

EXPERIMENT-8

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

THEORY:

Static application security testing or static analysis, is a testing methodology that analyzes source code to find security vulnerabilities that make your organization's applications susceptible to attack. SAST runs on an application before the code is compiled. It is also known as white box testing.

SAST tools give developers real-time feedback as they code, helping them fix issues before they pass the code to next phase of SDLC.

SAST tools automatically identify critical vulnerabilities - such as buffer overflows, SQL injection, cross-site scripting and others - with high confidence. Thus integrating static analysis into SDLC can yield dramatic results in overall quality of code developed.

There are six simple steps needed to perform SAST efficiently in organizations that have a very large number of applications built with different languages, frameworks and platforms.

- ① Finalize the tool
- ② creating scanning interface and deploy the tool
- ③ customize the tool
- ④ Prioritize and onboard applications
- ⑤ analyze scan results
- ⑥ provide governance and training

It is important to note that SAST tools must be run on application on regular basis, such as daily / monthly builds, everytime code is checked in or during a code release.

IMPLEMENTATION:

Step 1: Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.

S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	🔗	DeployWarExp-6	2 days 3 hr #7	2 days 4 hr #6	16 sec
✗	🔗	devops-expt-5	N/A	1 mo 4 days #1	0.21 sec
✗	🔗	mavenproject	N/A	2 mo 5 days #2	64 ms
✗	🔗	pipeline D15B	2 mo 5 days #3	2 mo 5 days #4	4.4 sec

Build Queue: No builds in the queue.

Build Executor Status:

- 1 idle
- 2 idle

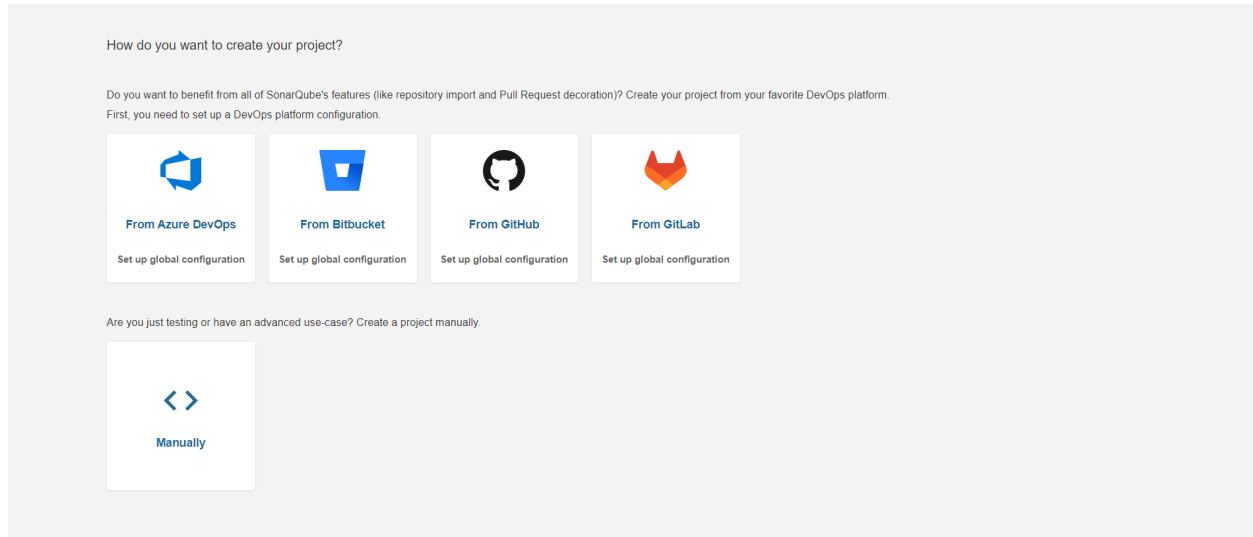
Step 2: Run SonarQube in a Docker container using this command -

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

```
C:\Users\mayuri>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonar:latest
```

```
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
9621f1afde84: Pull complete
0da9106727c7: Pull complete
129c5a3f9c32: Pull complete
Digest: sha256:3fa9a76948fab6faf41950bee256afea943773744723b5e4f38b340643516b9
Status: Downloaded newer image for sonarqube:latest
895ac0b29a9a624f8a4828b0cce1649d1ab0d32ea65874e34a24eff068a9bc93
```

Step 3: Once the container is up and running, you can check the status of SonarQube at localhost port 9000.

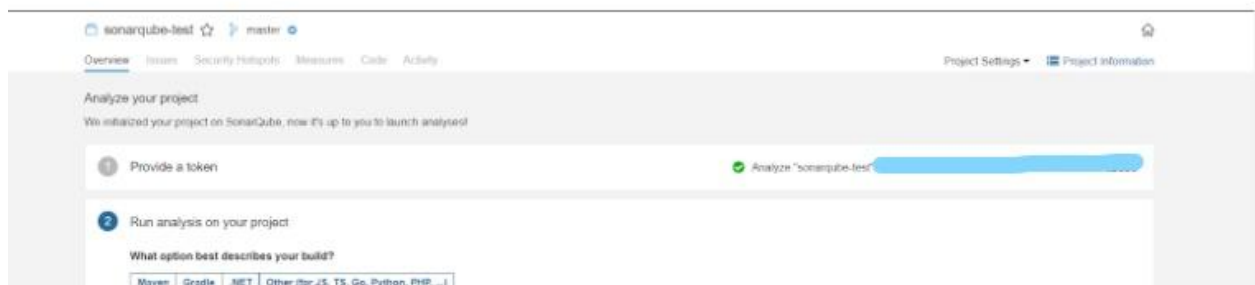


- Login to SonarQube using username admin and password admin.

Step 4: Create a manual project in SonarQube with the name sonarqube-test

The screenshot shows the "Create a project" form in SonarQube. The title "Create a project" is at the top. Below it, a note says "All fields marked with * are required". The first field is "Project display name *" with a text input containing "sonarqube-test" and a green checkmark icon to its right. Below this field, a note says "Up to 255 characters. Some scanners might override the value you provide." The second field is "Project key *" with a text input containing "sonarqube-test" and a green checkmark icon to its right. Below this field, a note says "The project key is a unique identifier for your project. It may contain up to 400 characters. Allowed characters are alphanumeric, '-' (dash), '_' (underscore), '.' (period) and ':' (colon), with at least one non-digit." At the bottom of the form is a "Set Up" button.

Step 5: Set a token name and generate it.
(Copy the token for the integration with jenkins)




Step 6: Setup the project and come back to Jenkins Dashboard.

- Add Credentials of SonarQube Project in Jenkins for your current user.

Credentials

T	P	Store	Domain	ID ↓	Name
---	---	-------	--------	------	------

Stores scoped to Jenkins

P	Store ↓	Domains
	Jenkins	(global)

- Go to Global Credentials

Global credentials (unrestricted)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name ↑	Kind	Description
This credential domain is empty. How about adding some credentials?			

Step 7: Add SonarQube Project Token

New credentials

Kind

Secret text

▼

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

▼

Secret

.....

ID ?

sonarqube-token

Description ?



sonarqube-token

Create

- As you can see the credentials have been added for the user.

Global credentials (unrestricted)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name ↑	Kind	Description
 sonarqube-token	sonarqube-token	Secret text	sonarqube-token 

Step 8: Under configuration system, tab mention the SonarQube installations

SonarQube installations

List of SonarQube installations

Name

sonarqube

Server URL

Default is http://localhost:9000

http://localhost:9000

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

sonarqube-token

+ Add

Step 9: Add SonarQube Scanner installation in Global Tool Configuration

SonarQube Scanner installations

List of SonarQube Scanner installations on this system

Add SonarQube Scanner

☰ SonarQube Scanner

Name

SonarQubeScanner

☒ Install automatically ?

☰ Install from Maven Central

Version

SonarQube Scanner 4.7.0.2747


Add Installer ▾


Step 10: Create a New Item in Jenkins, choose Pipeline.


Enter an item name


SonarPipeline


» Required field

**Freestyle project**
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

**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.

**OK**
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

Step 11: Under Pipeline Script, enter the following -

```
node {
  stage('Cloning the GitHub Repo') {
    git 'https://github.com/shazforiot/GOL.git'
  }
  stage('SonarQube analysis') {
    withSonarQubeEnv('sonarqube') {
      sh
"/c/ProgramData//Jenkins/.jenkins/tools/hudson.plugins.sonar.SonarRunnerInstallation/sonarqube/bin//sonar-scanner \
-D sonar.login=admin \
-D sonar.password=yourpassword \
-D sonar.projectKey=sonarqube-test \
-D sonar.exclusions=vendor/**,resources/**,**/*.java \
-D sonar.host.url=http://127.0.0.1:9000/"
    }
  }
}
```


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') {
7       sh "/c/ProgramData/Jenkins/.jenkins/tools/hudson.plugins.sonar.SonarRunnerInstallation/sonarqube/bin/sonar-scanner \
8         -D sonar.login=admin \
9         -D sonar.password= \
10        -D sonar.projectKey=sonarqube-test \
11        -D sonar.exclusions=vendor/**,resources/**,*/*.java \
12        -D sonar.host.url=http://127.0.0.1:9000/"
13     }
14   }
15 }
16 }
```

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

Step 12: Run The Build.

Dashboard > SonarPipeline >

Status

Changes

Build Now

Configure

Delete Pipeline

Full Stage View

SonarQube

Rename

Pipeline Syntax

Recent Changes

Stage View

Average stage times:
(Average full run time: ~5min 59s)

	Cloning the GitHub Repo	SonarQube analysis
Average	50s	3min 14s
#4 Sep 29 09:42 No Changes	1s	5min 57s

Step 13: Check the console output once the build is complete.

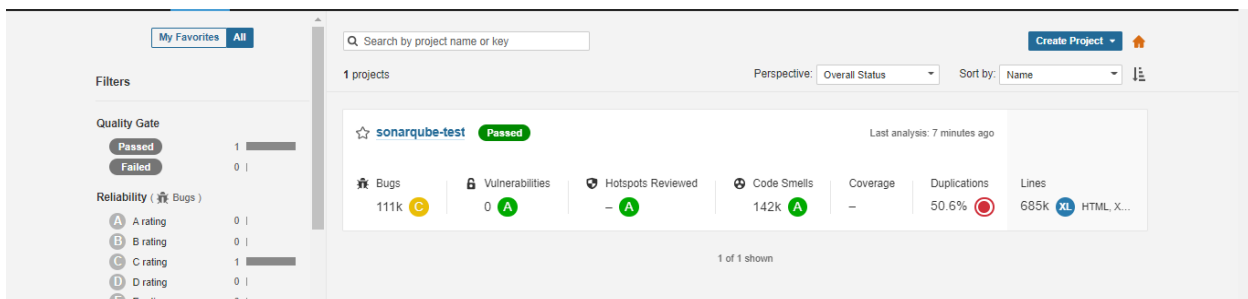
Console Output

Skipping 3,974 KB. [Full Log](#)

```
WARN: Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/modifiers/BeanShellPreProcessorBeanInfo.html
for block at line 230. Keep only the first 100 references.
WARN: Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/modifiers/BeanShellPreProcessorBeanInfo.html
for block at line 240. Keep only the first 100 references.
WARN: Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/modifiers/BeanShellPreProcessorBeanInfo.html
for block at line 32. Keep only the first 100 references.
WARN: Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/modifiers/BeanShellPreProcessorBeanInfo.html
for block at line 227. Keep only the first 100 references.
WARN: Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/modifiers/BeanShellPreProcessorBeanInfo.html
for block at line 230. Keep only the first 100 references.
WARN: Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/modifiers/BeanShellPreProcessorBeanInfo.html
for block at line 40. Keep only the first 100 references.
WARN: Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/modifiers/BeanShellPreProcessorBeanInfo.html
for block at line 41. Keep only the first 100 references.
...
line 75. Keep only the first 100 references.
INFO: CPD Executor CPD calculation finished (done) | time=78312ms
INFO: Analysis report generated in 1685ms, dir size=129.8 MB
INFO: Analysis report compressed in 7315ms, zip size=29.8 MB
INFO: Analysis report uploaded in 1809ms
INFO: ANALYSIS SUCCESSFUL, you can find the results at: http://127.0.0.1:9000/dashboard?id=sonarqube-test
INFO: Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
INFO: More about the report processing at http://127.0.0.1:9000/api/ce/task?id=AYOHd3nhvIxyw13qDg62
INFO: Analysis total time: 5:53.919 s
INFO: -----
INFO: EXECUTION SUCCESS
INFO: -----
INFO: Total time: 5:55.751s
INFO: Final Memory: 16M/74M
INFO: -----
[Pipeline] }
[Pipeline] // withSonarQubeEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Step 14: After that, check the project in SonarQube.

- Wait till the SAST process gets completed on SonarQube.



The screenshot displays the SonarQube web interface. On the left, there are filters for 'My Favorites' and 'All'. The main area shows a list of projects, with 'sonarqube-test' highlighted. The project status is 'Passed' (indicated by a green circle). Below the project name, there are several metrics: Bugs (111k, yellow circle), Vulnerabilities (0, green circle), Hotspots Reviewed (1, green circle), Code Smells (142k, green circle), Coverage (50.6%, red circle), Duplications (685k, blue circle), and Lines (HTML X...). The interface also shows a search bar at the top and a 'Create Project' button.

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

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

A background task is in progress. More details available on the [Background Tasks](#) page.

QUALITY GATE STATUS

Passed
All conditions passed.

MEASURES

New Code Overall Code

111k Bugs	Reliability
0 Vulnerabilities	Security
0 Security Hotspots	Reviewed Security Review
1477d Debt	142k Code Smells Maintainability

Code Problems -

Bugs and Code Smells

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

1 / 253,152 issues 3171d effort

Bulk Change

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

Insert a <!DOCTYPE> declaration to before this <html> tag. 2 years ago L1 user-experience

Bug Major Open Not assigned 5min effort Comment

Add "lang" and/or "xml:lang" attributes to this "<html>" element 2 years ago L1 accessibility, wcag2-a

Bug Major Open Not assigned 2min effort Comment

Add "<th>" headers to this "<table>". 2 years ago L9 accessibility, wcag2-a

Bug Major Open Not assigned 2min effort Comment

Remove this deprecated "width" attribute. 2 years ago L9 html5, obsolete

Code Smell Major Open Not assigned 5min effort Comment

Add a description to this table. 2 years ago L9 accessibility, wcag2-a

Bug Minor Open Not assigned 5min effort Comment

Remove this deprecated "align" attribute. 2 years ago L11 html5, obsolete

Code Smell Major Open Not assigned 5min effort Comment

Remove this deprecated "align" attribute. 2 years ago L12 html5, obsolete

Code Smell Major Open Not assigned 5min effort Comment

Remove this deprecated "size" attribute. 2 years ago L17

Filters

Period

New code

Type

Bug 111k

Vulnerability 0

Code Smell 142k

Severity

Blocker 0 Minor 65k

Critical 0 Info 2

Major 188k

Scope

Resolution

Status

Security Category

Creation Date

Language

Duplications -

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

sonarqube-test View as Tree to select files to navigate 6 files

Duplicated Lines (%) 50.6%

	Duplicated Lines (%)	Duplicated Lines
gameoflife-acceptance-tests	0.0%	0
gameoflife-build	0.0%	0
gameoflife-core	9.6%	374
gameoflife-deploy	0.0%	0
gameoflife-web	50.9%	383,633
pom.xml	0.0%	0

6 of 6 shown

Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 9.6.1 (build 59531) - LGPL v3 - Community - Documentation - Plugins - Web API

Cyclomatic Complexity -

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

sonarqube-test View as Tree to select files to navigate 6 files

Cyclomatic Complexity 1,113

gameoflife-acceptance-tests	-
gameoflife-build	-
gameoflife-core	18
gameoflife-deploy	-
gameoflife-web	1,095
pom.xml	-

6 of 6 shown

Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 9.6.1 (build 59531) - LGPL v3 - Community - Documentation - Plugins - Web API

CONCLUSIONS

We successfully created Jenkins CI/CD pipeline with SonarQube to perform static analysis of code to detect bugs, code smells and security vulnerabilities on sample application.