

## EXPERIMENT NO:7

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

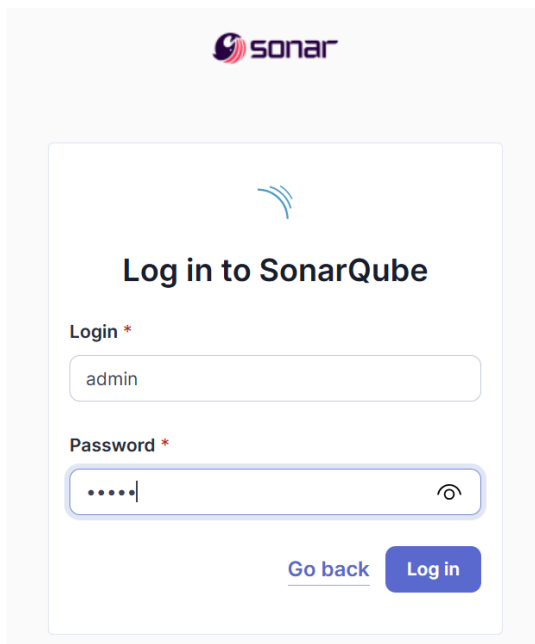
### Procedure:

#### 1.Creation of project on Sonarqube

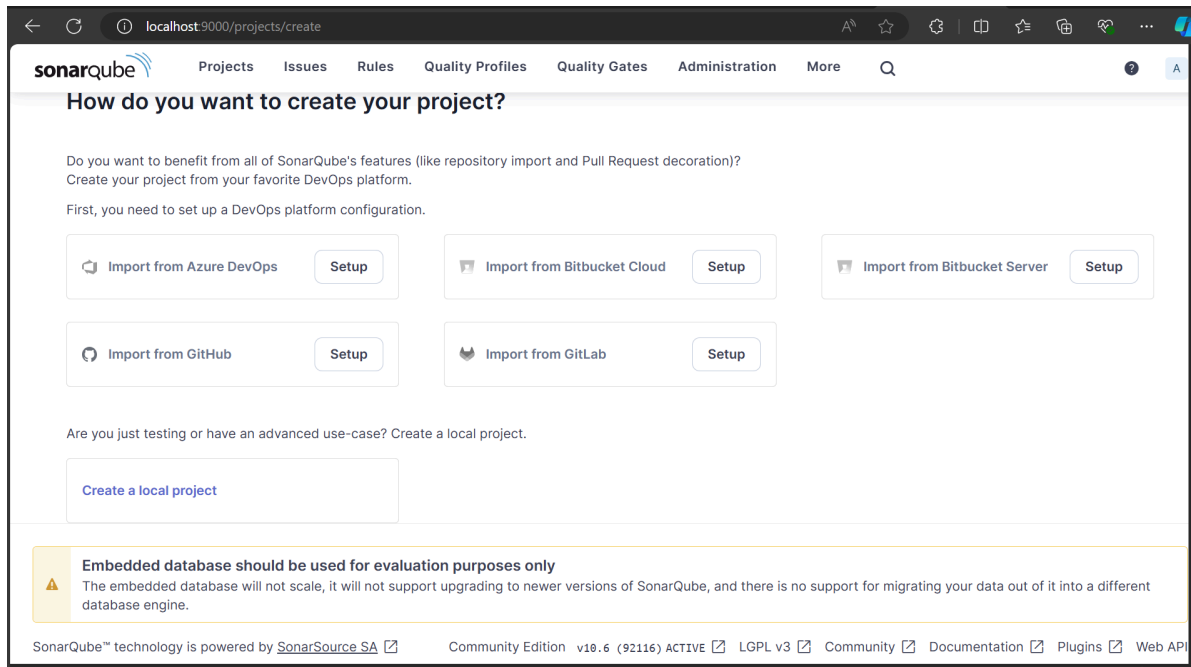
1. Open the command prompt and perform this command  
(\*\*Docker must be Installed before running this command.\*\*)  
docker run -d --name sonarqube -e SONAR\_ES\_BOOTSTRAP\_CHECKS\_DISABLE=true -p 9000:9000 sonarqube:latest

```
PS C:\Users\Sadneya> 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: sha256:72e9fec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecd
Status: Downloaded newer image for sonarqube:latest
19948c8162691fd8be61d035e355c4fe4a6a2fd0a15237e94f75c0857ff7e2ff
```

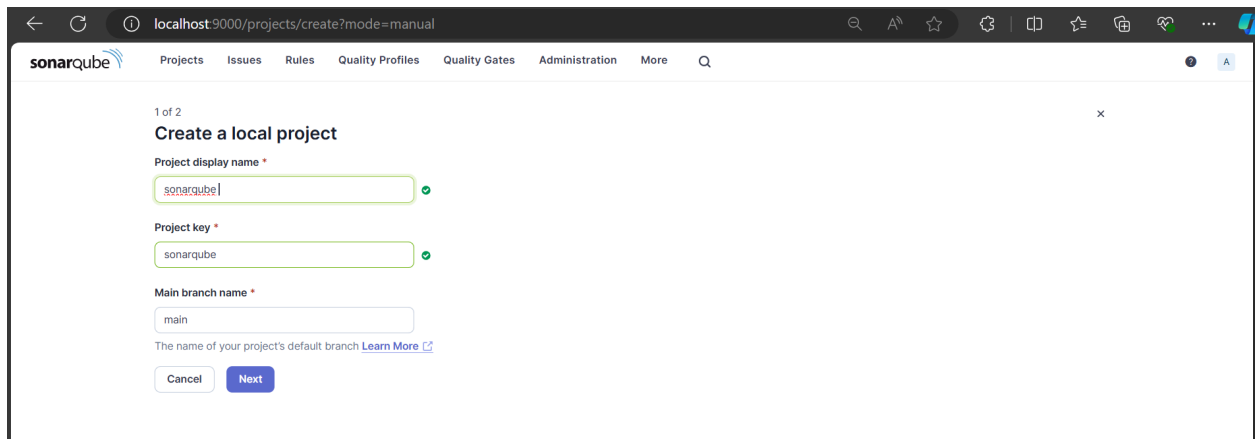
2. Then after its successful execution, run the sonarqube at <http://localhost:9000>
3. Then Login as Username admin and password admin.



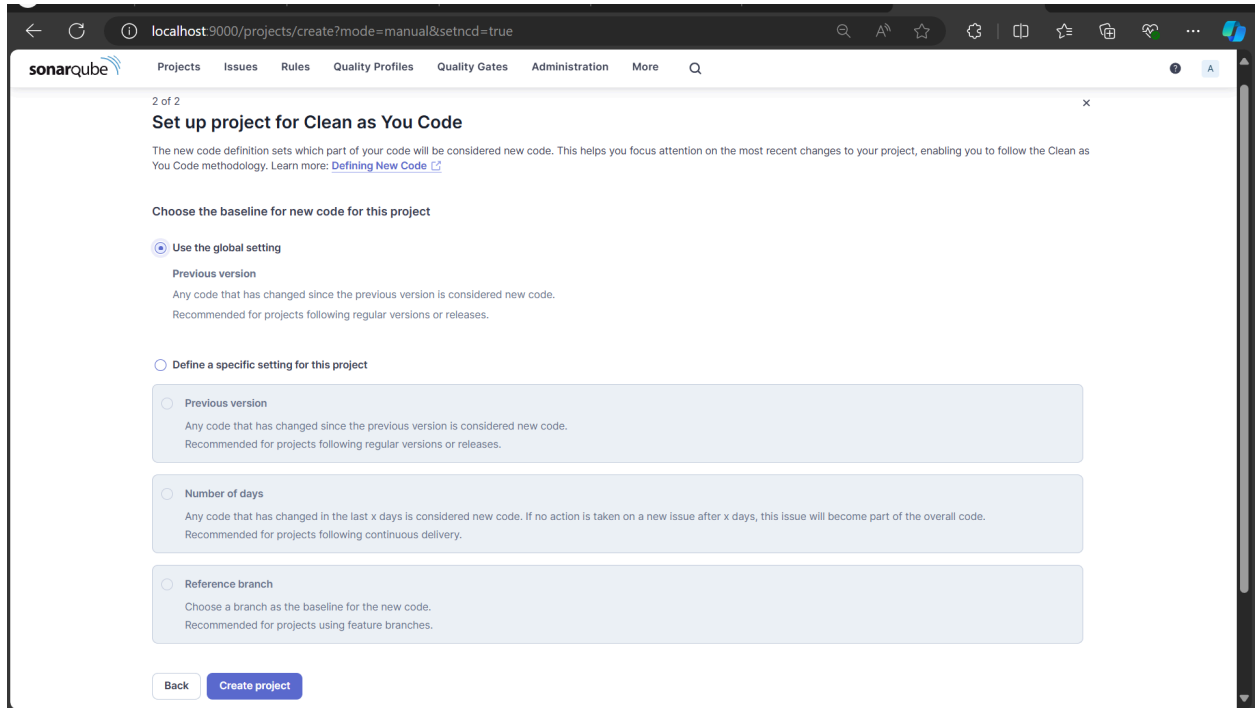
The image shows the SonarQube login page. At the top, there is the Sonar logo. Below it, the text "Log in to SonarQube" is displayed. There are two input fields: "Login \*" with the value "admin" and "Password \*" with masked characters ".....". To the right of the password field is an eye icon for toggling visibility. At the bottom, there are two buttons: "Go back" (a link) and "Log in" (a button).



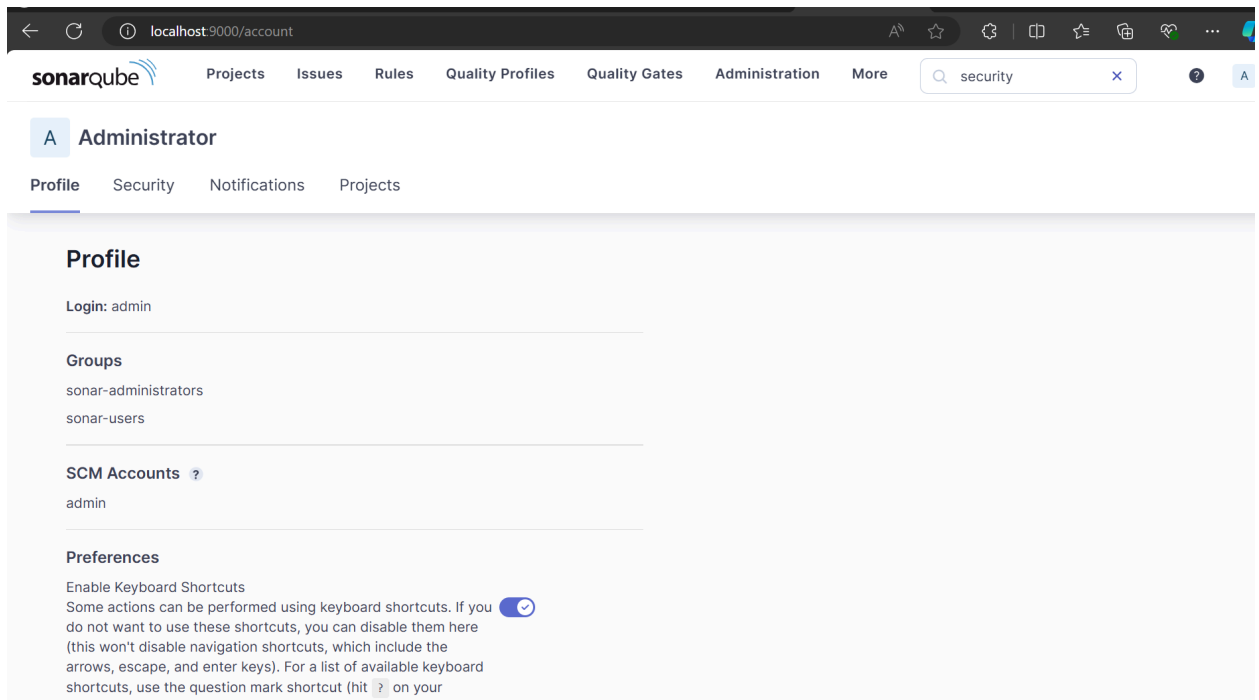
4. Then create a project. Here I have given the name “sonarqube”. Keep branch name main only and then click on next.



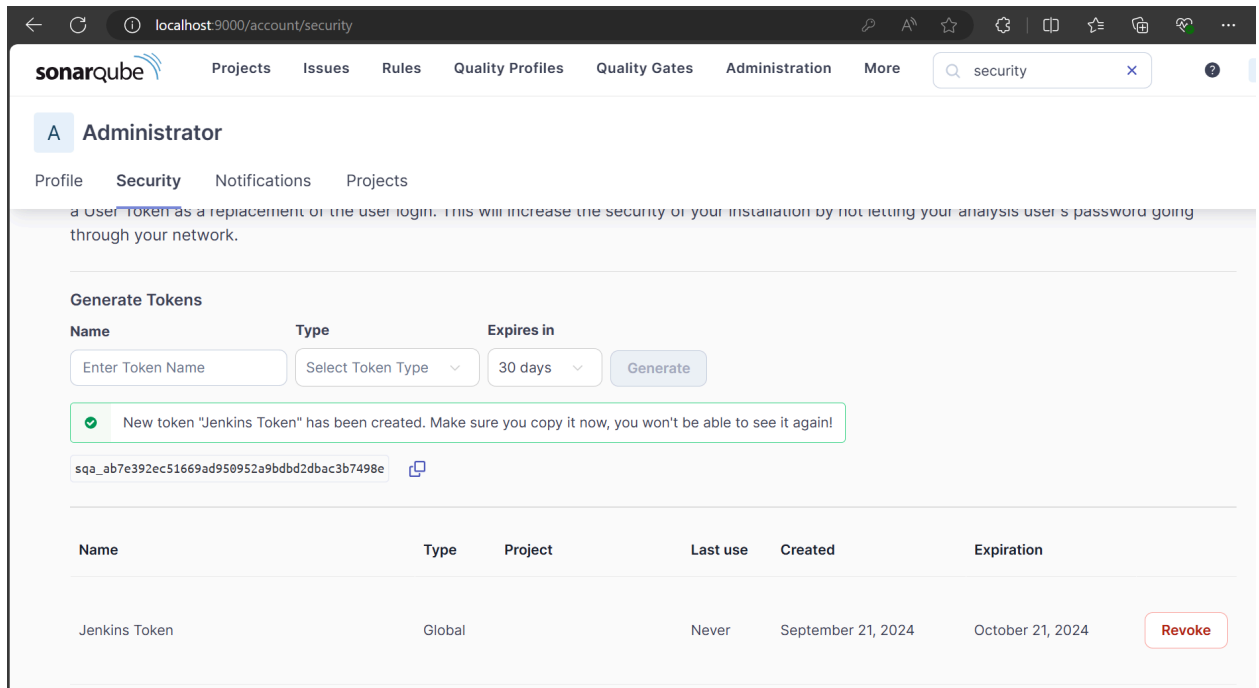
5. Then in the Setup project for clean as you code they will ask to choose the baseline for new code for this project. Choose Use the global setting. Then click on create project.



6. Then click on your account profile and select my account. Inside this click on security option.



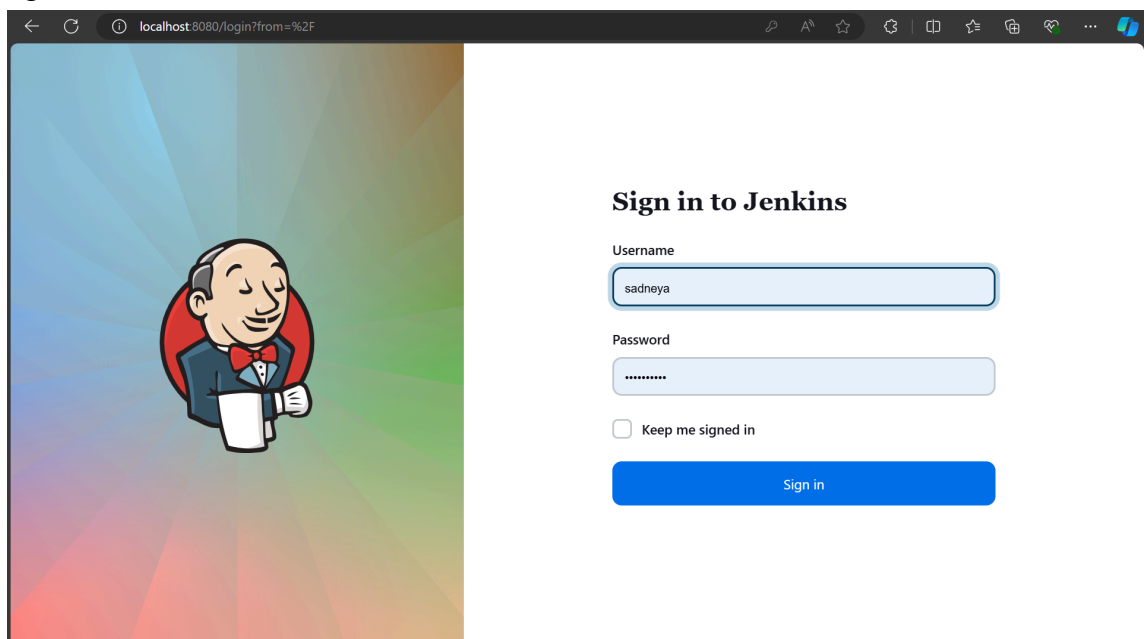
7. Then set the name of your token and here I have chosen the global scope of the token and then clicked on generate token, thus token created successfully. (Here I have given my token name as 'jenkins token'.)



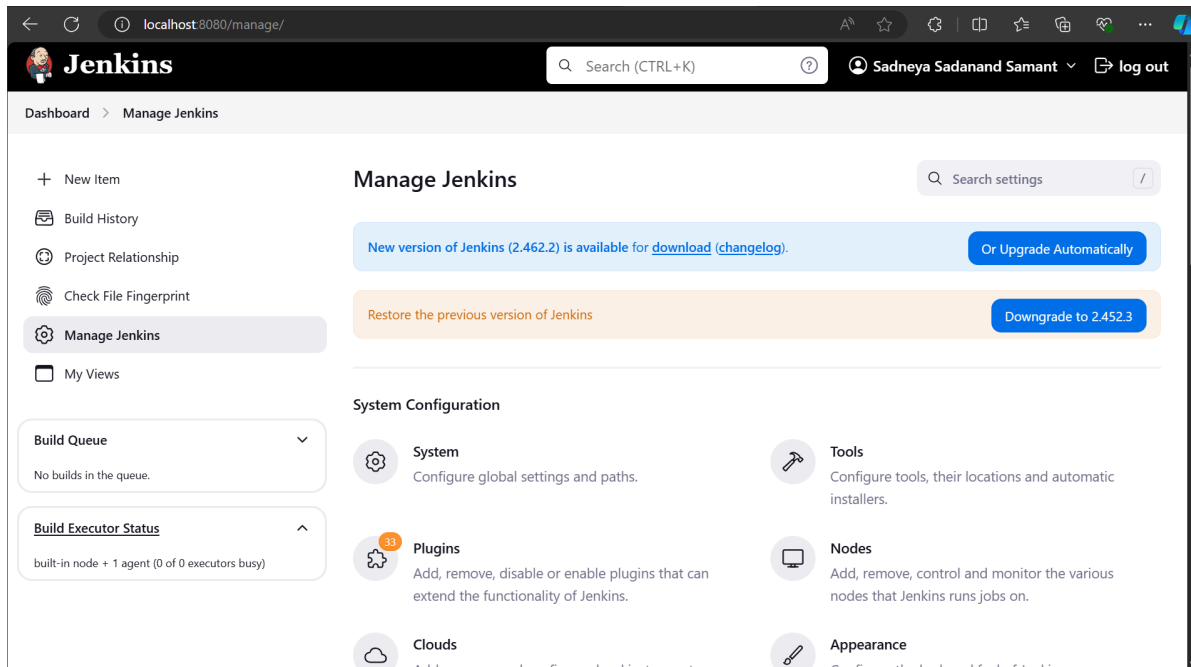
**\*\*Copy and paste this token as it will be used ahead in Jenkins.\*\***

## 2. Creation Freestyle project on Jenkins

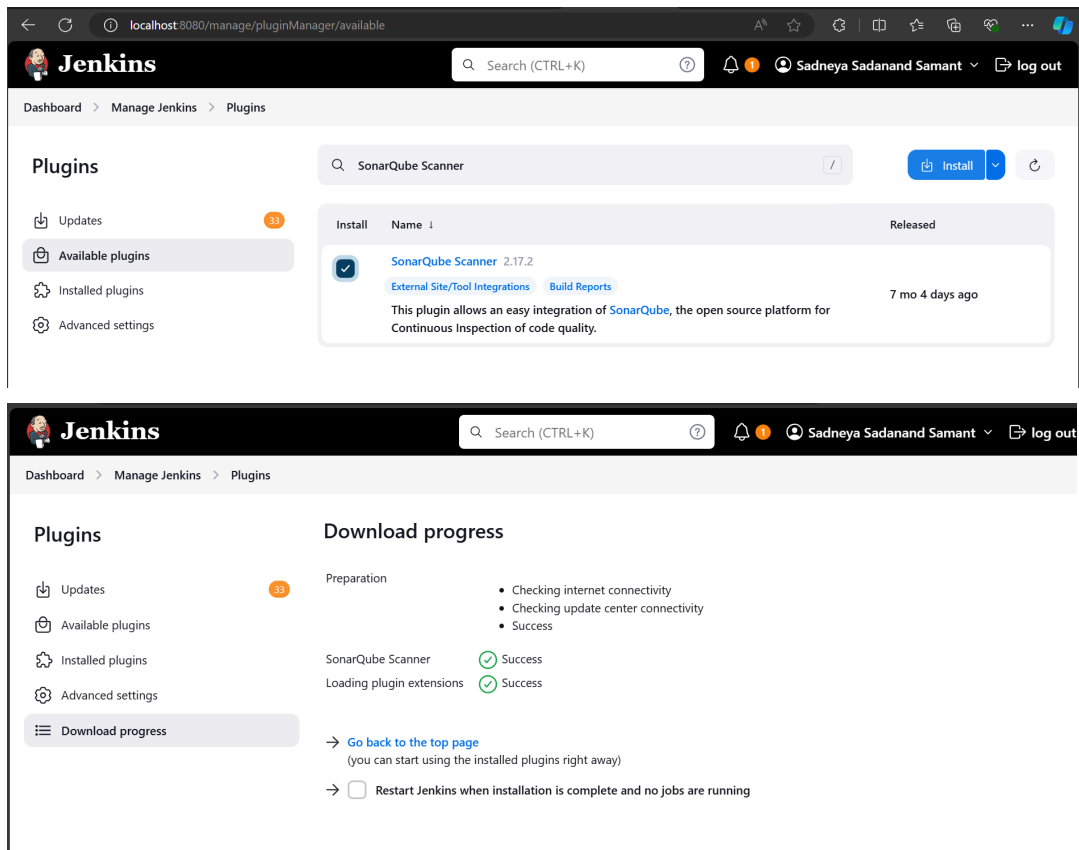
1. Run the jenkins at <http://localhost:8080> and enter username and password and click on sign in



2. Then Click on manage Jenkins.



3. Here click on Plugins. Inside Available Plugins search for SonarQube Scanner and then click on Install. Then again restart Jenkins.



- Then go back to Manage Jenkins and click on system and search for sonarqube server.

**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.

- none -

+ Add

Advanced

- Click on the environment variables and give name to SonarQube installation.(here I have given the name as “sonarqube local”) then give the server URL as <https://localhost:9000> then save and apply these changes.
- Then select item type Freestyle Project and give name to your project here I have given name “advdevops project”.Then click on ok.

localhost:8080/view/all/newJob

**Jenkins** Search (CTRL+K) Sadneya Sadanand Samant log out

Dashboard > All > New Item

**New Item**

Enter an item name

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.

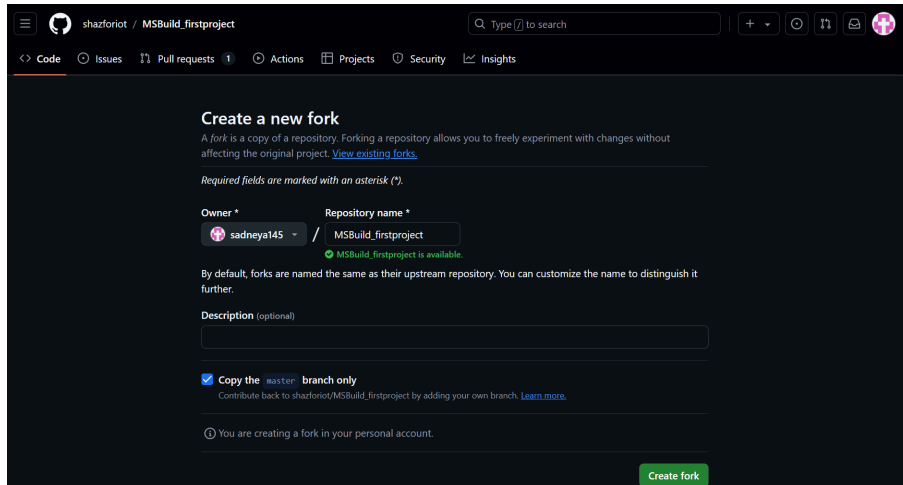
**Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

**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.

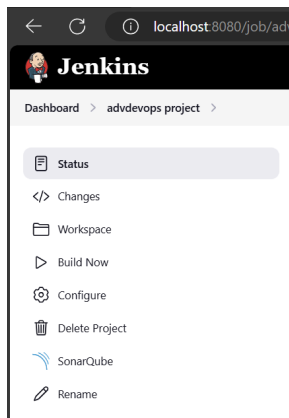
OK

- Choose the Github repository : [https://github.com/shazforiot/MSBuild\\_firstproject.git](https://github.com/shazforiot/MSBuild_firstproject.git)

It is a sample hello-world project with no vulnerabilities and issues.It is just for testing purposes.  
Fork This Repository.



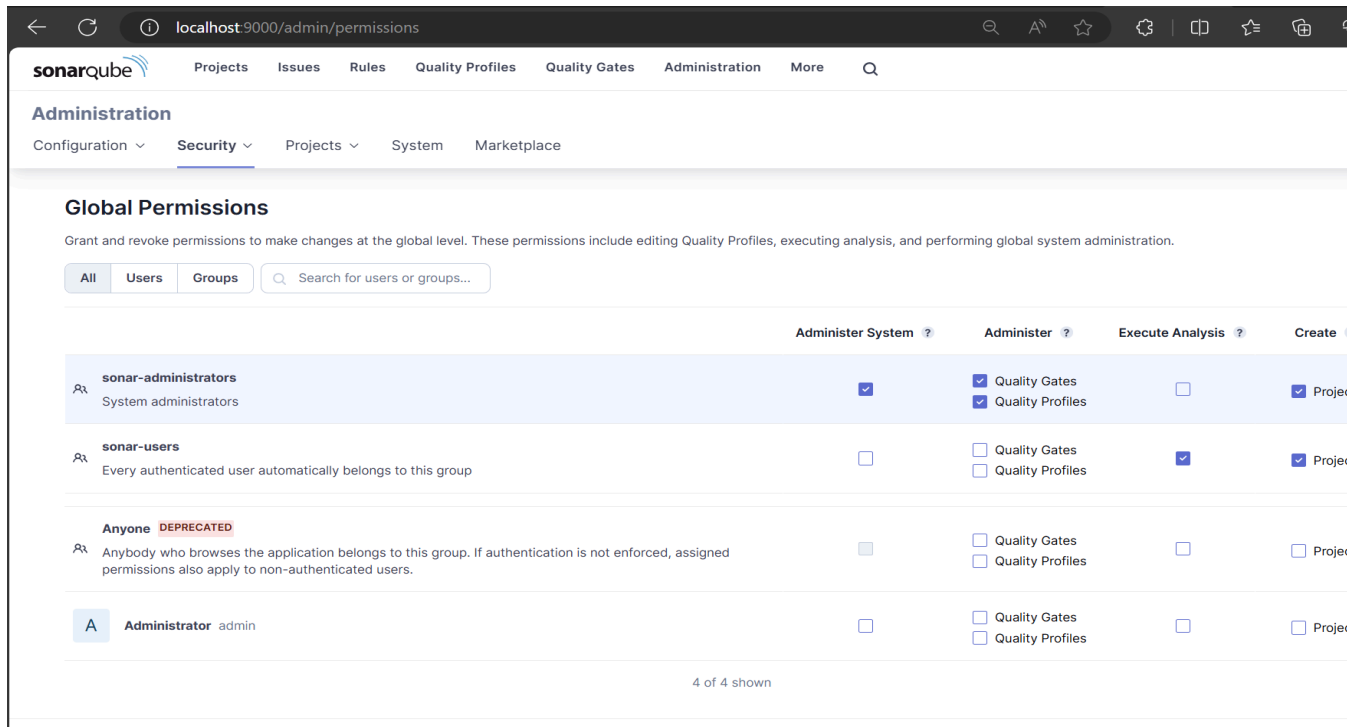
8. Then click on your project and then go to configure.



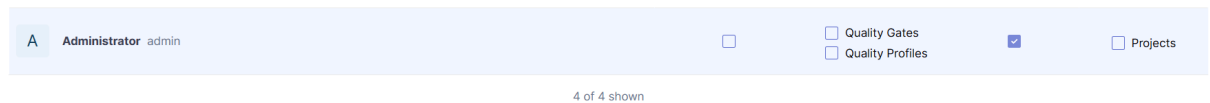
9. Then enable Git and add that github Repository URL.select \*/master branch.



10. Go to Sonarqube [http://localhost:9000/<user\\_name>/permissions](http://localhost:9000/<user_name>/permissions).



Then allow execute permissions to Admin User.



11. Then again Go inside the project and click on configure (addevops project->configure).Go there in the build steps section. Inside It add the analysis properties.

sonar.host.url=http://localhost:9000 (Your\_url)

sonar.login=sqa\_0d6700917b11bb2a57f2cc2189e162f6bbf26929 (login\_token that You have copied on clipboard created in step 7 of creation of project of sonarqube)

sonar.sources=.

sonar.projectKey=sonarqube(project name)

Then click on save and apply.



localhost:8080/job/advdevops%20project/configure

Dashboard > advdevops project > Configuration

### Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps**
- Post-build Actions

### Build Steps

**Execute SonarQube Scanner**

**JDK** ?  
JDK to be used for this SonarQube analysis  
(Inherit From Job)

**Path to project properties** ?

**Analysis properties** ?  
sonar.host.url=http://localhost:9000  
sonar.login=sqa\_0d6700917b11bb2a57f2cc2189e162f6bbf26929  
sonar.sources=/  
sonar.projectKey=sonarqube

**Additional arguments** ?

**JVM Options** ?

Add build step ▾

Save Apply

12. Then again go into manage Jenkins and check the number of executors if it is zero then set to two or more.If it is zero then it will not build successfully and give error.

Dashboard > System >

### Maven Project Configuration

**Global MAVEN\_OPTS** ?

**Local Maven Repository** ?  
Default ("~/m2/repository", or the value of 'localRepository' in Maven's settings file, if defined)

**# of executors**  
0

**Labels**

**Usage** ?  
Use this node as much as possible

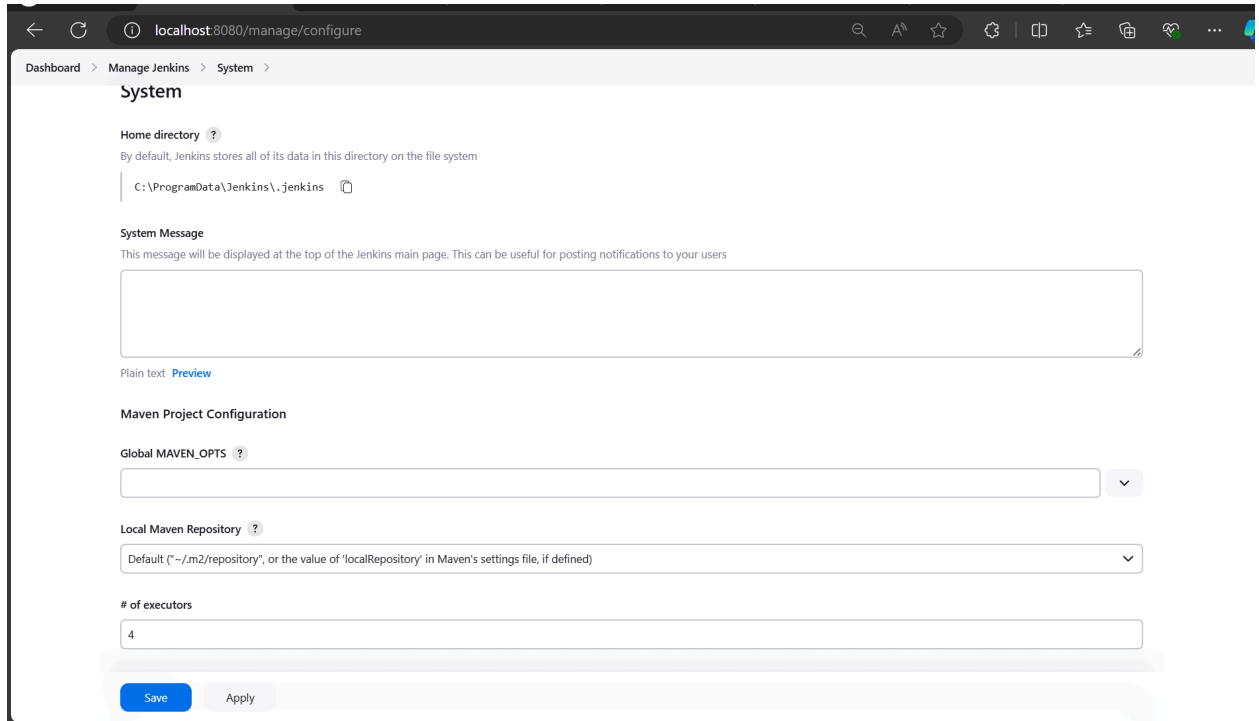
**Quiet period** ?  
5

**SCM checkout retry count**  
0

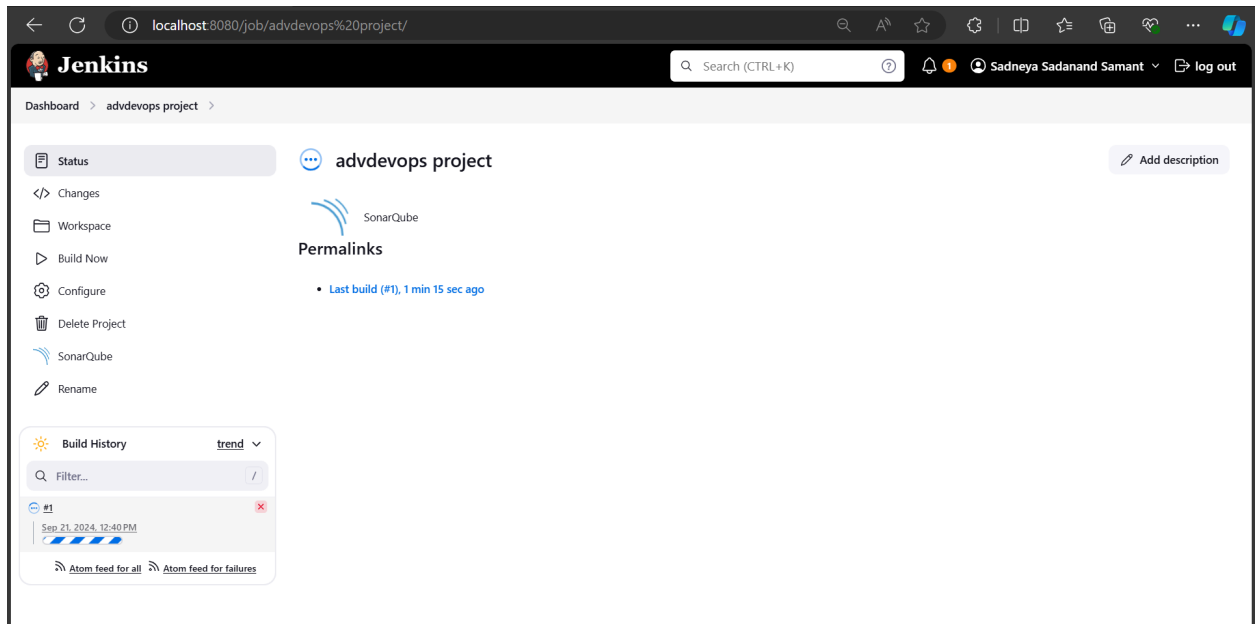
☐ Restrict request names

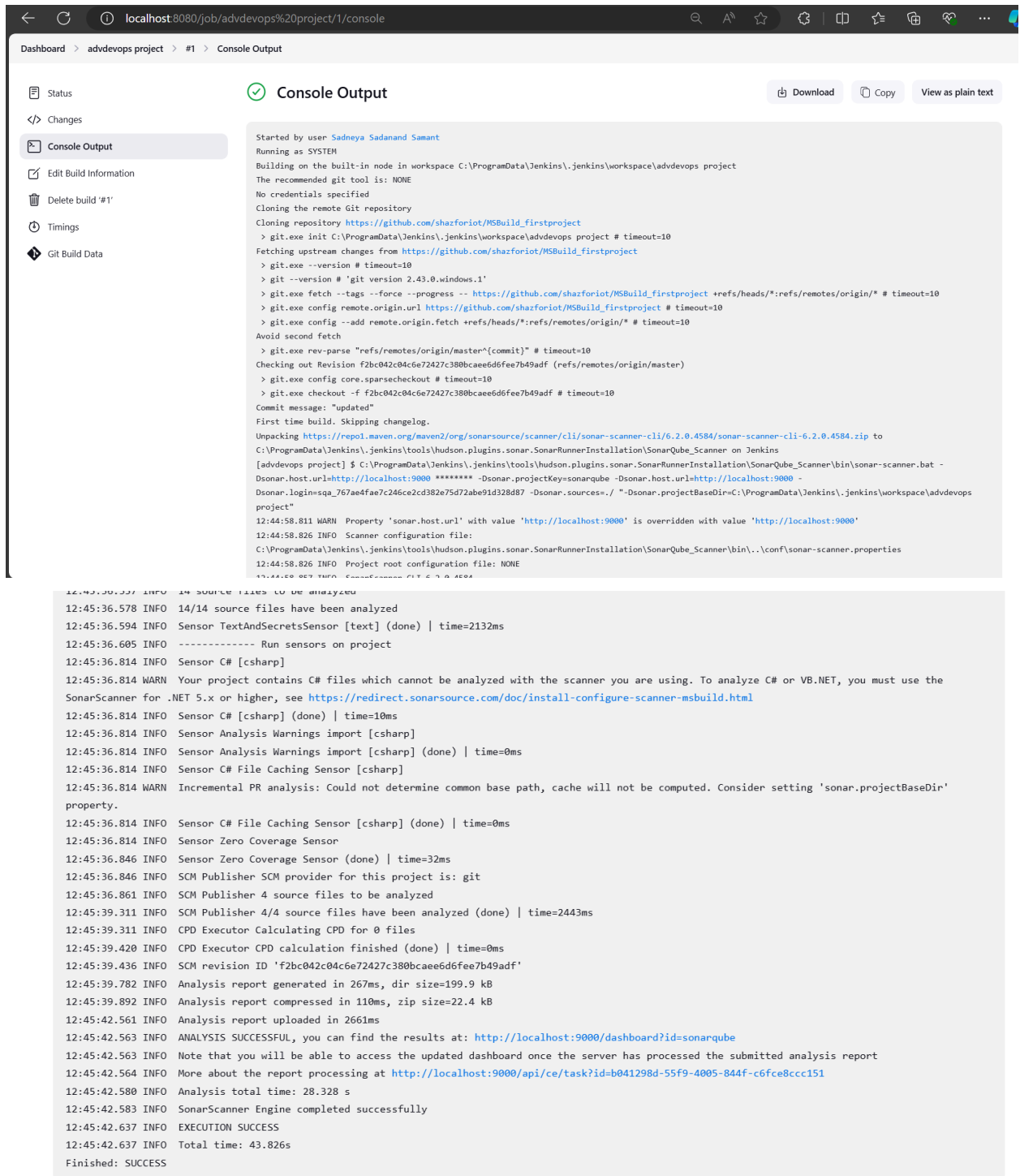
Save Apply

In the above image it was zero that's why my build was getting unsuccessful. Then I declared the value of # of executors to 4.



13. Then click on the build project. Thus the build is successful. see the console output.





```
Dashboard > advdevops project > #1 > Console Output

Status Console Output Download Copy View as plain text

Changes

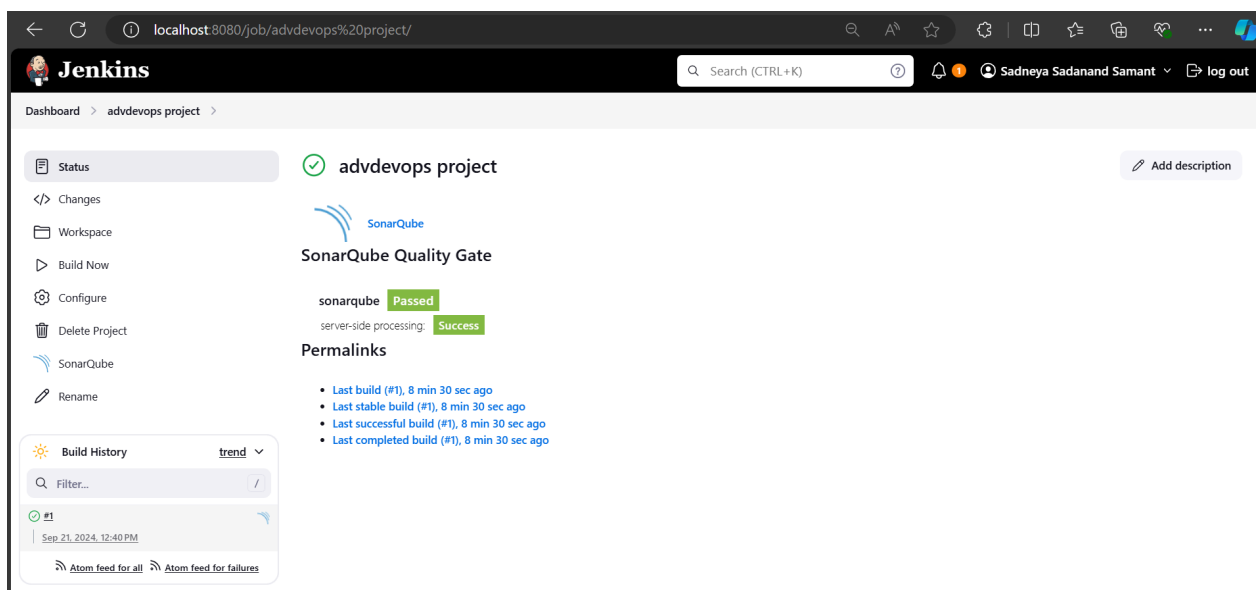
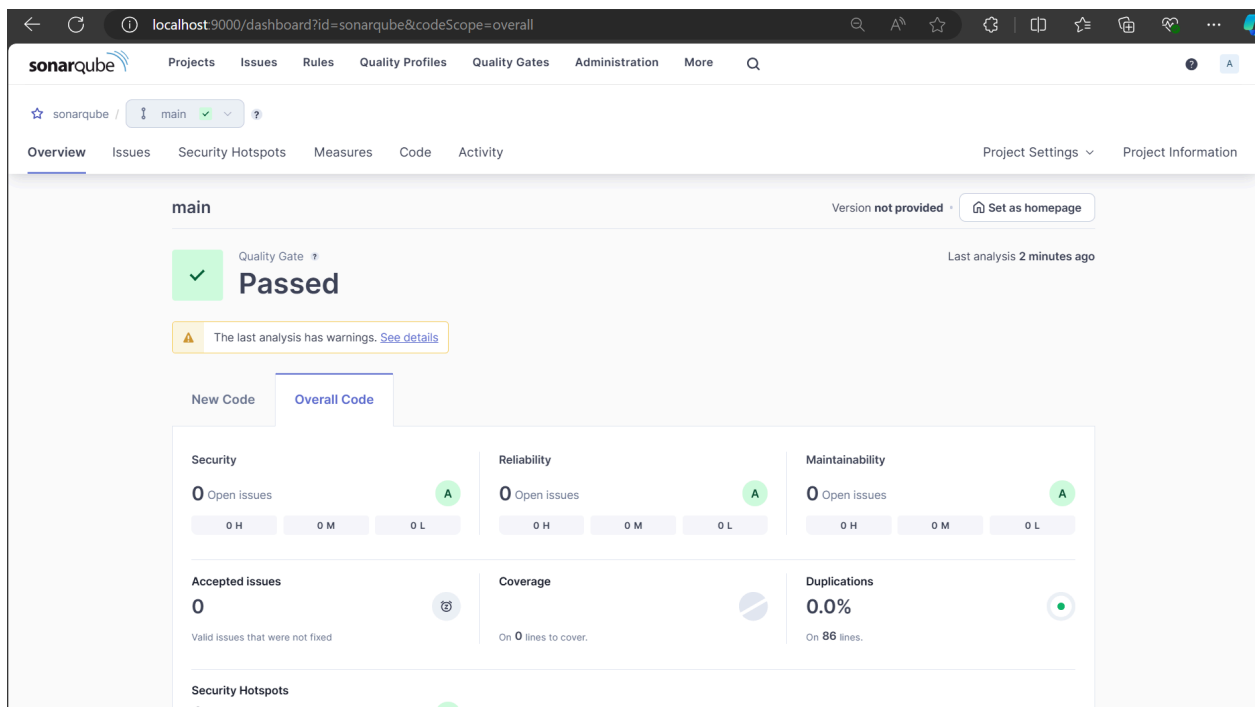
Delete build '#1'

Timings

Git Build Data

Started by user Sadneya Sadanand Samant
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\jenkins\workspace\advdevops project
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/shazforiot/MSBuild_firstproject
> git.exe init C:\ProgramData\Jenkins\jenkins\workspace\advdevops project # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject
> git.exe --version # timeout=10
> git --version # 'git version 2.43.0.windows.1'
> git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject # timeout=10
> git.exe config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> 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.
Unpacking https://repo1.maven.org/maven2/org/sonarsource/scanner/cli/sonar-scanner-cli/6.2.0.4584/sonar-scanner-cli-6.2.0.4584.zip to
C:\ProgramData\Jenkins\jenkins\tools\udson.plugins.sonar.SonarRunnerInstallation\SonarQube_Scanner on Jenkins
[advdevops project] $ C:\ProgramData\Jenkins\jenkins\tools\udson.plugins.sonar.SonarRunnerInstallation\SonarQube_Scanner\bin\sonar-scanner.bat -
Dsonar.host.url=http://localhost:9000 ***** -Dsonar.projectKey=sonarqube -Dsonar.host.url=http://localhost:9000 -
Dsonar.login=sqa_767ae4fae7c246ce2cd382e75d72abe91d328d87 -Dsonar.sources=./ "-Dsonar.projectBaseDir=C:\ProgramData\Jenkins\jenkins\workspace\advdevops
project"
12:44:58.811 WARN Property 'sonar.host.url' with value 'http://localhost:9000' is overridden with value 'http://localhost:9000'
12:44:58.826 INFO Scanner configuration file:
C:\ProgramData\Jenkins\jenkins\tools\udson.plugins.sonar.SonarRunnerInstallation\SonarQube_Scanner\bin\...\conf\sonar-scanner.properties
12:44:58.826 INFO Project root configuration file: NONE
12:44:58.827 INFO Execution file: C:\ProgramData\Jenkins\jenkins\workspace\advdevops project\
12:45:36.557 INFO 14 source files to be analyzed
12:45:36.578 INFO 14/14 source files have been analyzed
12:45:36.594 INFO Sensor TextAndSecretsSensor [text] (done) | time=2132ms
12:45:36.605 INFO ----- Run sensors on project
12:45:36.814 INFO Sensor C# [csharp]
12:45:36.814 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
12:45:36.814 INFO Sensor C# [csharp] (done) | time=10ms
12:45:36.814 INFO Sensor Analysis Warnings import [csharp]
12:45:36.814 INFO Sensor Analysis Warnings import [csharp] (done) | time=0ms
12:45:36.814 INFO Sensor C# File Caching Sensor [csharp]
12:45:36.814 WARN Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir'
property.
12:45:36.814 INFO Sensor C# File Caching Sensor [csharp] (done) | time=0ms
12:45:36.814 INFO Sensor Zero Coverage Sensor
12:45:36.846 INFO Sensor Zero Coverage Sensor (done) | time=32ms
12:45:36.846 INFO SCM Publisher SCM provider for this project is: git
12:45:36.861 INFO SCM Publisher 4 source files to be analyzed
12:45:39.311 INFO SCM Publisher 4/4 source files have been analyzed (done) | time=2443ms
12:45:39.311 INFO CPD Executor Calculating CPD for 0 files
12:45:39.420 INFO CPD Executor CPD calculation finished (done) | time=0ms
12:45:39.436 INFO SCM revision ID 'f2bc042c04c6e72427c380bcae6d6fee7b49adf'
12:45:39.782 INFO Analysis report generated in 267ms, dir size=199.9 kB
12:45:39.892 INFO Analysis report compressed in 110ms, zip size=22.4 kB
12:45:42.561 INFO Analysis report uploaded in 2661ms
12:45:42.563 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube
12:45:42.563 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
12:45:42.564 INFO More about the report processing at http://localhost:9000/api/ce/task?id=b041298d-55f9-4005-844f-c6fce8ccc151
12:45:42.580 INFO Analysis total time: 28.328 s
12:45:42.583 INFO SonarScanner Engine completed successfully
12:45:42.637 INFO EXECUTION SUCCESS
12:45:42.637 INFO Total time: 43.826s
Finished: SUCCESS
```

14. Then go to SonarQube. Then go inside the project that you created. where it will show output passed.



Thus Project Build successfully.

**Conclusion:** Here we created the sonarqube project locally successfully. Then created a freestyle project on Jenkins and installed a plugin named sonarqube scanner. Then we build that project by firstly adding github repository and mentioning Analysis properties which is nothing but name, key, token. Then after saving and applying the changes, the build executed successfully. Then we also checked that sonarqube given response about passing of analysis. Thus Build is done successfully.