Prajyot Shinde 57/D15A

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

Theory:

Static application security testing (SAST), or static analysis, is a testing methodology that analyzes source code to find security vulnerabilities that make your organization's applications susceptible to attack. SAST scans an application before the code is compiled It's also known as white box testing.

What problems does SAST solve?

SAST takes place very early in the software development life cycle (SDLC) as it does not require a working application and can take place without code being executed. It helps developers identify vulnerabilities in the initial stages of development and quickly resolve issues without breaking builds or passing on vulnerabilities to the final release of the application.

SAST tools give developers real-time feedback as they code, helping them fix issues before they pass the code to the next phase of the SDLC. This prevents security-related issues from being considered an afterthought. SAST tools also provide graphical representations of the issues found, from source to sink. These help you navigate the code easier. Some tools point out the exact location of vulnerabilities and highlight the risky code. Tools can also provide in-depth guidance on how to fix issues and the best place in the code to fix them, without requiring deep security domain expertise. It's important to note that SAST tools must be run on the application on a regular basis, such as during daily/monthly builds, every time code is checked in, or during a code release.

Why is SAST important?

Developers dramatically outnumber security staff. It can be challenging for an organization to find the resources to perform code reviews on even a fraction of its applications. A key strength of SAST tools is the ability to analyze 100% of the codebase. Additionally, they are much faster than manual secure code reviews performed by humans. These tools can scan millions of lines of code in a matter of minutes. SAST tools automatically identify critical vulnerabilities—such as buffer overflows, SQL injection, cross-site scripting, and others—with high confidence.

Thus, integrating static analysis into the SDLC can yield dramatic results in the overall quality of the code developed.

What are the key steps to run SAST effectively?

There are six simple steps needed to perform SAST efficiently in organizations that have a very large number of applications built with different languages, frameworks, and platforms.

1. Finalize the tool. Select a static analysis tool that can perform code reviews of applications written in the programming languages you use. The tool should also be able to comprehend the underlying framework used by your software.

2. Create the scanning infrastructure, and deploy the tool. This step involves

handling the licensing requirements, setting up access control and authorization, and procuring the resources required (e.g., servers and databases) to deploy the tool.

3. Customize the tool. Fine-tune the tool to suit the needs of the organization. For example, you might configure it to reduce false positives or find additional security

vulnerabilities by writing new rules or updating existing ones. Integrate the tool into the build environment, create dashboards for tracking scan results, and build custom reports.

4. Prioritize and onboard applications. Once the tool is ready, onboard your applications. If you have a large number of applications, prioritize the high-risk

applications to scan first. Eventually, all your applications should be onboarded and scanned regularly, with application scans synced with release cycles, daily or monthly builds, or code check-ins.

5. Analyze scan results. This step involves triaging the results of the scan to remove false positives. Once the set of issues is finalized, they should be tracked and provided to the deployment teams for proper and timely remediation.

6. Provide governance and training. Proper governance ensures that your development teams are employing the scanning tools properly. The software

security touchpoints should be present within the SDLC. SAST should be incorporated as part of your application development and deployment process.

Integrating Jenkins with SonarQube: Windows installation

Step 1 Install JDK 1.8

Step 2 download and install jenkins
installing-the-default-jre-jdk Step 1 Install JDK 1.8
sudo apt-get install openjdk-8-jre sudo apt install default-jre /

how-to-install-jenkins-on-ubuntu-20-04

Open SSH

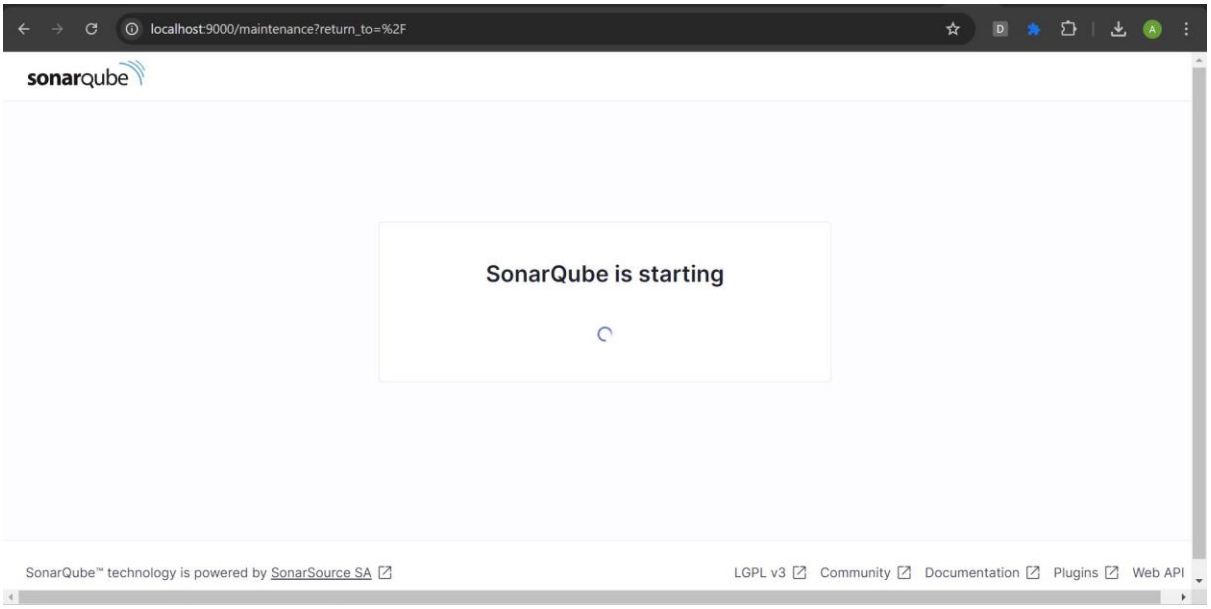
Prerequisites:

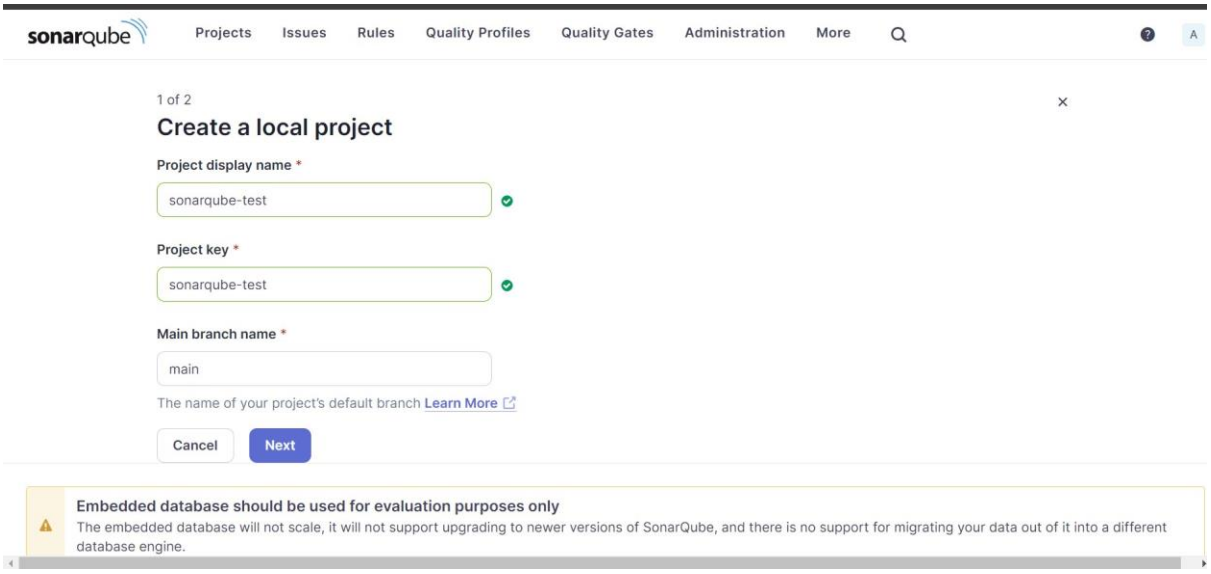
- Jenkins installed
- Docker Installed (for SonarQube)

(sudo apt-get install docker-ce=5:20.10.15~3-0~ubuntu-jammy docker-ce-cli=5:20.10.15~3-0~ubuntu-jammy containerd.io docker-compose-plugin)

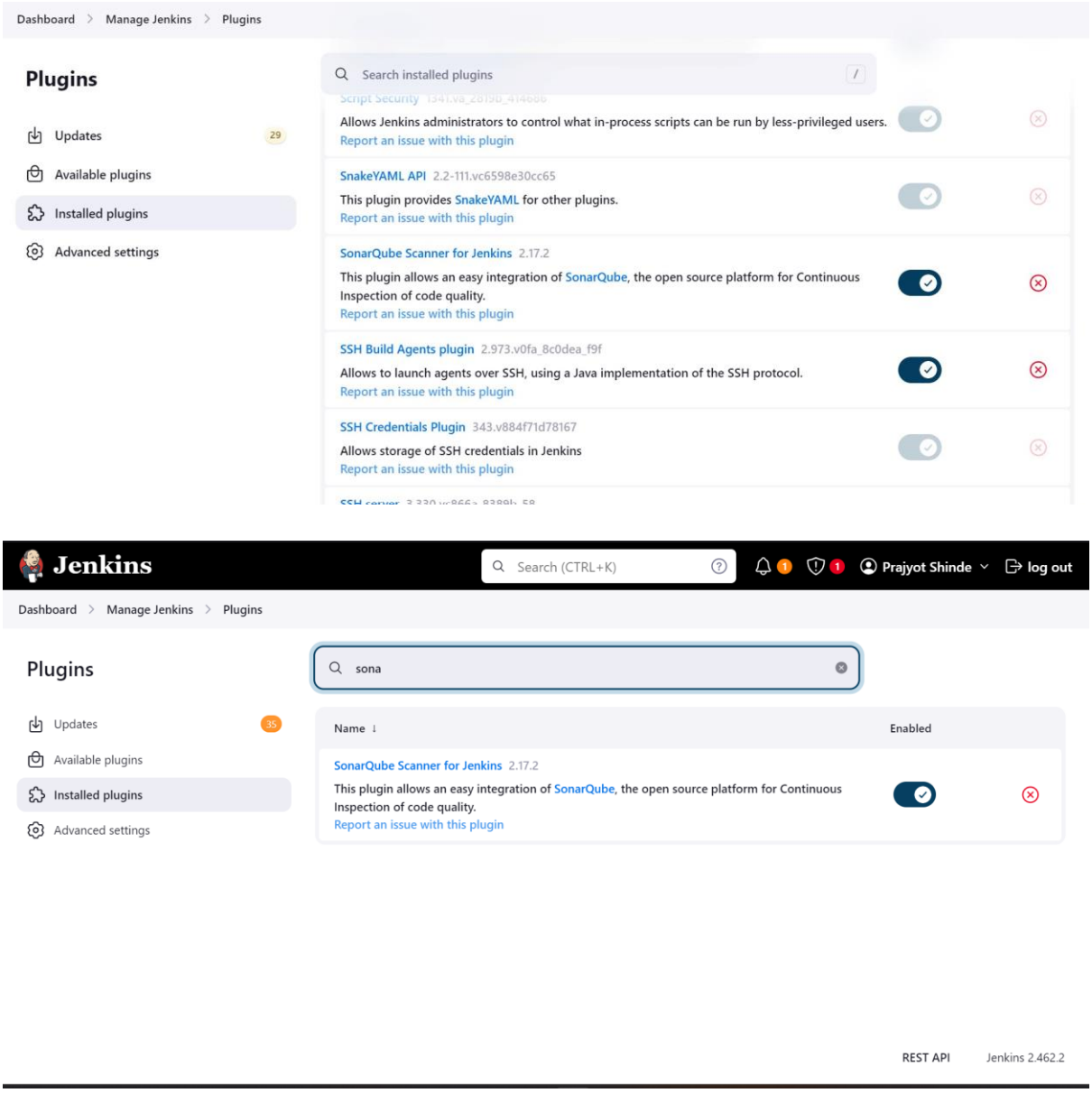
- SonarQube Docker Image

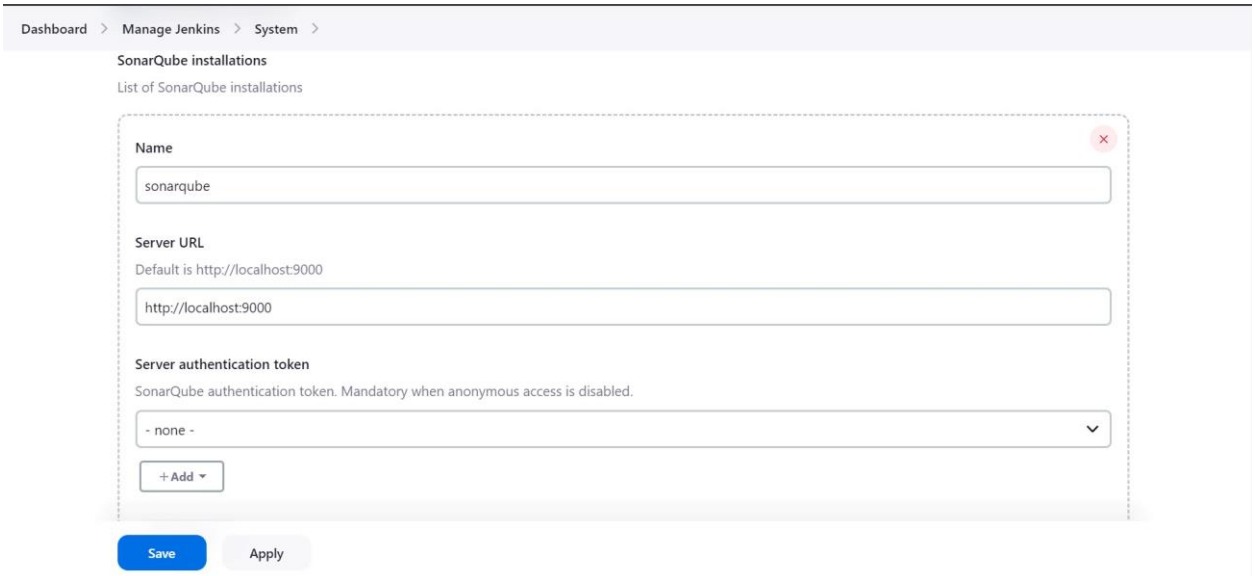
```
C:\Users\91952>docker run -d --name sonarqube -p 9000:9000 sonarqube
0392bc54d6d9ecd3e651c04433976aae2ab159b63bc07776b81a03ebf6e754b5

C:\Users\91952>docker image ls
REPOSITORY   TAG       IMAGE ID       CREATED         SIZE
postgres     latest    026d0ab72b34   7 weeks ago     611MB
sonarqube    latest    72e9feec7124   2 months ago    1.92GB

C:\Users\91952>
```

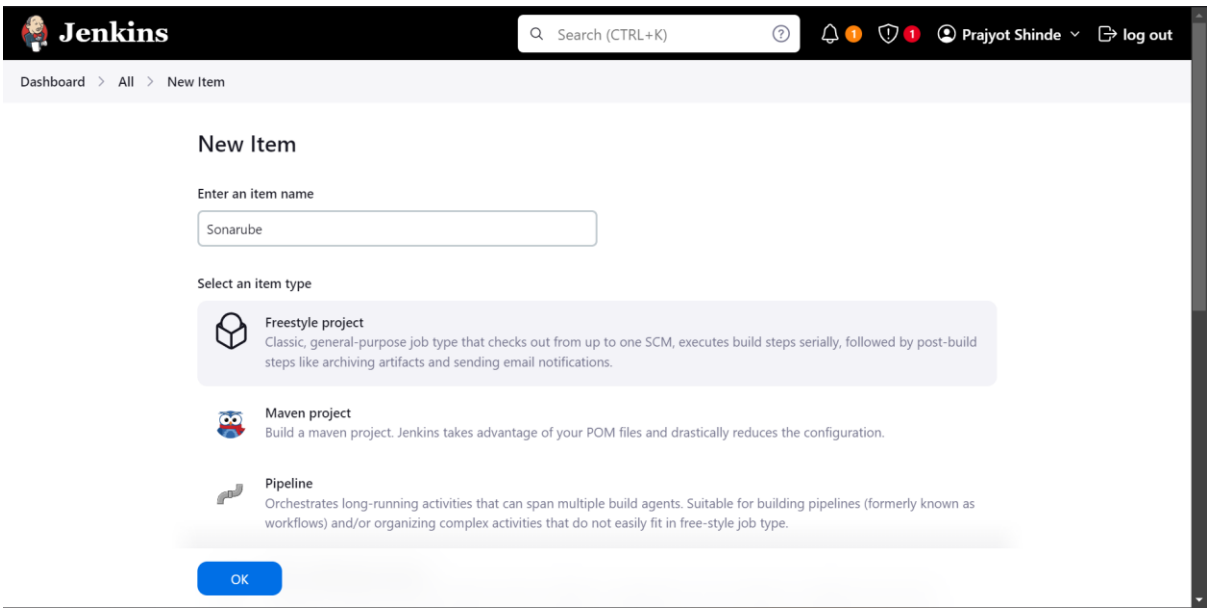


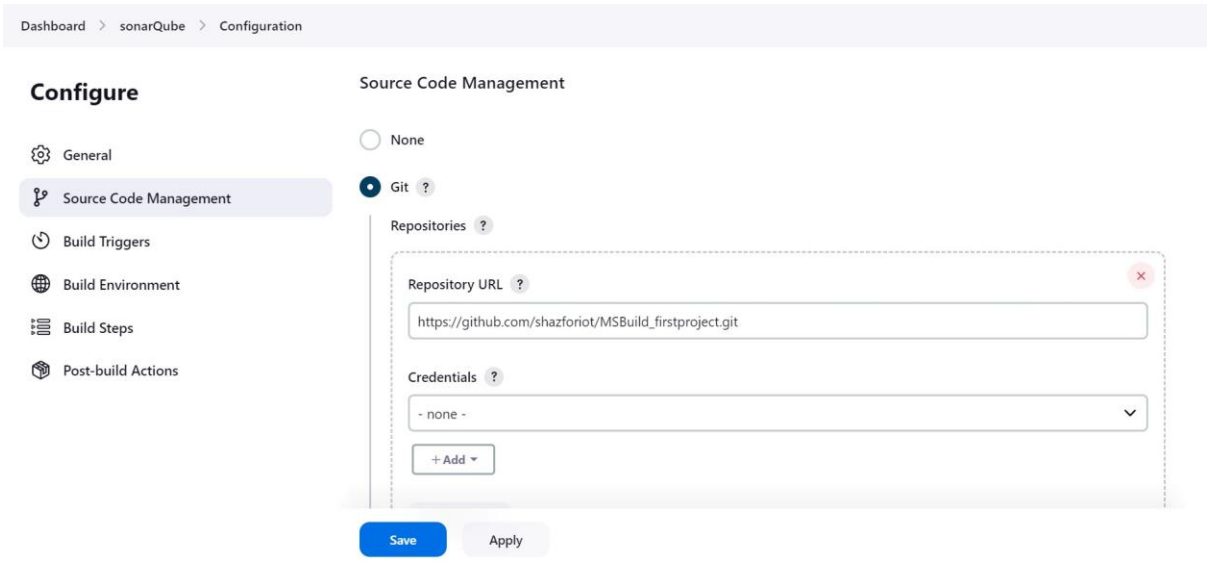


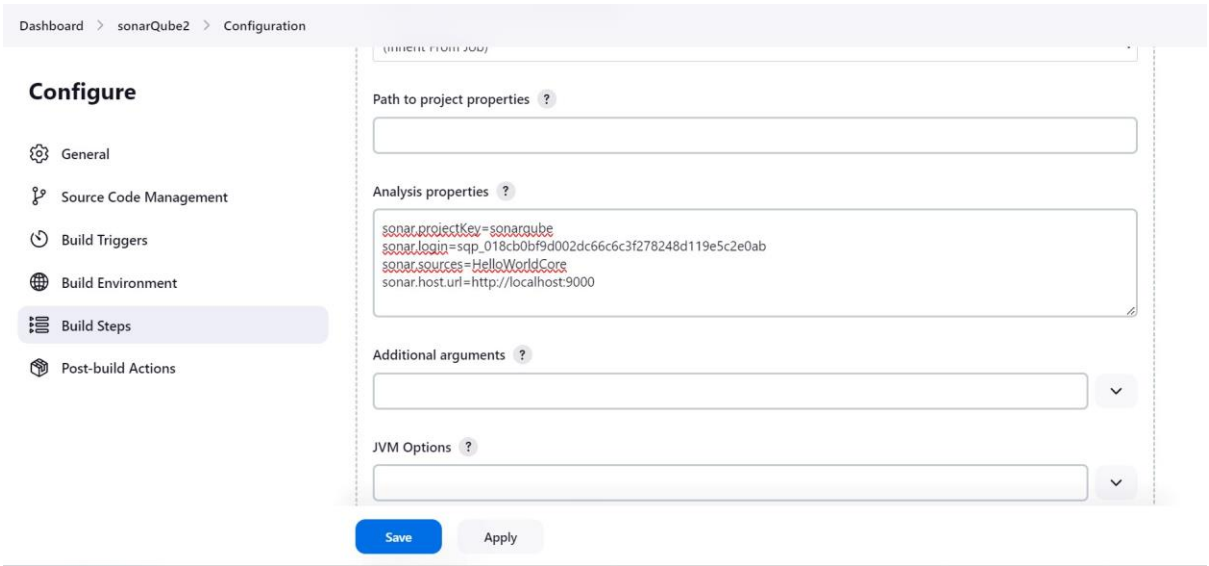
Ok so go to Jenkins dashboard and then go to Manage jenkins then plugins , install plugins and then install “sonarqube scanner”

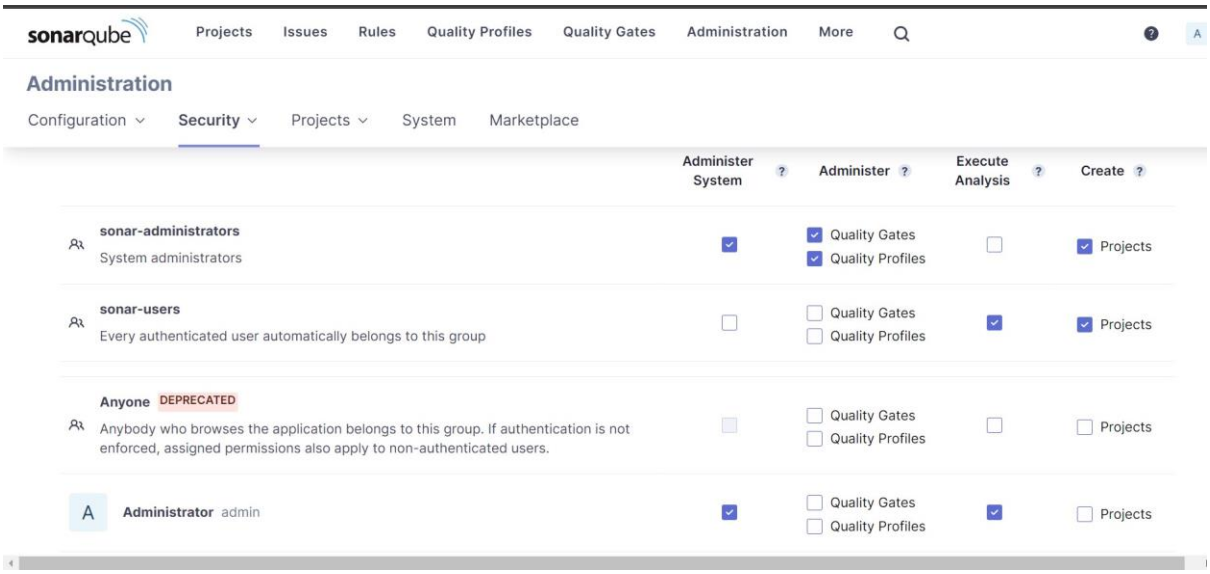



Dashboard > Manage Jenkins > System >
SonarQube installations
List of SonarQube installations
Name
sonarqube
Server URL
Default is http://localhost:9000
http://localhost:9000
Server authentication token
SonarQube authentication token. Mandatory when anonymous access is disabled.
- none -
+ Add
Save
Apply

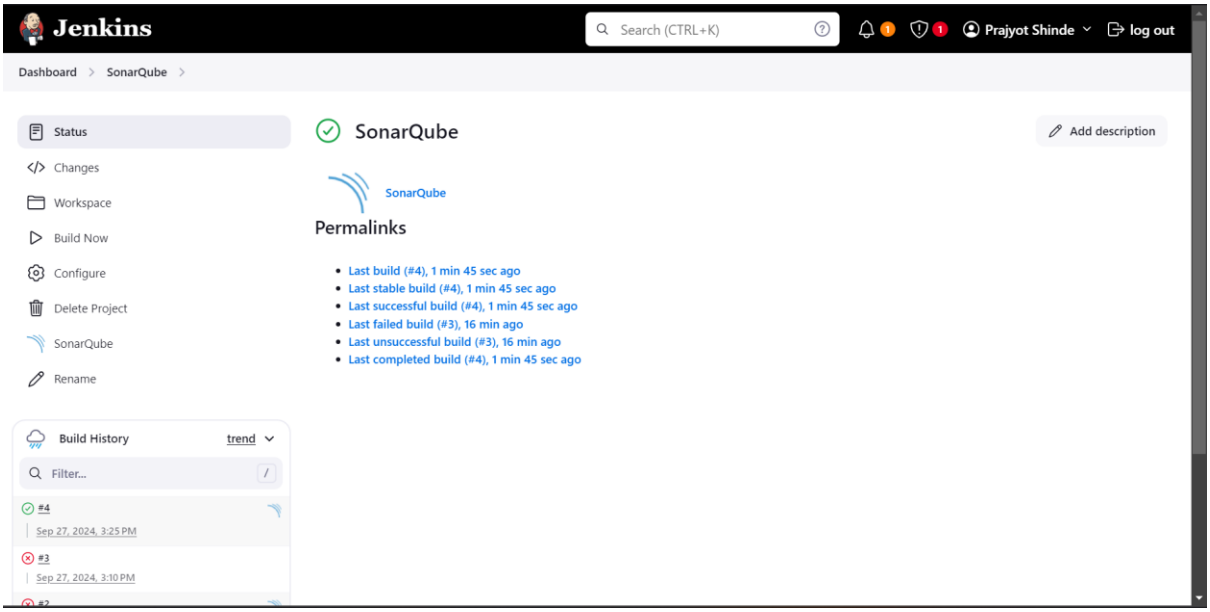

Jenkins
Search (CTRL+K)
Prajyot Shinde
log out
Dashboard > All > New Item
New Item
Enter an item name
Sonarube
Select an item type
Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts 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.
OK

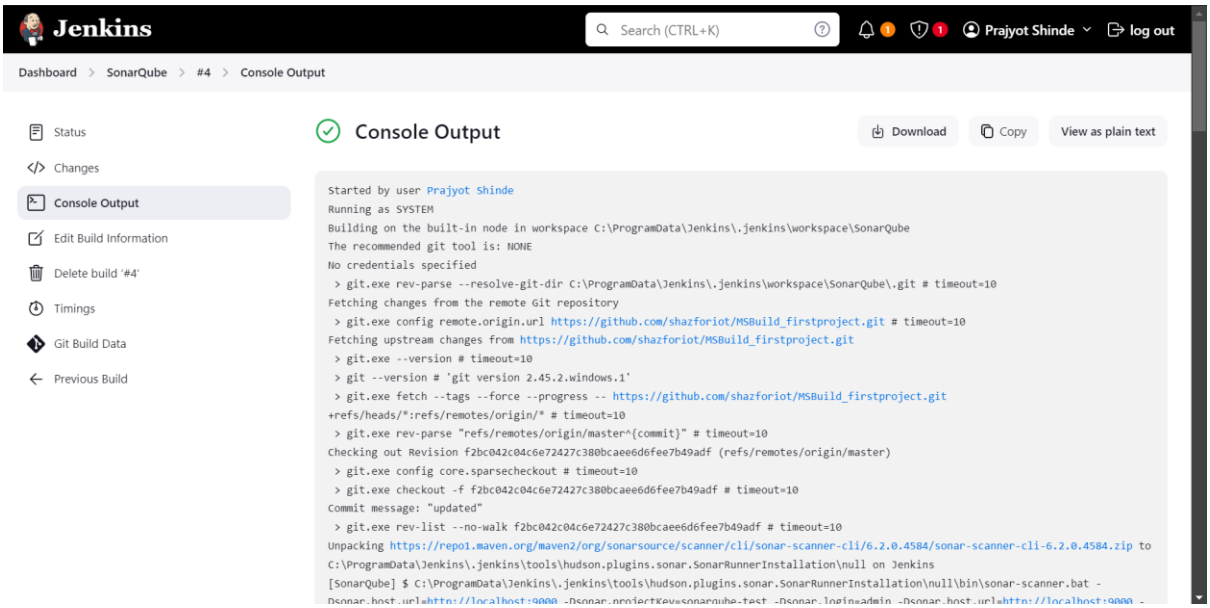

Dashboard > sonarQube > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
Source Code Management
None
Git
Repositories
Repository URL
https://github.com/shazforiot/MSBuild_firstproject.git
Credentials
- none -
+ Add
Save
Apply

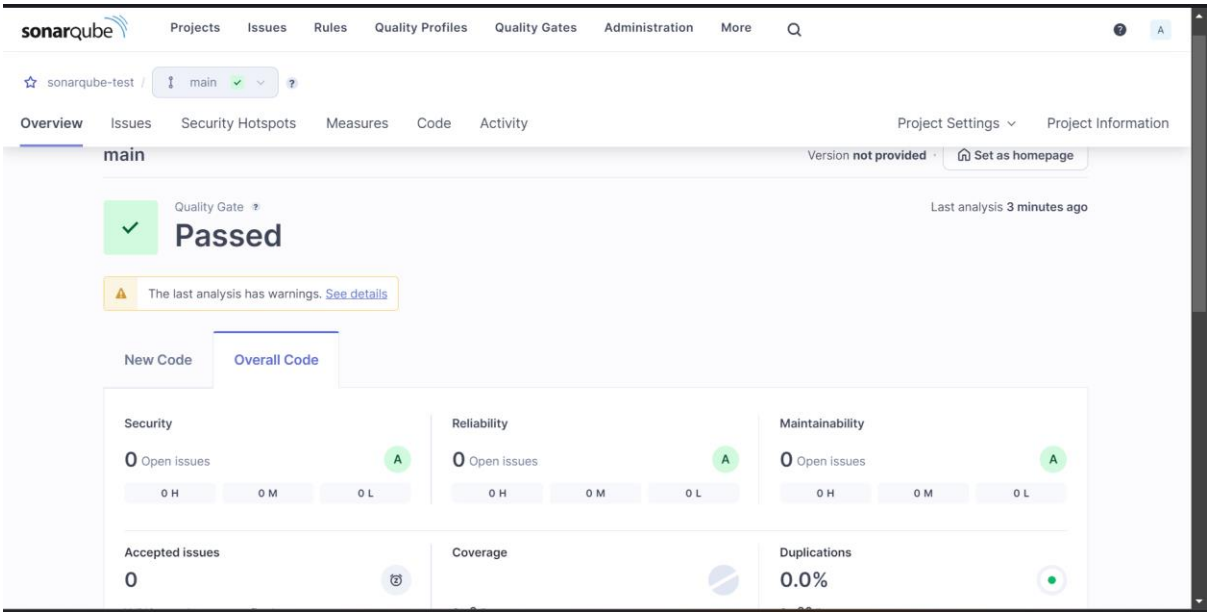

Dashboard > sonarQube2 > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
Path to project properties
Analysis properties
sonar.projectKey=sonarqube
sonar.login=sqp_018cb0bf9d002dc66c6c3f278248d119e5c2e0ab
sonar.sources=HelloWorldCore
sonar.host.url=http://localhost:9000
Additional arguments
JVM Options
Save
Apply


sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
Administration
Configuration
Security
Projects
System
Marketplace
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
A
Administrator admin
Quality Gates
Quality Profiles
Projects

After 3 failures finally there was 4th success which is shown down below in image






sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
sonarqube-test
main
Overview
Issues
Security Hotspots
Measures
Code
Activity
Project Settings
Project Information
main
Version not provided
Set as homepage
Quality Gate
Passed
Last analysis 3 minutes ago
The last analysis has warnings. See details
New Code
Overall Code
Security
0 Open issues
A
0 H
0 M
0 L
Reliability
0 Open issues
A
0 H
0 M
0 L
Maintainability
0 Open issues
A
0 H
0 M
0 L
Accepted issues
0
Coverage
Duplications
0.0%