

## 08 Advanced DevOps Lab

### 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.

### Steps

#### 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.

The screenshot shows the SonarScanner CLI download page. The left sidebar contains a search bar and a navigation menu with links to 'Homepage', 'Try out SonarQube', 'Server installation and setup', 'Analyzing source code', 'Scanners', 'Scanner environment', and 'SonarScanner CLI'. The main content area is titled 'SonarScanner CLI' and shows the latest version '6.1' with a release date of '2024-06-27'. It lists supported distributions: 'macOS and Linux AArch64 distributions'. Below this, it provides download links for 'Linux x64', 'Linux AArch64', 'Windows x64', 'macOS x64', 'macOS AArch64', and 'Docker'. A note states 'Any (Requires a pre-installed JVM)'. There is also a link to 'Release notes'. The text below the download links explains that the SonarScanner CLI is the scanner to use when there is no specific scanner for your build system, and that it does not yet officially support ARM architecture.

Extract the downloaded zip file in a folder.

The screenshot shows a Windows File Explorer window with the address bar displaying 'Downloads > sonar-scanner-cli-6.1.0.4477-windows-x64 >'. The search bar contains 'Search sonar-scanner-cli-6.1.0.4477-windows-x64'. The file list shows a single item: 'sonar-scanner-6.1.0.4477-windows-x64', which is a 'File folder' created on '18-09-2024 14:58'.

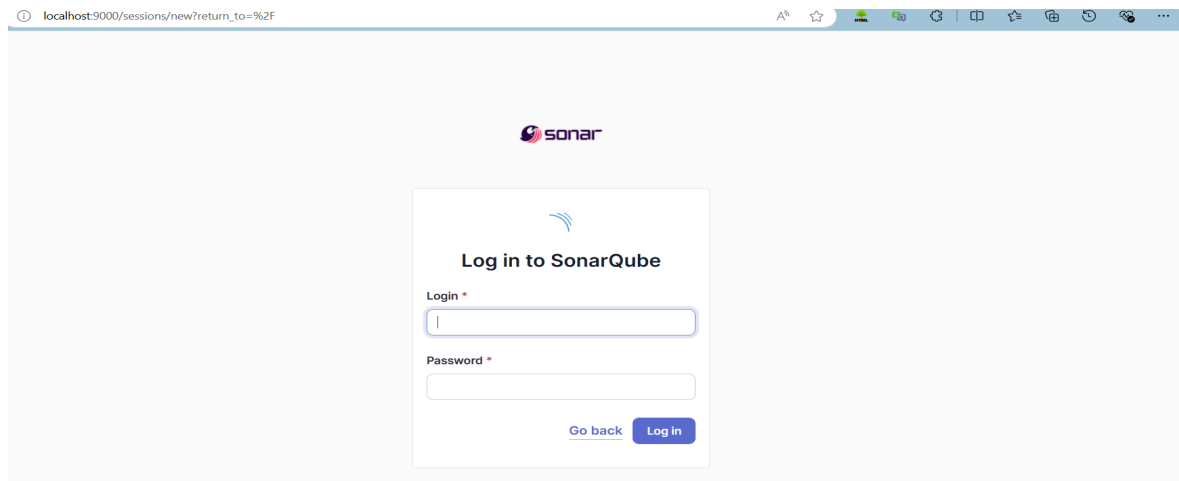
#### 1. Install sonarqube image

Command: **docker pull sonarqube** (skip if already installed we did install it in exp 7)

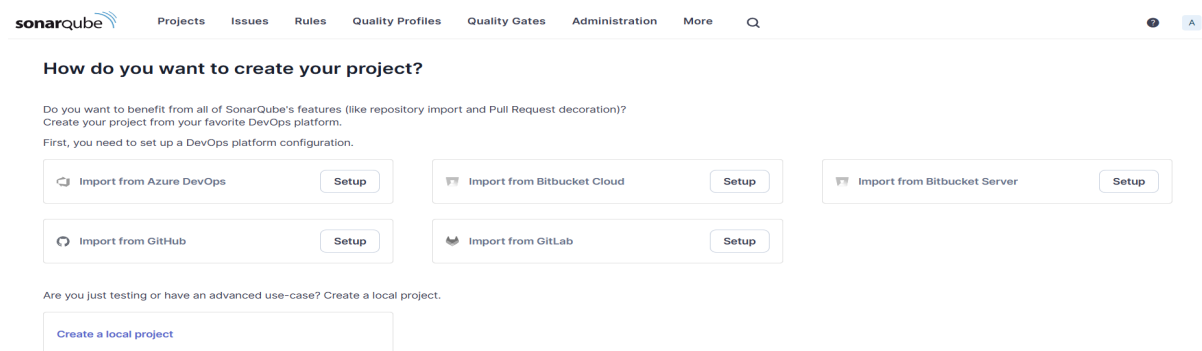
Then run the image

```
C:\Users\Lenovo>docker run -d -p 9000:9000 sonarqube
5007285df5d17d62fef087bc6b74409e37fff333d6308ee62bd323fed5716d5d
C:\Users\Lenovo>
```

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 local project in SonarQube with the name sonarqube

1 of 2

## Create a local project

**Project display name \***

sonarqube ✓

**Project key \***

sonarqube ✓

**Main branch name \***

main

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

Cancel

Next

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. This helps you focus at 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.

☐ Define a specific setting for this project☐ Previous version

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

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

The screenshot shows the Jenkins Dashboard. On the left, there's a sidebar with navigation links: New Item, Build History, Project Relationship, Check File Fingerprint, Manage Jenkins, and My Views. Below these are sections for 'Build Queue' (showing 'No builds in the queue') and 'Build Executor Status' (showing 'Built-In Node' with 1 idle and 2 offline executors, and 'Slave1' which is offline). The main area displays a table of build history with columns: S (Status), W (Icon), Name, Last Success, Last Failure, and Last Duration. The table lists several builds, including 'DevOps Pipeline', 'DevOps-NewJob', 'DevOpsPipeLineExp6', 'exp7', 'exp72', 'maven-project', 'MavenDemo', and 'webApp'. At the bottom, there are icons for 'S', 'M', and 'L'.

S	W	Name	Last Success	Last Failure	Last Duration
✓	☀	DevOps Pipeline	1 mo 23 days #4	N/A	6.9 sec
✓	☀	DevOps-NewJob	1 mo 14 days #1	N/A	0.67 sec
✓	☀	DevOpsPipeLineExp6	1 mo 7 days #1	N/A	2.7 sec
✓	☁	exp7	20 hr #5	22 hr #4	53 sec
✗	☁	exp72	N/A	20 hr #3	2 sec
⋯	☀	maven-project	N/A	N/A	N/A
✗	☁	MavenDemo	N/A	1 mo 23 days #2	25 sec
✓	☁	webApp	1 mo 0 days #5	1 mo 0 days #4	11 sec

6. Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.(we already installed it for exp 7 so you can skip )

The screenshot shows the 'Manage Jenkins' page, specifically the 'Plugins' tab. A search bar at the top contains 'sonarq'. Below the search bar, there's a table of plugins. The table has columns: Install, Name, and Released. The 'SonarQube Scanner 2.17.2' plugin is listed, with links for 'External Site/Tool Integrations' and 'Build Reports'. The description states: 'This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.' The release date is '6 mo 29 days ago'. There are 'Install', 'Uninstall', and 'Refresh' buttons at the top right of the plugin list.

Install	Name	Released
<input type="checkbox"/>	SonarQube Scanner 2.17.2 <a href="#">External Site/Tool Integrations</a> <a href="#">Build Reports</a> This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.	6 mo 29 days ago

Dashboard > Manage Jenkins > Plugins

### Plugins

- Updates 25
- Available plugins
- Installed plugins
- Advanced settings
- Download progress**

### Download progress

Preparation

- Checking internet connectivity
- Checking update center connectivity
- Success

SonarQube Scanner Success

Loading plugin extensions Success

→ [Go back to the top page](#)  
(you can start using the installed plugins right away)

→ ☐ Restart Jenkins when installation is complete and no jobs are running

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.(we dont need the token and this step was done in previous exp but jic)

In SonarQube installations: Under **Name** add <project name of sonarqube> for me **sonarqube\_exp8**  
In **Server URL** Default is **http://localhost:9000**

Name

sonarqube\_exp8

Server URL

Default is http://localhost:9000

http://localhost:9000

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

- none -

+ Add ▾

Advanced ▾

Add SonarQube

✓ Saved

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

**Dashboard > Manage Jenkins > Tools > SonarQube Scanner** (i kept the default setting from last experiment and just changed the name).

The screenshot shows the 'SonarQube Scanner installations' page in Jenkins. At the top, there's a header 'SonarQube Scanner installations' with a sub-header 'SonarQube Scanner Installations ^' and an 'Edited' status. Below this is a button 'Add SonarQube Scanner'. The main configuration area is a dashed box containing a 'SonarQube Scanner' section. Inside this section, there's a 'Name' field with the value 'sonarqube\_scanner\_exp8'. Below the name field is a checkbox 'Install automatically' which is checked. Underneath the checkbox is a sub-section 'Install from Maven Central' containing a 'Version' dropdown menu with the value 'SonarQube Scanner 6.2.0.4584'. At the bottom of this sub-section is a button 'Add Installer'. The entire configuration area is enclosed in a dashed box with a close button 'x' in the top right corner.

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


This screenshot is a closer view of the configuration area shown in the previous image. It shows the 'SonarQube Scanner' section with the 'Name' field set to 'sonarqube\_exp8'. The 'Install automatically' checkbox is checked. Below it, the 'Install from Maven Central' sub-section is visible, showing the 'Version' dropdown set to 'SonarQube Scanner 6.2.0.4584' and the 'Add Installer' button. The configuration area is enclosed in a dashed box with a close button 'x' in the top right corner.

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


Enter an item name

adDevOps\_exp8


» Required field


**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.


**Maven project**


Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.


**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.


**Multi-configuration project**


Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.


**Folder**

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.


**Multibranch Pipeline**

Creates a set of Pipeline projects according to detected branches in one SCM repository.


**Organization Folder**

[OK](#) a set of multibranch project subfolders by scanning for repositories.

10. Under Pipeline script, enter the following:

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

  stage('SonarQube Analysis') {
    withSonarQubeEnv('exp8') {
      bat """
      "C:\\Program Files\\Sonar
Scanner\\sonar-scanner-6.2.0.4584-windows-x64\\bin\\sonar-scanner.bat" ^
      -Dsonar.login=<username> ^
      -Dsonar.password=<password> ^
      -Dsonar.projectKey=<project-key> ^
      -Dsonar.exclusions=vendor/*,resources/,./java ^
    """
    }
  }
}
```

```

-Dsonar.host.url=http://127.0.0.1:9000/
    ""
}
}
}

```

\*Note that the code has placeholders

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

Definition

Pipeline script

Script ?

```

1 node {
2   stage('Cloning the GitHub Repo') {
3     git 'https://github.com/shazforiot/GOL.git'
4   }
5
6   stage('SonarQube Analysis') {
7     withSonarQubeEnv('sonarqube_exp8') {
8       bat """
9         "C:\sonar-scanner-6.2.0.4584-windows-x64\bin\sonar-scanner.bat" ^
10        -Dsonar.login=admin ^
11        -Dsonar.password=2923 ^
12        -Dsonar.projectKey=sonarqube ^
13        -Dsonar.exclusions=vendor/**/*.resources/**/*.java ^
14        -Dsonar.host.url=http://127.0.0.1:9000/
15        ""
16      }
17    }
18  }

```

☒ Use Groovy Sandbox ?

[Pipeline Syntax](#)

Save Apply

## 11. Build project

#5 Sep 26 12:24 No Changes	2s	
#4 Sep 26 12:21 No Changes	1s	
#3 Sep 26 12:13 No Changes	1s	1min 54s failed
#2 Sep 26 12:10 No Changes	2s	1s failed

Build links

## 12. Check console

Dashboard > adDevOps\_exp8 > #5

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete build #5

Timings

Git Build Data

Pipeline Overview

Pipeline Console

Replay

Pipeline Steps

Workspaces

Previous Build

Next Build

**Console Output**

Skipping 4.248 KB. [Full Log](#)

```
12:35:00.367 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html for block at line 810. Keep only references.
12:35:00.367 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html for block at line 512. Keep only references.
12:35:00.367 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html for block at line 789. Keep only references.
12:35:00.367 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html for block at line 512. Keep only references.
12:35:00.367 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html for block at line 248. Keep only references.
12:35:00.367 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html for block at line 886. Keep only references.
12:35:00.367 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html for block at line 249. Keep only references.
12:35:00.367 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html for block at line 662. Keep only references.
12:35:00.367 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html for block at line 615. Keep only references.
12:35:00.367 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html for block at line 664. Keep only references.
12:35:00.367 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html for block at line 913. Keep only references.
12:35:00.367 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html for block at line 810. Keep only references.
12:35:00.367 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html for block at line 668. Keep only references.
```

### 13. Now, check the project in SonarQube

sonarqube

Projects Issues Rules Quality Profiles Quality Gates Administration More Q

sonarqube main

Overview Issues Security Hotspots Measures Code Activity

Project Settings Project Information

main 683k Lines of Code - Version not provided Set as homepage

Quality Gate

**Passed** Last analysis 30 minutes ago

The last analysis has warnings. [See details](#)

New Code Overall Code

Security 0 Open issues 0 H 0 M 0 L

Reliability 68k Open issues 0 H 47% M 2% L

Maintainability 164k Open issues 7 H 143k M 2% L

Accepted issues 0 Valid issues that were not fixed

Coverage 0 On 0 lines to cover.

Duplications 50.6% On 759k lines.

Security Hotspots 3

### 14. Code Problems

#### Consistency

sonarqube

Projects Issues Rules Quality Profiles Quality Gates Administration More Q

sonarqube main

Overview Issues Security Hotspots Measures Code Activity

Project Settings Project Information

My Issues All

Filters Clear All Filters

Issues in new code

Clean Code Attribute 1 X

Consistency 19%

Intentionality 14k

Adaptability 0

Responsibility 0

Add to selection Ctrl + Click

Software Quality

Security 0

Reliability 54k

Maintainability 164k

gameoflife-core/build/reports/tests/all-tests.html

Insert a <IDOCATYPE> declaration to before this <html> tag. Consistency

Remove this deprecated "width" attribute. Consistency

Remove this deprecated "align" attribute. Consistency

Remove this deprecated "align" attribute. Consistency

Introducing Clean Code Attributes

Clean Code attributes are the characteristics that your code must have to be considered Clean Code.

You can now filter by these attributes to evaluate why your code is breaking away from being clean.

1 of 5

Next

L11 - Seen effort - 4 years ago - @ Code Small - @ Major

#### Intentionality



SonarQube interface showing Issues for project gameofflife-acceptance-tests/Dockerfile. The left sidebar shows filters for Clean Code Attribute (Intentionality: 14k) and Software Quality (Security: 0, Reliability: 14k, Maintainability: 15). The main area displays a list of issues, including 'Use a specific version tag for the image' and 'Surround this variable with double quotes; otherwise, it can lead to unexpected behavior'.

## Bugs

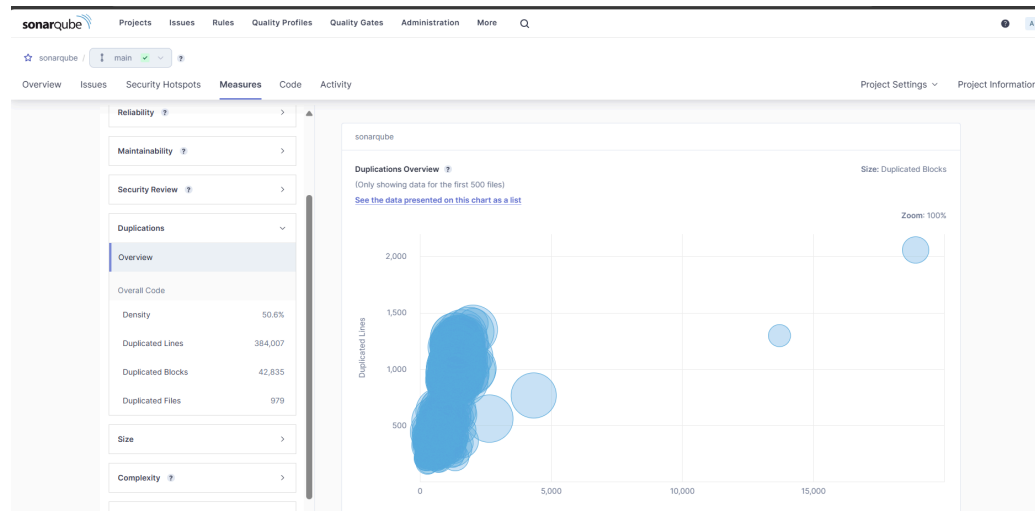
SonarQube interface showing Bugs for project gameofflife-core/build/reports/tests/all-tests.html. The left sidebar shows filters for Software Quality (Security: 0, Reliability: 14k, Maintainability: 0) and Type (Bug: 14k). The main area displays a list of bugs, including 'Add "lang" and/or "xmlns:lang" attributes to this <html> element' and 'Add "tbody" headers to this <table>'.

## Code Smells

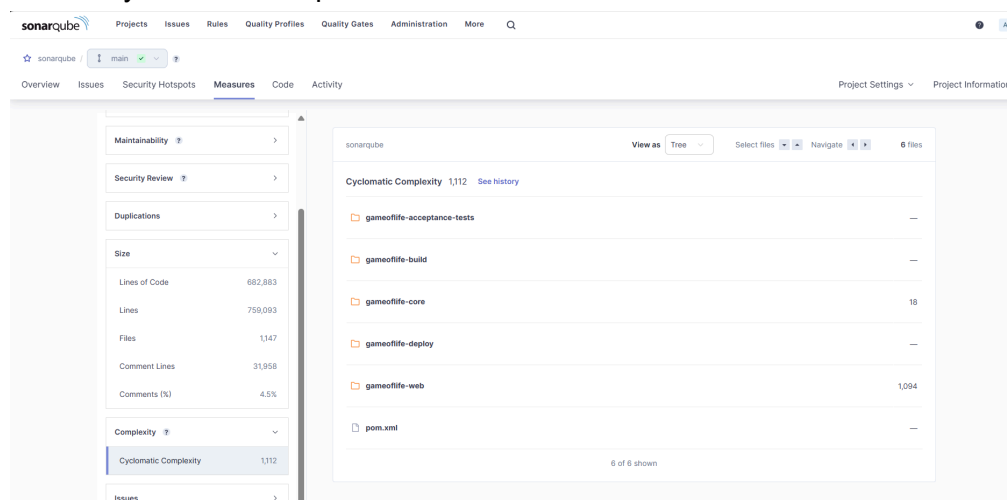
SonarQube interface showing Code Smells for project gameofflife-acceptance-tests/Dockerfile. The left sidebar shows filters for Software Quality (Security: 0, Reliability: 253, Maintainability: 15) and Type (Bug: 14k, Vulnerability: 0, Code Smell: 268). The main area displays a list of code smells, including 'Use a specific version tag for the image' and 'Surround this variable with double quotes; otherwise, it can lead to unexpected behavior'.

Embedded database should be used for evaluation purposes only

## Duplications



## ● Cyclomatic Complexities



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

## Conclusion:

In this experiment, we successfully integrated Jenkins with SonarQube to automate continuous code quality monitoring within our CI/CD pipeline. This process involved deploying SonarQube via Docker, setting up a project for analysis, and configuring Jenkins with the SonarQube Scanner plugin. After configuring the necessary tools and adding SonarQube server details, we created a Jenkins pipeline that automates cloning from GitHub and running static analysis on the code. This integration allows us to detect potential bugs, code smells, and security vulnerabilities at every stage of development, ensuring improved code quality and streamlined development workflows.