Name: Shivam.R.Prajapati **Div:D15C** **Roll No: 41**

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
```

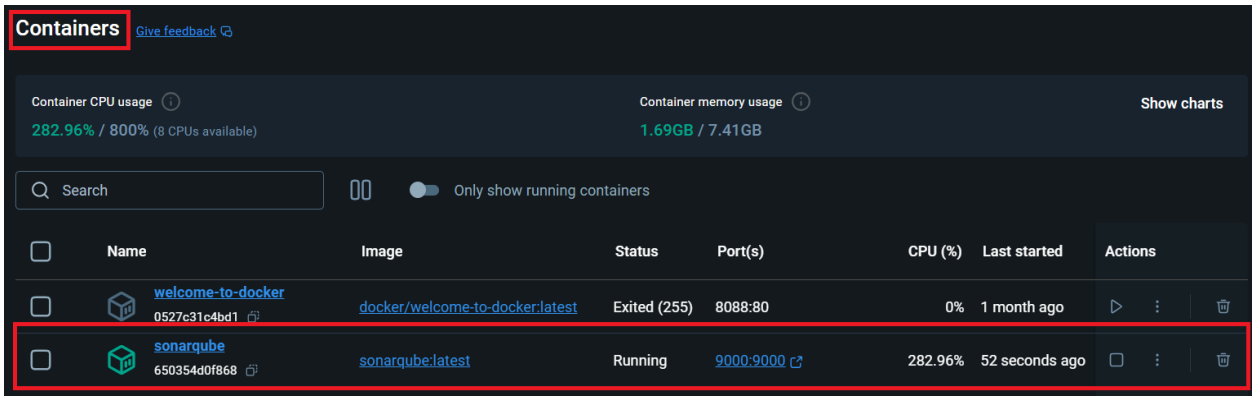
3) Make sure **Jenkins** is already installed on your system before starting the process. Jenkins will be used to automate tasks, like running SonarQube for code analysis. If Jenkins isn't installed yet, you can download and set it up from the official Jenkins website.

Name: Shivam.R.Prajapati **Div:D15C** **Roll No: 41**

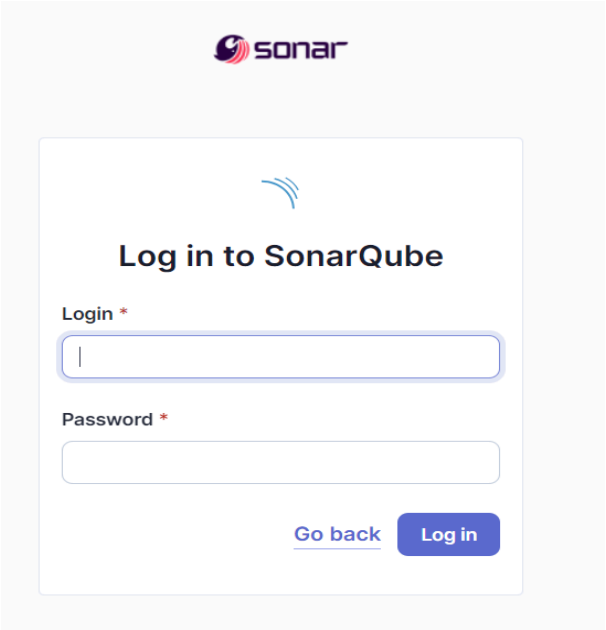
STEPS:

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

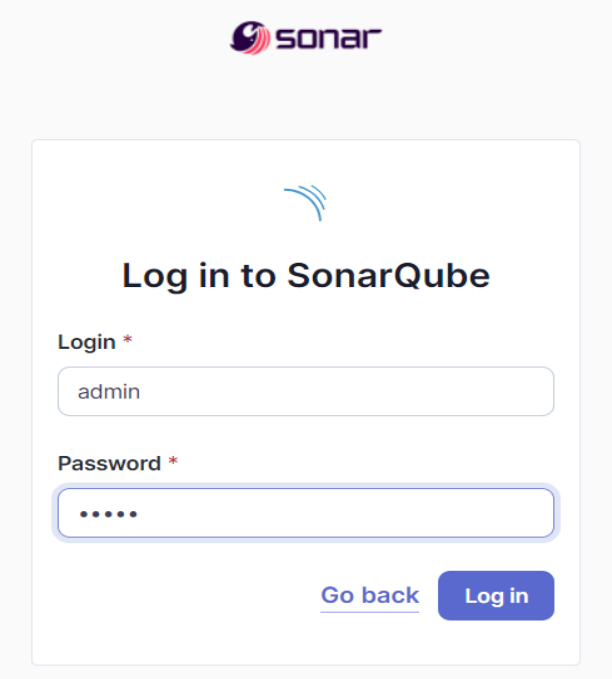
```
C:\Users\praja>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
650354d0f868ae4ad2d800426080076c604eb09f29b10d4a251aee70f51ce907

C:\Users\praja>
```



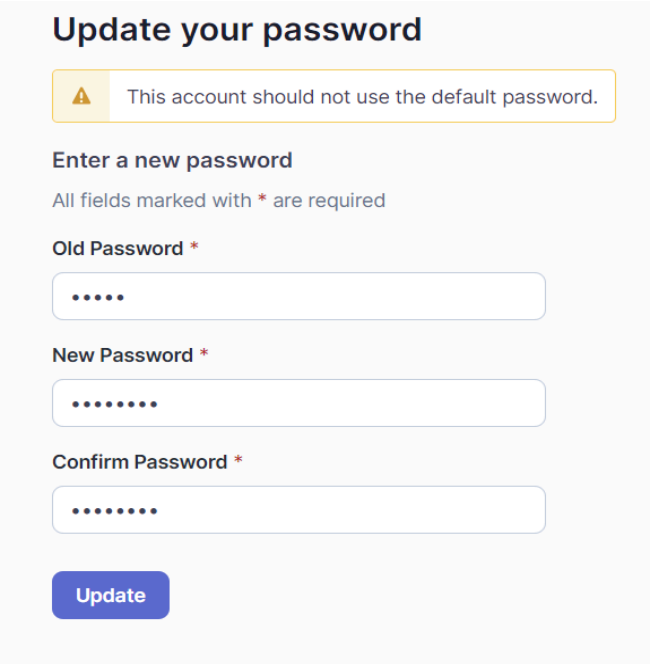
Step2: After starting the SonarQube image, open your browser and go to **http://localhost:9000** to access SonarQube.



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.



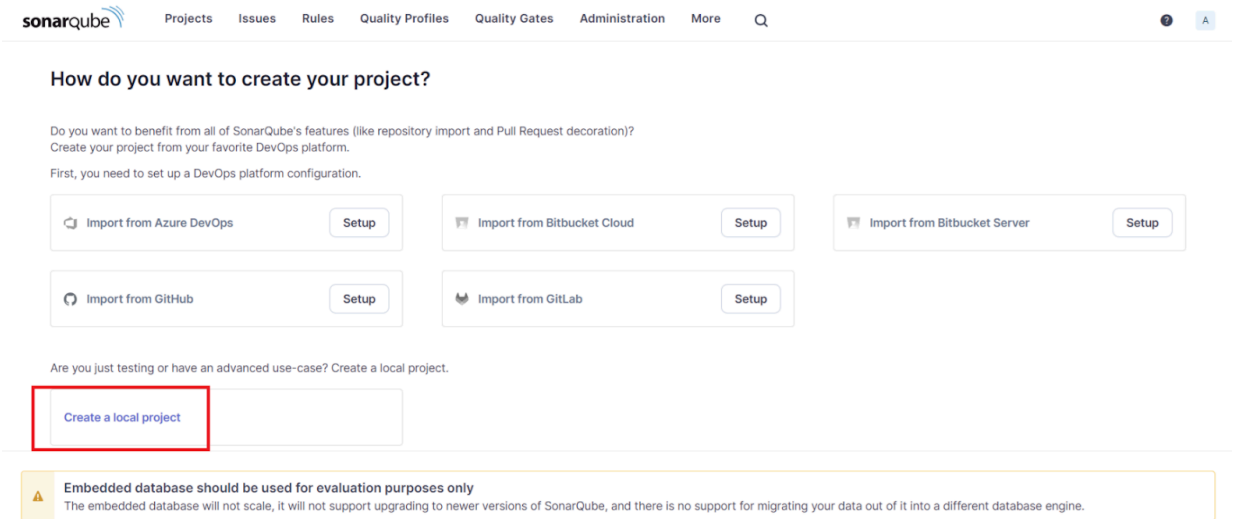
Click on Log in



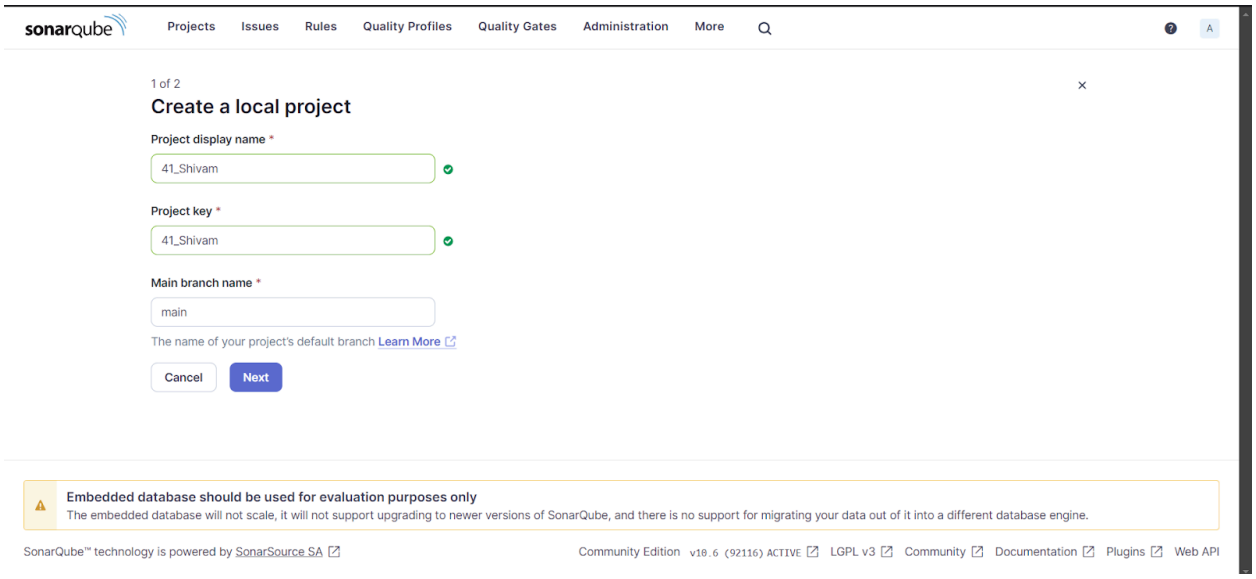
Click on Update .

Name: Shivam.R.Prajapati **Div:D15C** **Roll No: 41**

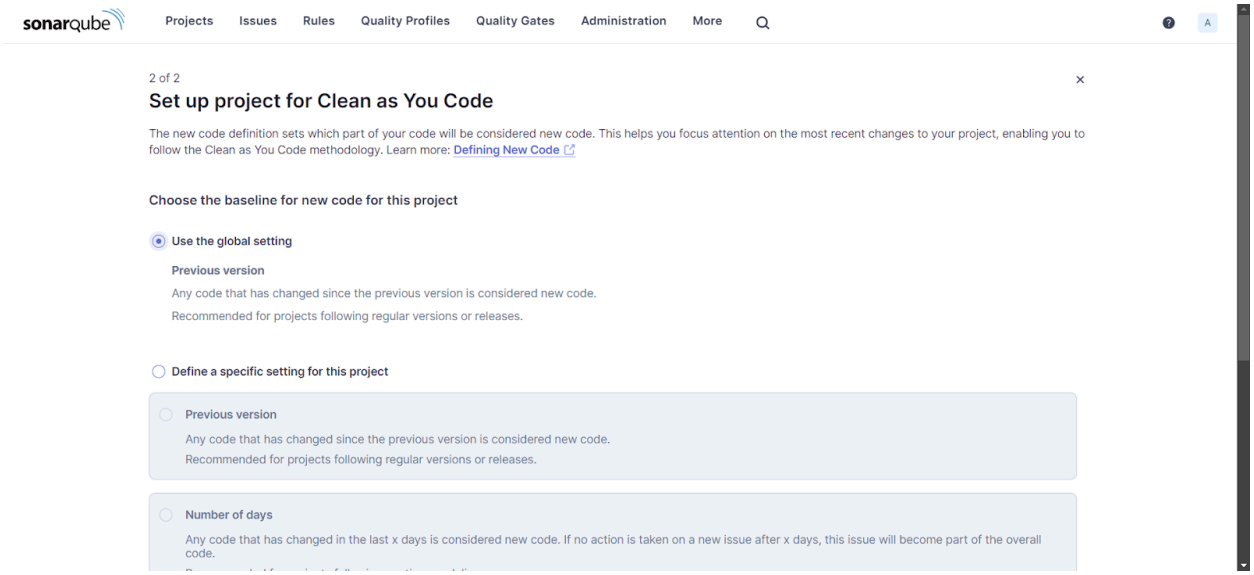
Step 4: After changing the password, you will be directed to this screen. Click on **Create a Local Project**.



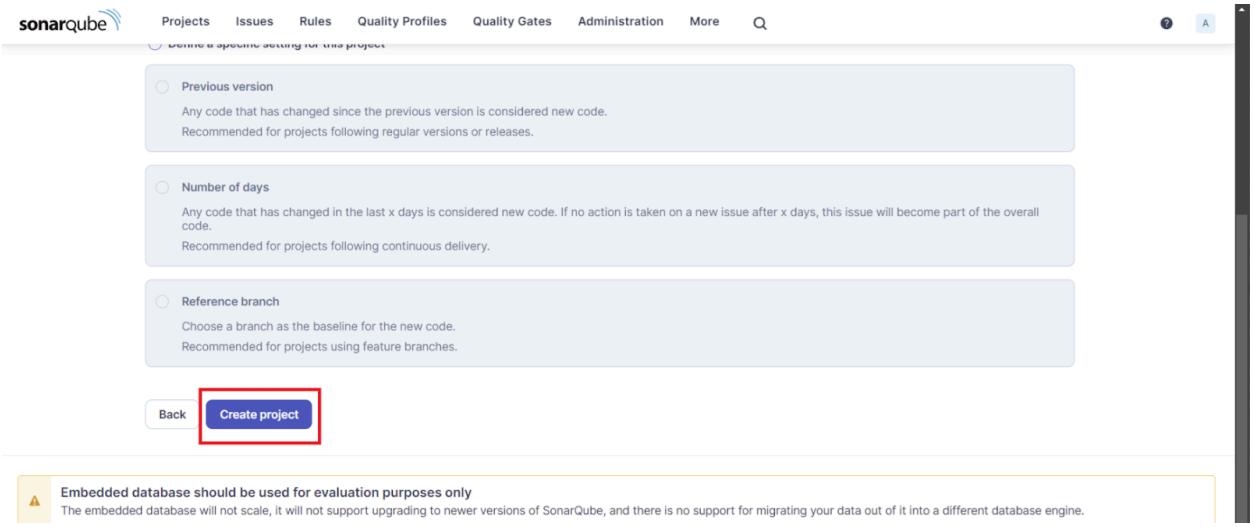
Step 5: Give your project , a display name and project key



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.



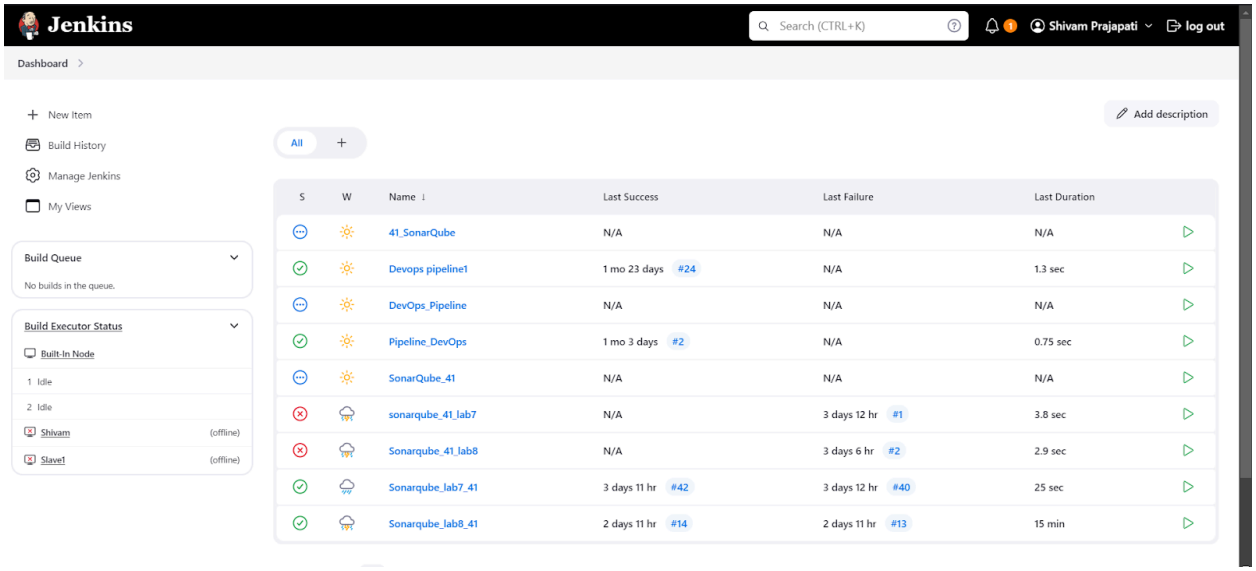
Scroll Down



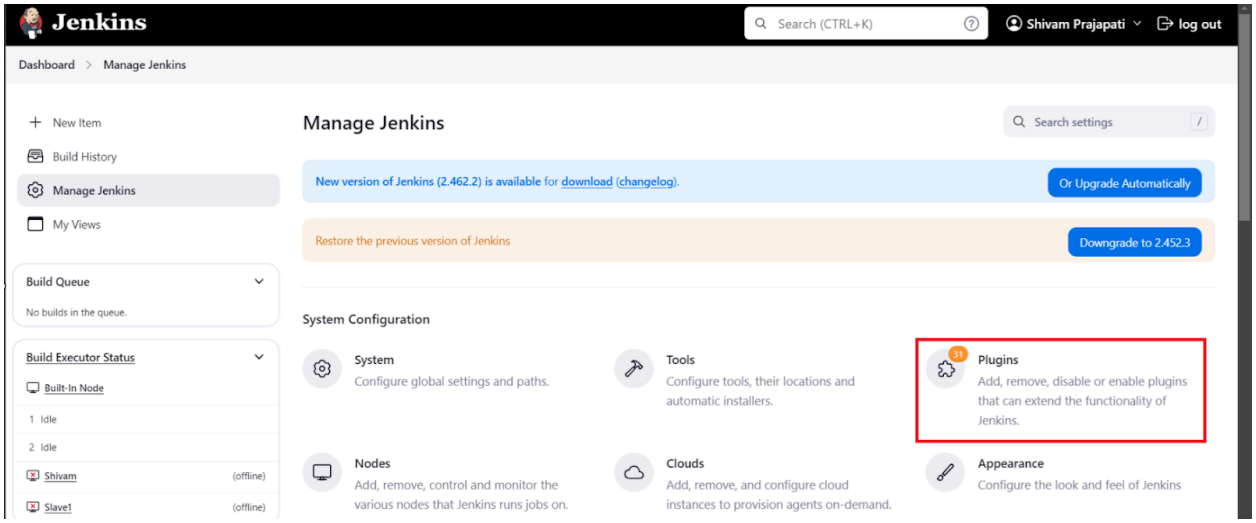
Click on Create project

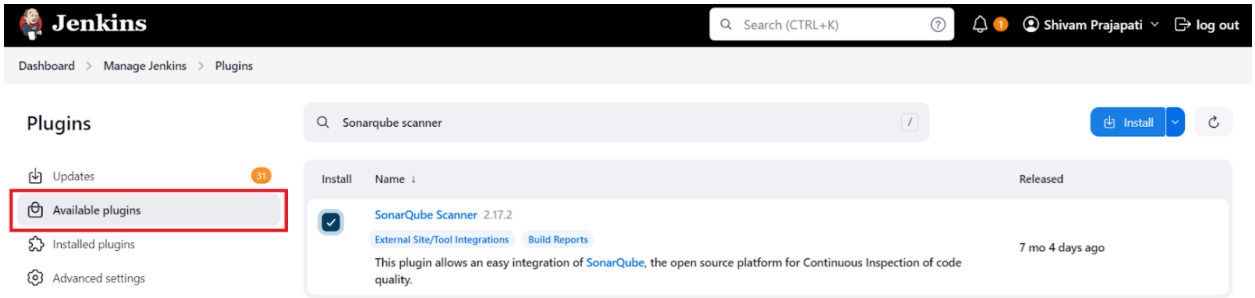
Name: Shivam.R.Prajapati **Div:D15C** **Roll No: 41**

Step 7: Open Jenkins by going to **http://localhost:<port_number>** in your browser, replacing <port_number> with the specific port Jenkins is running on.

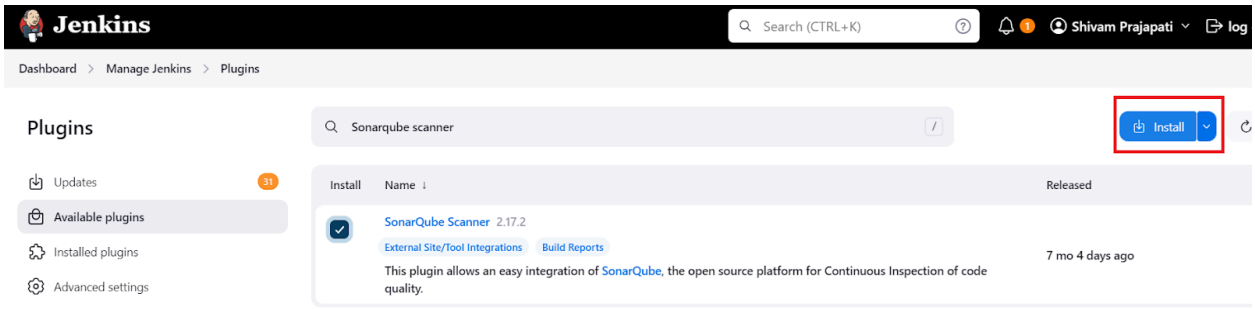


Step 8: Now go to Manage Jenkin then go for Plugins followed by Available plugins search for **Sonarqube Scanner** where we are going to install it as a plugin.

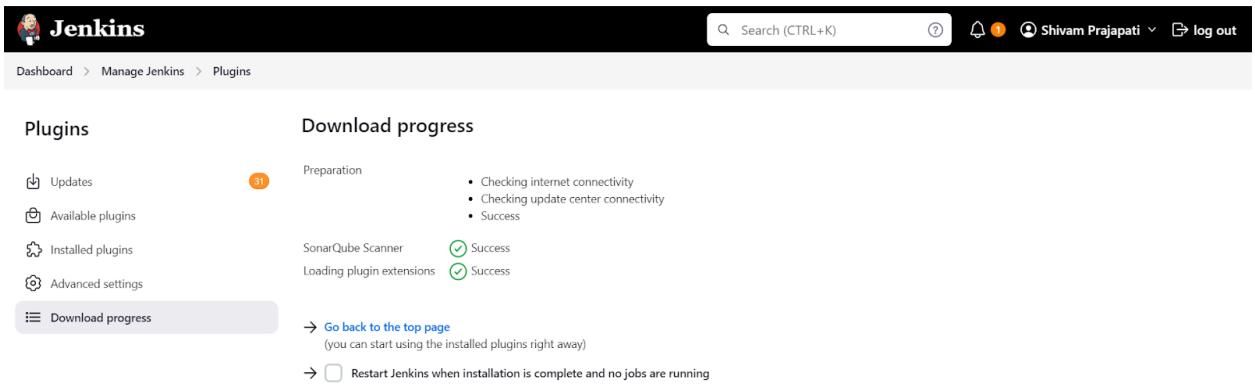




Click on Available Plugins.



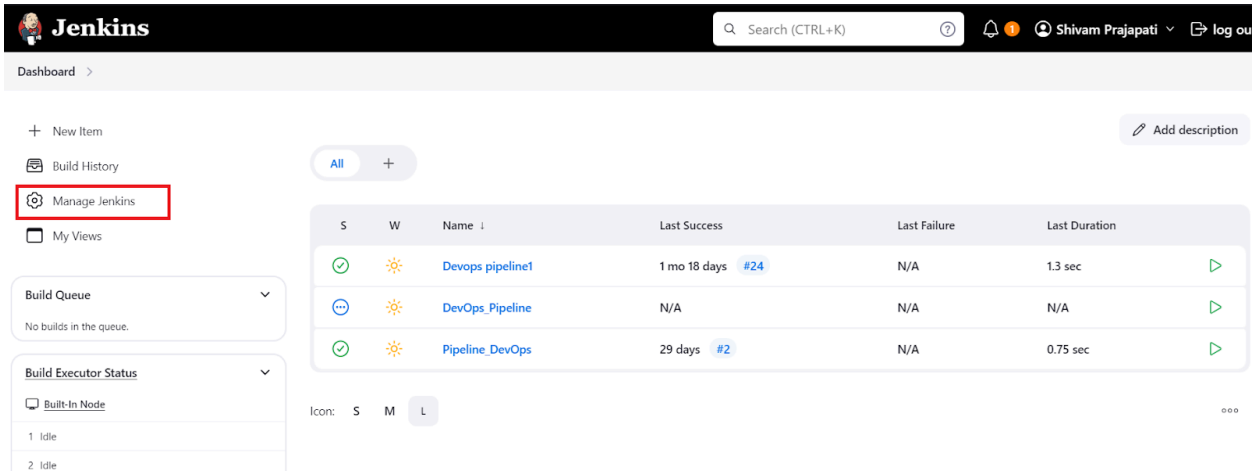
Search in the Search bar the required Plugin Name and click on Install.



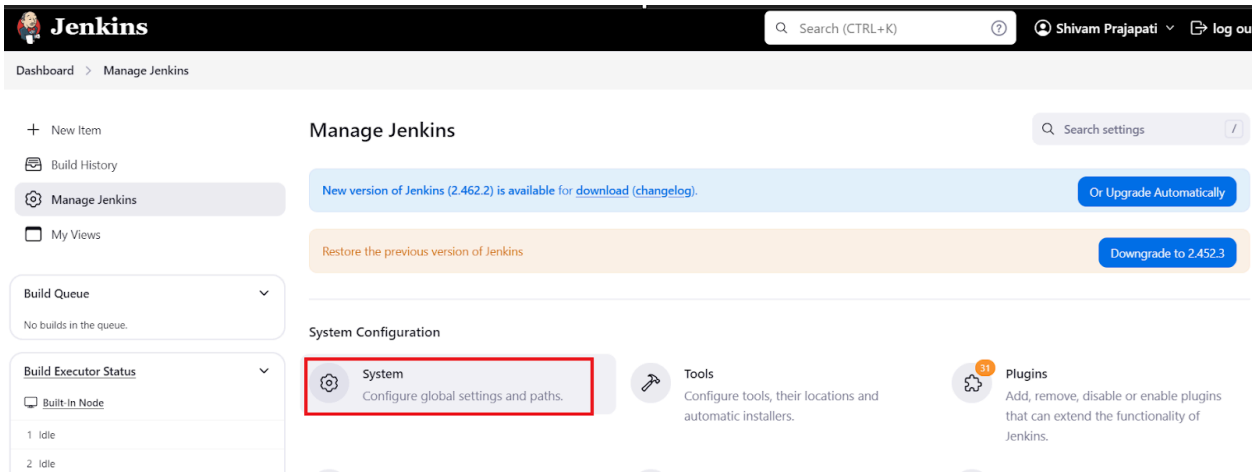
Plugin Installed Successfully.

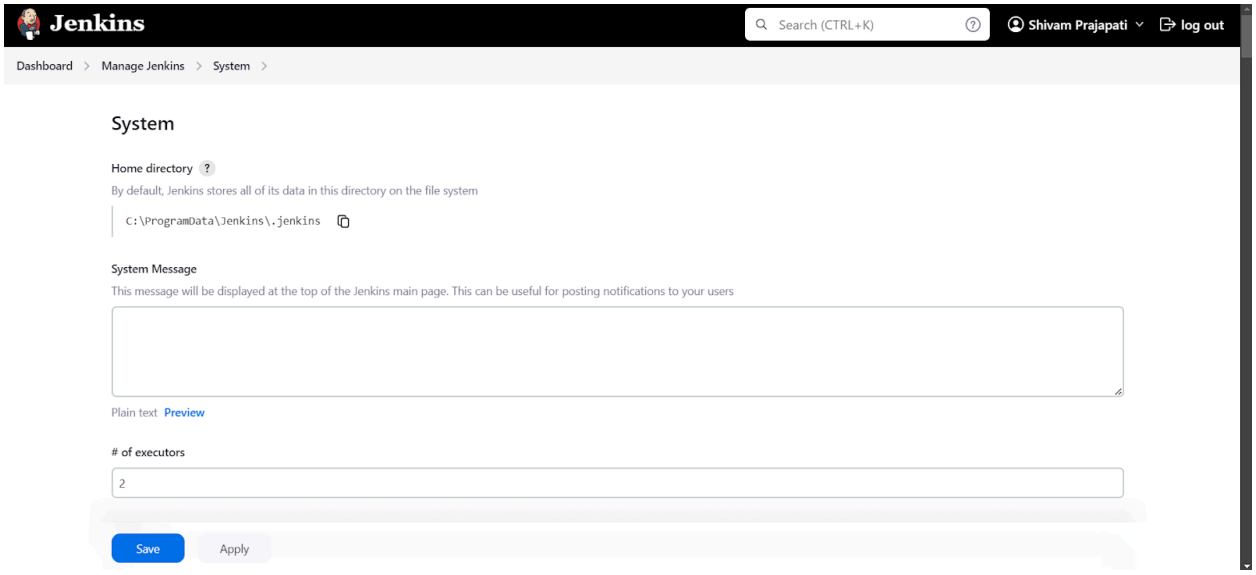
Name: Shivam.R.Prajapati **Div:D15C** **Roll No: 41**

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

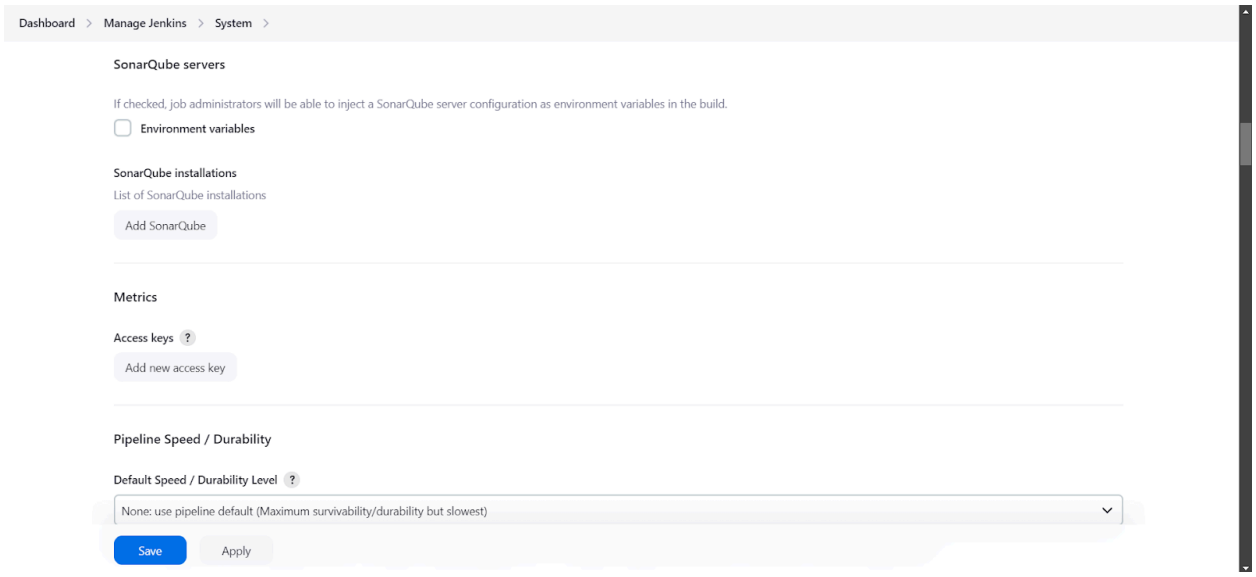


Go to system

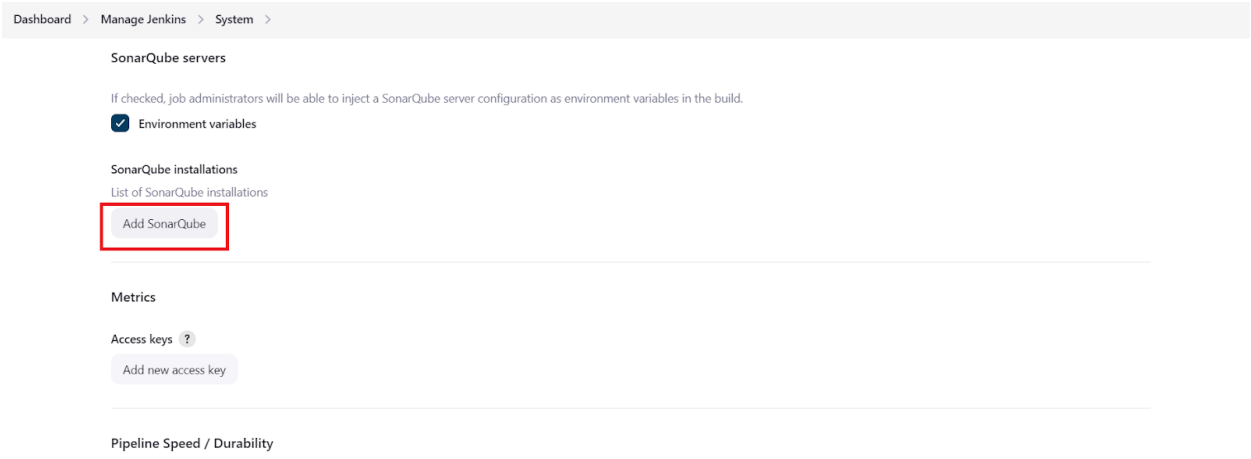




Scroll Down



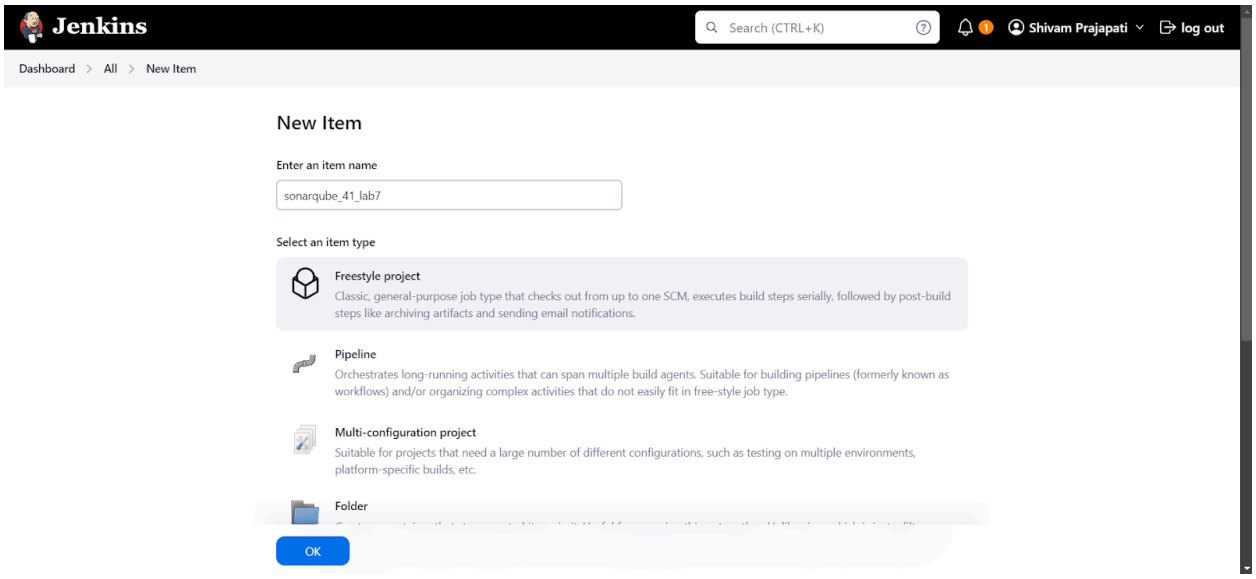
Select Environment Variable and Click on Add Sonar Qube button in order to Add SonarQube Server to Jenkin



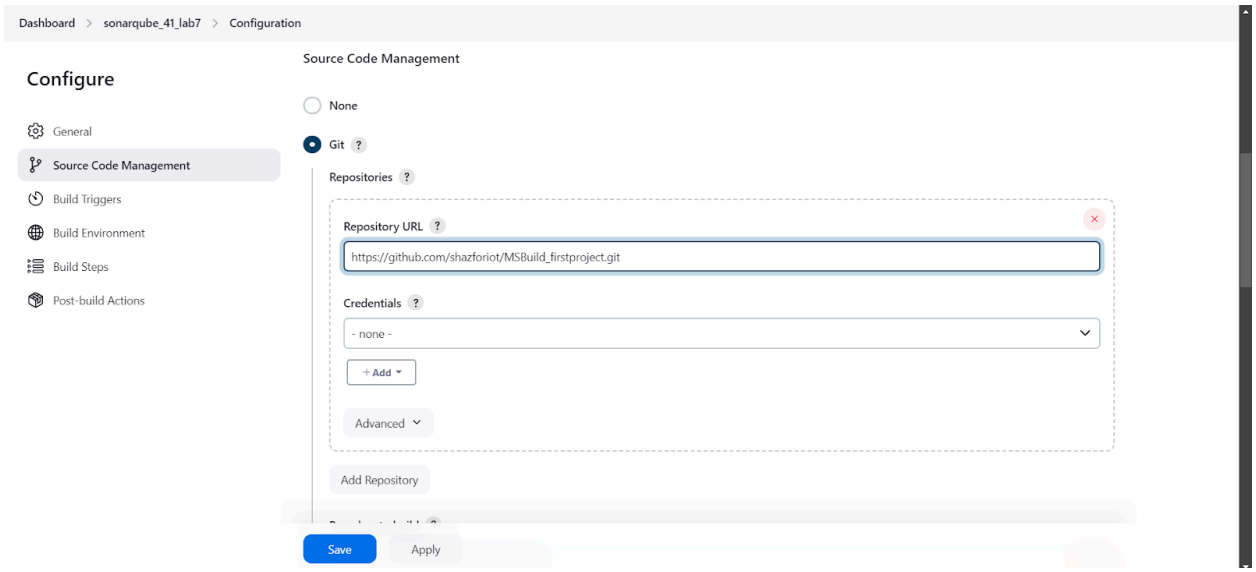
Do the required entries as shown below

Click on save

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

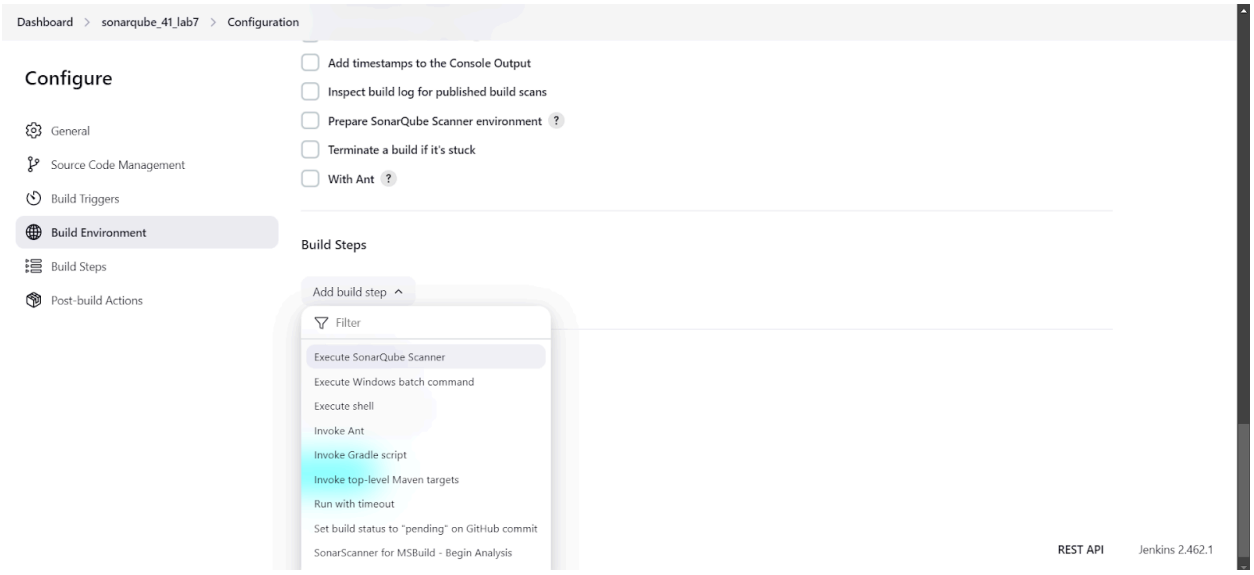


Step 11: Use this github repository in Source Code Management. **https://github.com/shazforiot/MSBuild_firstproject.** It is a sample hello-world project with no vulnerabilities.

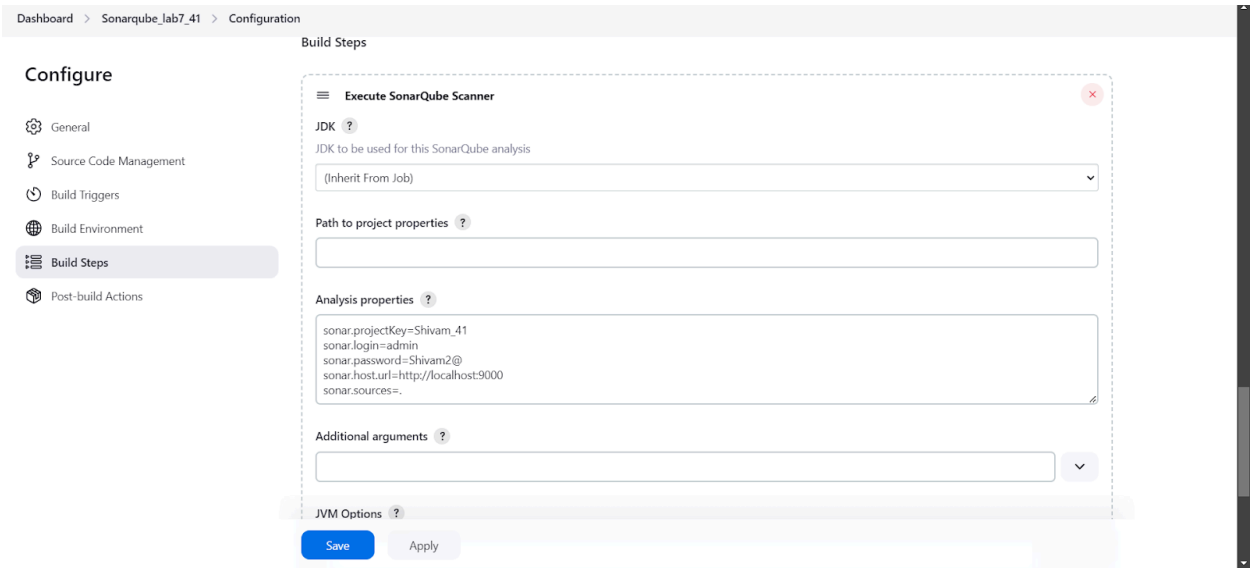


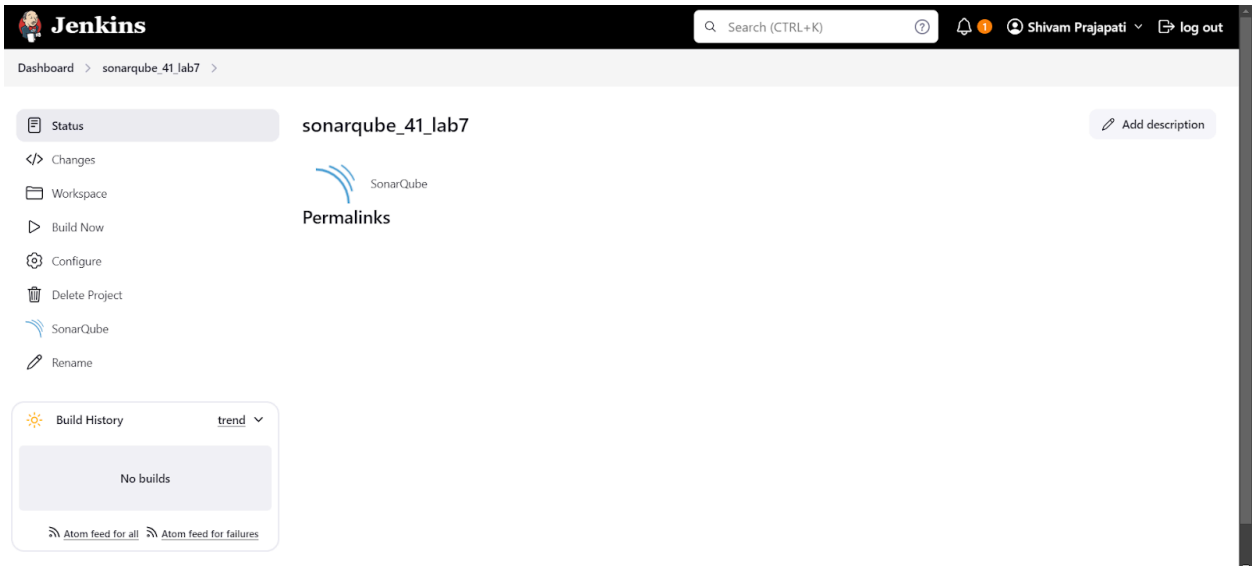
Step 12: Under Build Steps, enter Sonarqube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.

Name: Shivam.R.Prajapati **Div:D15C** **Roll No: 41**

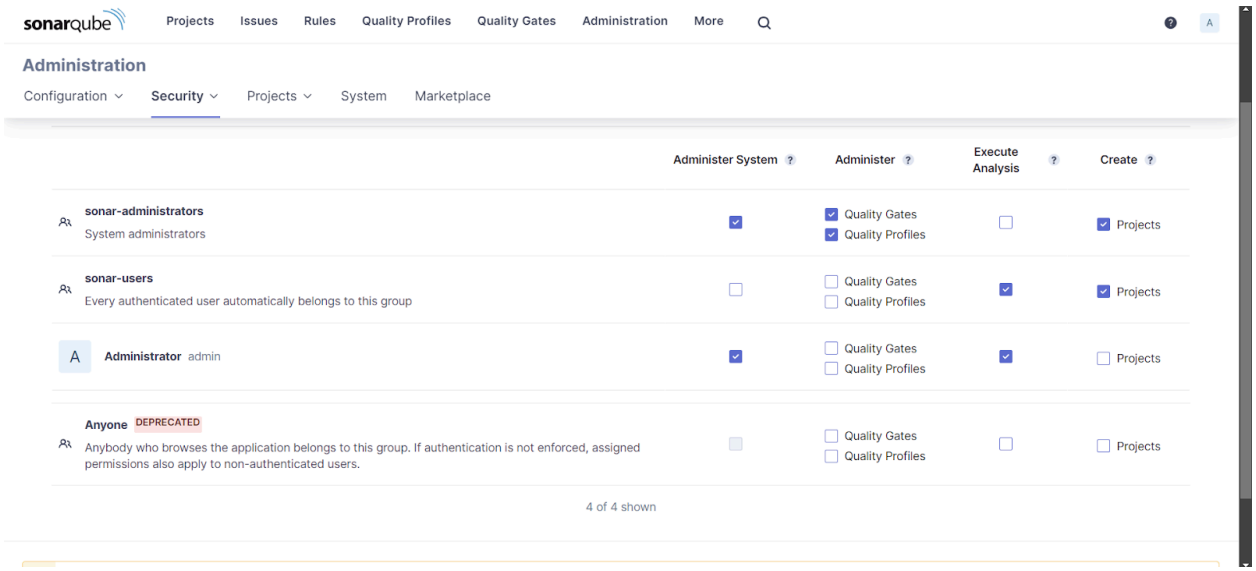


Click on execute sonar scanner

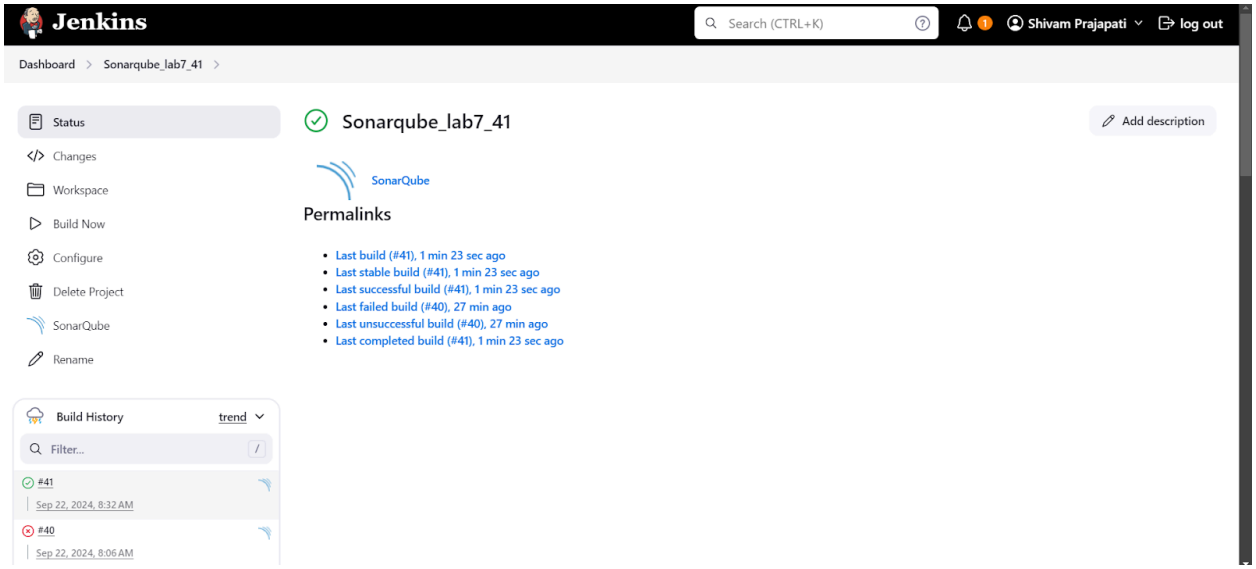




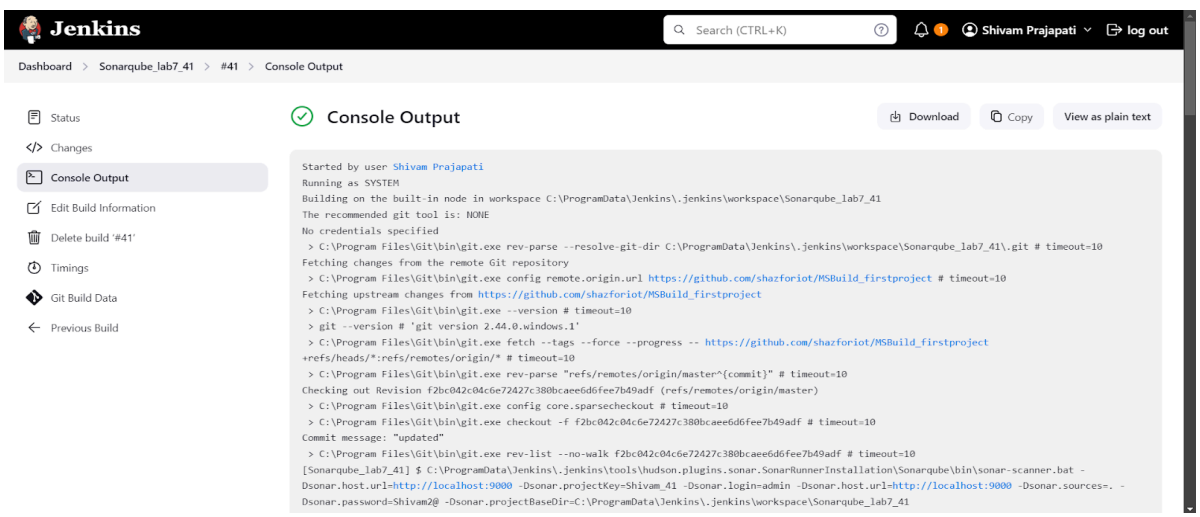
Step 13: Now, you need to grant the local user (here admin user) permissions to Execute the Analysis stage on SonarQube. For this, go to **http://loaclhost:<port_number>/admin/permissions** and check the 'Execute Analysis' checkbox under Administrator.



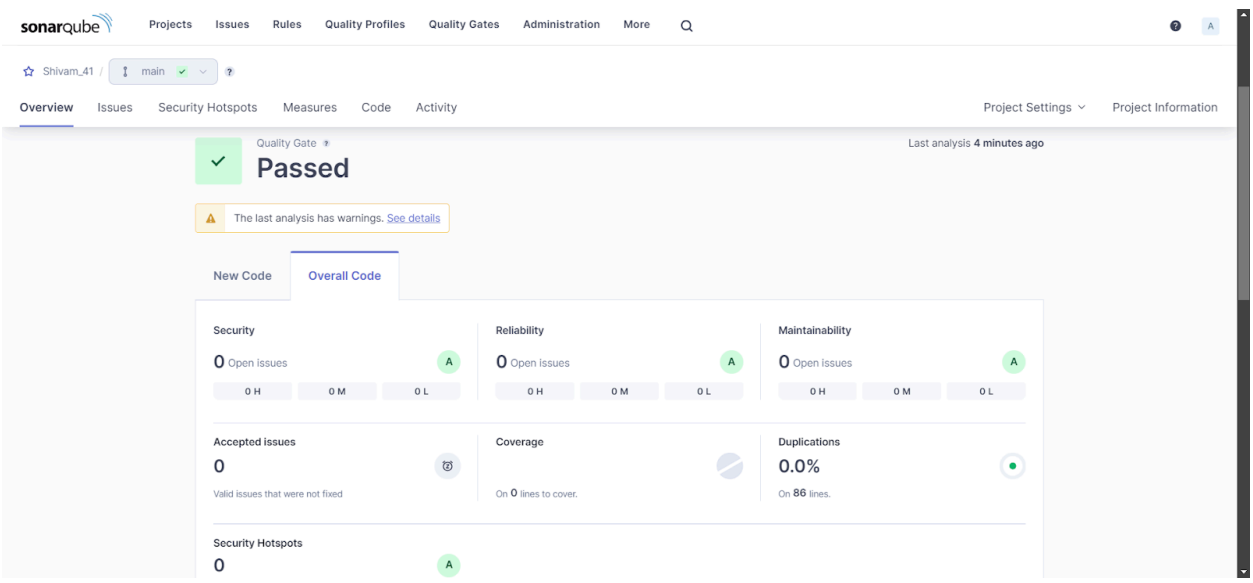
Step 14: Go back to jenkins. Go to the job you had just built and click on Build Now.



Check the Console Output



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.