

Exp:8

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.

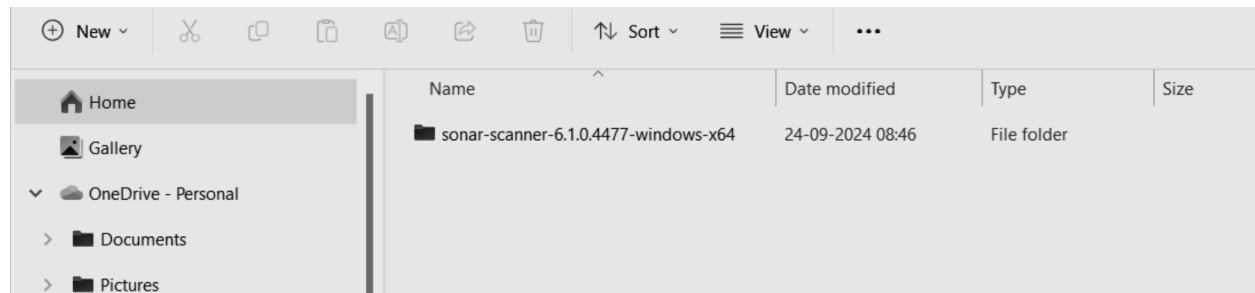
Step 1: Download sonar scanner

<https://docs.sonarsource.com/sonarqube/latest/analyzing-source-code/scanners/sonarscanner/>

The screenshot shows the SonarScanner CLI documentation page. The left sidebar contains a navigation menu with links like 'Homepage', 'Try out SonarQube', 'Server installation and setup', 'Analyzing source code', 'Scanners', and 'SonarScanner CLI'. The main content area is titled 'SonarScanner CLI' and features a '6.1' version section with download links for various operating systems and architectures. A 'START FREE' button is visible in the top right corner.

Visit this link and download the sonarqube scanner CLI.

Extract the downloaded zip file in a folder.



1. Install sonarqube image

Command: `docker pull sonarqube`

```
C:\Users\HP\Desktop\sem5\advdevops8>docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Image is up to date for sonarqube:latest
docker.io/library/sonarqube:latest
```

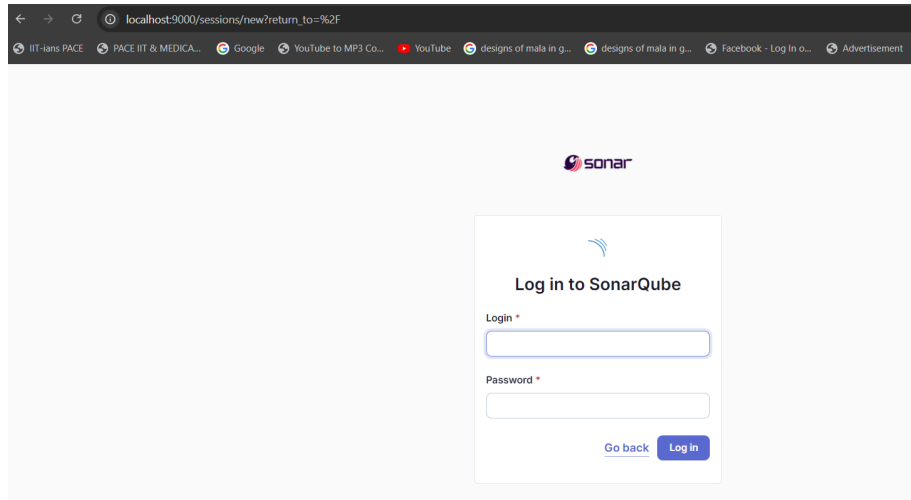
What's next:

View a summary of image vulnerabilities and recommendations → `docker scout quickview sonarqube`

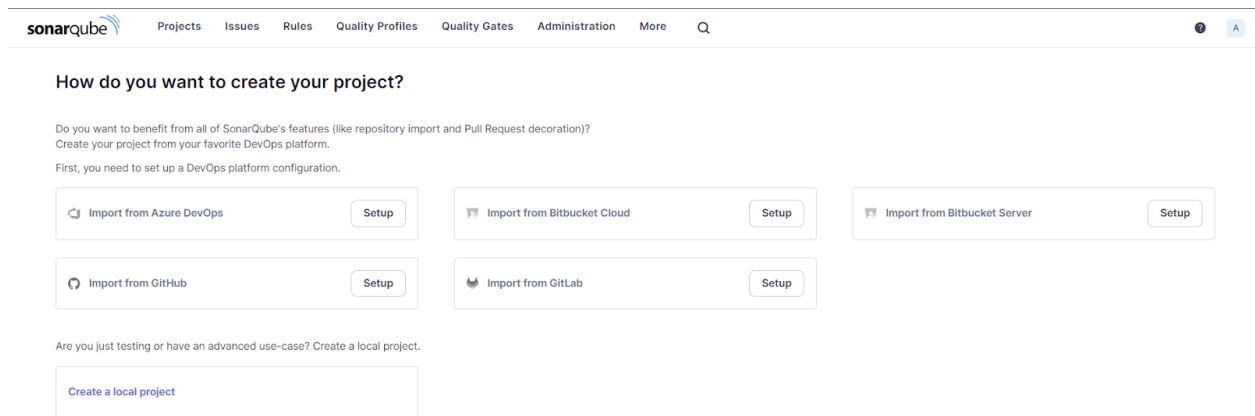
```
C:\Users\HP\Desktop\sem5\advdevops8>|
```

```
C:\Users\HP\Desktop\sem5\advdevops8>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
a57154161e14bed00ec141b755fa197a52321bf5c0688b825ff4dfbeaf712099
```

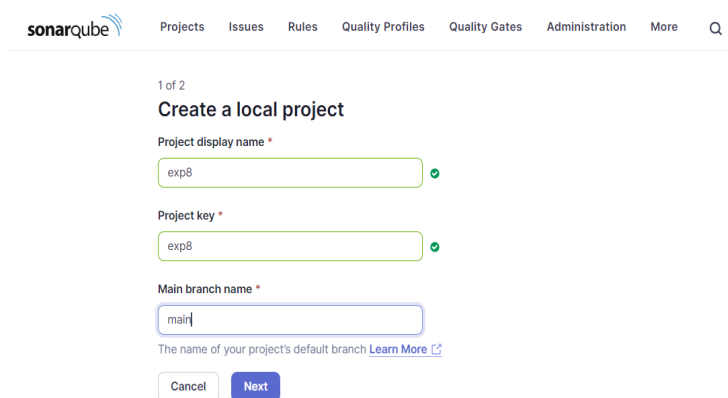
2. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.



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



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





2 of 2

Set up project for Clean as You Code

The new code definition sets which part of your code will be considered new code. T
You Code methodology. Learn more: [Defining New Code](#)

Choose the baseline for new code for this project

☒ Use the global setting

Previous version

Any code that has changed since the previous version is considered new code.

Recommended for projects following regular versions or releases.

5. Open up Jenkins Dashboard on localhost, port 8090 or whichever port it is at for you.

The Jenkins Dashboard shows the following components:

- Build Queue:** No builds in the queue.
- Build Executor Status:** 1 Idle, 2 Idle, 1 Offline (Sahil).
- Build History Table:**

S	W	Name	Last Success	Last Failure	Last Duration
✓	☁	exp7	22 hr #20	22 hr #18	33 sec
✓	☀	sahil 7	25 days #2	N/A	96 ms
✓	☀	Sahil exp6	25 days #3	N/A	1 sec
⌚	☀	SahilExp6	N/A	N/A	N/A
✗	☁	sahiljob	N/A	25 days #1	1.5 sec

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

The Jenkins Manage Jenkins Plugins page shows the following search results for "sonarq":

Install	Name	Released
<input checked="" type="checkbox"/>	SonarQube Scanner 2.17.2 External Site/Tool Integrations Build Reports This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.	7 mo 6 days ago
<input type="checkbox"/>	Sonar Gerrit 388.v9b_f1cb_e42306 External Site/Tool Integrations This plugin allows to submit issues from SonarQube to Gerrit as comments directly.	3 mo 20 days ago
<input type="checkbox"/>	SonarQube Generic Coverage 1.0 TODO	5 yr 1 mo ago

7. Under Jenkins 'Manage Jenkins' then go to 'system', scroll and look for SonarQube Servers and enter the details.

Enter the Server Authentication token if needed.

In SonarQube installations: Under Name add <project name of sonarqube> for me
adv_devops_7_sonarqube

In Server URL Default is <http://localhost:9000>

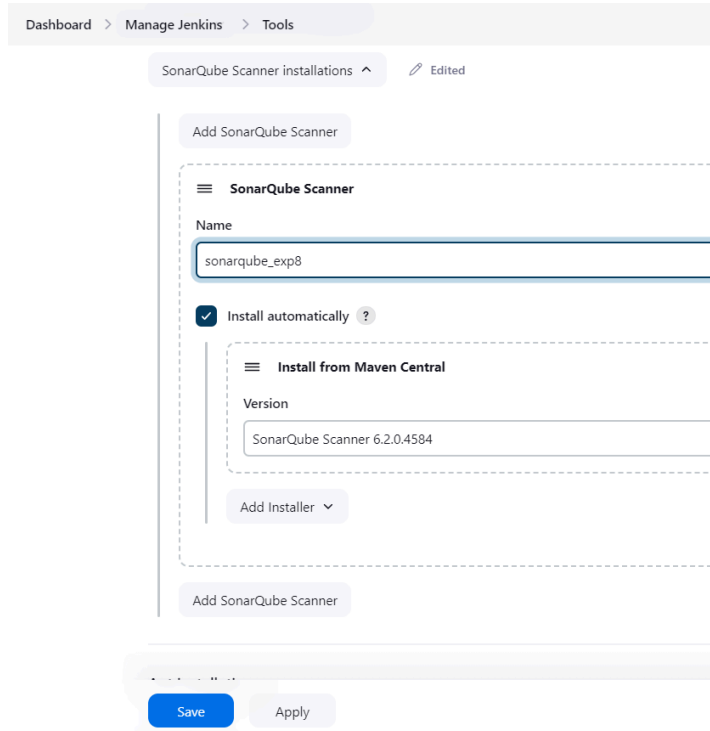
The screenshot shows the Jenkins 'System' configuration page under 'Manage Jenkins'. The breadcrumb trail is 'Dashboard > Manage Jenkins > System >'. The section is titled 'SonarQube installations' with a subtitle 'List of SonarQube installations'. A dashed box contains a form for adding a new installation. The form has three main sections: 'Name' with a text input field containing 'exp8'; 'Server URL' with a text input field containing 'http://localhost:9000' and a note 'Default is http://localhost:9000'; and 'Server authentication token' with a dropdown menu set to '- none -' and a note 'SonarQube authentication token. Mandatory when anonymous access is disabled.' Below the dropdown is a '+ Add' button. At the bottom of the dashed box is an 'Advanced' dropdown. Below the dashed box is an 'Add SonarQube' button. At the very bottom of the page are 'Save' and 'Apply' buttons.

8. Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.

Dashboard > Manage Jenkins > Tools

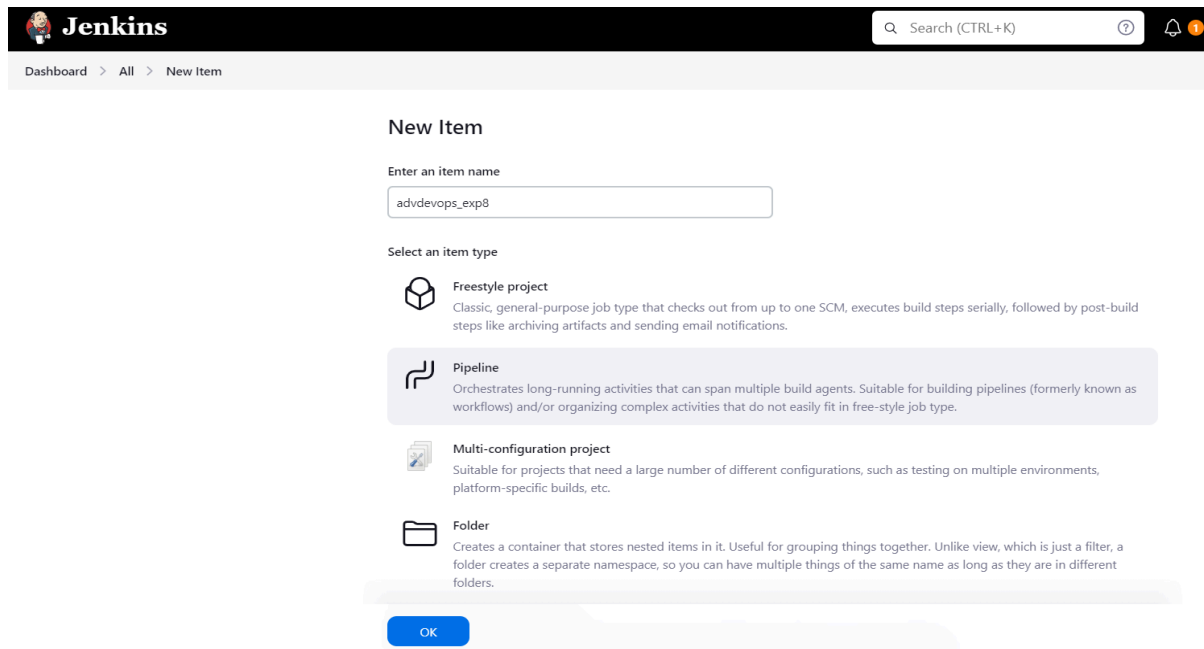
The screenshot shows the Jenkins 'Tools' configuration page under 'Manage Jenkins'. The breadcrumb trail is 'Dashboard > Manage Jenkins > Tools'. The page has several sections, each with an 'Add' button: 'Add Git', 'Gradle installations' with 'Add Gradle', 'SonarScanner for MSBuild installations' with 'Add SonarScanner for MSBuild', 'SonarQube Scanner installations' with a dropdown menu set to 'SonarQube Scanner installations' and an 'Edited' link, and 'Ant installations' with 'Add Ant'. At the bottom of the page are 'Save' and 'Apply' buttons.

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



The screenshot shows the Jenkins 'Manage Jenkins' > 'Tools' > 'SonarQube Scanner installations' page. A new installation is being added. The 'Name' field is filled with 'sonarqube_exp8'. The 'Install automatically' checkbox is checked. Under the 'Install from Maven Central' section, the 'Version' field is filled with 'SonarQube Scanner 6.2.0.4584'. At the bottom, there are 'Save' and 'Apply' buttons.

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



The screenshot shows the Jenkins 'New Item' page. The 'Enter an item name' field is filled with 'advdevops_exp8'. Under 'Select an item type', the 'Pipeline' option is selected and highlighted. The 'Pipeline' description reads: '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.' At the bottom, there is an 'OK' button.

10. Under Pipeline script, enter the following:

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

stage('SonarQube analysis') {
    withSonarQubeEnv('<Name_of_SonarQube_environment_on_Jenkins>') {
        sh """
            <PATH_TO_SONARQUBE_SCANNER_FOLDER>/bin/sonar-scanner \
            -D sonar.login=<SonarQube_USERNAME> \
            -D sonar.password=<SonarQube_PASSWORD> \
            -D sonar.projectKey=<Project_KEY> \
            -D sonar.exclusions=vendor/**,resources/**,**/*.java \
            -D sonar.host.url=<SonarQube_URL>(default: http://localhost:9000/)
        """
    }
}
}
```

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

Dashboard > advdevops_exp8 > Configuration

Configure

- General
- Advanced Project Options
- Pipeline**

Pipeline

Definition

Pipeline script

```
1 * node {
2   stage('Cloning the Github Repo') {
3     git 'https://github.com/shazforiot/GOL.git'
4   }
5   stage('SonarQube analysis') {
6     withSonarQubeEnv('exp8') {
7       bat
8       """
9       "C:\Users\HP\Desktop\sem5\advdevops8\sonar-scanner-6.1.0.4477-windows-x64\bin\sonar-scanner."
10      -D sonar.login=admin ^
11      -D sonar.password=anandpur ^
12      -D sonar.projectKey=exp8 ^
13      -D sonar.exclusions=vendor/**,resources/**,**/*.java ^
14      -D sonar.host.url=http://localhost:9000/
15      """
16     }
17   }
18 }
19 }
```

☒ Use Groovy Sandbox ?

[Pipeline Syntax](#)

[Save](#) [Apply](#)

11. Build project

Jenkins

Dashboard > advdevops_exp8 >

Status

Changes

Build Now

Configure

Delete Pipeline

SonarQube

Stages

Rename

Pipeline Syntax

advdevops_exp8

Permalinks

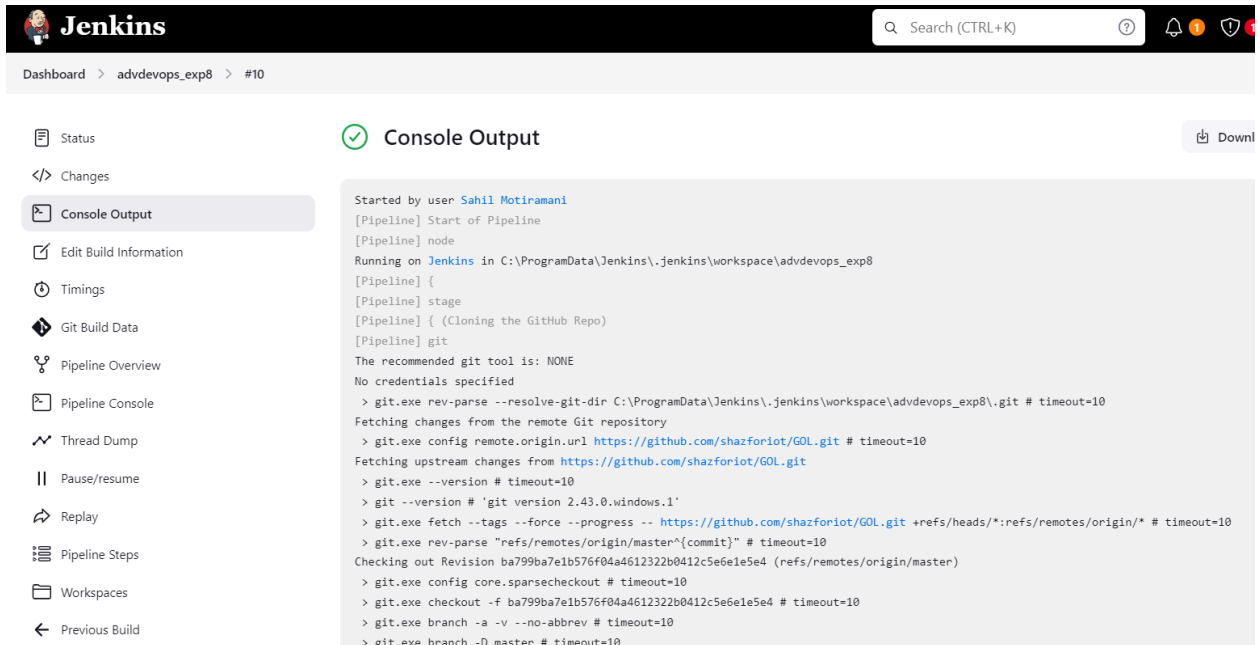
- Last build (#10), 17 min ago
- Last stable build (#10), 17 min ago
- Last successful build (#10), 17 min ago
- Last failed build (#7), 27 min ago
- Last unsuccessful build (#9), 19 min ago
- Last completed build (#10), 17 min ago

Build History trend ▾

#10

Sep 24, 2024, 7:01 PM

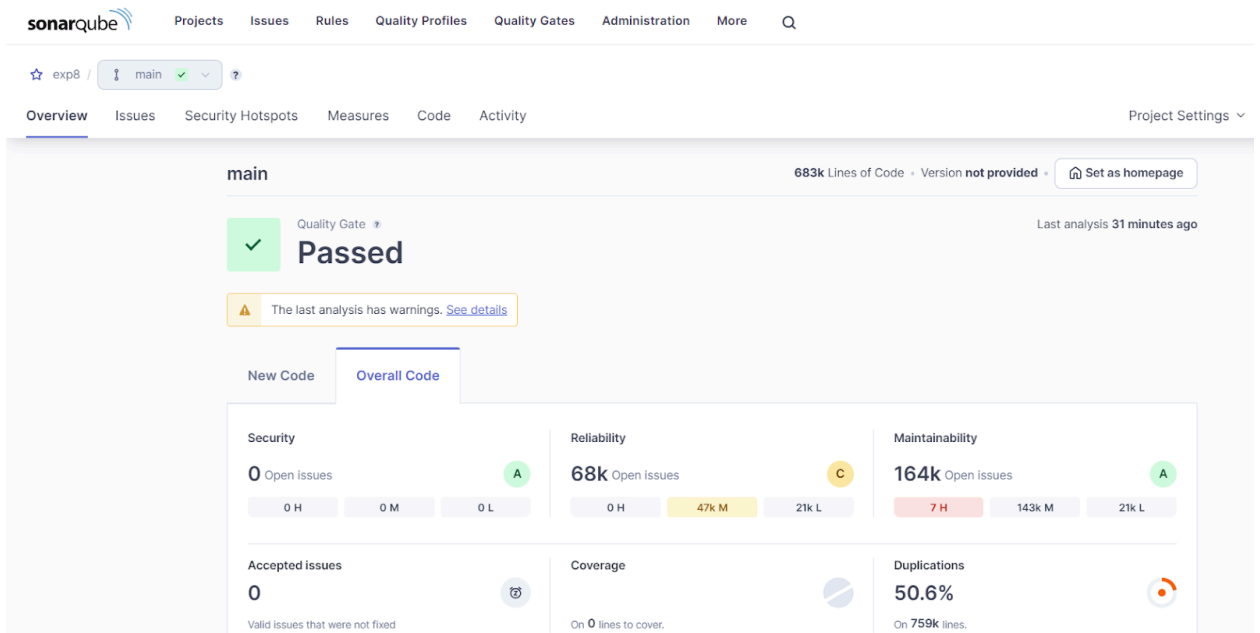
12. Check console



The screenshot shows the Jenkins web interface. The top navigation bar includes the Jenkins logo, a search bar, and notification icons. The breadcrumb trail is 'Dashboard > advdevops_exp8 > #10'. The left sidebar contains a list of links: Status, Changes, Console Output (selected), Edit Build Information, Timings, Git Build Data, Pipeline Overview, Pipeline Console, Thread Dump, Pause/resume, Replay, Pipeline Steps, Workspaces, and Previous Build. The main content area is titled 'Console Output' with a green checkmark icon and a 'Download' button. The console log shows the following output:

```
Started by user Sahil Motiramani
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in C:\ProgramData\Jenkins\.jenkins\workspace\advdevops_exp8
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Cloning the GitHub Repo)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\advdevops_exp8\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/shazforiot/GOL.git # timeout=10
Fetching upstream changes from https://github.com/shazforiot/GOL.git
> git.exe --version # timeout=10
> git --version # 'git version 2.43.0.windows.1'
> git.exe fetch --tags --force --progress -- https://github.com/shazforiot/GOL.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision ba799ba7e1b576f04a4612322b0412c5e6e1e5e4 (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f ba799ba7e1b576f04a4612322b0412c5e6e1e5e4 # timeout=10
> git.exe branch -a -v --no-abbrev # timeout=10
> git.exe branch -D master # timeout=10
```

13. Now, check the project in SonarQube



The screenshot shows the SonarQube web interface for a project named 'main'. The top navigation bar includes the SonarQube logo and links to Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, and a search bar. The breadcrumb trail is 'exp8 / main'. The main content area shows the 'main' branch with a 'Passed' status and a 'Quality Gate' icon. A warning message states: 'The last analysis has warnings. See details'. The 'Overview' tab is selected, showing various metrics:

Security	Reliability	Maintainability
0 Open issues	68k Open issues	164k Open issues
0 H, 0 M, 0 L	0 H, 47k M, 21k L	7 H, 143k M, 21k L

Accepted issues	Coverage	Duplications
0	50.6%	50.6%
Valid issues that were not fixed	On 0 lines to cover.	On 759k lines.

14. Code Problems

- Consistency

The screenshot shows the SonarQube web interface. The top navigation bar includes 'sonarqube', 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', 'Administration', and 'More'. Below the navigation bar, there's a search bar and a dropdown menu set to 'main'. The main content area is titled 'Issues' and shows a list of issues. On the left, there's a sidebar with filters: 'My Issues', 'All', 'Filters', 'Clear All Filters', 'Issues in new code', 'Clean Code Attribute' (1), 'Consistency' (197k), 'Intentionality' (14k), 'Adaptability' (0), 'Responsibility' (0), 'Add to selection Ctrl + click', 'Software Quality'. The main area displays three issues related to 'Intentionality':

- Insert a <!DOCTYPE> declaration to before this <html> tag.** (Consistency, user-experience, L1, 5min effort, 4 years ago, Bug, Major)
- Remove this deprecated "width" attribute.** (Consistency, html5, obsolete, L9, 5min effort, 4 years ago, Code Smell, Major)
- Remove this deprecated "align" attribute.** (Consistency, html5, obsolete, L9, 5min effort, 4 years ago, Code Smell, Major)

● Intentionality

The screenshot shows the SonarQube web interface. The top navigation bar includes 'sonarqube', 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', 'Administration', and 'More'. Below the navigation bar, there's a search bar and a dropdown menu set to 'main'. The main content area is titled 'Issues' and shows a list of issues. On the left, there's a sidebar with filters: 'My Issues', 'All', 'Filters', 'Clear All Filters', 'Issues in new code', 'Clean Code Attribute' (1), 'Consistency' (197k), 'Intentionality' (14k), 'Adaptability' (0), 'Responsibility' (0), 'Add to selection Ctrl + click', 'Software Quality'. The main area displays three issues related to 'Intentionality':

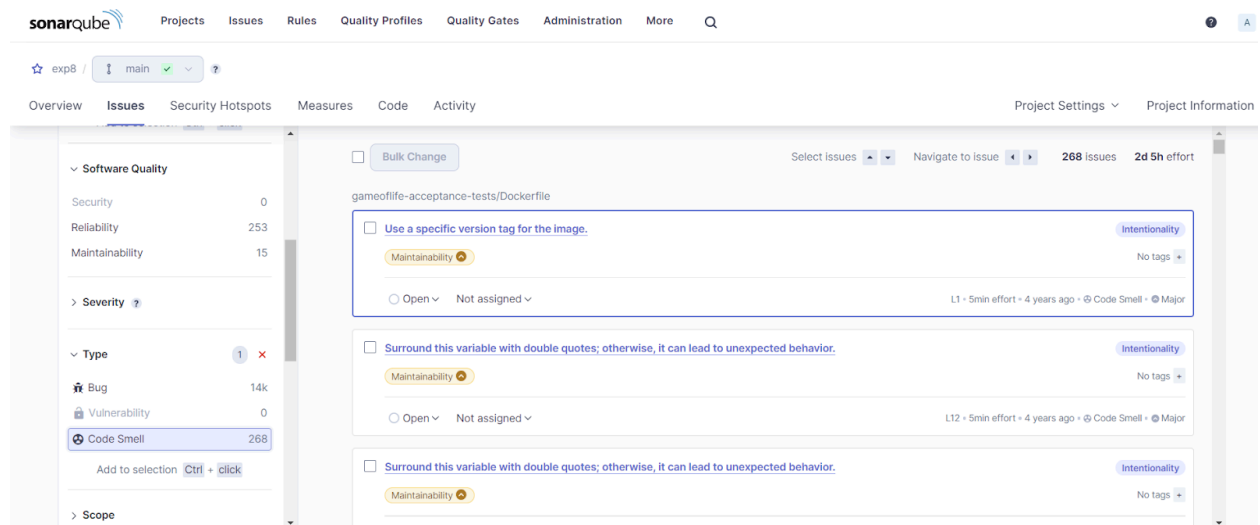
- Use a specific version tag for the image.** (Intentionality, No tags, L1, 5min effort, 4 years ago, Code Smell, Major)
- Surround this variable with double quotes; otherwise, it can lead to unexpected behavior.** (Intentionality, No tags, L12, 5min effort, 4 years ago, Code Smell, Major)
- Surround this variable with double quotes; otherwise, it can lead to unexpected behavior.** (Intentionality, No tags, L12, 5min effort, 4 years ago, Code Smell, Major)

● Bugs

The screenshot shows the SonarQube web interface. The top navigation bar includes 'sonarqube', 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', 'Administration', and 'More'. Below the navigation bar, there's a search bar and a dropdown menu set to 'main'. The main content area is titled 'Issues' and shows a list of issues. On the left, there's a sidebar with filters: 'My Issues', 'All', 'Filters', 'Clear All Filters', 'Issues in new code', 'Clean Code Attribute' (1), 'Consistency' (197k), 'Intentionality' (14k), 'Adaptability' (0), 'Responsibility' (0), 'Add to selection Ctrl + click', 'Software Quality'. The main area displays three issues related to 'Bugs':

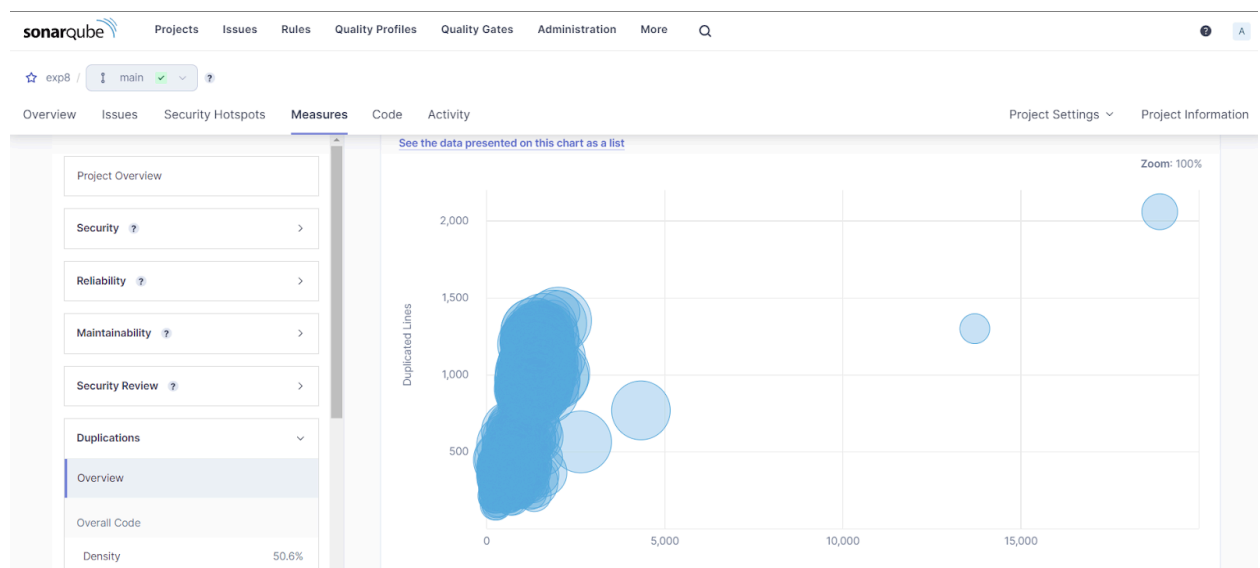
- Add "lang" and/or "xml:lang" attributes to this "html" element** (Intentionality, accessibility, wcag2-a, L1, 2min effort, 4 years ago, Bug, Major)
- Add "<th>" headers to this "<table>".** (Intentionality, accessibility, wcag2-a, L9, 2min effort, 4 years ago, Bug, Major)
- Add "lang" and/or "xml:lang" attributes to this "html" element** (Intentionality, accessibility, wcag2-a, L1, 2min effort, 4 years ago, Bug, Major)

● Code Smells



The screenshot shows the SonarQube interface for the 'exp8' project, specifically the 'Issues' tab. The left sidebar displays 'Software Quality' metrics: Security (0), Reliability (253), and Maintainability (15). Under 'Severity', there are 14k Bugs, 0 Vulnerabilities, and 268 Code Smells. The 'Type' filter is set to 'Code Smell'. The main area shows a list of issues for the file 'gameoflife-acceptance-tests/Dockerfile'. Three issues are visible, all categorized as 'Maintainability' and 'Intentionality'. The first issue is 'Use a specific version tag for the image.' with a severity of 'L1', 5min effort, and 4 years ago. The second and third issues are 'Surround this variable with double quotes; otherwise, it can lead to unexpected behavior.' with a severity of 'L12', 5min effort, and 4 years ago. The interface includes a 'Bulk Change' button, 'Select issues' dropdown, and 'Navigate to issue' controls. The top right shows '268 Issues' and '2d 5h effort'.

● Duplications



● Cyclomatic Complexities

The screenshot displays the SonarQube web interface for project 'exp8'. The 'Measures' tab is active, showing a table of code quality metrics. On the left, a sidebar provides an overview of duplications and a detailed view of the 'Cyclomatic Complexity' measure, which has a value of 1,112. The main content area shows a tree view of the project's structure, listing modules like 'gameoflife-acceptance-tests', 'gameoflife-build', 'gameoflife-core', 'gameoflife-deploy', and 'gameoflife-web', along with the 'pom.xml' file. The 'gameoflife-core' module is highlighted with a value of 18, and 'gameoflife-web' has a value of 1,094.

Measure	Value
Density	50.6%
Duplicated Lines	384,007
Duplicated Blocks	42,801
Duplicated Files	979
Cyclomatic Complexity	1,112

Module	Value
gameoflife-acceptance-tests	—
gameoflife-build	—
gameoflife-core	18
gameoflife-deploy	—
gameoflife-web	1,094
pom.xml	—

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

Conclusion:

In this experiment, we integrated Jenkins with SonarQube to enable automated code quality checks within our CI/CD pipeline. We started by deploying SonarQube using Docker, setting up a project, and configuring it to analyze code quality. Next, we configured Jenkins by installing the SonarQube Scanner plugin, adding SonarQube server details, and setting up the scanner tool. We then developed a Jenkins pipeline to automate the process of cloning a GitHub repository and running SonarQube analysis on the code. This integration helps ensure continuous monitoring of code quality, detecting issues such as bugs, code smells, and security vulnerabilities throughout the development process.