Advanced DevOps Lab
Experiment 08

Roll No.	24
Name	Iyer Sreekesh Subramanian
Class	D15-A
Subject	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.

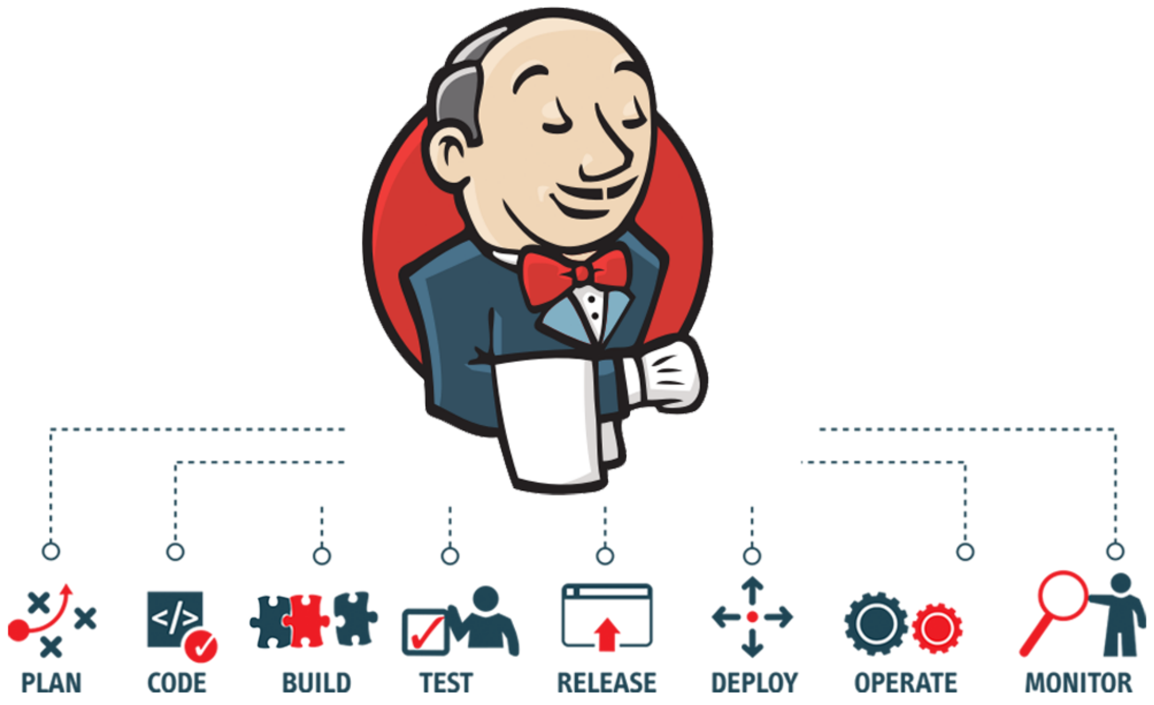
Why is SAST important?

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:

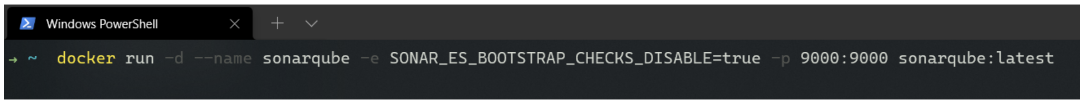
Prerequisites:

- Jenkins installed
- Docker Installed (for SonarQube)
- SonarQube Docker Image

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

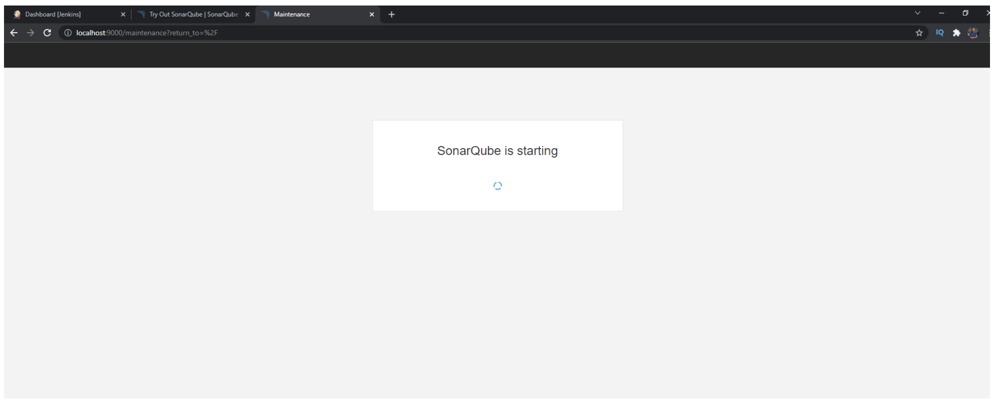
1. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.

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 sonarqube:latest
```

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

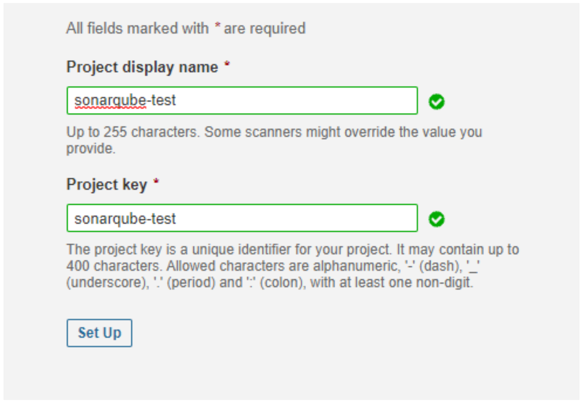


Dashboard [Jenkins]
Try Out SonarQube | SonarQube
Maintenance
localhost:9000/maintenance?return_to=%2F

SonarQube is starting

4. Login to SonarQube using username *admin* and password *admin*.

5. Create a manual project in SonarQube with the name **sonarqube-test**



All fields marked with * are required

Project display name *

sonarqube-test

Up to 255 characters. Some scanners might override the value you provide.

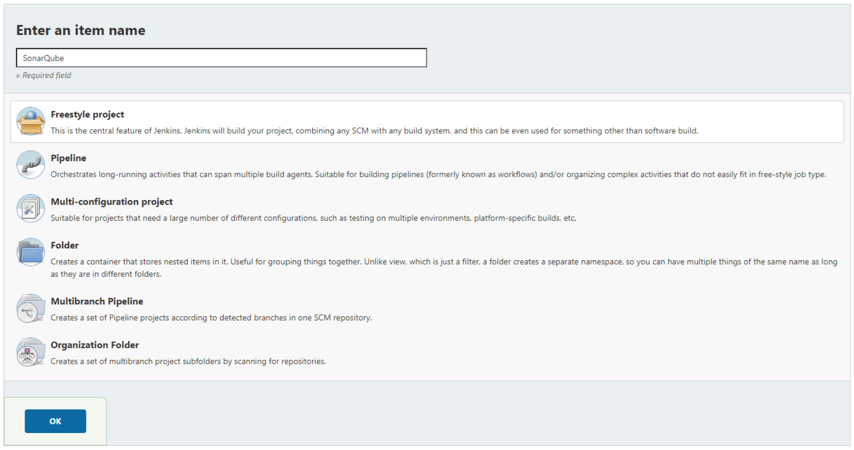
Project key *

sonarqube-test

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.

Setup the project and come back to Jenkins Dashboard.

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



Enter an item name

SonarQube

» 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.

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.

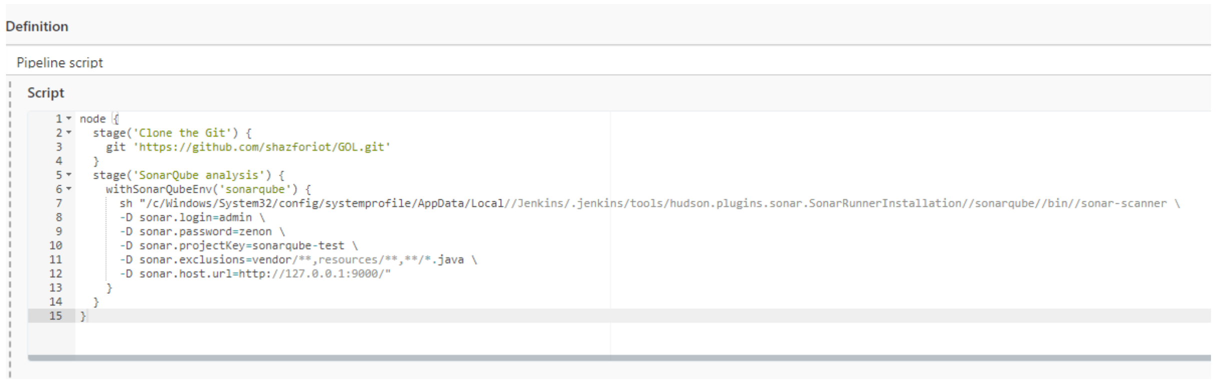
Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

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

Organization Folder
Creates a set of multibranch project subfolders by scanning for repositories.

7. 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 "<PATH_TO_SONARQUBE_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=http://127.0.0.1:9000/"
    }
  }
}
```



Definition

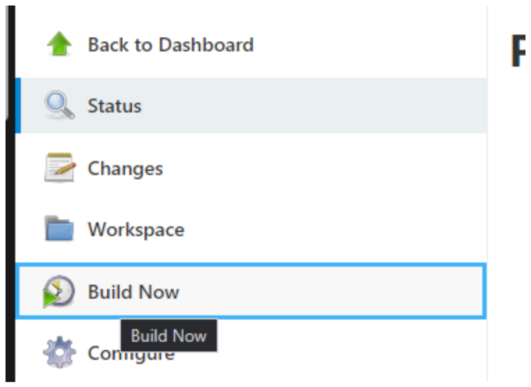
Pipeline script

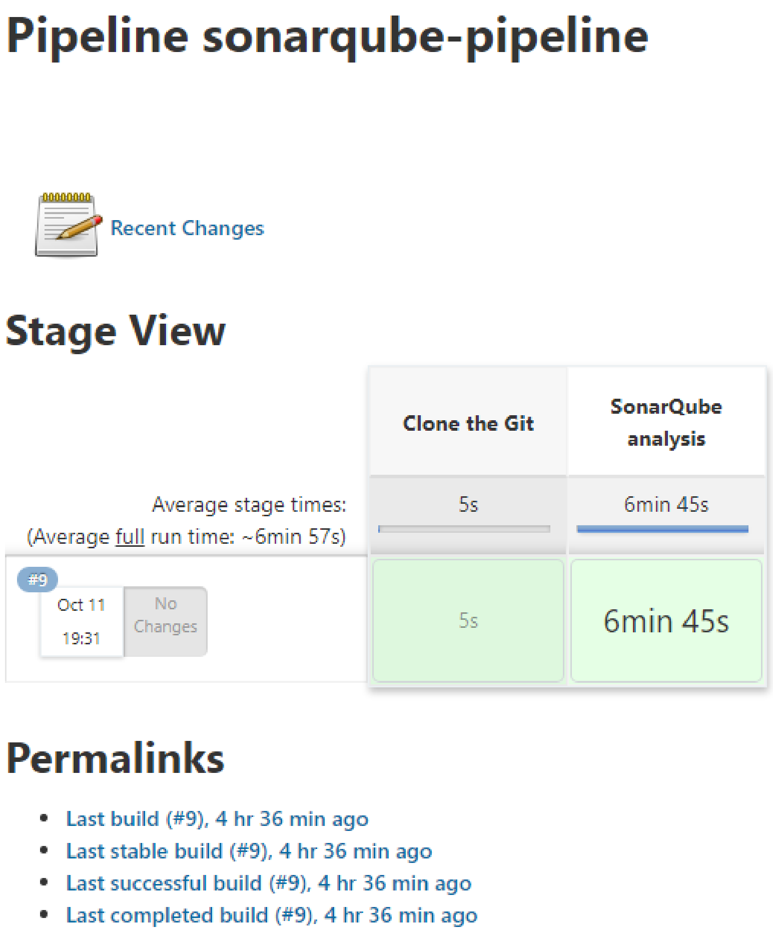
Script

```
node {
  stage('Clone the Git') {
    git 'https://github.com/shazforiot/GOL.git'
  }
  stage('SonarQube analysis') {
    withSonarQubeEnv('sonarqube') {
      sh "/c/Windows/System32/config/systemprofile/AppData/Local//Jenkins/.jenkins/tools/hudson.plugins.sonar.SonarRunnerInstallation//sonarqube//bin//sonar-scanner \
      -D sonar.login=admin \
      -D sonar.password=zenon \
      -D sonar.projectKey=sonarqube-test \
      -D sonar.exclusions=vendor/**,resources/**,**/*.java \
      -D sonar.host.url=http://127.0.0.1:9000/"
    }
  }
}
```

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

8. Run The Build.





Pipeline sonarqube-pipeline

Recent Changes

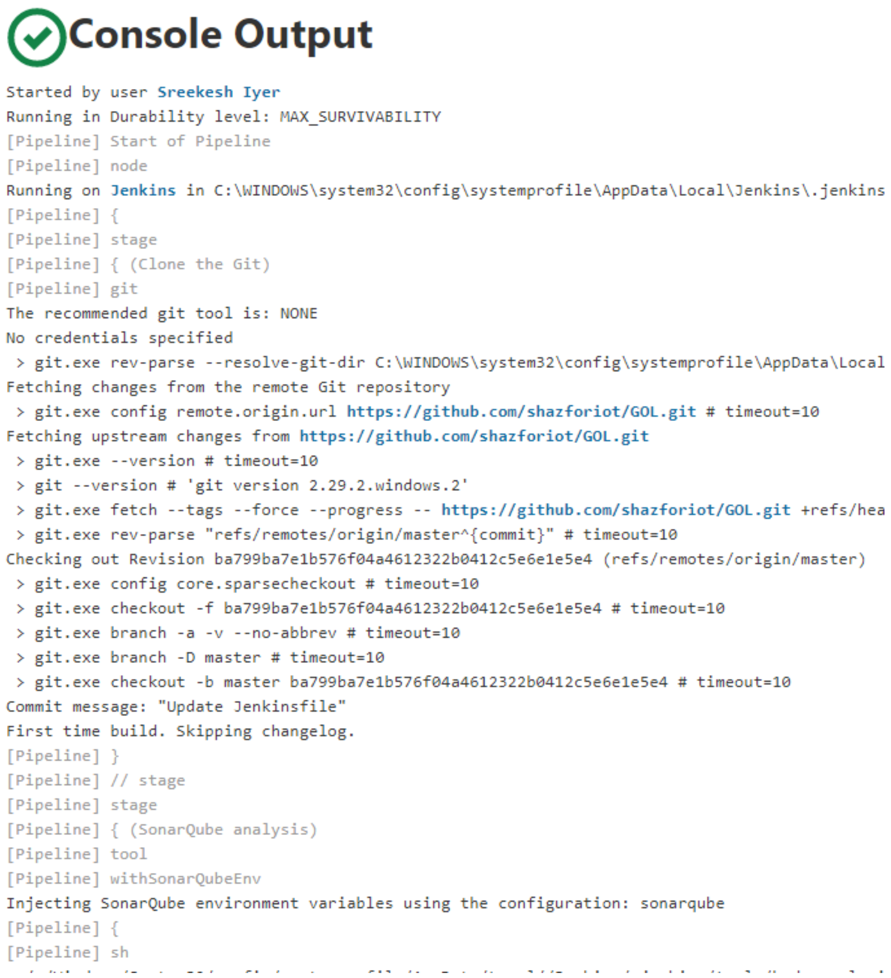
Stage View

	Clone the Git	SonarQube analysis
Average stage times: (Average full run time: ~6min 57s)	5s	6min 45s
#9 Oct 11 19:31 No Changes	5s	6min 45s

Permalinks

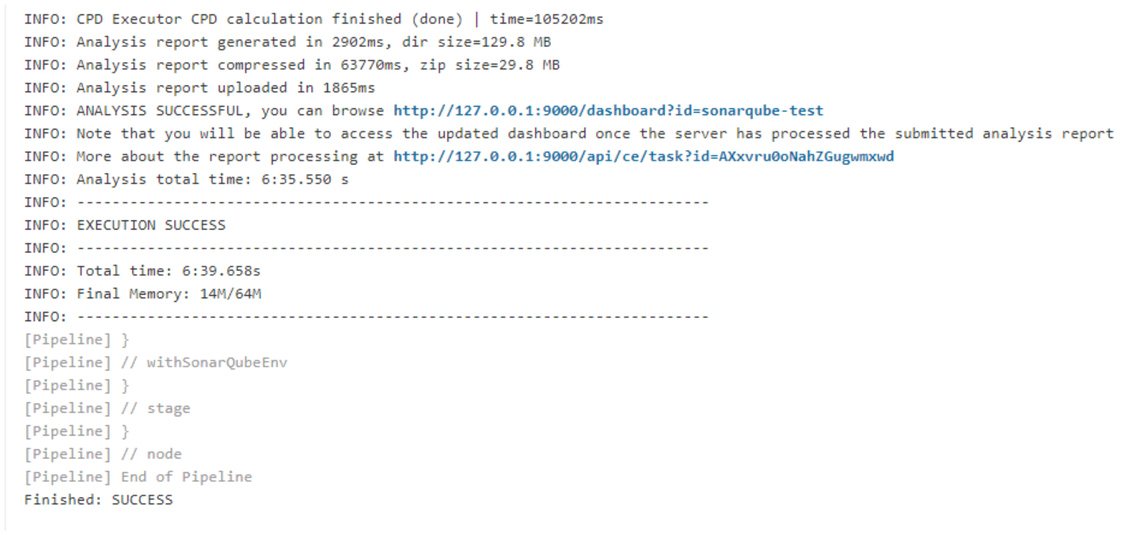
- Last build (#9), 4 hr 36 min ago
- Last stable build (#9), 4 hr 36 min ago
- Last successful build (#9), 4 hr 36 min ago
- Last completed build (#9), 4 hr 36 min ago

9. Check the console output once the build is complete.



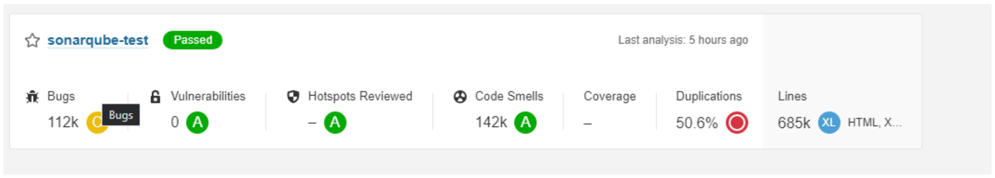
Console Output

```
Started by user Sreekesh Iyer
Running in Durability level: MAX_SURVIVABILITY
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in C:\WINDOWS\system32\config\systemprofile\AppData\Local\Jenkins\.jenkins
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Clone the Git)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
 > git.exe rev-parse --resolve-git-dir C:\WINDOWS\system32\config\systemprofile\AppData\Local
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 2.29.2.windows.2'
 > git.exe fetch --tags --force --progress -- https://github.com/shazforiot/GOL.git +refs/hea
 > 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
 > git.exe checkout -b master ba799ba7e1b576f04a4612322b0412c5e6e1e5e4 # timeout=10
Commit message: "Update Jenkinsfile"
First time build. Skipping changelog.
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (SonarQube analysis)
[Pipeline] tool
[Pipeline] withSonarQubeEnv
Injecting SonarQube environment variables using the configuration: sonarqube
[Pipeline] {
[Pipeline] sh
```



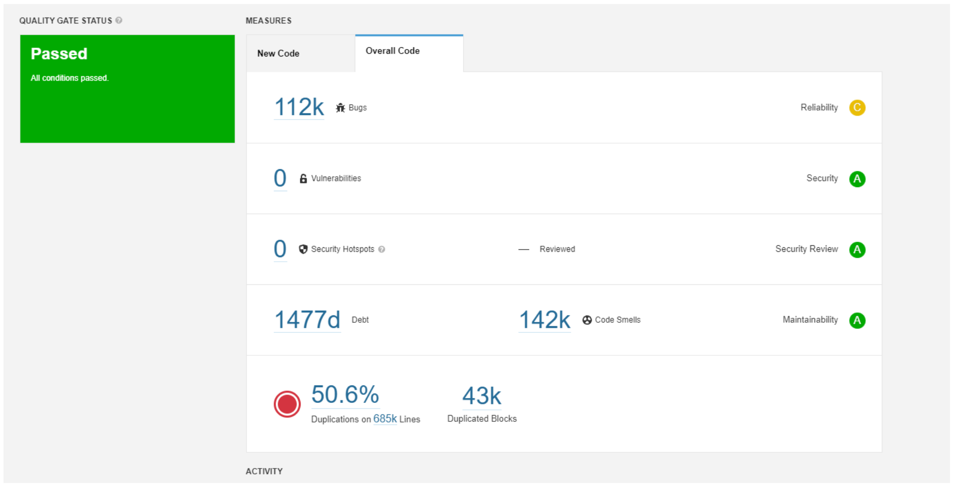
```
INFO: CPD Executor CPD calculation finished (done) | time=105202ms
INFO: Analysis report generated in 2902ms, dir size=129.8 MB
INFO: Analysis report compressed in 63770ms, zip size=29.8 MB
INFO: Analysis report uploaded in 1865ms
INFO: ANALYSIS SUCCESSFUL, you can browse http://127.0.0.1:9000/dashboard?id=sonarqube-test
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=AXxvru0oNahZGugwmxwd
INFO: Analysis total time: 6:35.550 s
INFO: ------------------------------------------------------------------------
INFO: EXECUTION SUCCESS
INFO: ------------------------------------------------------------------------
INFO: Total time: 6:39.658s
INFO: Final Memory: 14M/64M
INFO: ------------------------------------------------------------------------
[Pipeline] }
[Pipeline] // withSonarQubeEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

10. After that, check the project in SonarQube.



sonarqube-test Passed Last analysis: 5 hours ago

Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications	Lines
112k C	0 A	– A	142k A	–	50.6%	685k XL HTML, X...



QUALITY GATE STATUS

Passed
All conditions passed.

MEASURES

New Code | Overall Code

112k Bugs — Reliability C

0 Vulnerabilities — Security A

0 Security Hotspots — Reviewed — Security Review A

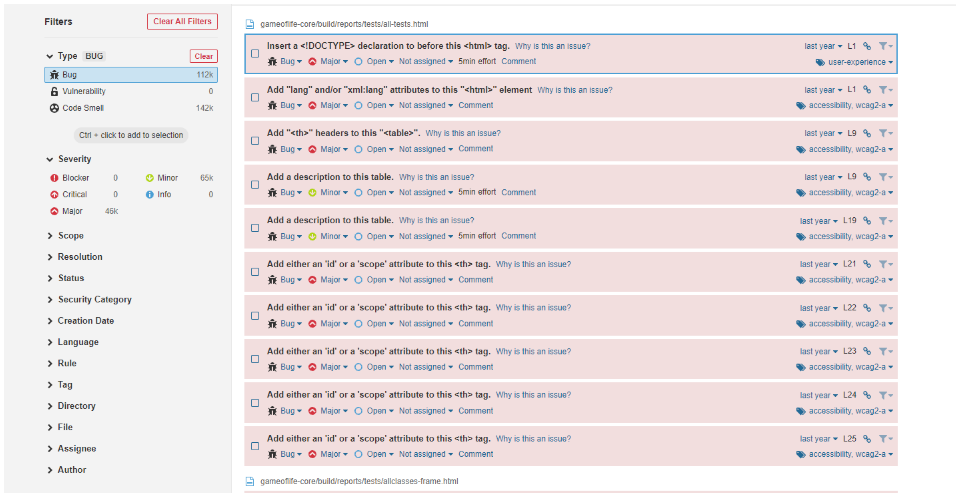
1477d Debt — 142k Code Smells — Maintainability A

50.6% Duplications on 685k Lines — 43k Duplicated Blocks

ACTIVITY

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

11. **Code Problems -**



Filters Clear All Filters

Type BUG Clear
- Bug 112k
- Vulnerability 0
- Code Smell 142k

Ctrl + click to add to selection

Severity
- Blocker 0
- Critical 0
- Major 46k
- Minor 65k
- Info 0

Scope
Resolution
Status
Security Category
Creation Date
Language
Rule
Tag
Directory
File
Assignee
Author

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

Insert a <!DOCTYPE> declaration to before this <html> tag. Why is this an issue? last year L1
Bug Major Open Not assigned 5min effort Comment user-experience

Add "lang" and/or "xml:lang" attributes to this "<html>" element Why is this an issue? last year L1
Bug Major Open Not assigned Comment accessibility, wcag2-a

Add "<th>" headers to this "<table>". Why is this an issue? last year L9
Bug Major Open Not assigned Comment accessibility, wcag2-a

Add a description to this table. Why is this an issue? last year L9
Bug Minor Open Not assigned 5min effort Comment accessibility, wcag2-a

Add a description to this table. Why is this an issue? last year L19
Bug Minor Open Not assigned 5min effort Comment accessibility, wcag2-a

Add either an 'id' or a 'scope' attribute to this <th> tag. Why is this an issue? last year L21
Bug Major Open Not assigned Comment accessibility, wcag2-a

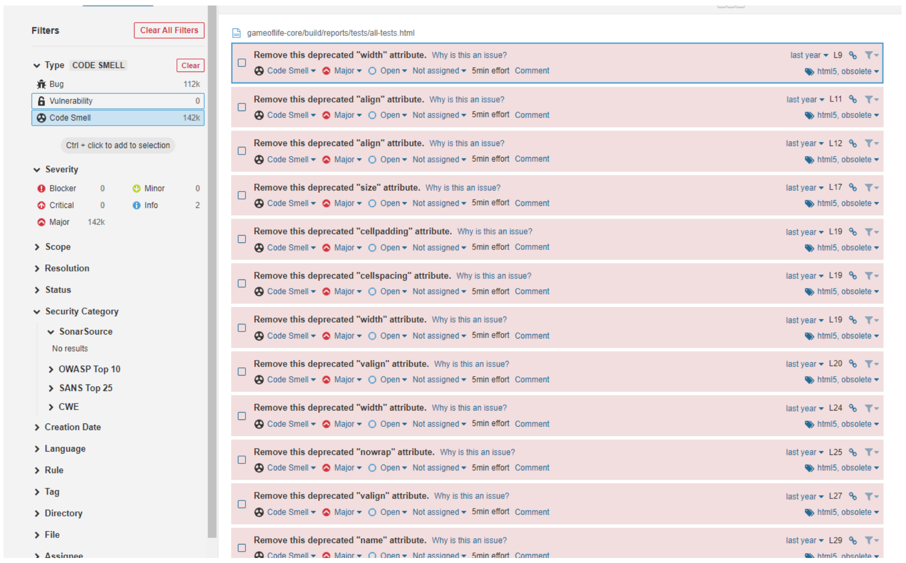
Add either an 'id' or a 'scope' attribute to this <th> tag. Why is this an issue? last year L22
Bug Major Open Not assigned Comment accessibility, wcag2-a

Add either an 'id' or a 'scope' attribute to this <th> tag. Why is this an issue? last year L23
Bug Major Open Not assigned Comment accessibility, wcag2-a

Add either an 'id' or a 'scope' attribute to this <th> tag. Why is this an issue? last year L24
Bug Major Open Not assigned Comment accessibility, wcag2-a

Add either an 'id' or a 'scope' attribute to this <th> tag. Why is this an issue? last year L25
Bug Major Open Not assigned Comment accessibility, wcag2-a

gameoflife-core/build/reports/tests/allclasses-frame.html



Filters Clear All Filters

Type CODE SMELL Clear
- Bug 112k
- Vulnerability 0
- Code Smell 142k

Ctrl + click to add to selection

Severity
- Blocker 0
- Critical 0
- Major 142k
- Minor 0
- Info 2

Scope
Resolution
Status
Security Category
- SonarSource
  - No results
- OWASP Top 10
- SANS Top 25
- CWE

Creation Date
Language
Rule
Tag
Directory
File
Assignee

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

Remove this deprecated "width" attribute. Why is this an issue? last year L9
Code Smell Major Open Not assigned 5min effort Comment html5, obsolete

Remove this deprecated "align" attribute. Why is this an issue? last year L11
Code Smell Major Open Not assigned 5min effort Comment html5, obsolete

Remove this deprecated "align" attribute. Why is this an issue? last year L12
Code Smell Major Open Not assigned 5min effort Comment html5, obsolete

Remove this deprecated "size" attribute. Why is this an issue? last year L17
Code Smell Major Open Not assigned 5min effort Comment html5, obsolete

Remove this deprecated "cellpadding" attribute. Why is this an issue? last year L19
Code Smell Major Open Not assigned 5min effort Comment html5, obsolete

Remove this deprecated "cellspacing" attribute. Why is this an issue? last year L19
Code Smell Major Open Not assigned 5min effort Comment html5, obsolete

Remove this deprecated "width" attribute. Why is this an issue? last year L19
Code Smell Major Open Not assigned 5min effort Comment html5, obsolete

Remove this deprecated "valign" attribute. Why is this an issue? last year L20
Code Smell Major Open Not assigned 5min effort Comment html5, obsolete

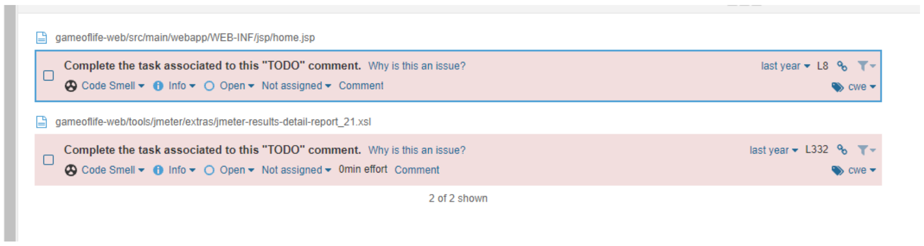
Remove this deprecated "width" attribute. Why is this an issue? last year L24
Code Smell Major Open Not assigned 5min effort Comment html5, obsolete

Remove this deprecated "nowrap" attribute. Why is this an issue? last year L25
Code Smell Major Open Not assigned 5min effort Comment html5, obsolete

Remove this deprecated "valign" attribute. Why is this an issue? last year L27
Code Smell Major Open Not assigned 5min effort Comment html5, obsolete

Remove this deprecated "name" attribute. Why is this an issue? last year L29
Code Smell Major Open Not assigned 5min effort Comment html5, obsolete

Bugs and Code Smells



gameoflife-web/src/main/webapp/WEB-INF/jsp/home.jsp

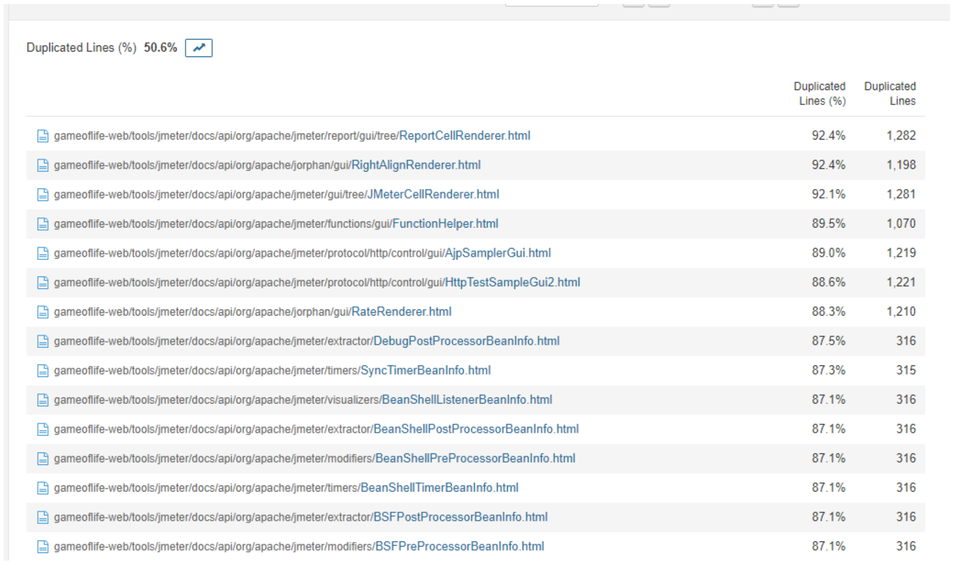
Complete the task associated to this "TODO" comment. Why is this an issue? last year L8
Code Smell Info Open Not assigned Comment cwe

gameoflife-web/tools/jmeter/extras/jmeter-results-detail-report_21.xsl

Complete the task associated to this "TODO" comment. Why is this an issue? last year L332
Code Smell Info Open Not assigned 0min effort Comment cwe

2 of 2 shown

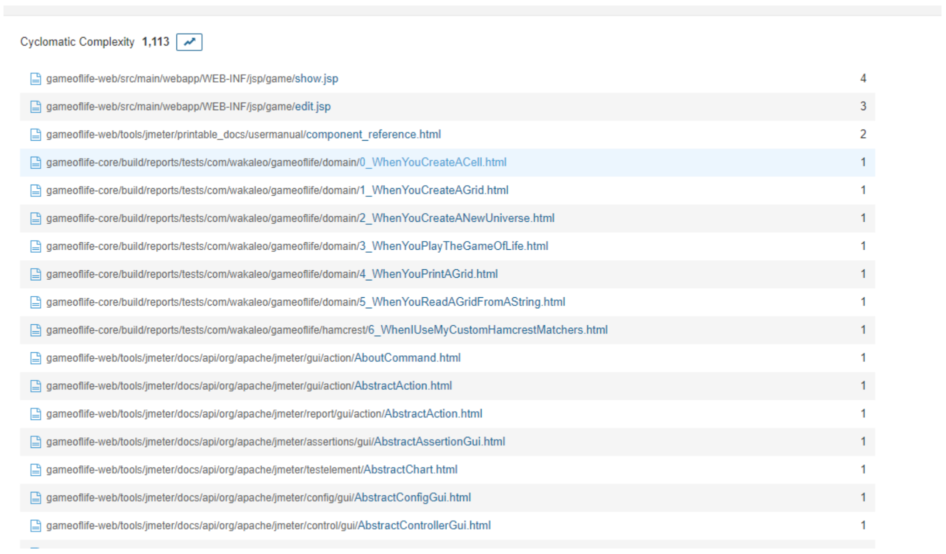
Unfinished TODOs



Duplicated Lines (%) 50.6%

	Duplicated Lines (%)	Duplicated Lines
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/report/gui/tree/ReportCellRenderer.html	92.4%	1,282
gameoflife-web/tools/jmeter/docs/api/org/apache/jorphan/gui/RightAlignRenderer.html	92.4%	1,198
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/tree/JMeterCellRenderer.html	92.1%	1,281
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/functions/gui/FunctionHelper.html	89.5%	1,070
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/control/gui/AjpSamplerGui.html	89.0%	1,219
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/control/gui/HttpTestSampleGui2.html	88.6%	1,221
gameoflife-web/tools/jmeter/docs/api/org/apache/jorphan/gui/RateRenderer.html	88.3%	1,210
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/extractor/DebugPostProcessorBeanInfo.html	87.5%	316
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/timers/SyncTimerBeanInfo.html	87.3%	315
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/BeanShellListenerBeanInfo.html	87.1%	316
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/extractor/BeanShellPostProcessorBeanInfo.html	87.1%	316
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/modifiers/BeanShellPreProcessorBeanInfo.html	87.1%	316
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/timers/BeanShellTimerBeanInfo.html	87.1%	316
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/extractor/BSFPostProcessorBeanInfo.html	87.1%	316
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/modifiers/BSFPreProcessorBeanInfo.html	87.1%	316

Duplicates



Cyclomatic Complexity 1,113

gameoflife-web/src/main/webapp/WEB-INF/jsp/game/show.jsp	4
gameoflife-web/src/main/webapp/WEB-INF/jsp/game/edit.jsp	3
gameoflife-web/tools/jmeter/printable_docs/usermanual/component_reference.html	2
gameoflife-core/build/reports/tests/com/wakaleo/gameoflife/domain/0_WhenYouCreateACell.html	1
gameoflife-core/build/reports/tests/com/wakaleo/gameoflife/domain/1_WhenYouCreateAGrid.html	1
gameoflife-core/build/reports/tests/com/wakaleo/gameoflife/domain/2_WhenYouCreateANewUniverse.html	1
gameoflife-core/build/reports/tests/com/wakaleo/gameoflife/domain/3_WhenYouPlayTheGameOfLife.html	1
gameoflife-core/build/reports/tests/com/wakaleo/gameoflife/domain/4_WhenYouPrintAGrid.html	1
gameoflife-core/build/reports/tests/com/wakaleo/gameoflife/domain/5_WhenYouReadAGridFromAString.html	1
gameoflife-core/build/reports/tests/com/wakaleo/gameoflife/hamcrest/6_WhenIUseMyCustomHamcrestMatchers.html	1
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/action/AboutCommand.html	1
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/action/AbstractAction.html	1
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/report/gui/action/AbstractAction.html	1
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/assertions/gui/AbstractAssertionGui.html	1
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/testelement/AbstractChart.html	1
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/config/gui/AbstractConfigGui.html	1
gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/control/gui/AbstractControllerGui.html	1

Cyclomatic Complexities

In this way, we have created a CI/CD Pipeline with Jenkins and integrated it with SonarQube to find issues in the code like bugs, code smells, duplicates, cyclomatic complexities, etc.

<u>Conclusion:</u>

In this experiment, we performed a static analysis of the code to detect bugs, code smells, and security vulnerabilities on our sample Java application.