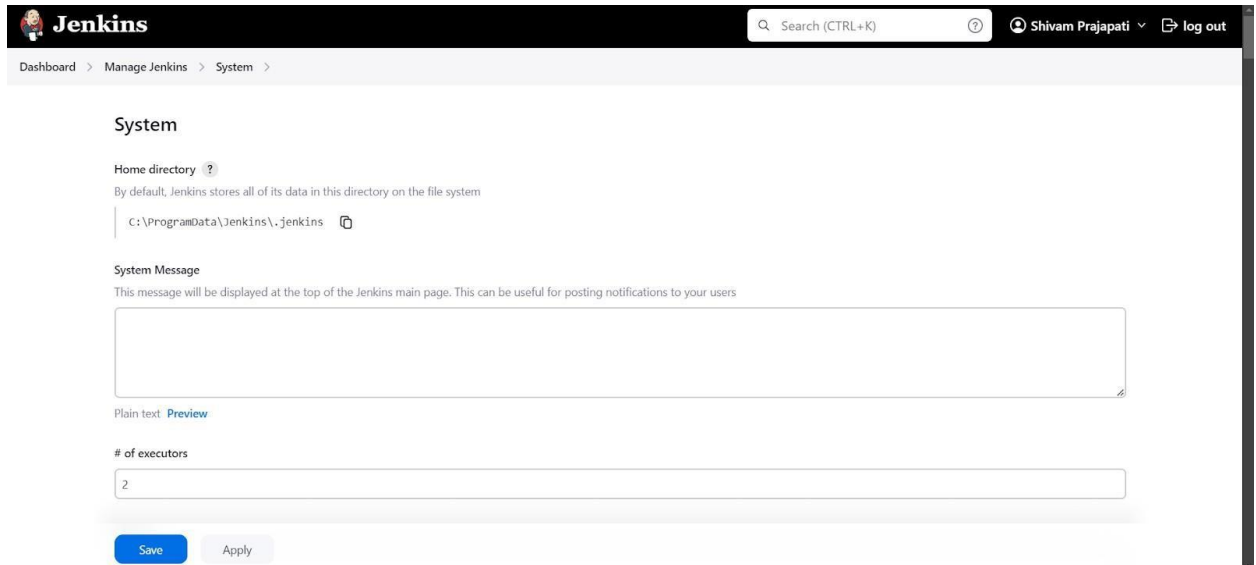
Below the table, there is a section for 'Icon: S M L'.

Go to system



The screenshot shows the Jenkins 'Manage Jenkins' page. The left sidebar is the same as the dashboard. The main content area has a 'Manage Jenkins' header with a search bar. Below the header, there are two banners: 'New version of Jenkins (2.462.2) is available for download (changelog)' with a button 'Or Upgrade Automatically', and 'Restore the previous version of Jenkins' with a button 'Downgrade to 2.452.3'. The 'System Configuration' section is visible, with 'System' (Configure global settings and paths) highlighted by a red box. Other sections like 'Tools' and 'Plugins' are also visible.





The image shows the Jenkins 'System' configuration page. At the top, there's a header with the Jenkins logo, a search bar, and user information (Shivam Prajapati). Below the header, a breadcrumb trail shows 'Dashboard > Manage Jenkins > System >'. The main section is titled 'System'. It contains several configuration options: 'Home directory' with a help icon and a text input field containing 'C:\ProgramData\jenkins\jenkins'; 'System Message' with a large text area and a 'Preview' link; and '# of executors' with a text input field containing '2'. At the bottom, there are 'Save' and 'Apply' buttons.

**Jenkins**

Search (CTRL+K) Shivam Prajapati log out

Dashboard > Manage Jenkins > System >

### System

**Home directory** ?  
By default, Jenkins stores all of its data in this directory on the file system  
C:\ProgramData\jenkins\jenkins

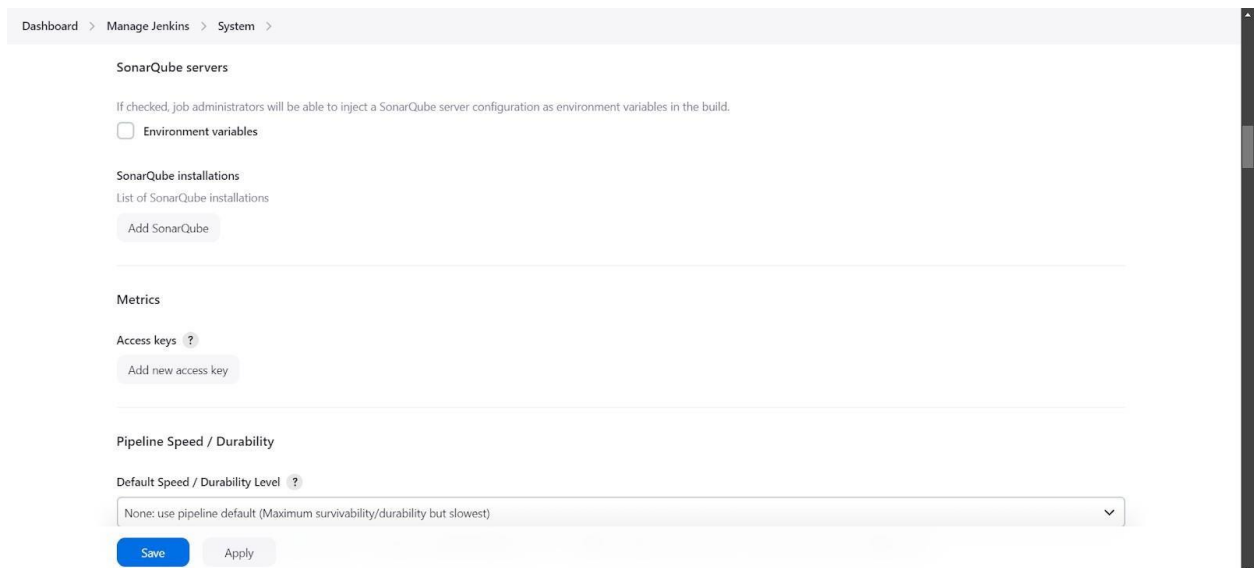
**System Message**  
This message will be displayed at the top of the Jenkins main page. This can be useful for posting notifications to your users

Plain text [Preview](#)

**# of executors**  
2

[Save](#) [Apply](#)

## Scroll Down



The image shows the continuation of the Jenkins 'System' configuration page. It includes sections for 'SonarQube servers' with a checkbox for 'Environment variables', 'SonarQube installations' with an 'Add SonarQube' button, 'Metrics' with an 'Add new access key' button, and 'Pipeline Speed / Durability' with a dropdown menu for 'Default Speed / Durability Level' set to 'None: use pipeline default (Maximum survivability/durability but slowest)'. 'Save' and 'Apply' buttons are at the bottom.

Dashboard > Manage Jenkins > System >

### 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

[Add SonarQube](#)

### Metrics

**Access keys** ?  
[Add new access key](#)

### Pipeline Speed / Durability

**Default Speed / Durability Level** ?  
None: use pipeline default (Maximum survivability/durability but slowest)

[Save](#) [Apply](#)

Select Environment Variable and Click on Add Sonar Qube button in order to Add SonarQube Server to Jenkin

Dashboard > Manage Jenkins > System >

### 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

[Add SonarQube](#)

---

### Metrics

Access keys ?

[Add new access key](#)

---

Pipeline Speed / Durability

Do the required entries as shown below

Dashboard > Manage Jenkins > System >

### 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

Server URL

Default is http://localhost:9000

Server authentication token

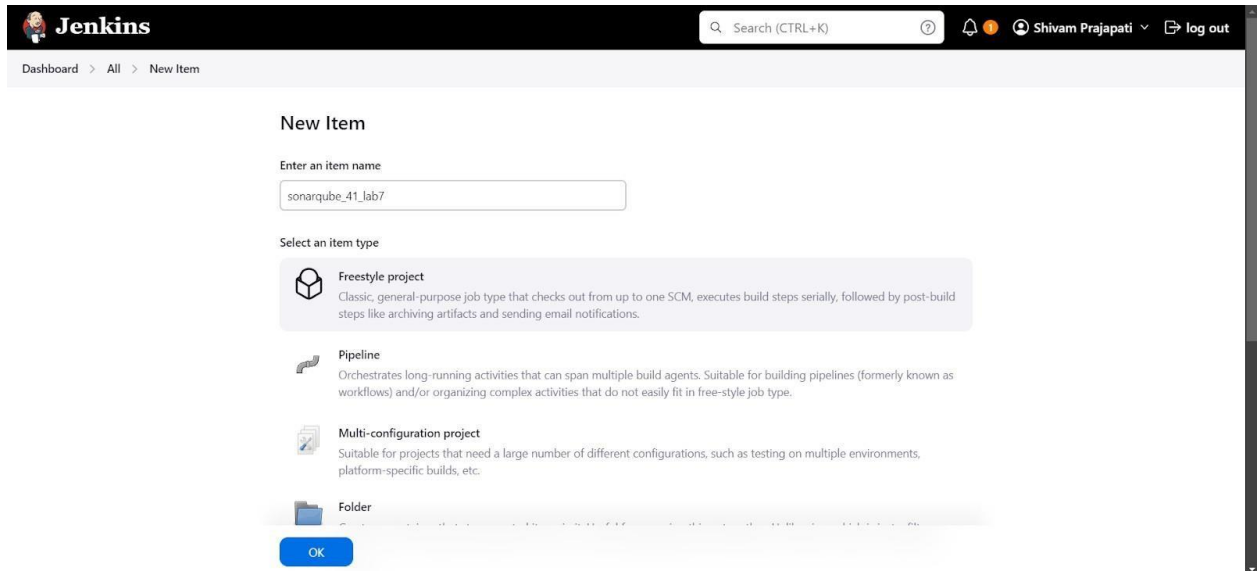
SonarQube authentication token. Mandatory when anonymous access is disabled.

[+ Add](#)

[Save](#) [Apply](#)

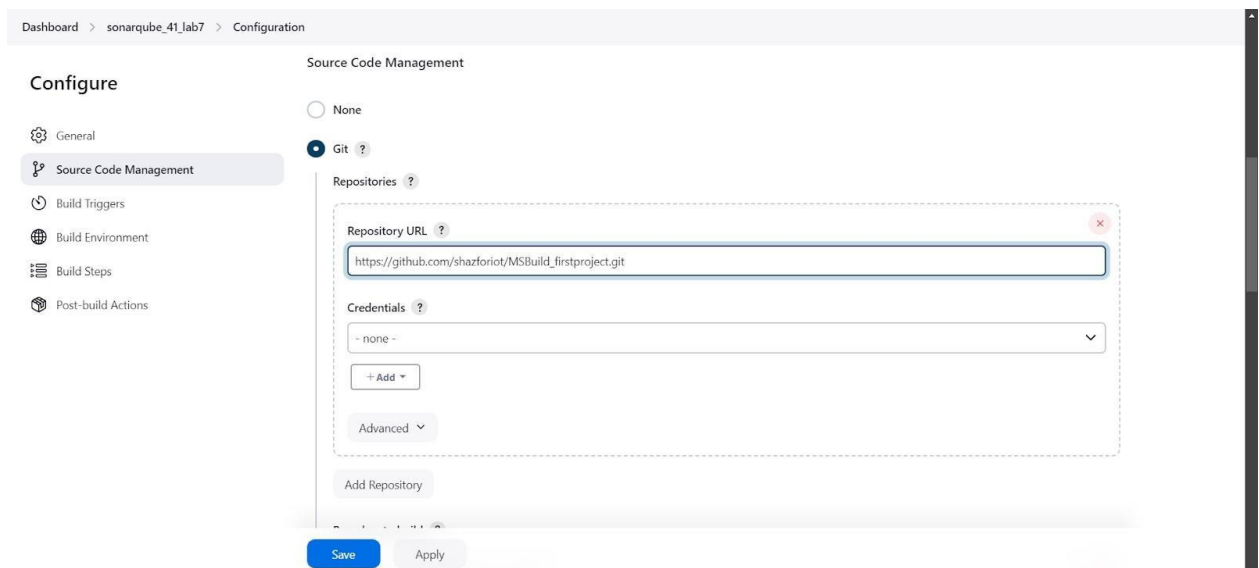
Click on save

**Step 10:** After configuration, create a New Item → choose a freestyle project.

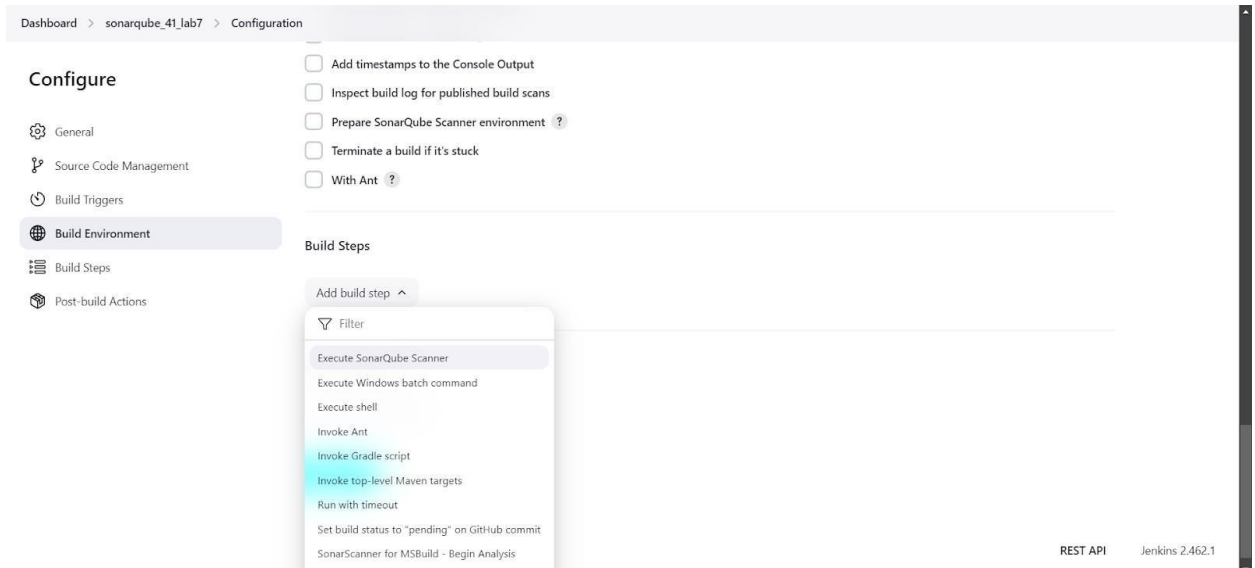


**Step 11:** Use this github repository in Source Code Management.

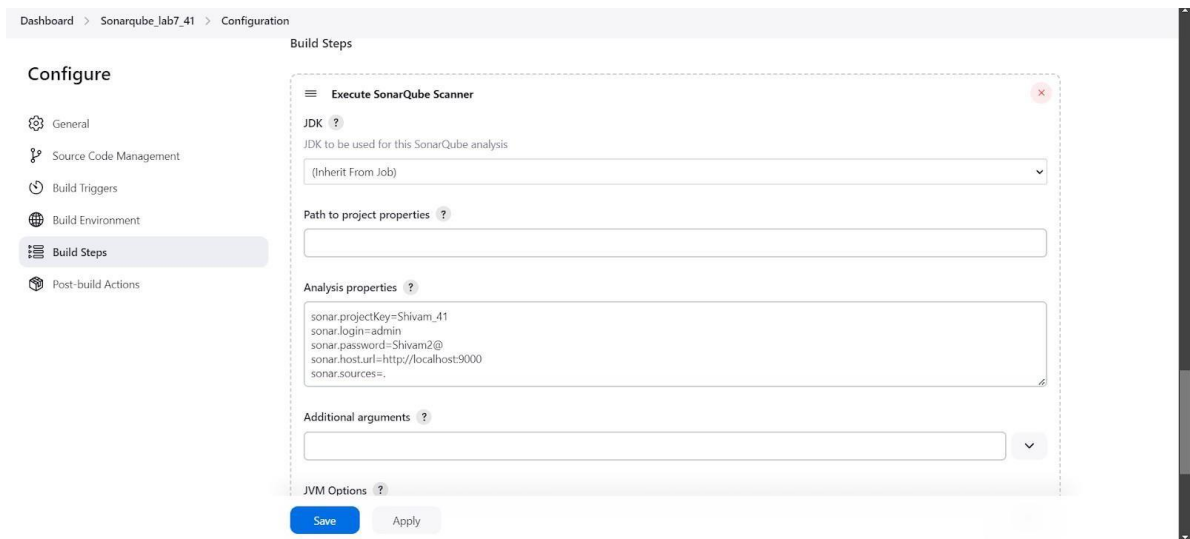
[https://github.com/shazforiot/MSBuild\\_firstproject](https://github.com/shazforiot/MSBuild_firstproject). It is a sample hello-world project with no vulnerabilities.

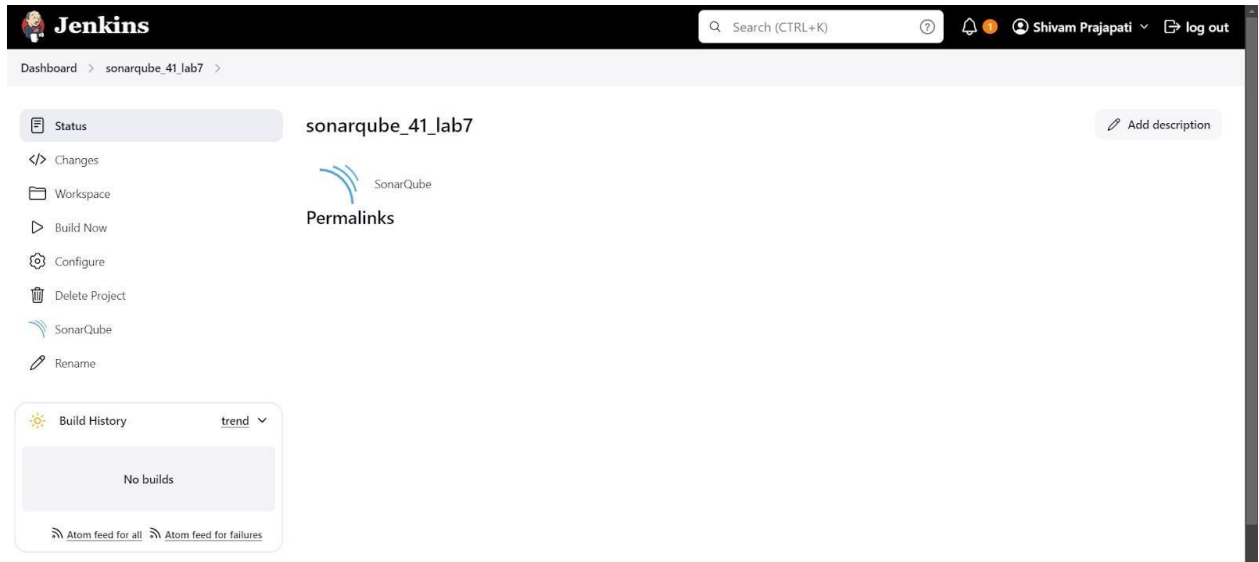


**Step 12:** Under Build Steps, enter Sonarqube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.

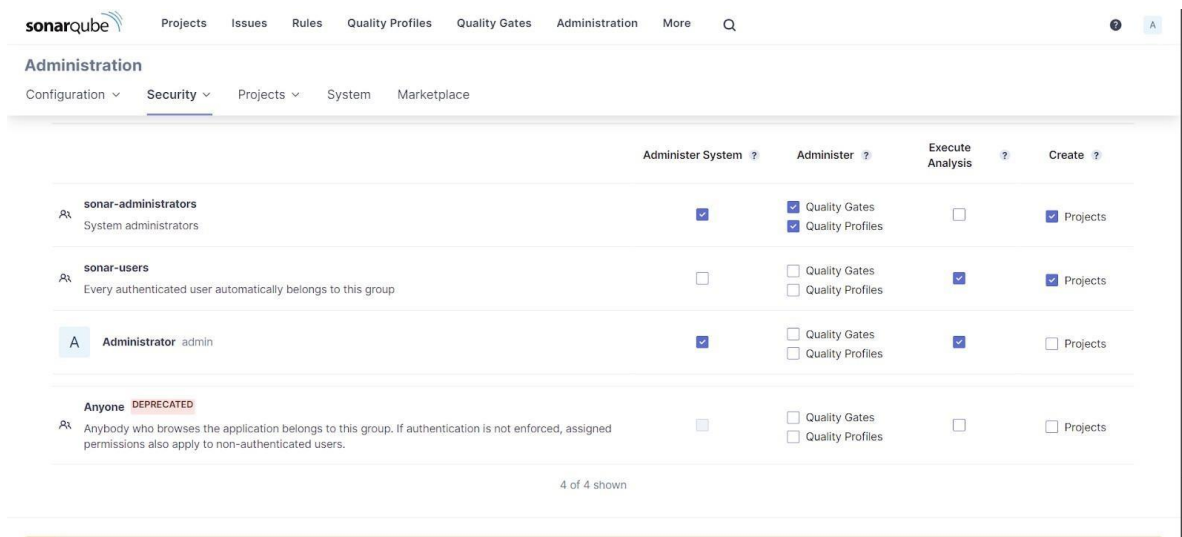


Click on execute sonar scanner

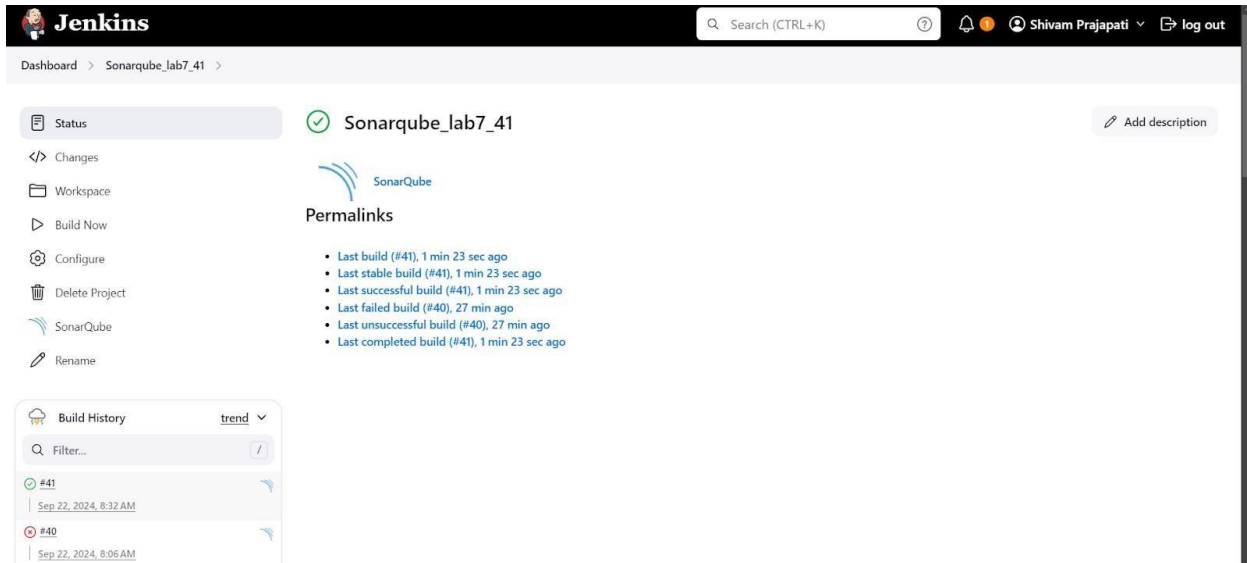




**Step 13:** Now, you need to grant the local user (here admin user) permissions to Execute the Analysis stage on SonarQube. For this, go to <http://localhost:9000/admin/permissions> and check the 'Execute Analysis' checkbox under Administrator.





**Step 14:** Go back to Jenkins. Go to the job you had just built and click on Build Now.



**Jenkins** Search (CTRL+K) Shivam Prajapati log out

Dashboard > Sonarqube\_lab7\_41 >

**Status**  **Sonarqube\_lab7\_41** [Add description](#)



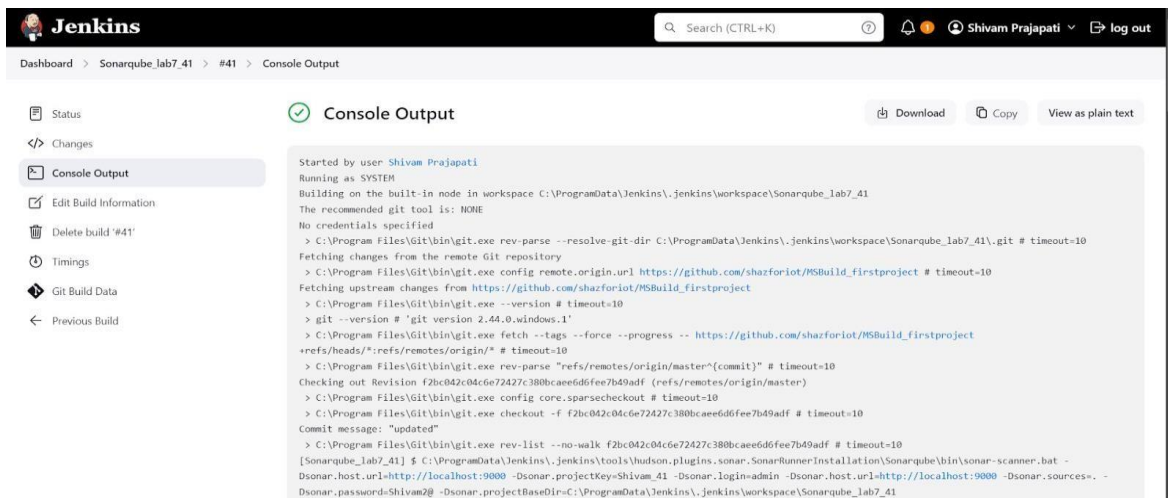
**Permalinks**

- Last build (#41), 1 min 23 sec ago
- Last stable build (#41), 1 min 23 sec ago
- Last successful build (#41), 1 min 23 sec ago
- Last failed build (#40), 27 min ago
- Last unsuccessful build (#40), 27 min ago
- Last completed build (#41), 1 min 23 sec ago

**Build History** trend

Build	Time
#41	Sep 22, 2024, 8:32 AM
#40	Sep 22, 2024, 8:06 AM

Check the Console Output



**Jenkins** Search (CTRL+K) Shivam Prajapati log out

Dashboard > Sonarqube\_lab7\_41 > #41 > Console Output

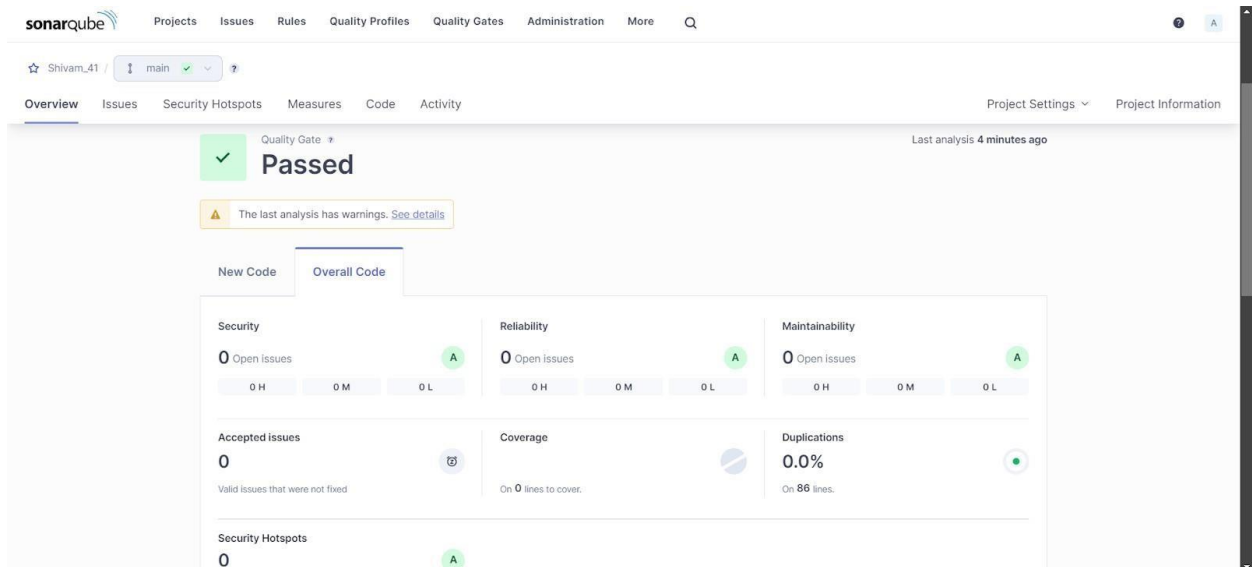
**Console Output** [Download](#) [Copy](#) [View as plain text](#)

```
Started by user Shivam Prajapati
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\jenkins\jenkins\workspace\Sonarqube_lab7_41
The recommended git tool is: NONE
No credentials specified
> C:\Program Files\Git\bin\git.exe rev-parse --resolve-git-dir C:\ProgramData\jenkins\jenkins\workspace\Sonarqube_lab7_41\.git # timeout=10
Fetching changes from the remote Git repository
> C:\Program Files\Git\bin\git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject
> C:\Program Files\Git\bin\git.exe --version # timeout=10
> git --version # 'git version 2.44.0.windows.1'
> C:\Program Files\Git\bin\git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject
+refs/heads/*:refs/remotes/origin/* # timeout=10
> C:\Program Files\Git\bin\git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcae6d6fee7b49adf (refs/remotes/origin/master)
> C:\Program Files\Git\bin\git.exe config core.sparsecheckout # timeout=10
> C:\Program Files\Git\bin\git.exe checkout -f f2bc042c04c6e72427c380bcae6d6fee7b49adf # timeout=10
Commit message: "updated"
> C:\Program Files\Git\bin\git.exe rev-list --no-walk f2bc042c04c6e72427c380bcae6d6fee7b49adf # timeout=10
[Sonarqube_lab7_41] $ C:\ProgramData\jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\Sonarqube\bin\sonar-scanner.bat -
Dsonar.host.url=http://localhost:9000 -Dsonar.projectKey=Shivam_41 -Dsonar.login=admin -Dsonar.host.url=http://localhost:9000 -Dsonar.sources=. -
Dsonar.password=Shivam2@ -Dsonar.projectBaseDir=C:\ProgramData\jenkins\jenkins\workspace\Sonarqube_lab7_41
```

```
Dashboard > Sonarqube_lab7_41 > #41 > Console Output
08:32:55.264 INFO Sensor Analysis Warnings import [csharp] (done) | time=4ms
08:32:55.264 INFO Sensor C# File Caching Sensor [csharp]
08:32:55.264 WARN Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir'
property.
08:32:55.264 INFO Sensor C# File Caching Sensor [csharp] (done) | time=0ms
08:32:55.264 INFO Sensor Zero Coverage Sensor
08:32:55.279 INFO Sensor Zero Coverage Sensor (done) | time=15ms
08:32:55.288 INFO SCM Publisher SCM provider for this project is: git
08:32:55.290 INFO SCM Publisher 4 source files to be analyzed
08:32:55.515 INFO SCM Publisher 4/4 source files have been analyzed (done) | time=225ms
08:32:55.522 INFO CPD Executor Calculating CPD for 0 files
08:32:55.523 INFO CPD Executor CPD calculation finished (done) | time=0ms
08:32:55.524 INFO SCM revision ID 'f2bc042c04cbe72427c380bcaee6d6fee7b49adf'
08:32:55.809 INFO Analysis report generated in 125ms, dir size=201.0 kB
08:32:55.858 INFO Analysis report compressed in 40ms, zip size=22.5 kB
08:32:56.061 INFO Analysis report uploaded in 201ms
08:32:56.062 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=Shivam_41
08:32:56.063 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
08:32:56.063 INFO More about the report processing at http://localhost:9000/api/ce/task?id=94824f79-1689-41ea-99d7-abfee5815e63
08:32:56.077 INFO Analysis total time: 28.732 s
08:32:56.078 INFO SonarScanner Engine completed successfully
08:32:56.158 INFO EXECUTION SUCCESS
08:32:56.158 INFO Total time: 38.798s
Finished: SUCCESS
```

REST API Jenkins 2.462.1

**Step 15:** 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 the SonarQube project displays that the code has no errors.