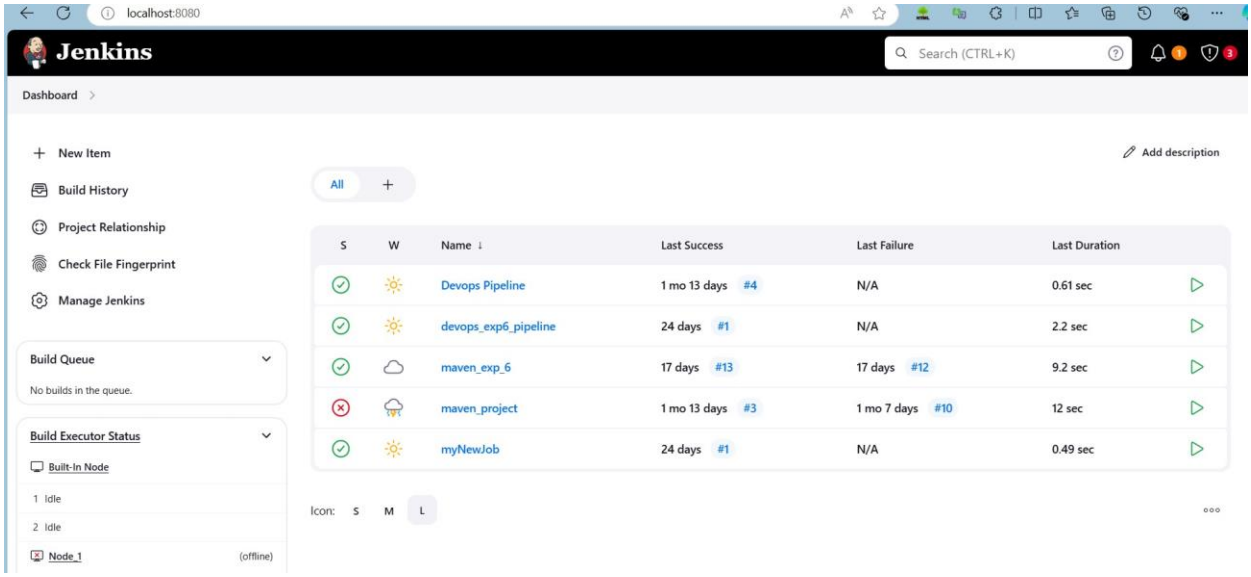
Name : Aditya Kirtane , Div:D15C , Roll No.26

Adv DevOps Practical 7

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

Steps to integrate Jenkins with SonarQube

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



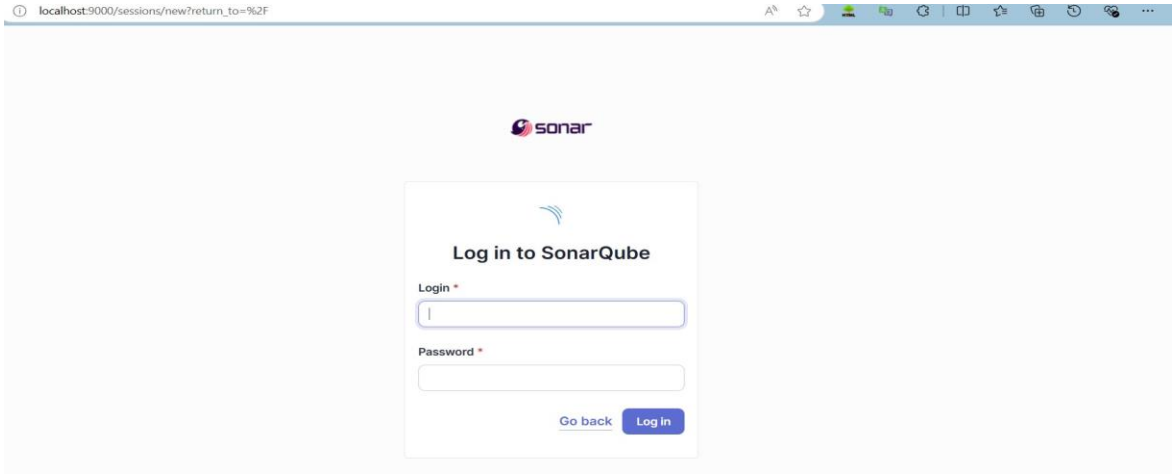
2. Run SonarQube in a Docker container using this command -
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

----------Warning: run below command only once

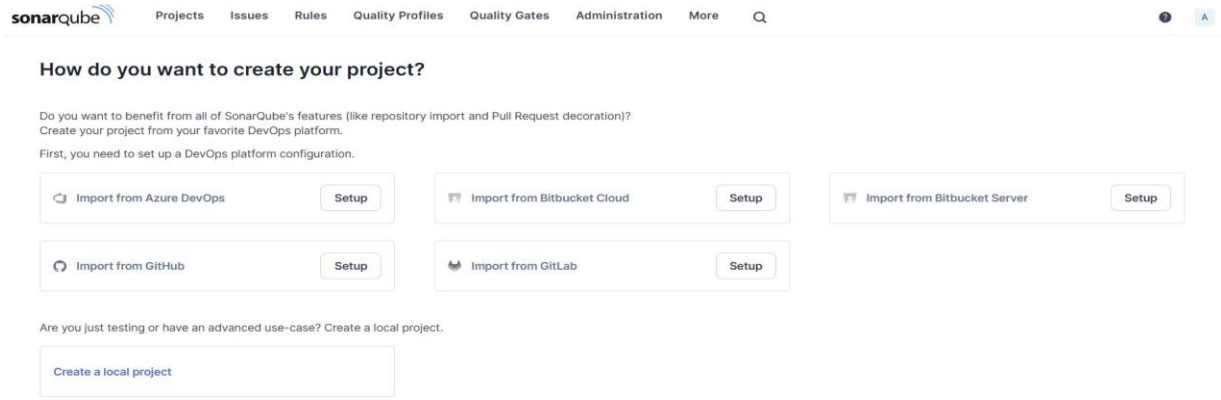
```
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Ayush Maurya> docker run -d --name sonarqube -e SINAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
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:bb444c58c1e04d8a147a3bb12af941c57e0100a5b21d10e599384d59bed36c86
Status: Downloaded newer image for sonarqube:latest
4af48468290f95b22362652ee37b96c935b0bed754945c62cf3b0d5d51a2ac0c
PS C:\Users\Ayush Maurya>
```

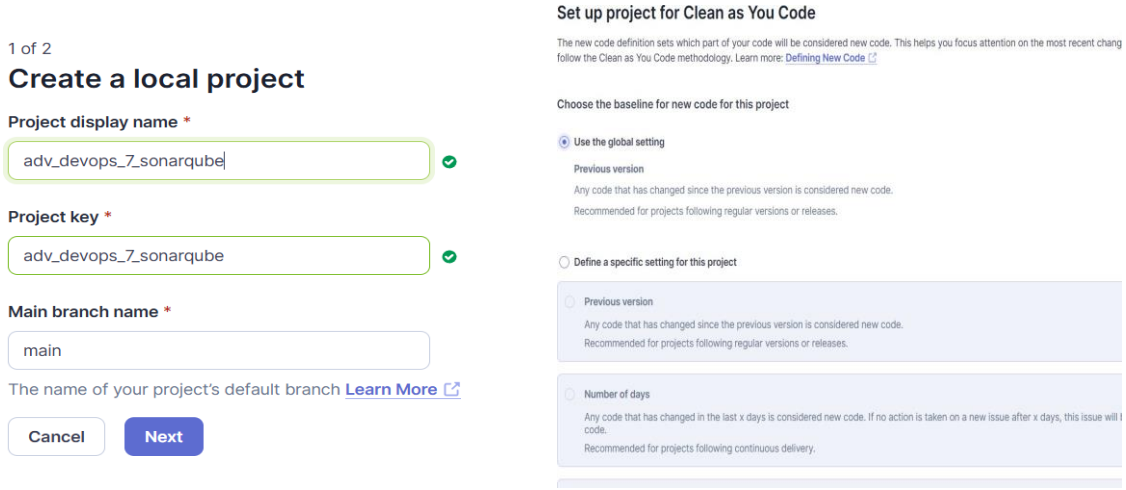
3. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.



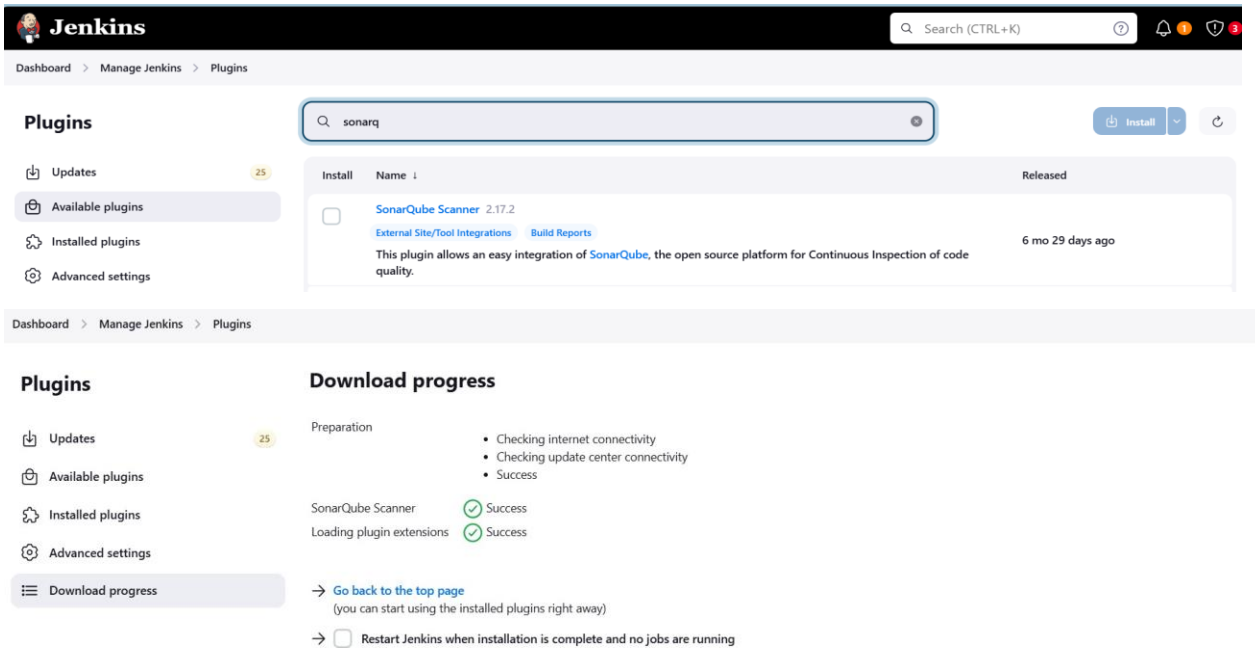
4. Login to SonarQube using username admin and password admin.



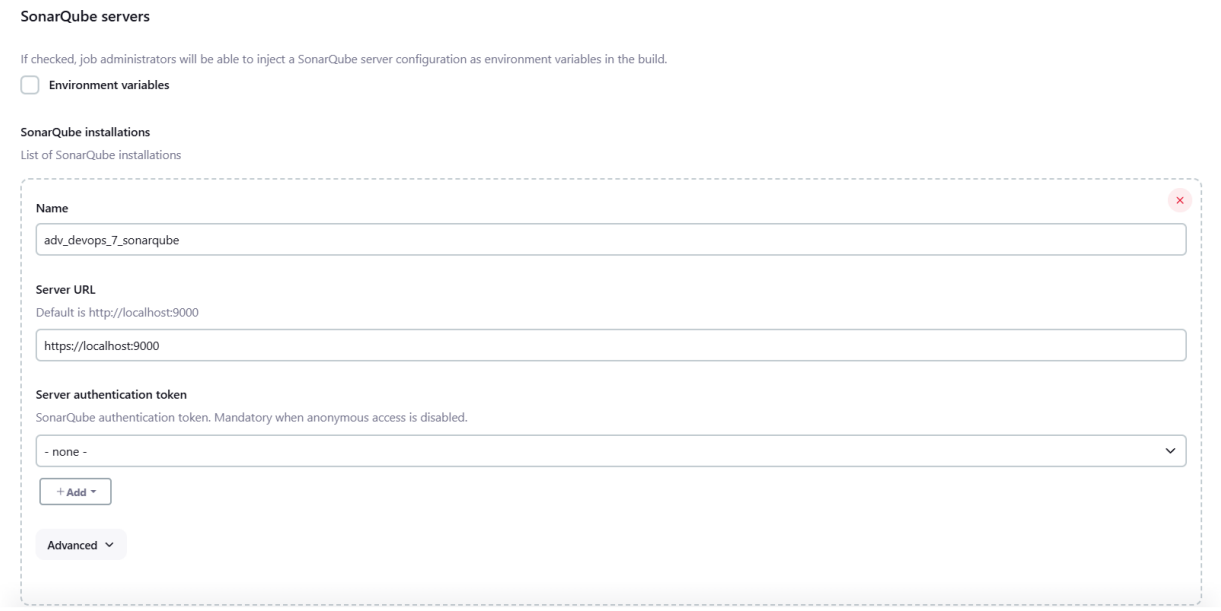
5. Create a manual project in SonarQube with the name sonarqube



Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.

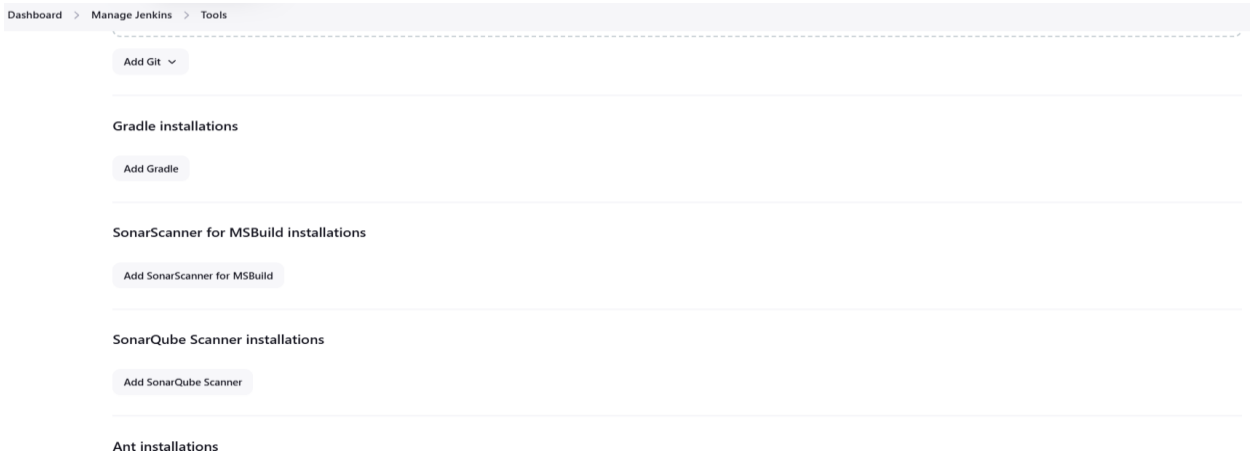


6. Under Jenkins ‘Manage Jenkins’ then go to ‘system’, scroll and look for **SonarQube Servers** and enter the details.
In SonarQube installations: Under **Name** add <project name of sonarqube> for me **adv_devops_7_sonarqube**
In **Server URL** Default is **http://localhost:9000**

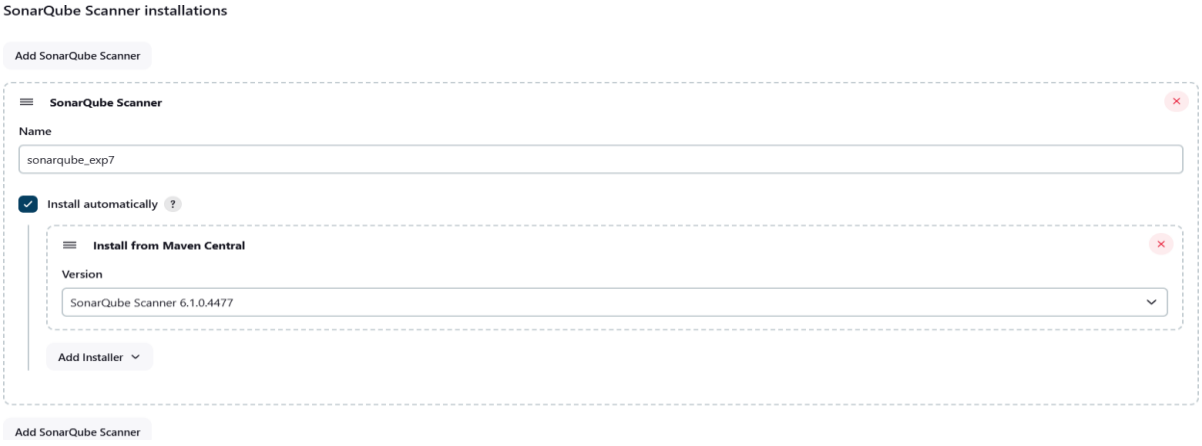


7. Search for SonarQube Scanner under Global Tool Configuration.

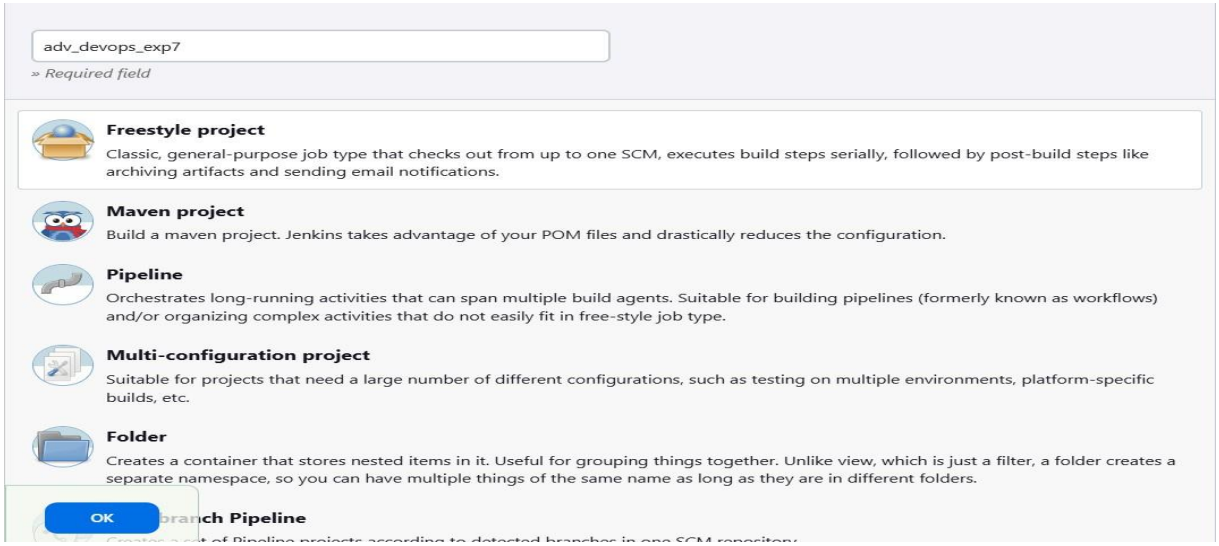
Dashboard > Manage Jenkins > Tools



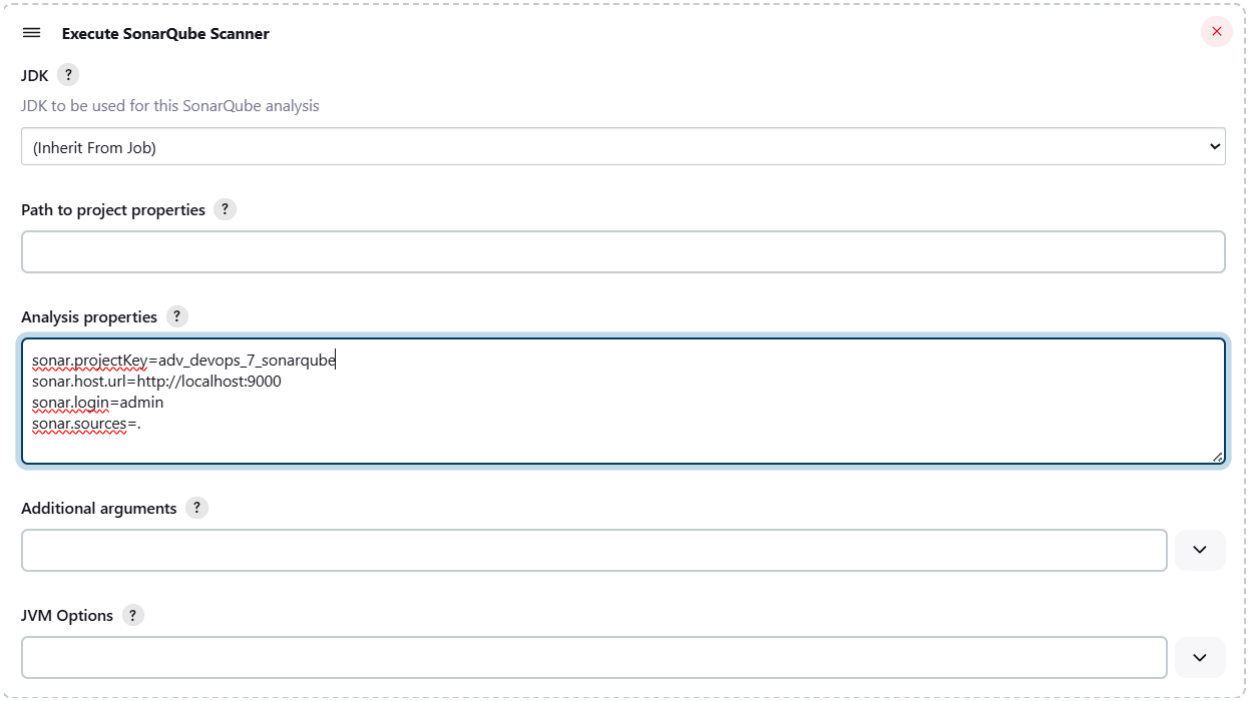
Check the “Install automatically” option. → Under name any name as identifier → Check the “Install automatically” option.



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

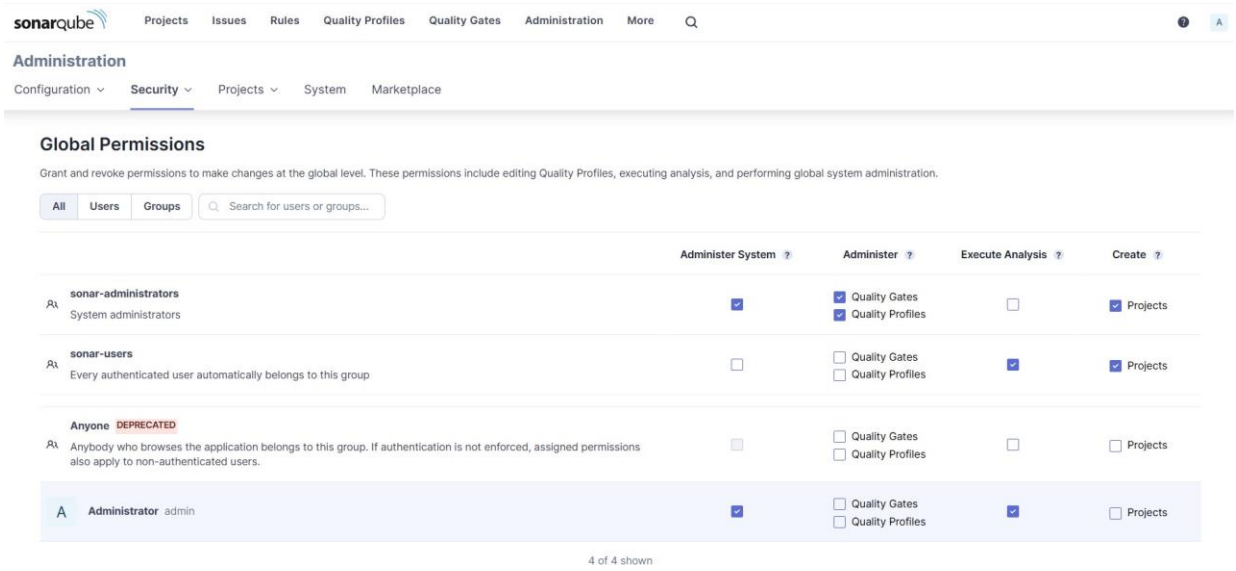


9. Choose this GitHub repository in Source Code Management.
https://github.com/shazforiot/MSBuild_firstproject.git
It is a sample hello-world project with no vulnerabilities and issues, just to test the integration.



Then save

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



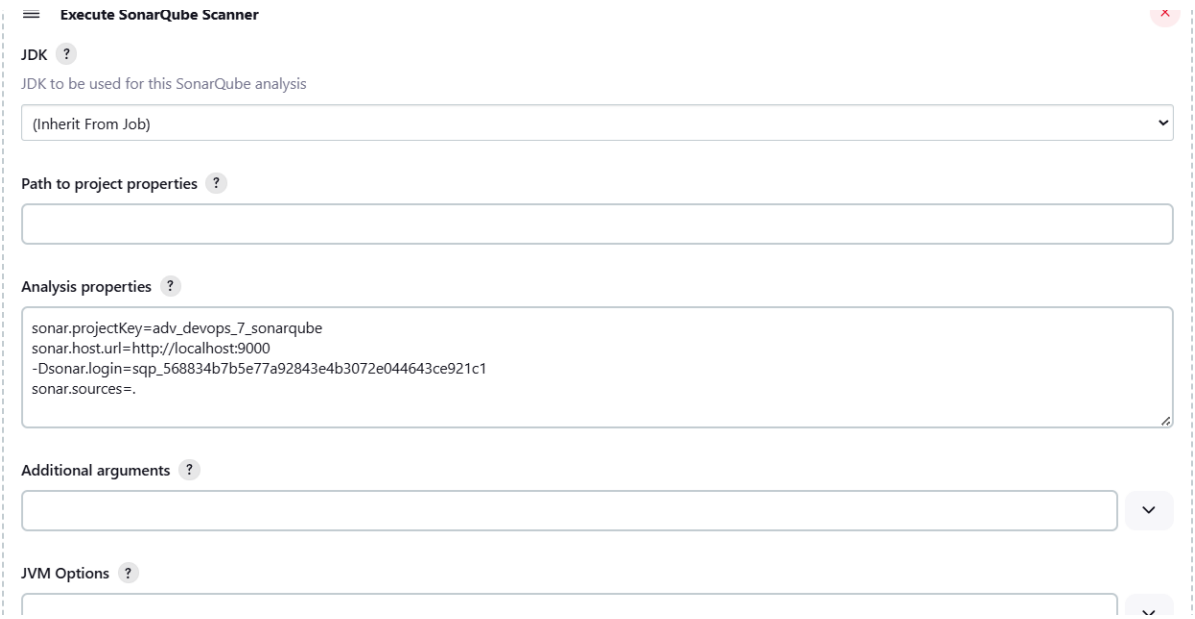
IF CONSOLE OUTPUT FAILED:

Step 1: Generate a New Authentication Token in SonarQube

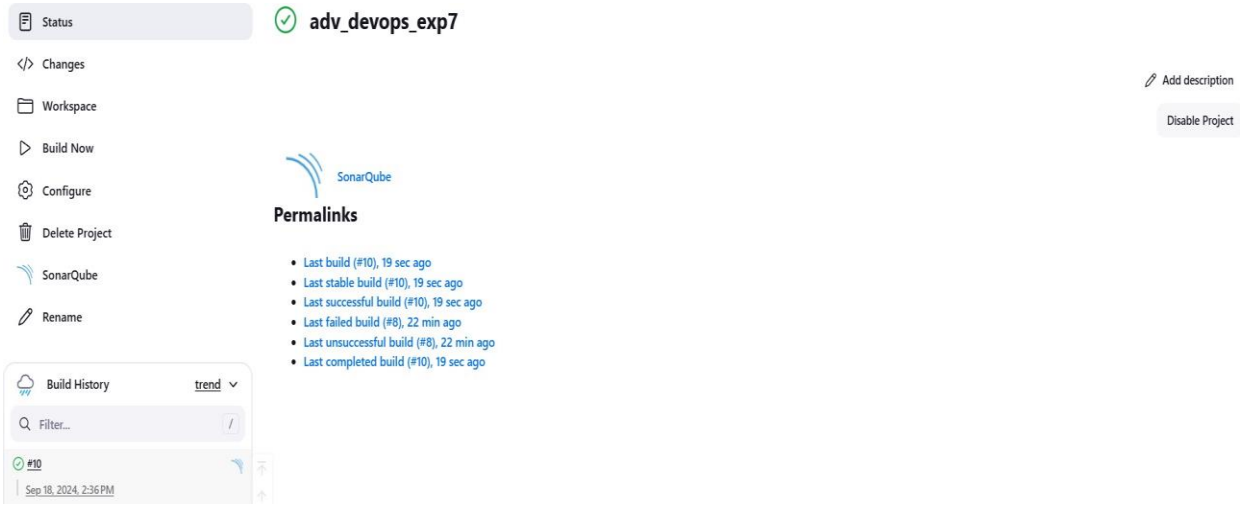
1. **Login to SonarQube:**
    - Open your browser and go to `http://localhost:9000`.
    - Log in with your admin credentials (default username is `admin`, and the password is either `admin` or your custom password if it was changed).
2. **Generate a New Token:**
    - Click on your **username** in the top-right corner of the SonarQube dashboard.
    - Select **My Account** from the dropdown menu.
    - Go to the **Security** tab.
    - Under **Generate Tokens**, type a name for the token (e.g., "Jenkins-SonarQube").
    - Click **Generate**.
    - Copy the token and save it securely. You will need it in Jenkins.

Step 2: Update the Token in Jenkins

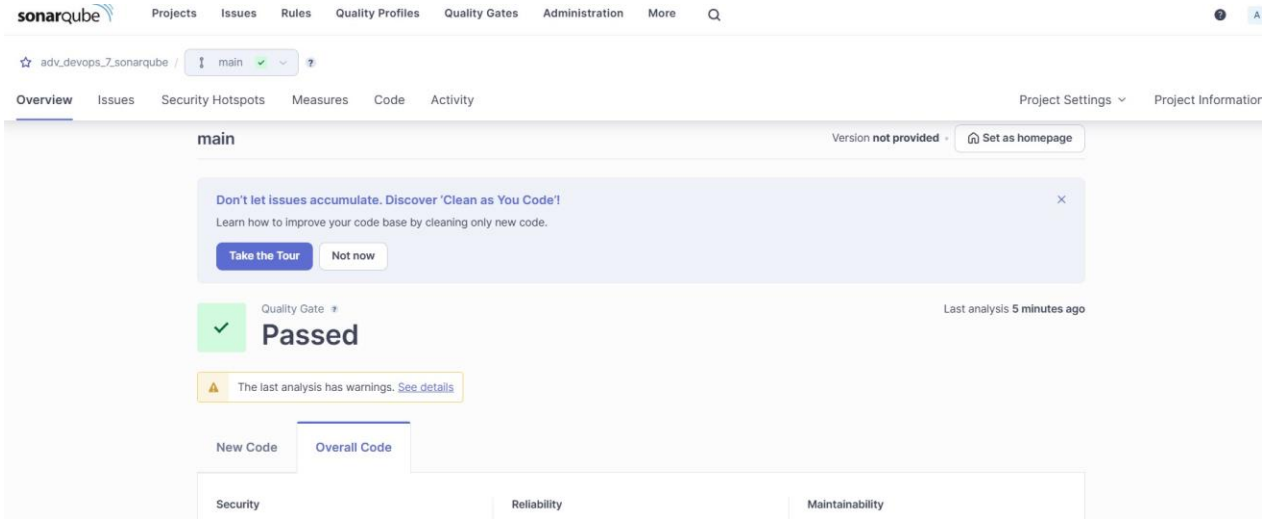
1. **Go to Jenkins Dashboard:**
2. **Configure the Jenkins Job:**
3. **Update the SonarQube Token:**



11. Run the Jenkins build.



12. Once the build is complete, check project on SonarQube



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

Conclusion:

In this project, we integrated Jenkins with SonarQube for automated static application security testing (SAST). We set up SonarQube using Docker, configured Jenkins with the necessary plugins and authentication, and linked it to a GitHub repository. The SonarQube scanner was added as a build step, enabling continuous code analysis for vulnerabilities, code smells, and quality issues, ensuring automated reporting and continuous code quality improvement.