

Name: Tarunkumar Sharma
Roll no: 56

Expt No. 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.

Theory:

What is SAST?

Static application security testing (SAST), 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 scans an application before the code is compiled. It's also known as white box testing.

What problems does SAST solve?

SAST takes place very early in the software development life cycle (SDLC) as it does not require a working application and can take place without code being executed. It helps developers identify vulnerabilities in the initial stages of development and quickly resolve issues without breaking builds or passing on vulnerabilities to the final release of the application.

SAST tools give developers real-time feedback as they code, helping them fix issues before they pass the code to the next phase of the SDLC. This prevents security-related issues from being considered an afterthought. SAST tools also provide graphical representations of the issues found, from source to sink. These help you navigate the code easier. Some tools point out the exact location of vulnerabilities and highlight the risky code. Tools can also provide in-depth guidance on how to fix issues and the best place in the code to fix them, without requiring deep security domain expertise.

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

Why is SAST important?

Developers dramatically outnumber security staff. It can be challenging for an organization to

find the resources to perform code reviews on even a fraction of its applications. A key strength of SAST tools is the ability to analyze 100% of the codebase. Additionally, they are much faster than manual secure code reviews performed by humans. These tools can scan millions of lines of code in a matter of minutes. SAST tools automatically identify critical vulnerabilities—such as buffer overflows, SQL injection, cross-site scripting, and others—with high confidence.

What is a CI/CD Pipeline?

CI/CD pipeline refers to the Continuous Integration/Continuous Delivery pipeline. Before we dive deep into this segment, let's first understand what is meant by the term 'pipeline'?

A pipeline is a concept that introduces a series of events or tasks that are connected in a sequence to make quick software releases. For example, there is a task, that task has got five different stages, and each stage has got some steps. All the steps in phase one have to be completed, to mark the latter stage to be complete.



Now, consider the CI/CD pipeline as the backbone of the DevOps approach. This Pipeline is responsible for building codes, running tests, and deploying new software versions. The Pipeline executes the job in a defined manner by first coding it and then structuring it inside several

blocks that may include several steps or tasks.

What is SonarQube?

SonarQube is an open-source platform developed by SonarSource for continuous inspection of code quality. Sonar does static code analysis, which provides a detailed report of bugs, code smells, vulnerabilities, code duplications.

It supports 25+ major programming languages through built-in rulesets and can also be extended with various plugins.

Benefits of SonarQube

- **Sustainability** - Reduces complexity, possible vulnerabilities, and code duplications, optimising the life of applications.
- **Increase productivity** - Reduces the scale, cost of maintenance, and risk of the application; as such, it removes the need to spend more time changing the code
- **Quality code** - Code quality control is an inseparable part of the process of software development.
- **Detect Errors** - Detects errors in the code and alerts developers to fix them automatically before submitting them for output.
- **Increase consistency** - Determines where the code criteria are breached and enhances the quality
- **Business scaling** - No restriction on the number of projects to be evaluated
- **Enhance developer skills** - Regular feedback on quality problems helps developers to improve their coding skills

Integrating Jenkins with SonarQube:

Prerequisites:

- Jenkins installed

- Docker Installed (for SonarQube)
- SonarQube Docker Image

Steps to create a Jenkins CI/CD Pipeline and use SonarQube to perform SAST

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

```
C:\Users\tarun>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
docker: Error response from daemon: Conflict. The container name "/sonarqube" is already in use by container "53d4af904582e15d29003562ae68704002c13d93fb8c2256b6d198f5a0724743". You have to remove (or rename) that container to be able to reuse that name.
See 'docker run --help'.
```

```
C:\Users\tarun>docker run -d --name sonarqube8 -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
fe8be66efe7847b23fe6b5ca7846a8ce8685ae741f933c0dc073034b0db4380a
```

```
C:\Users\tarun>
```

2. Run SonarQube in a Docker container using this command -

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

4. Create project manually

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

6. Create a New Item in Jenkins, choose Pipeline.

New Item

Enter an item name

expi8

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.



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

7. Enter the following in pipeline script:

Definition

Pipeline script

Script ?

```
1 node {
2   stage('Cloning the Github Repo') {
3     git 'https://github.com/PrajaktaUpadhye6/MSBuild_firstproject.git'
4   }
5   stage('SonarQube analysis') {
6     withSonarQubeEnv('sonarqube') {
7       bat "D:/sonar-scanner-cli-5.0.1.3006-windows/sonar-scanner-5.0.1.3006-windows/bin/sonar-scanner.bat \
8         -D sonar.login=admin \
9         -D sonar.password=abc \
10        -D sonar.projectKey=AdDevops \
11        -D sonar.exclusions=vendor/**,resources/**,**/*.java \
12        -D sonar.host.url=http://127.0.0.1:9000/"
13     }
14   }
15 }
16
```

try sample Pipeline...



Use Groovy Sandbox ?

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

Build and run:

Status

</> Changes

Build Now

Configure

Delete Pipeline

Full Stage View

GitHub

SonarQube

Rename

Pipeline Syntax

Build History

trend

Filter builds...

#8

Sep 20, 2023, 12:36 PM

Atom feed for all

Atom feed for failures

Pipeline SonarQube

Lab 8

Stage View

Average stage times:
(Average full run time: ~36s)

#8

Sep 20
12:36

No Changes

Cloning the GitHub Repo	SonarQube analysis
2s	33s
2s	33s

Permalinks

- Last build (#8), 1 min 44 sec ago
- Last stable build (#8), 1 min 44 sec ago
- Last successful build (#8), 1 min 44 sec ago
- Last completed build (#8), 1 min 44 sec ago

Console output:

```
17:49:39.413 INFO   CPD Executor CPD calculation finished (done) | time=159837ms
17:49:39.428 INFO   SCM revision ID 'ba799ba7e1b576f04a4612322b0412c5e6e1e5e4'
17:52:16.075 INFO   Analysis report generated in 4781ms, dir size=127.2 MB
17:52:33.696 INFO   Analysis report compressed in 17599ms, zip size=29.6 MB
17:52:34.987 INFO   Analysis report uploaded in 1290ms
17:52:34.989 INFO   ANALYSIS SUCCESSFUL, you can find the results at: http://127.0.0.1:9000/dashboard?id=sonarqube
17:52:34.989 INFO   Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
17:52:34.989 INFO   More about the report processing at http://127.0.0.1:9000/api/ce/task?id=8c04859e-050f-41e5-b320-ee32d44d4259
17:52:43.609 INFO   Analysis total time: 16:49.875 s
17:52:43.612 INFO   SonarScanner Engine completed successfully
17:52:44.416 INFO   EXECUTION SUCCESS
17:52:44.419 INFO   Total time: 17:04.839s
[Pipeline] }
[Pipeline] // withSonarQubeEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
Pipeline completed successfully!
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Sonarqube:

☆ AdDevops PUBLIC

✓ Passed

Last analysis: 3 minutes ago • 543 Lines of Code • HTML, CSS

E 7

A 0

E 0.0%

A 8

—

22.3%

Bugs

Vulnerabilities

Hotspots Reviewed

Code Smells

Coverage

Duplications


☆ AdDevops / main ✓

The last analysis has warnings. [See details](#) Version not provided

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

Quality Gate Status

✓ Quality Gate Passed



Enjoy your sparkling clean code!

Measures

New Code Overall Code

Reliability E

7 Bugs

Maintainability A

8 Code Smells

Security A

0 Vulnerabilities

Security Review E

6 Security Hotspots

Duplications

Reliability:

☆ AdDevops / main ✓

The last analysis has warnings. [See details](#) Version not provided

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

Project Overview

Reliability

Overview

New Code

Bugs 0

Rating A

Remediation Effort 0

Overall Code

Bugs 7

AdDevops

View as List Select files Navigate 15 files

Reliability Rating E See history

New Code: Since September 20, 2023

scroll.css E

trial.css C

verticaltable.html C

demo.html B

index.html B

payment.html B

There are 9 hidden components with a score of A. [Show Them](#)

Bugs:

AdDevops / main [?](#) [The last analysis has warnings. See details](#) Version not provided [?](#)

Overview Issues Security Hotspots **Measures** Code Activity Project Settings Project Information

Bugs 0

Rating **A**

Remediation Effort 0

Overall Code

Bugs 7

Rating **E**

Remediation Effort 28min

Security [?](#) >

Security Review [?](#) >

Bugs 7 [See history](#) **New Code:** Since September 20, 2023

box.html	0
class.css	0
demo.html	1
element.css	0
float property.html	0
home.html	0
id.css	0
index.html	1
payment.html	1
scroll.css	1

Security:

AdDevops / main [?](#) [The last analysis has warnings. See details](#) Version not provided [?](#)

Overview Issues Security Hotspots **Measures** Code Activity Project Settings Project Information

Project Overview

Reliability [?](#) >

Security [?](#) >

Security Review [?](#) v

New Code

Security Hotspots 0

Rating **A**

Overall Code

Security Hotspots 6

AdDevops **View as** List Select files Navigate 15 files

Security Review Rating E [See history](#) **New Code:** Since September 20, 2023

demo.html	E
home.html	E
index.html	E

There are 12 hidden components with a score of A. Show Them

Duplications:

AdDevops / main [?](#) [The last analysis has warnings. See details](#) Version not provided [?](#)

Overview Issues Security Hotspots **Measures** Code Activity Project Settings Project Information

Project Overview

Reliability [?](#) >

Security [?](#) >

Security Review [?](#) >

Maintainability [?](#) >

Coverage >

Duplications v

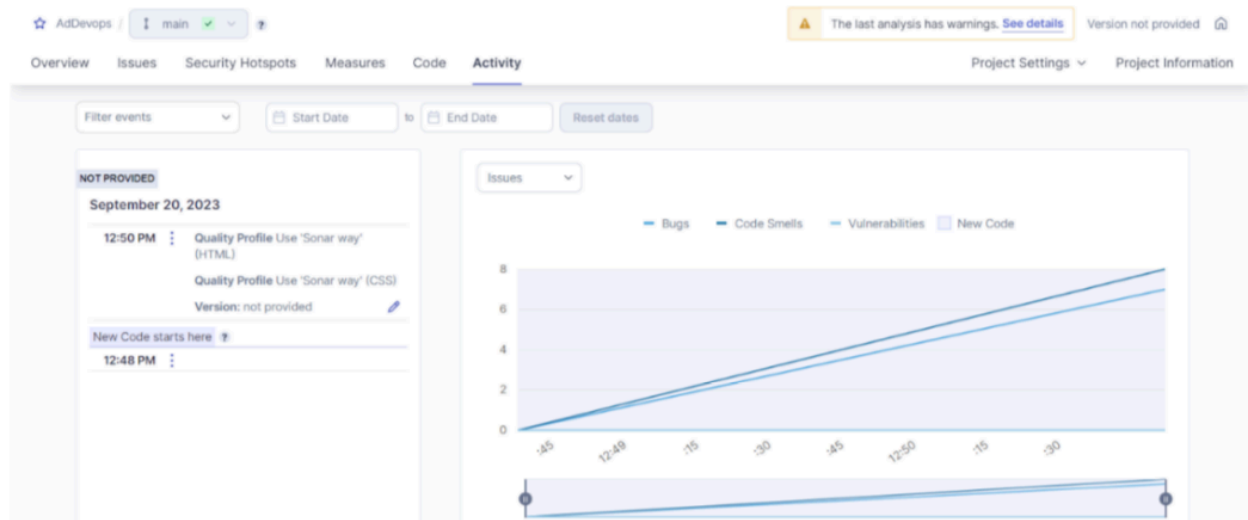
Overview

AdDevops **View as** List Select files Navigate 15 files

Duplicated Lines (%) 22.3% [See history](#) **New Code:** Since September 20, 2023

	Duplicated Lines (%)	Duplicated Lines
demo.html	100%	67
index.html	100%	67
box.html	0.0%	0
class.css	0.0%	0
element.css	0.0%	0
float property.html	0.0%	0

Activity:



Conclusion: Thus, we have successfully integrated Jenkins with SonarQube.