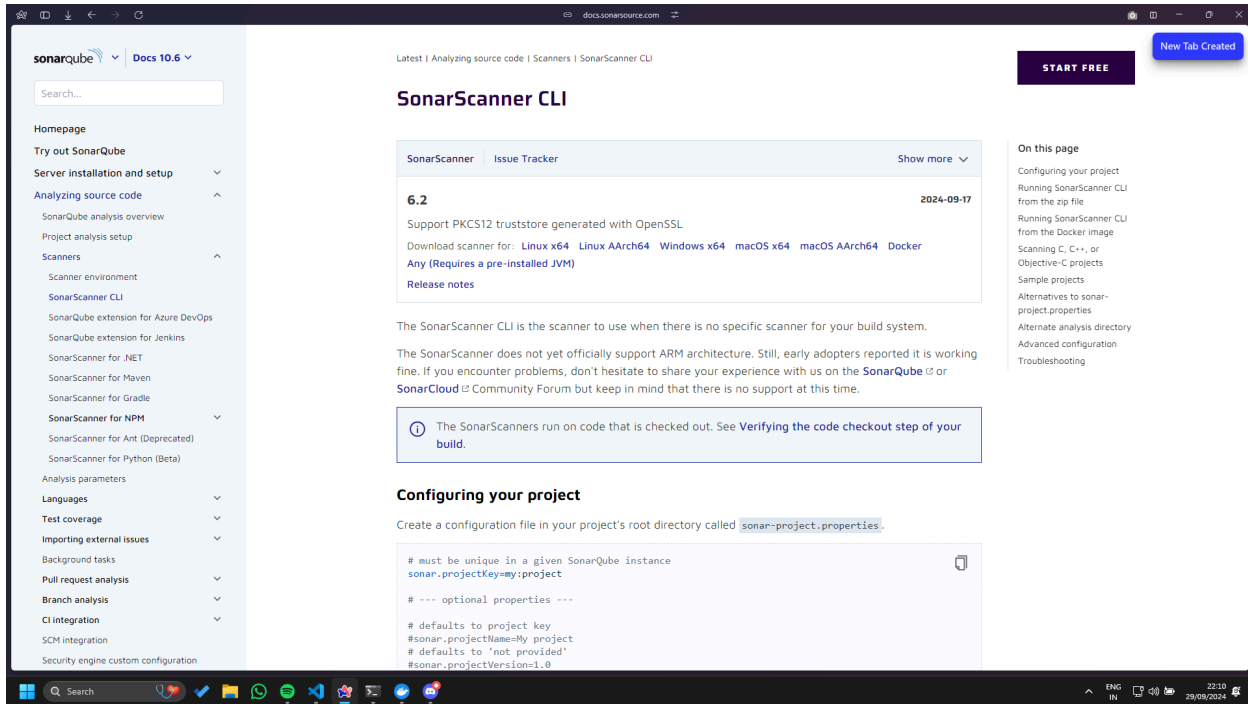


Aim: Create a Jenkins CI/CD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web / Java / Python application.

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



Run docker -v command.

Run docker pull sonarqube command

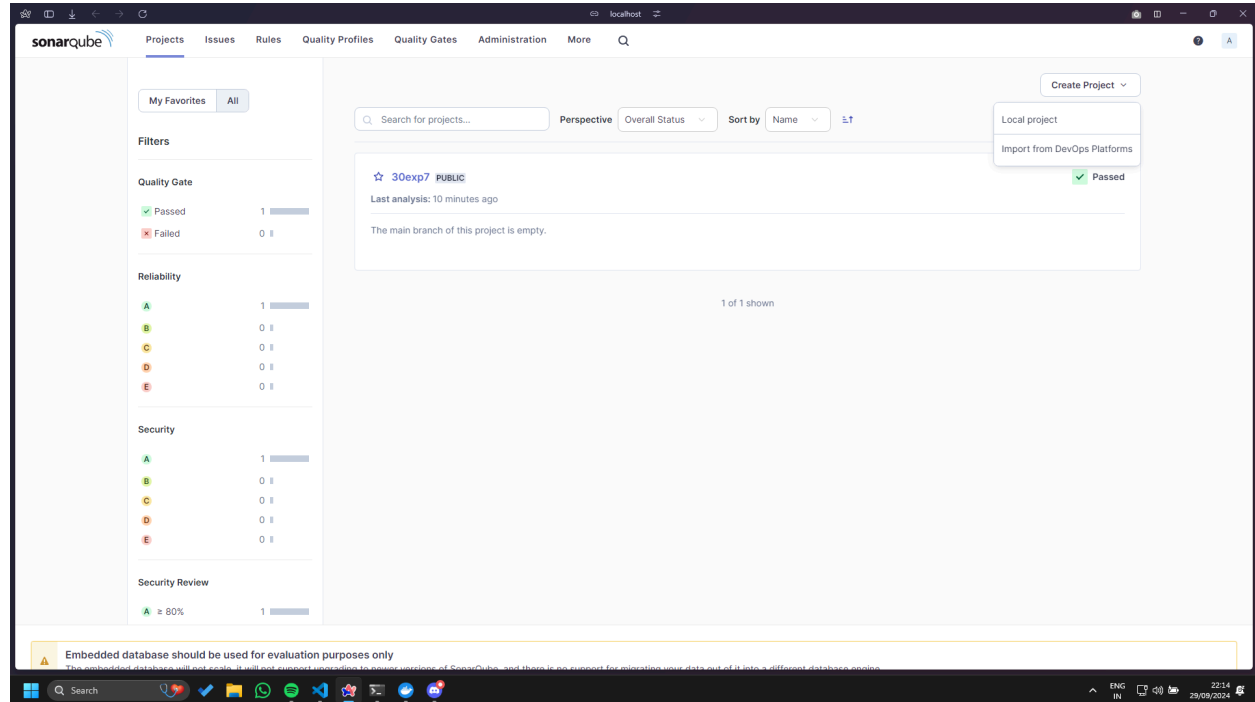
```
C:\Users\Aditya>docker -v
Docker version 27.2.0, build 3ab4256

C:\Users\Aditya>docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
4f4fb700ef54: Download complete
bd819c9b5ead: Download complete
80338217a4ab: Download complete
7478e0ac0f23: Download complete
90a925ab929a: Download complete
7b87d6fa783d: Download complete
1a5fd5c7e184: Download complete
7d9a34308537: Download complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest
```

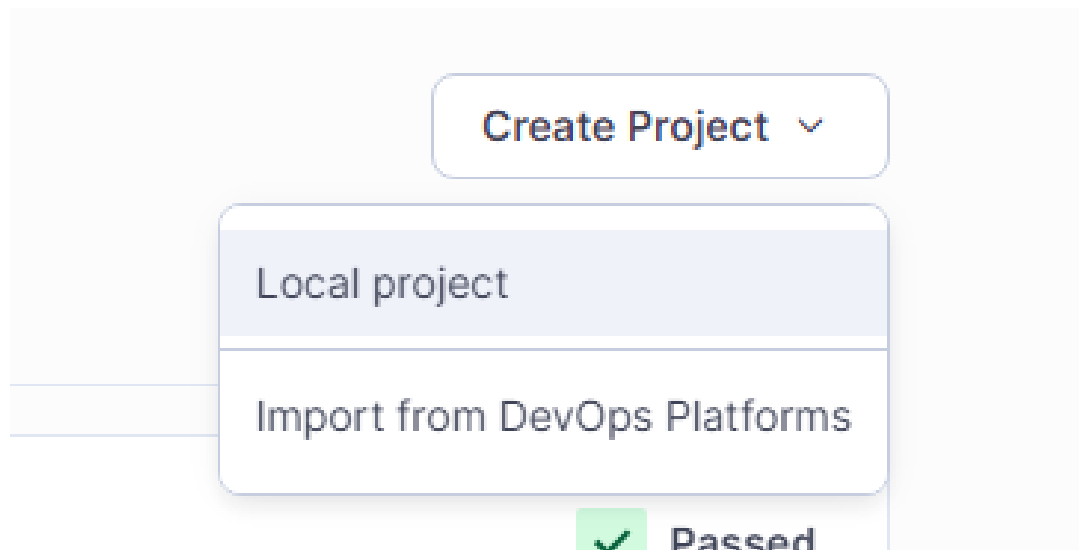
Run SonarQube image `docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest` This command will run the SonarQube image that was just installed using docker.

```
C:\Users\Aditya>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
102a6e51b67968db1df41bc8c6c3ce068f11506acadcab2d306470aede30908
```

Go to <http://localhost:9000>



Create a new local project



1 of 2

Create a local project

Project display name *



Project key *

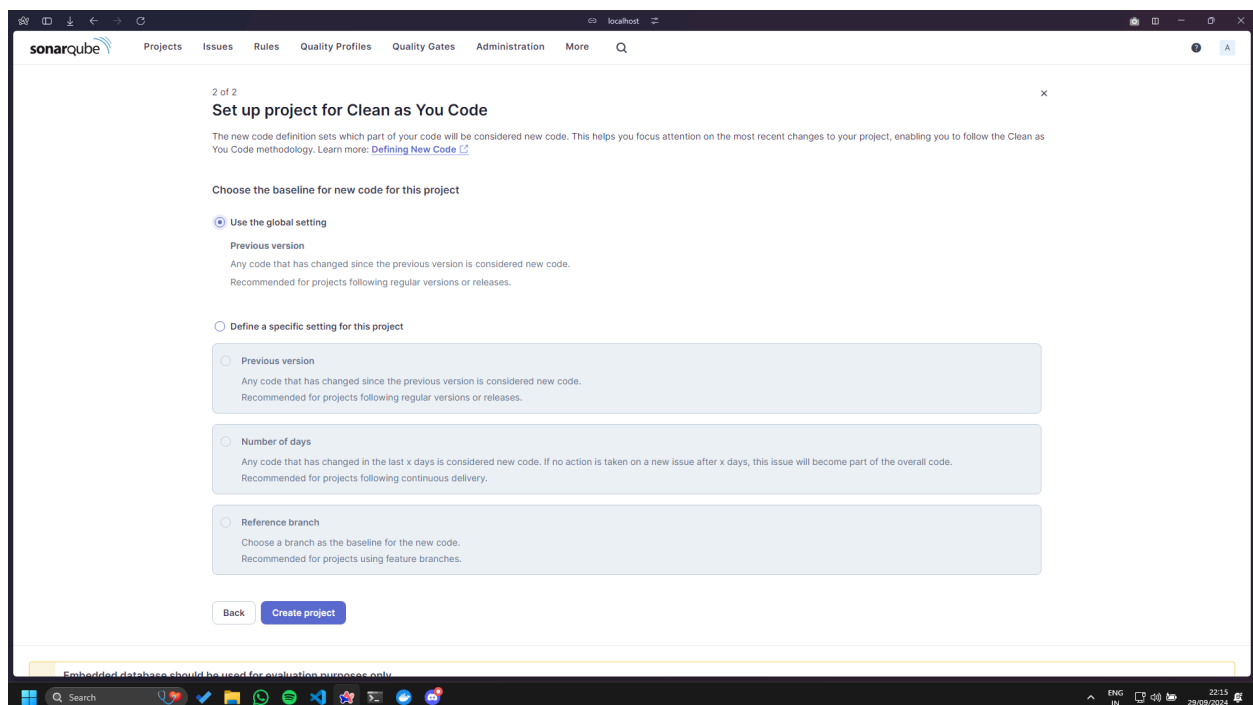


Main branch name *

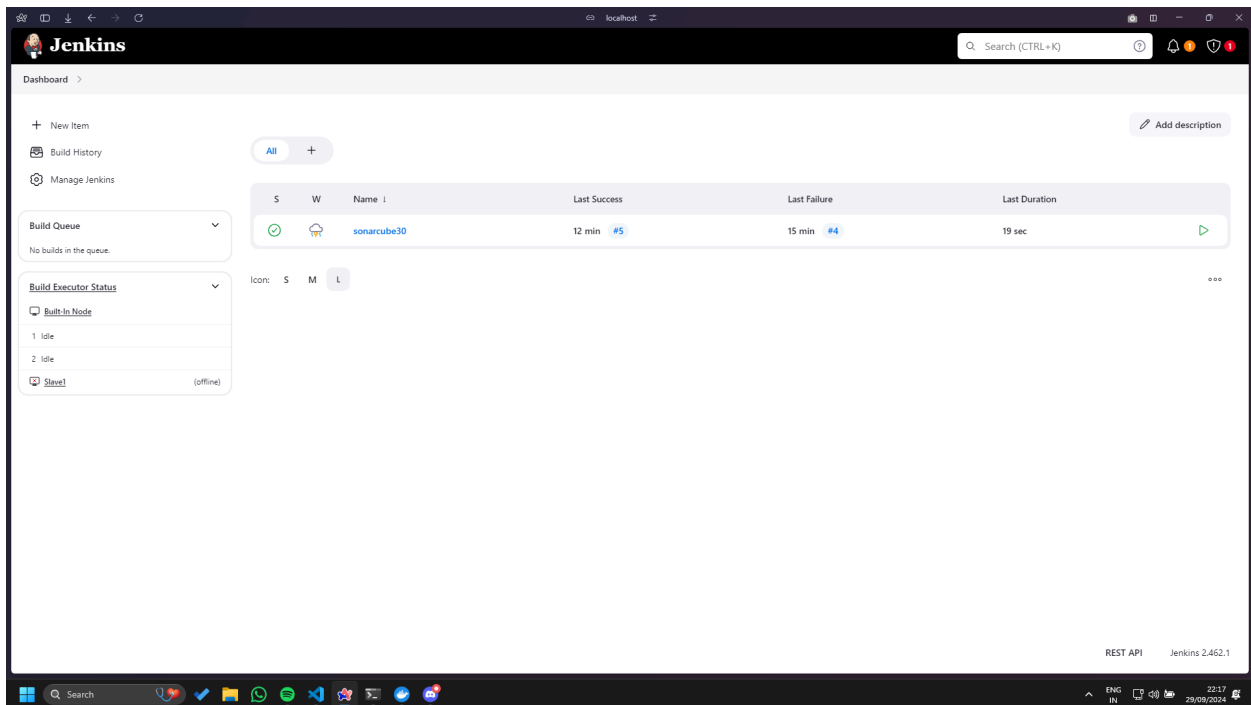
The name of your project's default branch [Learn More](#)

Cancel

Next



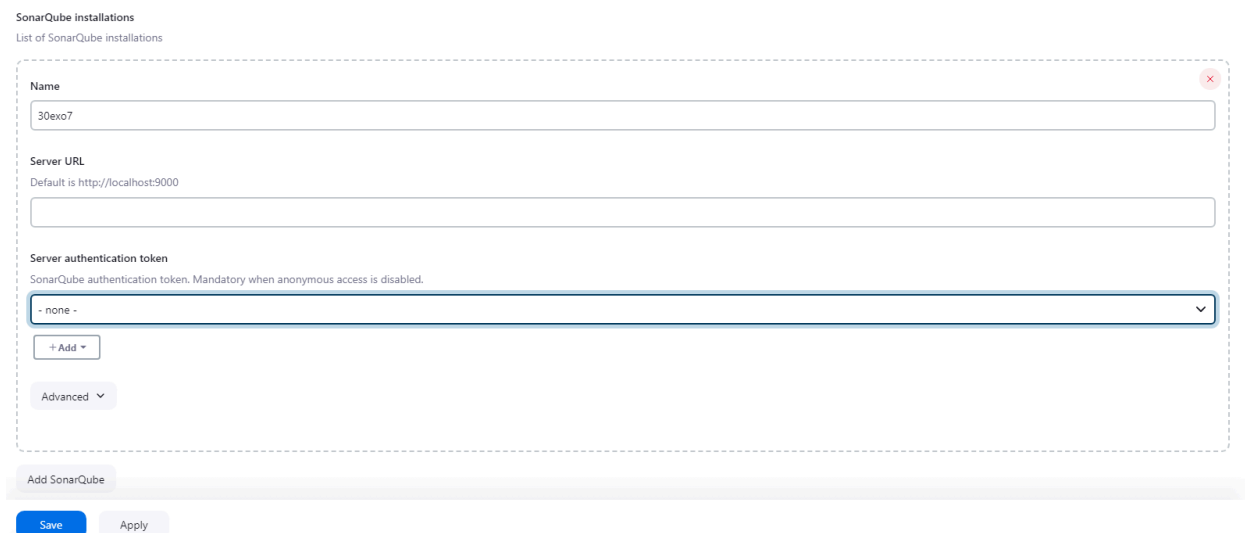
Open jenkins



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



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



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

SonarQube Scanner installations

Add SonarQube Scanner

☰

SonarQube Scanner

✕

Name

30exp7

☒ Install automatically

?

☰

Install from Maven Central

✕

Version

SonarQube Scanner 6.2.0.4584

▼

Add Installer

▼

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

New Item

Enter an item name

30exp8

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.



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.

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 ""
```

```
C:\Users\Aditya\Downloads\sonar-scanner-6.2.0.4584-windows-x64\bin\sonar-scanner.bat ^
-D sonar.login=admin ^
-D sonar.password=aditya@123 ^
-D sonar.projectKey=30exp8 ^
```

```
-D sonar.exclusions=vendor/**,resources/**,**/*.java ^
-D sonar.host.url=http://localhost:9000/
""""
}
}
}
```

Pipeline

Definition

Pipeline script

Script ?

```
1 node {
2   stage('Cloning the GitHub Repo') {
3     git 'https://github.com/shazforiot/GOL.git'
4   }
5   stage('SonarQube analysis') {
6     withSonarQubeEnv('sonarqube lab') {
7       bat
8       C:\Users\Aditya\Downloads\sonar-scanner-6.2.0.4584-windows-x64\bin\sonar-scanner.bat ^
9       -D sonar.login=admin ^
10      -D sonar.password=aditya@123 ^
11      -D sonar.projectKey=30exp8 ^
12      -D sonar.exclusions=vendor/**,resources/**,**/*.java ^
13      -D sonar.host.url=http://localhost:9000/
14      """"
15    }
16  }
17 }
```

try sample Pipeline... ▼

☒ Use Groovy Sandbox ?[Pipeline Syntax](#)

Save

Apply

Go to the job you had just built and click on Build Now.

 30exp8

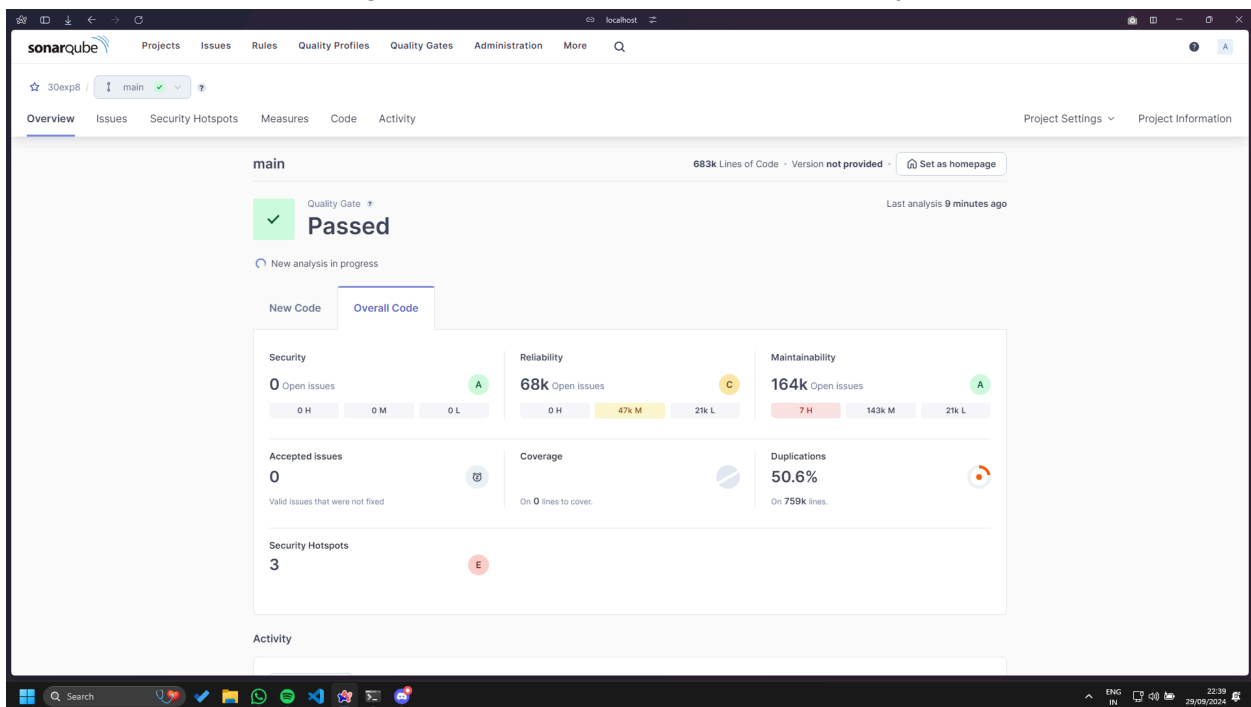
Permalinks

- [Last build \(#4\), 6 min 59 sec ago](#)
- [Last stable build \(#4\), 6 min 59 sec ago](#)
- [Last successful build \(#4\), 6 min 59 sec ago](#)
- [Last failed build \(#3\), 7 min 26 sec ago](#)
- [Last unsuccessful build \(#3\), 7 min 26 sec ago](#)
- [Last completed build \(#4\), 6 min 59 sec ago](#)

Check the console output once

```
22:36:56.460 INFO Analysis total time: 6:26.571 s
22:36:56.477 INFO SonarScanner Engine completed successfully
22:36:57.109 INFO EXECUTION SUCCESS
22:36:57.193 INFO Total time: 6:30.536s
[Pipeline] }
[Pipeline] // withSonarQubeEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

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



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

• Consistency

The screenshot displays the SonarQube web interface for a project named '30exp8'. The 'Issues' tab is active, showing a list of issues under the 'Consistency' category. The left sidebar shows the 'Clean Code Attribute' section with 'Consistency' selected, showing 197k issues. The main panel lists four issues related to deprecated attributes and missing DOCTYPE declarations. Each issue includes a description, a severity level (L1, L9, L11, L12), an effort (5min), and a time ago (4 years ago). The issues are categorized as 'Bug' or 'Code Smell' with a 'Major' severity.

| Issue Description | Category | Severity | Effort | Time Ago |
|--|-----------------|------------|--------|-------------|
| Insert a <!DOCTYPE> declaration to before this <html> tag. | Reliability | Bug | 5min | 4 years ago |
| Remove this deprecated "width" attribute. | Maintainability | Code Smell | 5min | 4 years ago |
| Remove this deprecated "align" attribute. | Maintainability | Code Smell | 5min | 4 years ago |
| Remove this deprecated "align" attribute. | Maintainability | Code Smell | 5min | 4 years ago |

• Intentionality

The screenshot displays the SonarQube web interface for the same project '30exp8', but with the 'Intentionality' category selected. The left sidebar shows 'Intentionality' selected under 'Clean Code Attribute', with 14k issues. The main panel lists four issues related to missing version tags and double quotes around variables. Each issue includes a description, a severity level (L1, L12, L13), an effort (5min), and a time ago (4 years ago). The issues are categorized as 'Code Smell' with a 'Major' severity.

| Issue Description | Category | Severity | Effort | Time Ago |
|---|-----------------|------------|--------|-------------|
| Use a specific version tag for the image. | Maintainability | Code Smell | 5min | 4 years ago |
| Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. | Maintainability | Code Smell | 5min | 4 years ago |
| Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. | Maintainability | Code Smell | 5min | 4 years ago |
| Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. | Maintainability | Code Smell | 5min | 4 years ago |

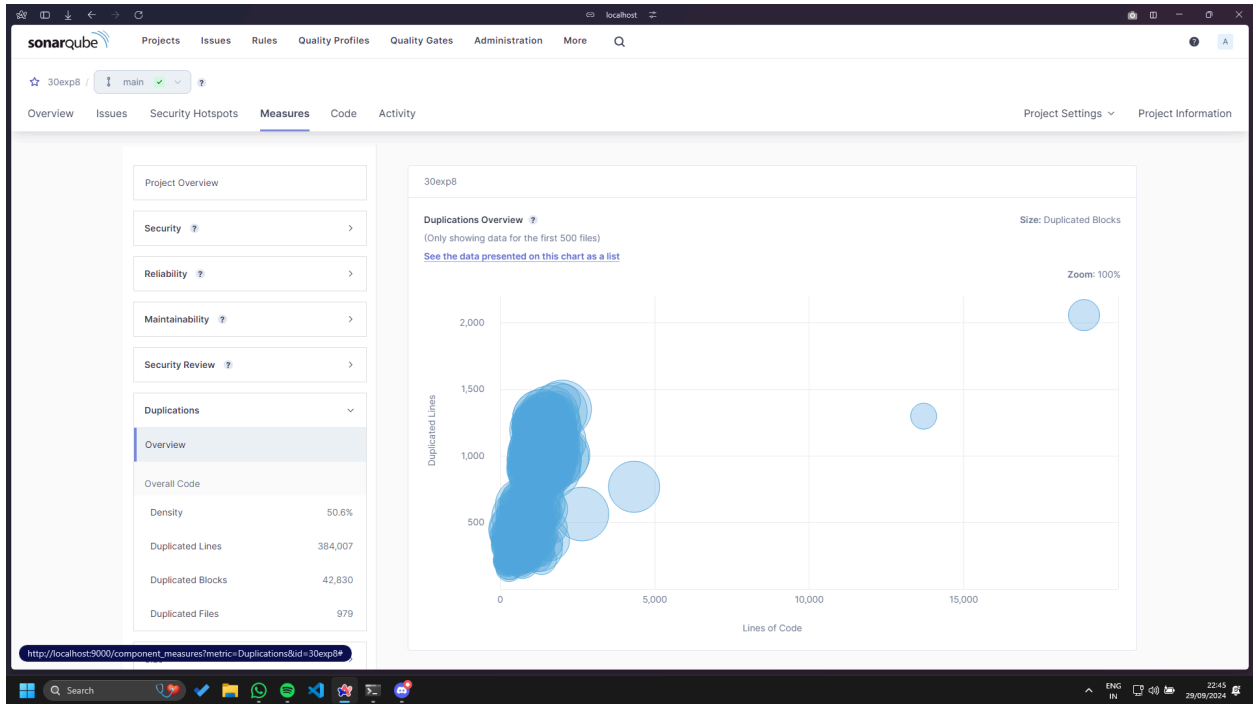
• Bugs

SonarQube interface showing Issues for project 30exp8. The left sidebar shows filters for Severity (High: 0, Medium: 14k, Low: 0) and Type (Bug: 14k, Vulnerability: 0, Code Smell: 268). The main area displays four issues related to HTML attributes and table headers, all marked as 'Intentionally' ignored. A warning at the bottom states: 'Embedded database should be used for evaluation purposes only'.

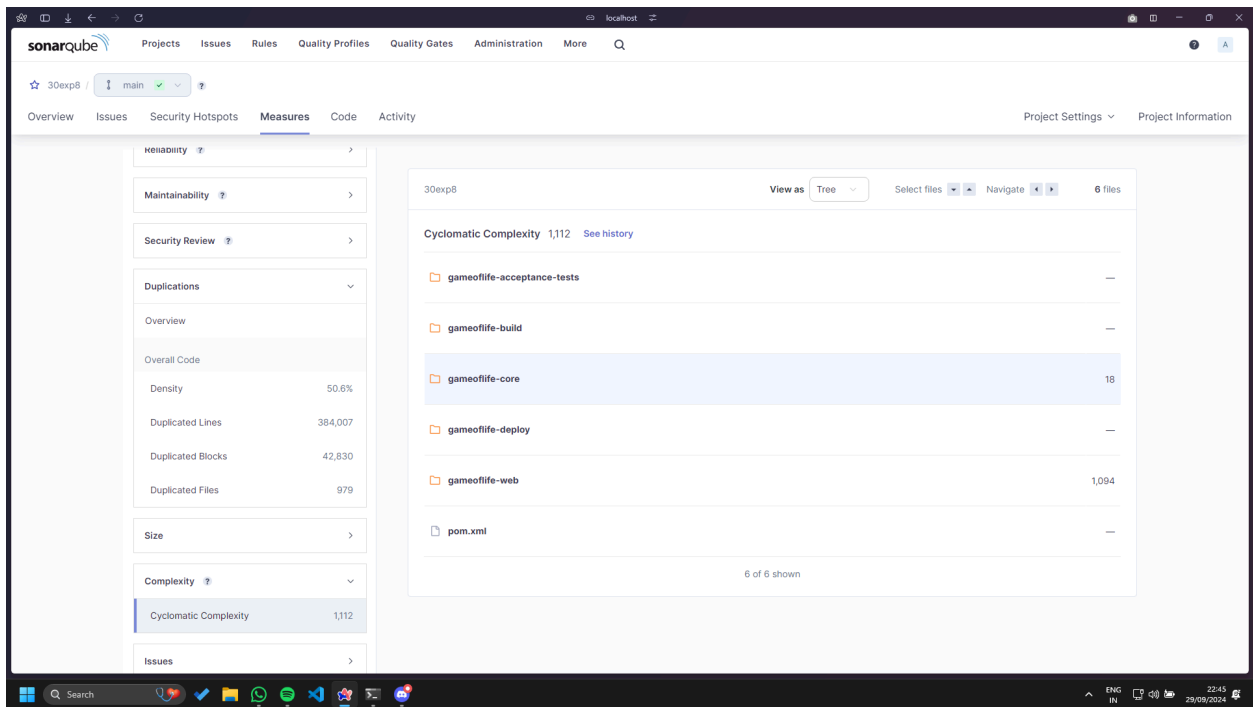
- Code Smells

SonarQube interface showing Code Smells for project 30exp8. The left sidebar shows filters for Severity (High: 7, Medium: 6, Low: 255) and Type (Bug: 14k, Vulnerability: 0, Code Smell: 268). The main area displays four code smells related to image version tags and variable quoting, all marked as 'Intentionally' ignored. A warning at the bottom states: 'Embedded database should be used for evaluation purposes only'.

- 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