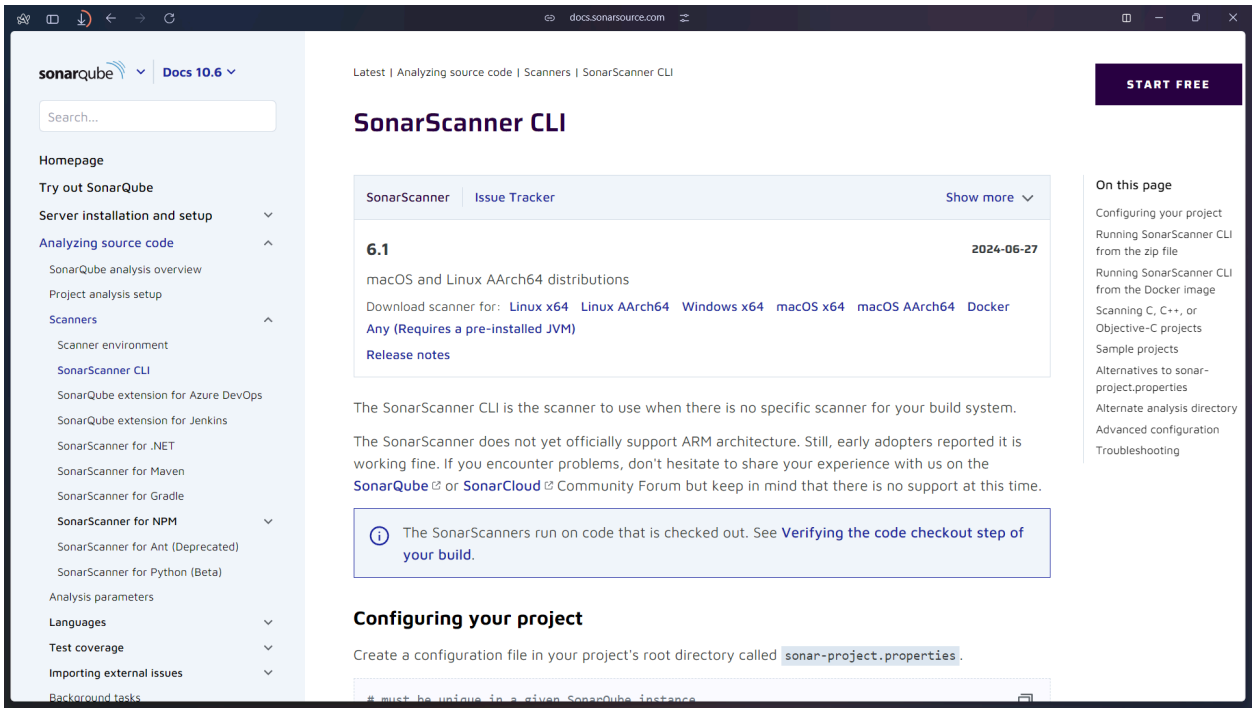
Sairam Konar D15C 27 2024-25

Aim: Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web / Java / Python application.

Prerequisites:

Step 1: Download sonar scanner

https://docs.sonarsource.com/sonarqube/latest/analyzing-source-code/scanners/sonarscanner/
Visit this link and download the sonarqube scanner CLI.



Extract the downloaded zip file in a folder.

1) Docker
   Run docker -v command.

```
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab7> docker -v
Docker version 27.0.3, build 7d4bcd8
```

2) Install sonarqube image
   Command: docker pull sonarqube

```
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab7> docker pull sonarqube
Using default tag: latest
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
docker.io/library/sonarqube:latest

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview sonarqube
```

3) Keep Jenkins installed on your system.

Experiment Steps:

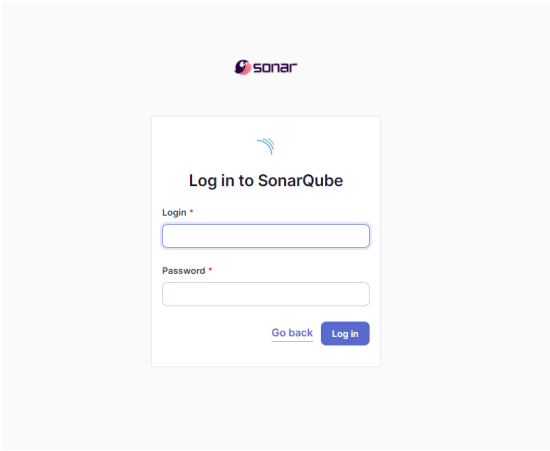
Step 1)Run SonarQube image

docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

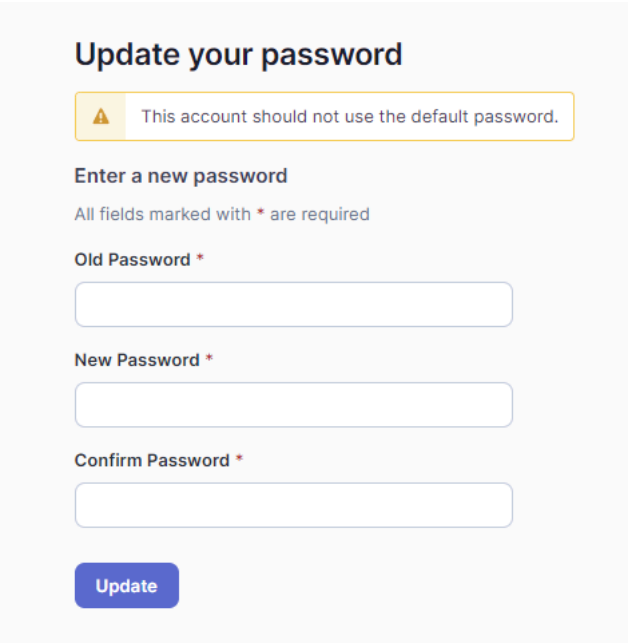
This command will run the SonarQube image that was just installed using docker.

```
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab7> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=t
rue -p 9000:9000 sonarqube:latest
36ff8a656bd28857ba9a28bf2bb0174099ae3232a9fc9ba2766d46f0c14d08a6
```

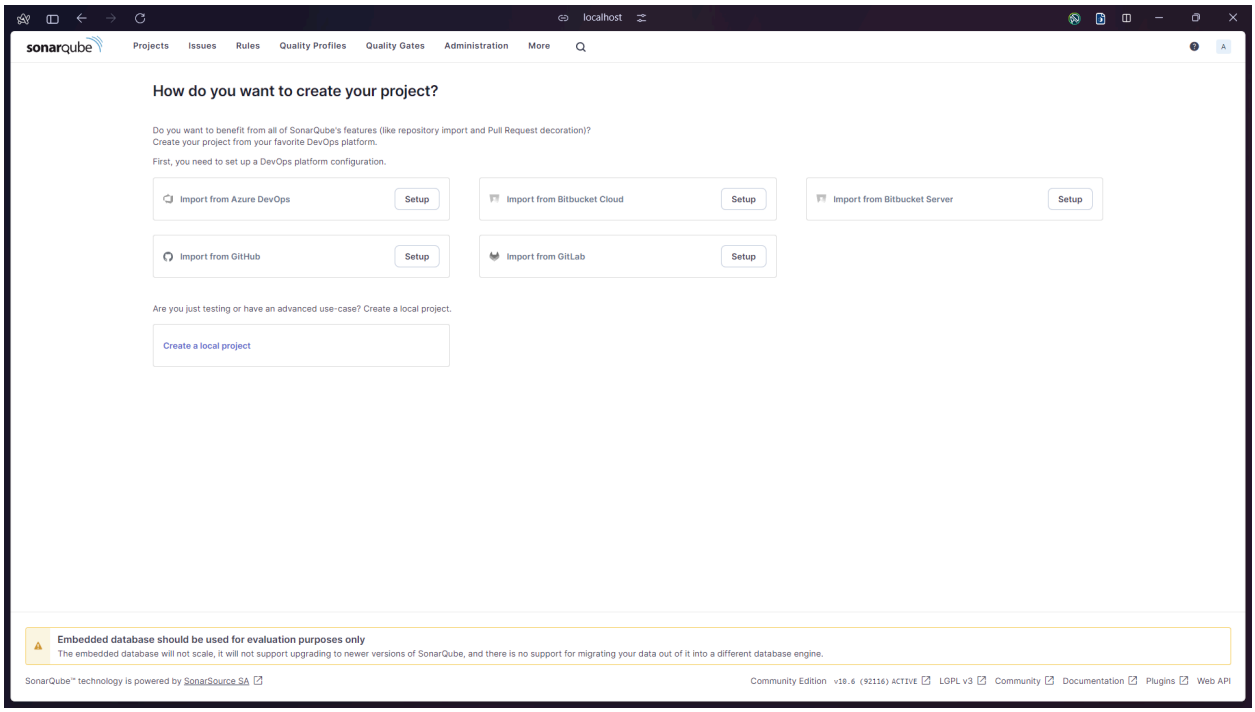
Step 2) Once the SonarQube image is started, you can go to http://localhost:9000 to find the SonarQube that has started



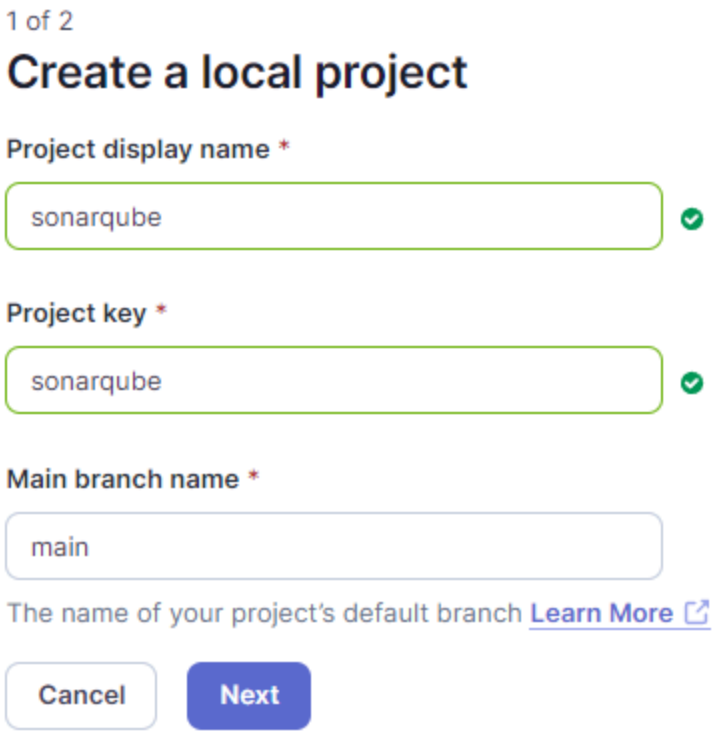
Step 3) On this interface, login with username = ‘admin’ and password = ‘admin’. Once logged in successfully, SonarQube will ask you to reset this password. Reset it and remember this password.



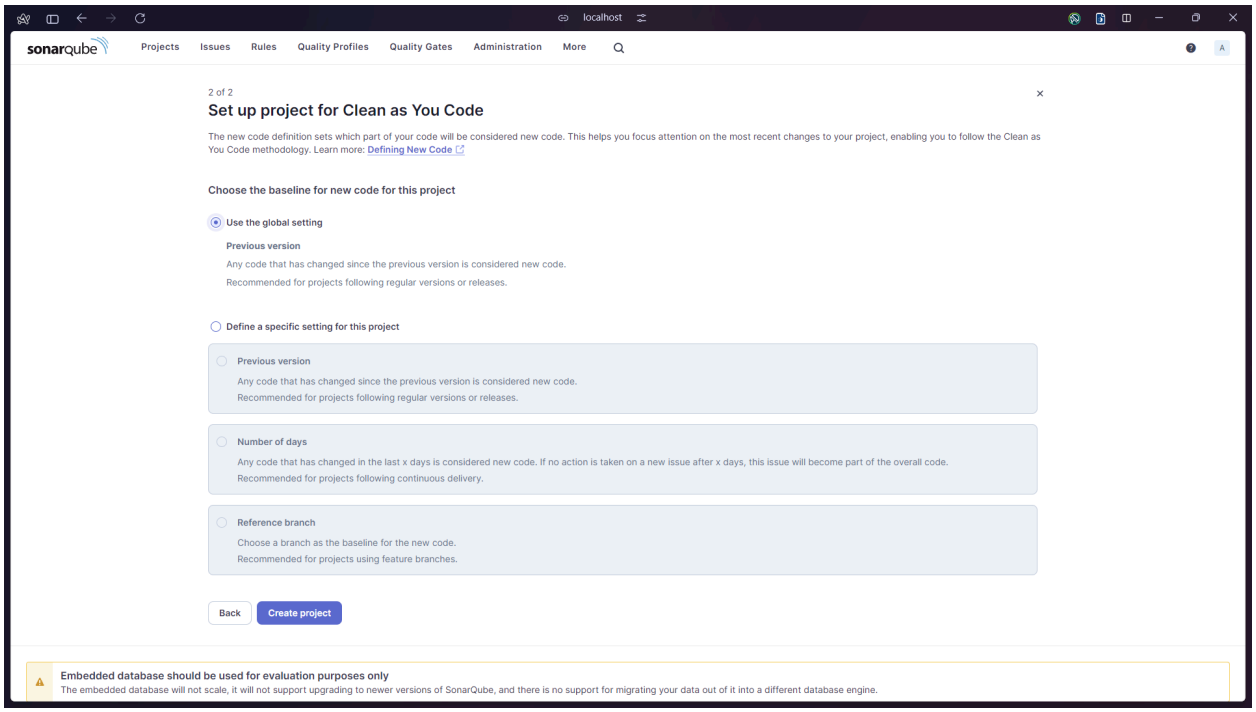
Step 4) After changing the password, you will be directed to this screen. Click on Create a Loal Project.



Give the project a display name and project key

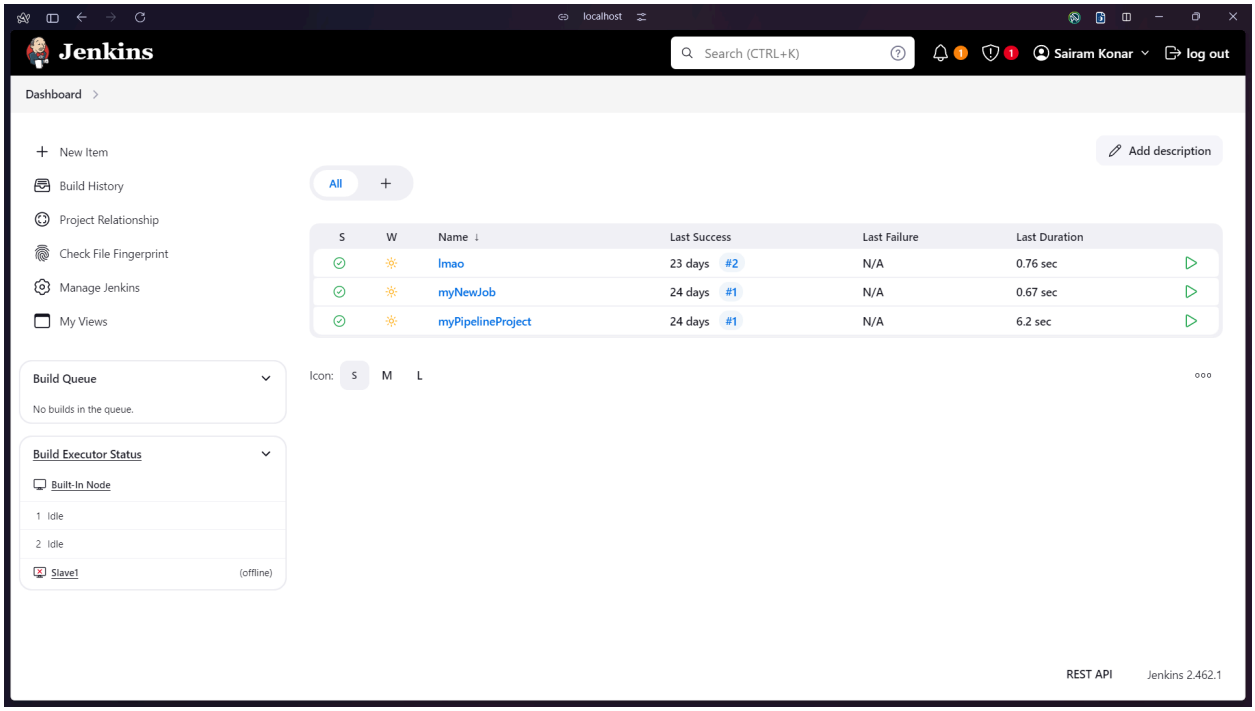


Set up the project as required and click on create.

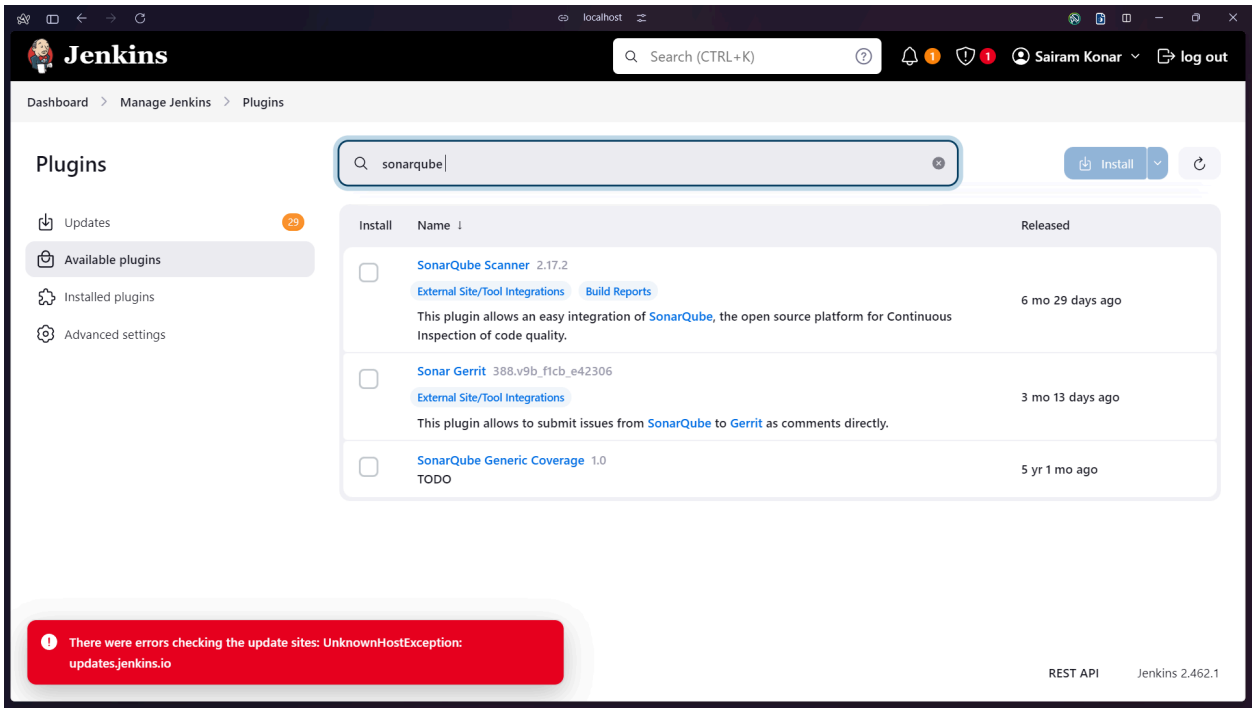


Sairam Konar D15C 27 2024-25

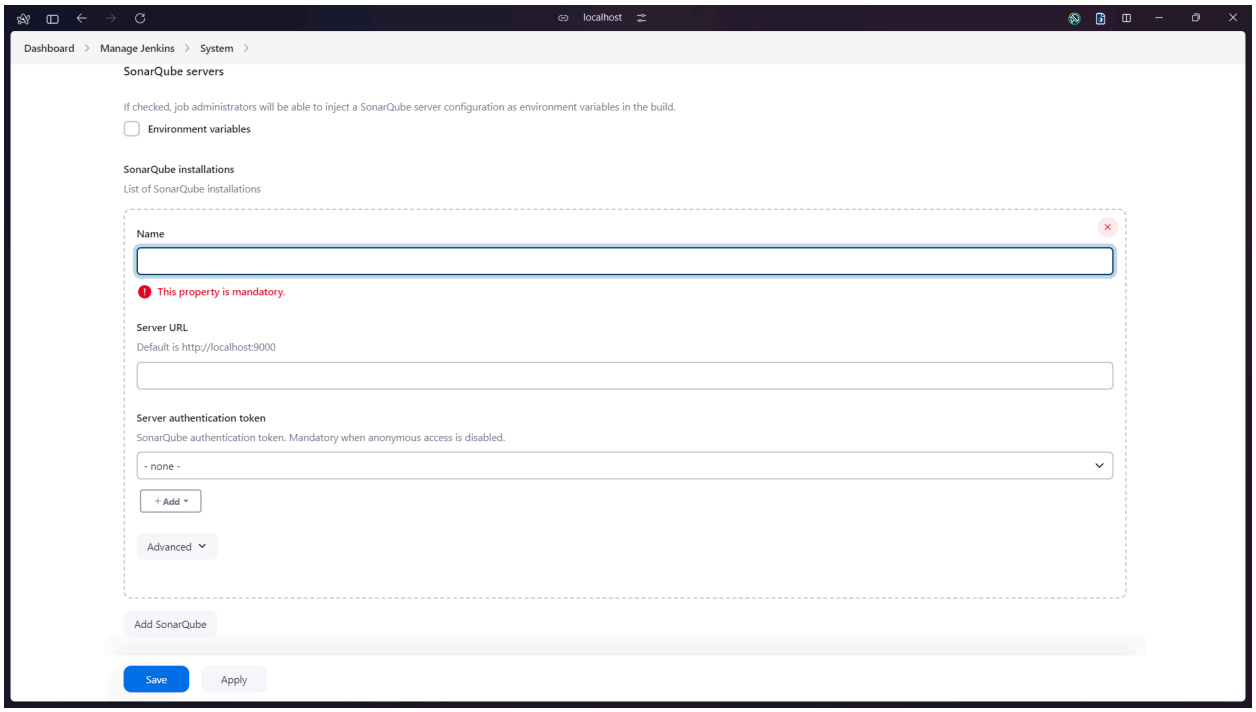
Step 5) Open Jenkins on whichever port it is installed. (http://loaclhost:<port_number>).



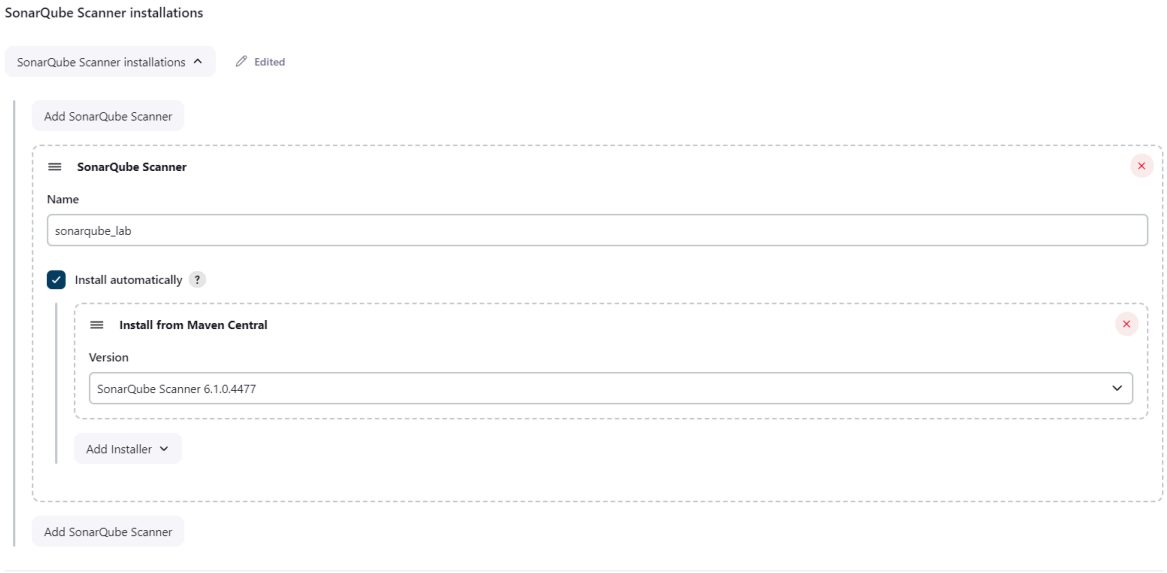
Step 6) Go to manage jenkins → Search for Sonarqube Scanner for Jenkins and install it.



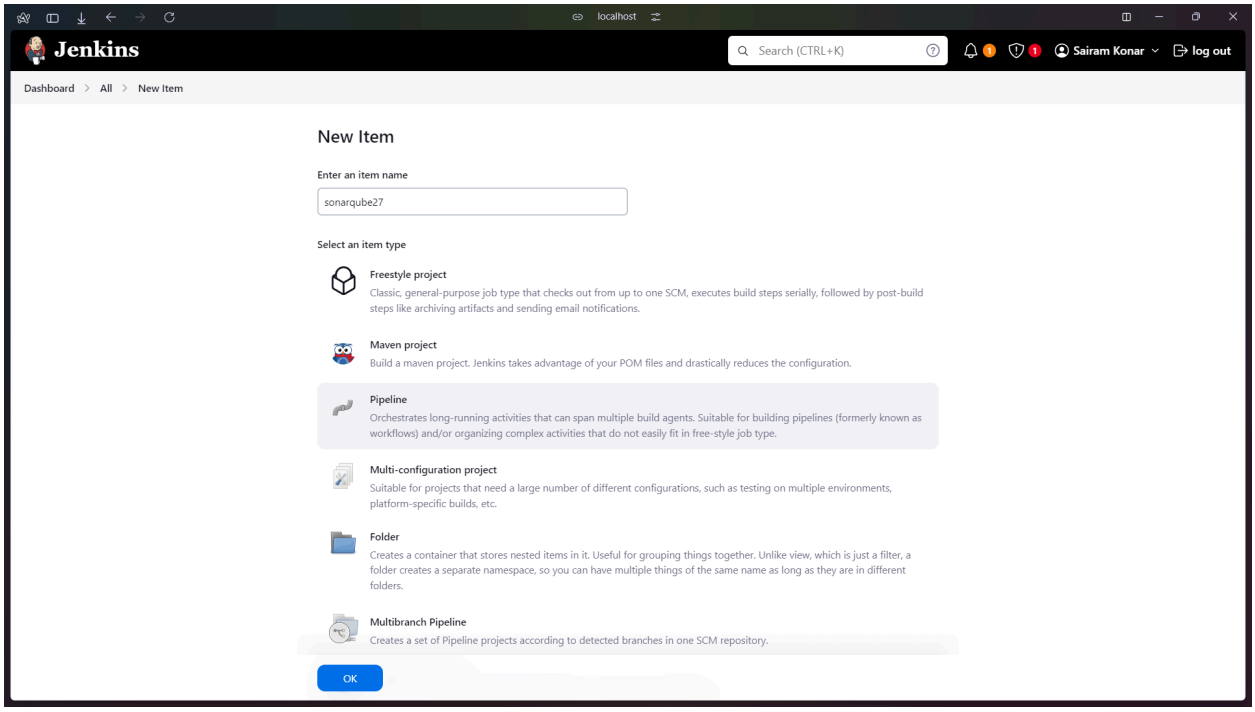
Step 7) Now, go to Manage Jenkins → System. Under Sonarqube servers, add a server. Add server authentication token if needed.



Step 8) Go to Manage Jenkins → Tools. Go to SonarQube scanner, choose the latest configuration and choose install automatically.



Step 9) After configuration, create a New Item → choose a pipeline project.



Step 10) Under Pipeline script, enter the following:

```
node {
        stage('Cloning the GitHub Repo') {
        git 'https://github.com/shazforiot/GOL.git'
        }

         stage('SonarQube analysis') {
                withSonarQubeEnv('sonarqube lab') {
                bat """
                        <PATH_TO_SONARSCANNER_FOLDER>\\bin\\sonar-scanner.bat ^
        -D sonar.login=<SONARQUBE_LOGIN> ^
        -D sonar.password=<SONARQUBE_PASSWORD> ^
        -D sonar.projectKey=<PROJECT_KEY> ^
        -D sonar.exclusions=vendor/**,resources/**,**/*.java ^
        -D sonar.host.url=http://localhost:9000/
         """
        }
 }
}
```

It is a java sample project which has a lot of repetitions and issues that will be detected by SonarQube.

Definition

Pipeline script

Script

try sample Pipeline...

```
node {
    stage('Cloning the GitHub Repo') {
        git 'https://github.com/shazforiot/GOL.git'
    }

    stage('SonarQube analysis') {
        withSonarQubeEnv('sonarqube lab') {
            bat """
                C:\\Users\\saira\\Downloads\\sonar-scanner-cli-6.1.0.4477-windows-x64\\sonar-scanner-6.1.0.4477-windows-x64\\bin\\sonar-scanner.bat ^
                -D sonar.login=admin ^
                -D sonar.password=123456 ^
                -D sonar.projectKey=sonarqube27 ^
                -D sonar.exclusions=vendor/**,resources/**,**/*.java ^
                -D sonar.host.url=http://localhost:9000/
            """
        }
    }
}
```

Use Groovy Sandbox

Pipeline Syntax

Save Apply

Step 11) Go back to jenkins. Go to the job you had just built and click on Build Now.

sonarqube27

Stage View

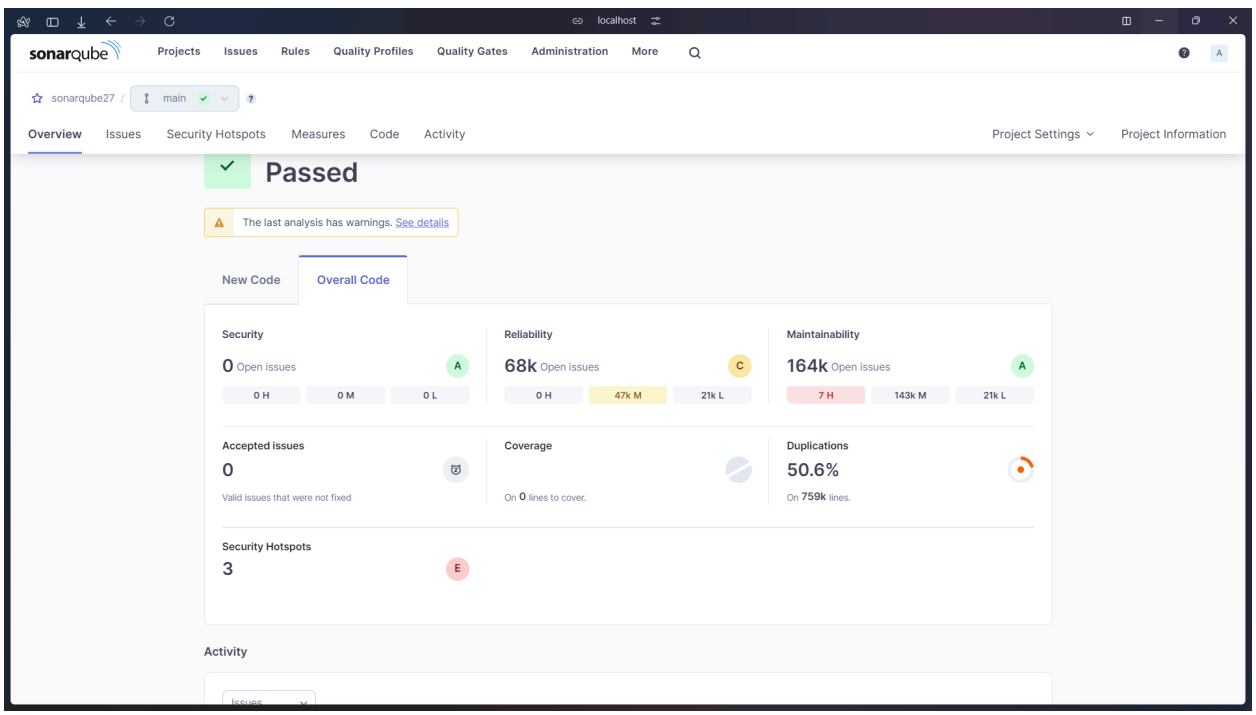
	Cloning the GitHub Repo	SonarQube analysis
Average stage times: (Average full run time: ~15min 35s)	2s	15min 32s
#19 Sep 18 09:23 No Changes	2s	15min 32s

Permalinks

Check the console output once

```
first 100 references.
09:36:08.013 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/action/Cut.html for block at line 75. Keep only the
first 100 references.
09:36:08.029 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/jdbc/config/package-summary.html for block at
line 39. Keep only the first 100 references.
09:36:08.029 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/jdbc/config/package-summary.html for block at
line 40. Keep only the first 100 references.
09:36:08.054 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/util/accesslog/Generator.html for block at
line 40. Keep only the first 100 references.
09:36:08.054 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/util/accesslog/Generator.html for block at
line 41. Keep only the first 100 references.
09:36:08.055 INFO  CPD Executor CPD calculation finished (done) | time=163405ms
09:36:08.069 INFO  SCM revision ID 'ba799ba7e1b576f04a4612322b0412c5e6e1e5e4'
09:38:20.750 INFO  Analysis report generated in 4390ms, dir size=127.2 MB
09:38:45.857 INFO  Analysis report compressed in 25089ms, zip size=29.6 MB
09:38:46.532 INFO  Analysis report uploaded in 675ms
09:38:46.533 INFO  ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube27
09:38:46.533 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
09:38:46.533 INFO  More about the report processing at http://localhost:9000/api/ce/task?id=46576333-cbde-4277-89d7-471ee554de32
09:39:00.038 INFO  Analysis total time: 15:24.256 s
09:39:00.041 INFO  SonarScanner Engine completed successfully
09:39:00.810 INFO  EXECUTION SUCCESS
09:39:00.811 INFO  Total time: 15:29.301s
[Pipeline] }
[Pipeline] // withSonarQubeEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Step 12) Once the build is complete, go back to SonarQube and check the project linked.

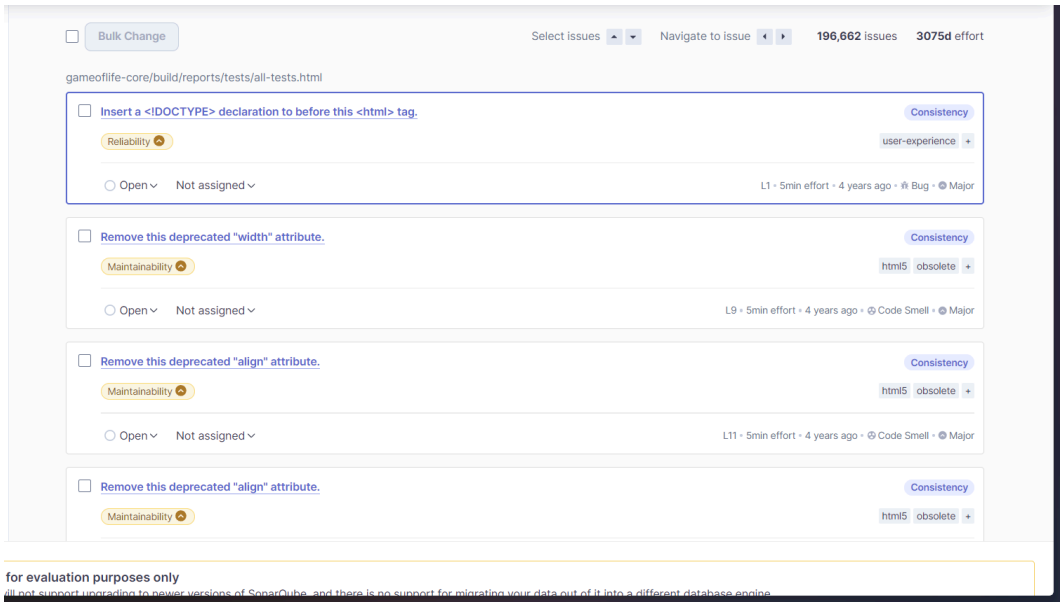


Under different tabs, check all the issues with the code.

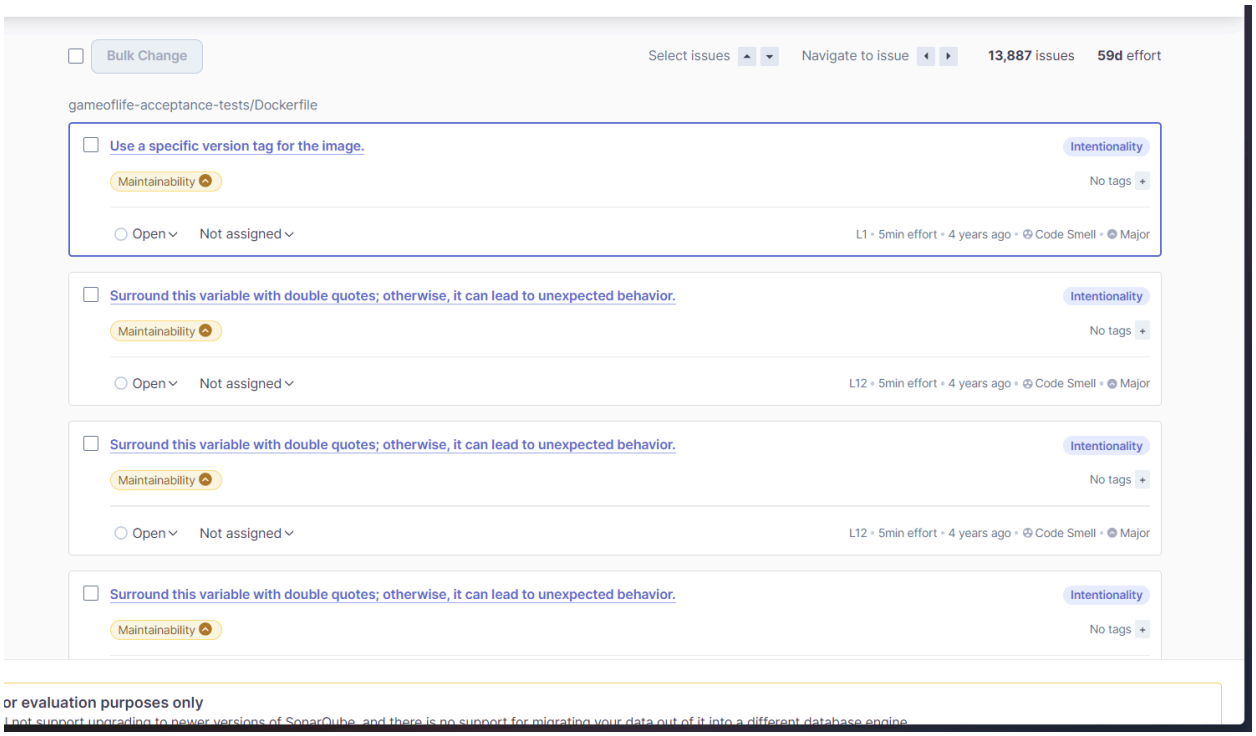
- Code Problems

Sairam Konar D15C 27 2024-25

- Consistency

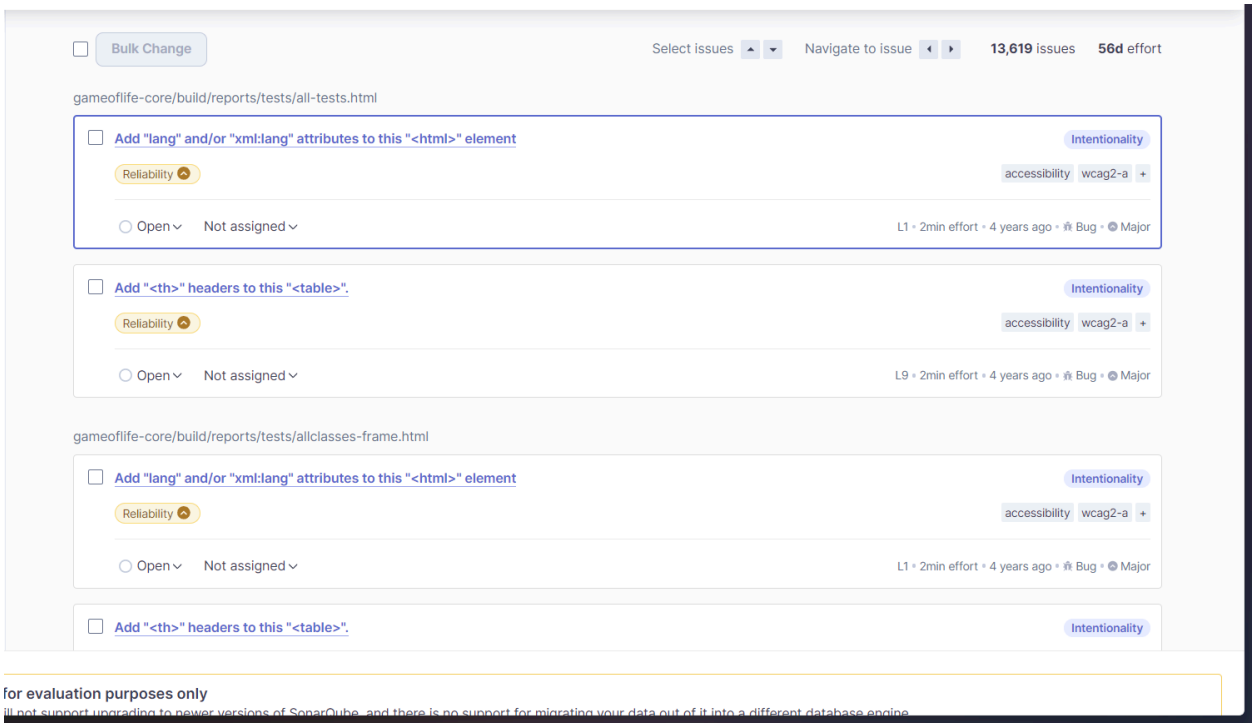


- Intentionality

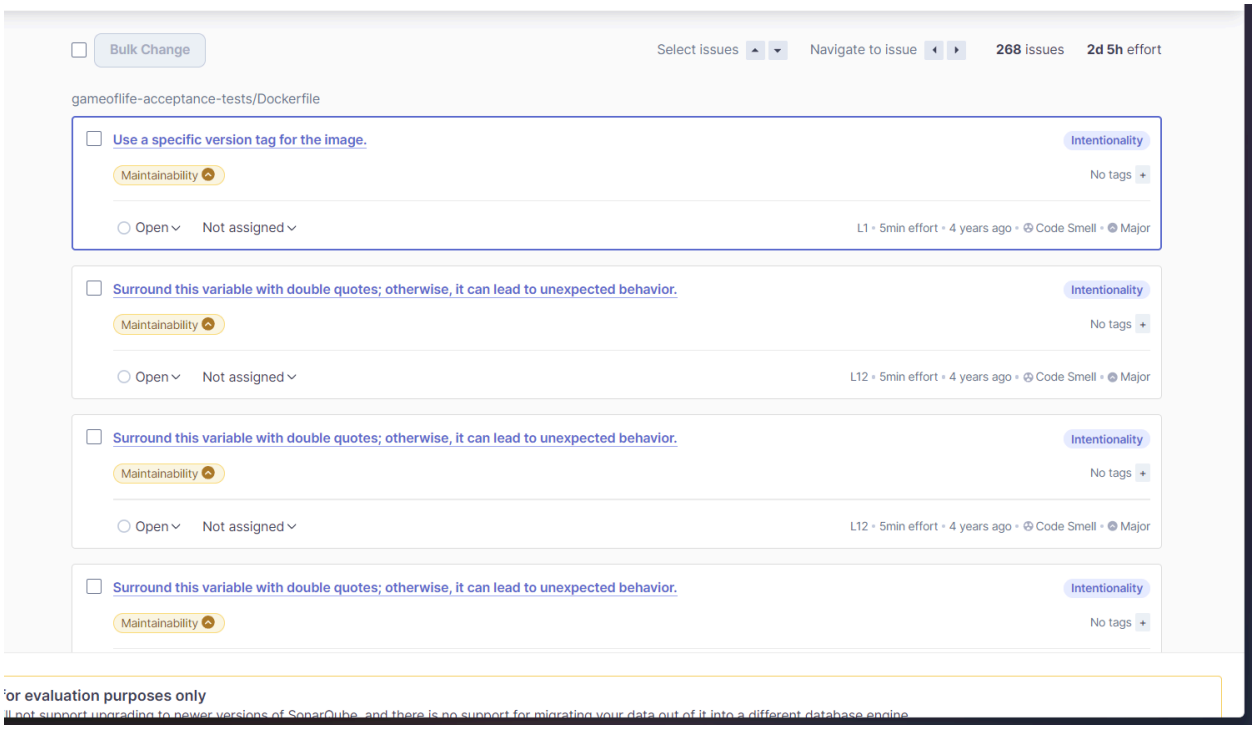


Sairam Konar D15C 27 2024-25

- Bugs

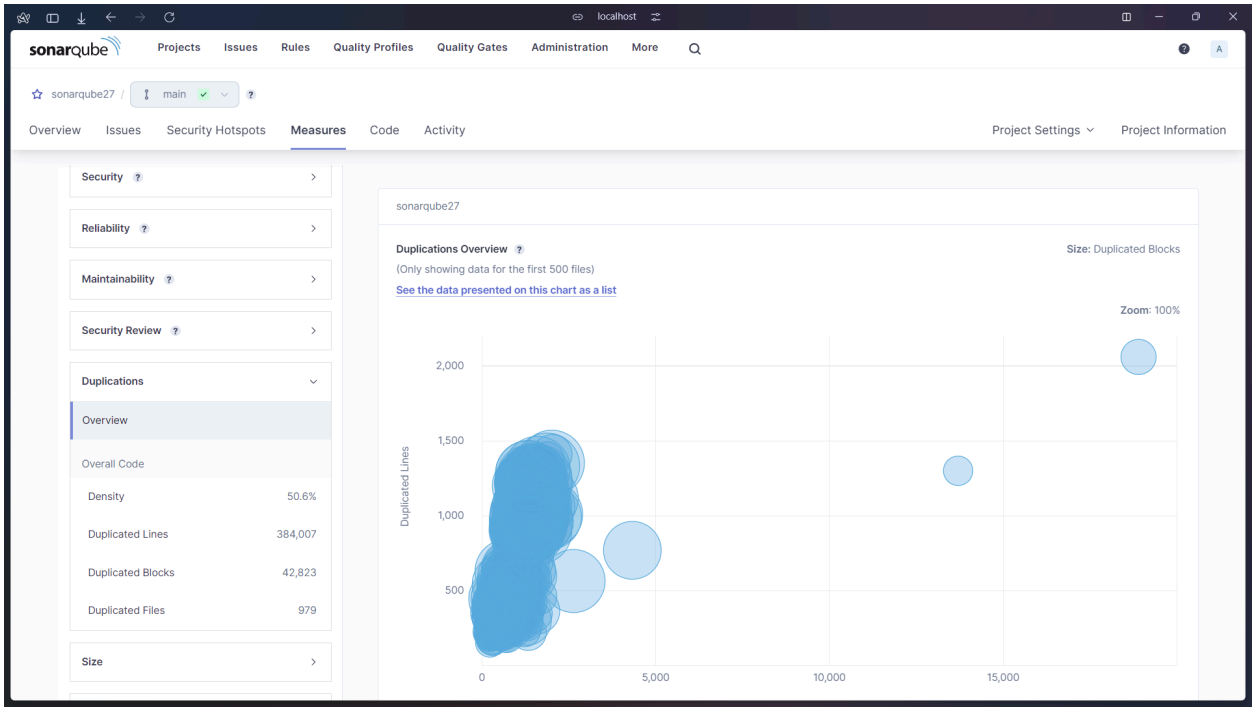


- Code Smells

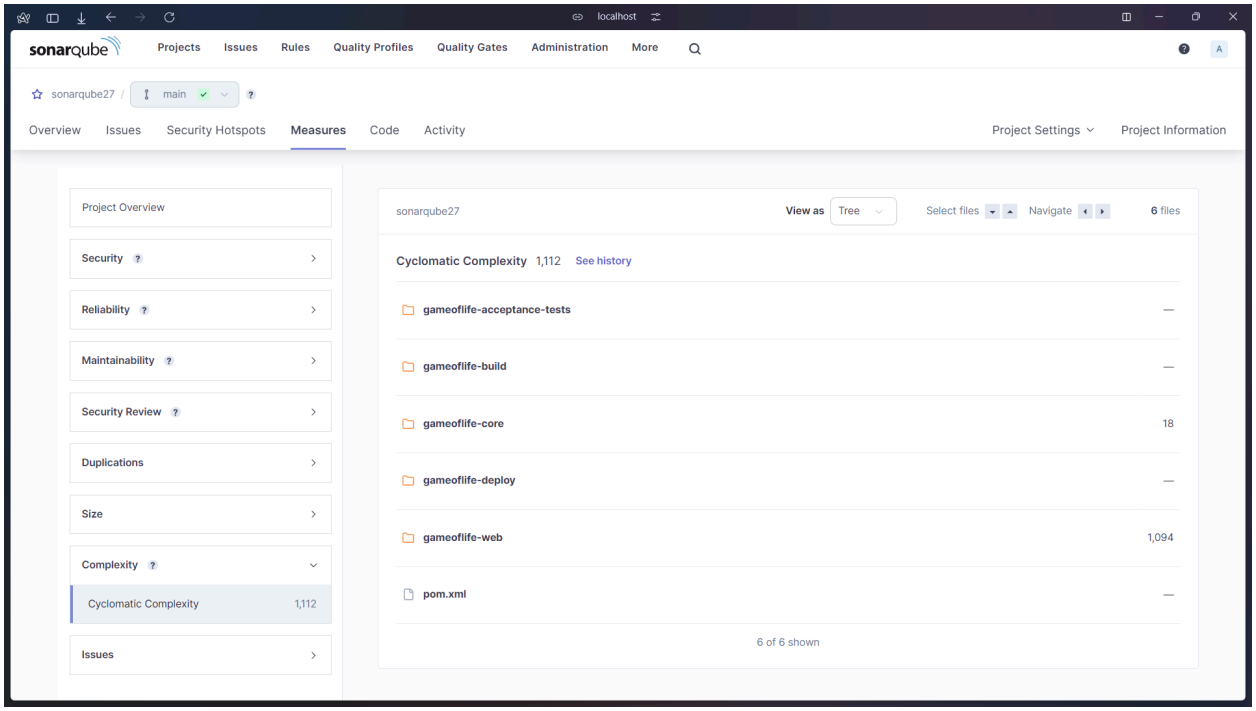


Sairam Konar D15C 27 2024-25

- Duplications



- Cyclomatic Complexities



Conclusion: In this experiment, we have learned how to perform static analysis of a code using Jenkins CI/CD Pipeline with SonarQune analysis. A pipeline project is to be created which is given a pipeline script. This script contains all the information needed for the project to run the SonarQube analysis. After the necessary configurations are made on jenkins, the Jenkins project is built. The code provided in this experiment contains lots of error, bugs, duplications which can be checked on the SonarQube project linked with this build.