DevOps

Lesson 04-Sonar(SonarQube)

Lesson Objectives

Introduction of Sonar

- Architecture
- Integration

Analyzing the Java code with Sonar

Integrating Jenkins with Sonar

Analyzing Maven, Java Code with Sonar

© 2017 Capgemini. All rights reserved.

5.1: introduction of Sonar
Sonar

Sonar is an open source platform used by development teams to manage source code quality. Sonar has been made with a main objective in mind: make code quality management accessible to everyone with minimal effort.

SonarQube (formerly known as *Sonar*) is an open source tool suite to measure and analyze the quality of source code. It is written in Java but is able to analyze code in about 20 different programming languages.

Code analysis may be started manually by executing a so-called sonar runner but SonarQube' s full potential is especially revealed when used in combination with continuous integration such as a Jenkins server.

© 2017 Capgemini. All rights reserved.

5.1: introduction of Sonar
Why Code Analyzer tool

Why we are using SonarQube(Code analyzer tool)

Code quality analysis helps to make your code:

- less error-prone
- more sustainable
- more reliable
- more readable
- more welcoming to new contributors

© 2017 Capgemini. All rights reserved.

5.1: introduction of Sonar
Features of Sonar

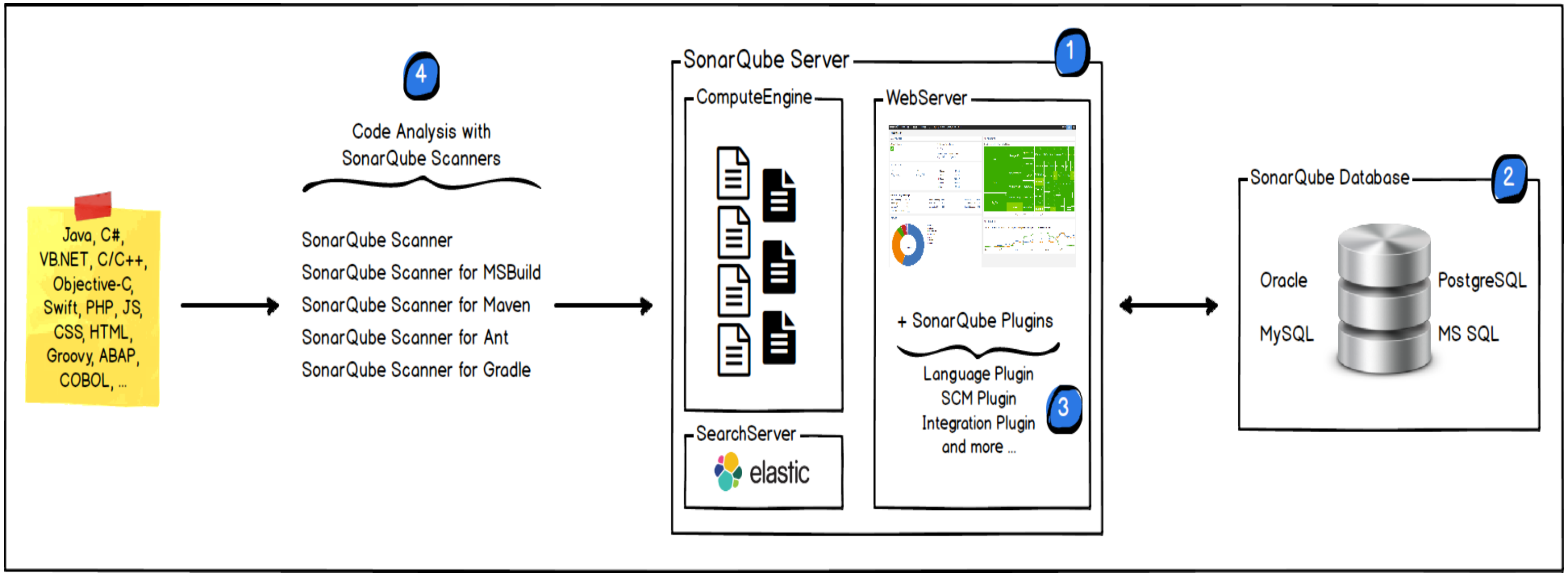
- Write clean Code
- DevOps Integration
- Centralize Quality
- Support 20+ languages

© 2017 Capgemini. All rights reserved.

5.1: introduction of Sonar
Sonar-Architecture

The SonarQube Platform is made of 4 components

- SonarQube Server
- SonarQube Database
- SonarQube Plugins
- SonarQube Scanners


Java, C#, VB.NET, C/C++, Objective-C, Swift, PHP, JS, CSS, HTML, Groovy, ABAP, COBOL, ...
4
Code Analysis with SonarQube Scanners
SonarQube Scanner
SonarQube Scanner for MSBuild
SonarQube Scanner for Maven
SonarQube Scanner for Ant
SonarQube Scanner for Gradle
SonarQube Server
1
ComputeEngine
WebServer
+ SonarQube Plugins
Language Plugin
SCM Plugin
Integration Plugin
and more ...
3
SearchServer
elastic
SonarQube Database
2
Oracle
MySQL
PostgreSQL
MS SQL

© 2017 Capgemini. All rights reserved.

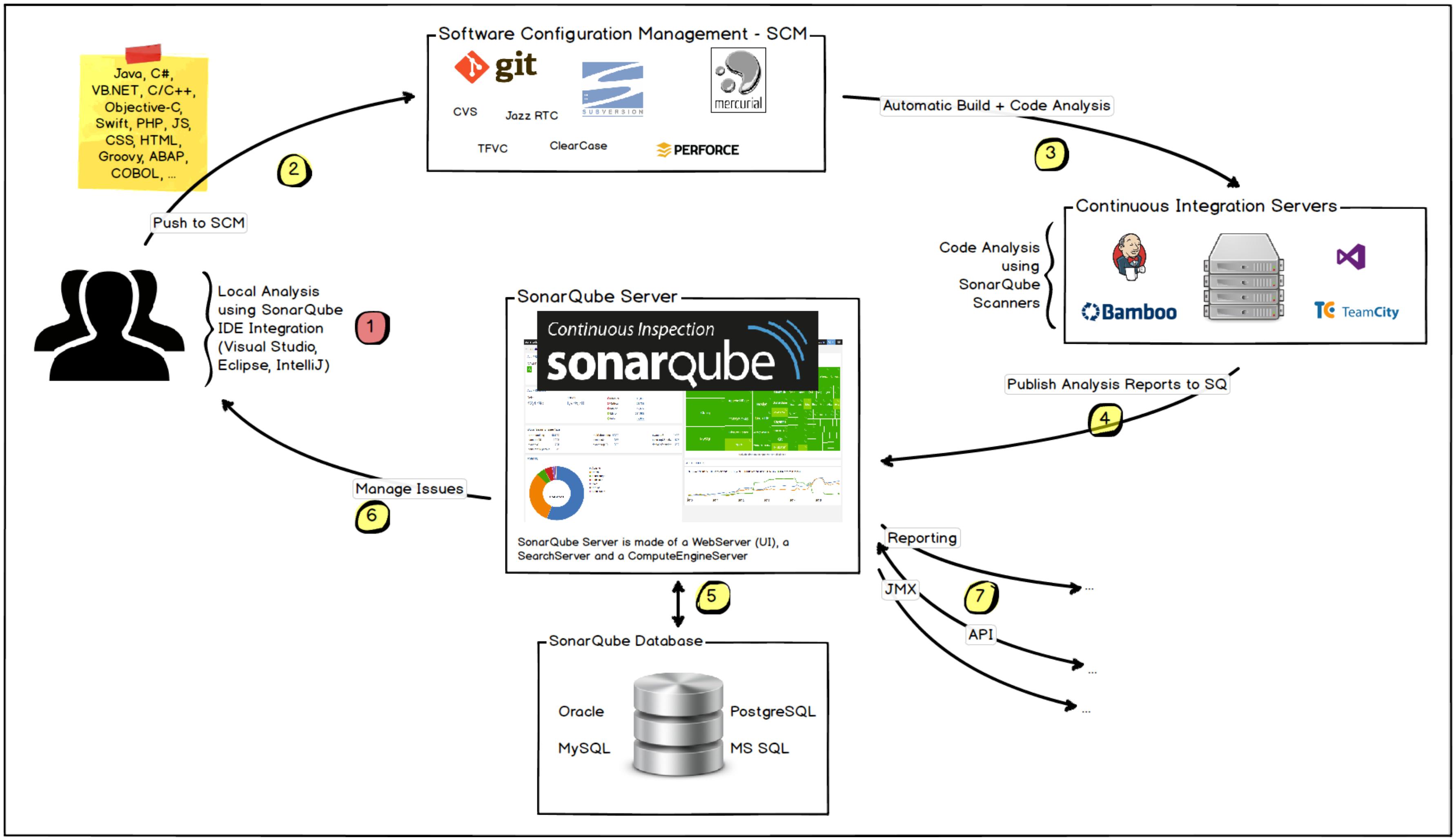
5.1: introduction of Sonar
Sonar-Integration

The following schema shows how SonarQube integrates with other ALM tools where the various components of SonarQube are used.

- Developers code in their IDEs and use SonarLint to run local analysis.
- Developers push their code into their favorite SCM : git, SVN, TFVC, ...
- The Continuous Integration Server triggers an automatic build, and the execution of the SonarQube Scanner required to run the SonarQube analysis.
- The analysis report is sent to the SonarQube Server for processing.
- SonarQube Server processes and stores the analysis report results in the SonarQube Database, and displays the results in the UI.
- Developers review, comment, challenge their issues to manage and reduce their Technical Debt through the SonarQube UI.
- Managers receive Reports from the analysis. Ops use APIs to automate configuration and extract data from SonarQube. Ops use JMX to monitor SonarQube Server.

© 2017 Capgemini. All rights reserved.

5.1: introduction of Sonar
Sonar-Integration


Java, C#, VB.NET, C/C++, Objective-C, Swift, PHP, JS, CSS, HTML, Groovy, ABAP, COBOL, ...
Push to SCM
2
Software Configuration Management - SCM
git
CVS
Jazz RTC
SUBVERSION
mercurial
TFVC
ClearCase
PERFORCE
Automatic Build + Code Analysis
3
Continuous Integration Servers
Code Analysis using SonarQube Scanners
Bamboo
TeamCity
Local Analysis using SonarQube IDE Integration (Visual Studio, Eclipse, IntelliJ)
1
SonarQube Server
Continuous Inspection
sonarqube
SonarQube Server is made of a WebServer (UI), a SearchServer and a ComputeEngineServer
Publish Analysis Reports to SQ
4
Manage Issues
6
5
SonarQube Database
Oracle
PostgreSQL
MySQL
MS SQL
Reporting
JMX
7
API
...
...
...

© 2017 Capgemini. All rights reserved.

5.1: introduction of Sonar
Sonar-Rules

254 rules written – identify atleast 10 rules

- "equals(Object obj)" and "hashCode()" should be overridden in pairs
- "final" classes should not have "protected" members
- "for" loop incrementers should modify the variable being tested in the loop's stop condition
- "Iterator.hasNext()" should not call "Iterator.next()"
- "Iterator.next()" methods should throw "NoSuchElementException"
- "main" should not "throw" anything
- "NullPointerException" should not be caught
- "entrySet()" should be iterated when both the key and value are needed

We can see all the rules in Sonar Dashboard

© 2017 Capgemini. All rights reserved.

5.1: introduction of Sonar
Sonar Installation

Sonar is easy to install & use .

Download Sonar -Sonarqube-x.xx & Sonar-scanner-x.xx:

- https://www.sonarqube.org/downloads/

Sonar can be installed in different ways:

- As a standalone application
- Windows Service

For starting sonar server use -StartSonar.bat

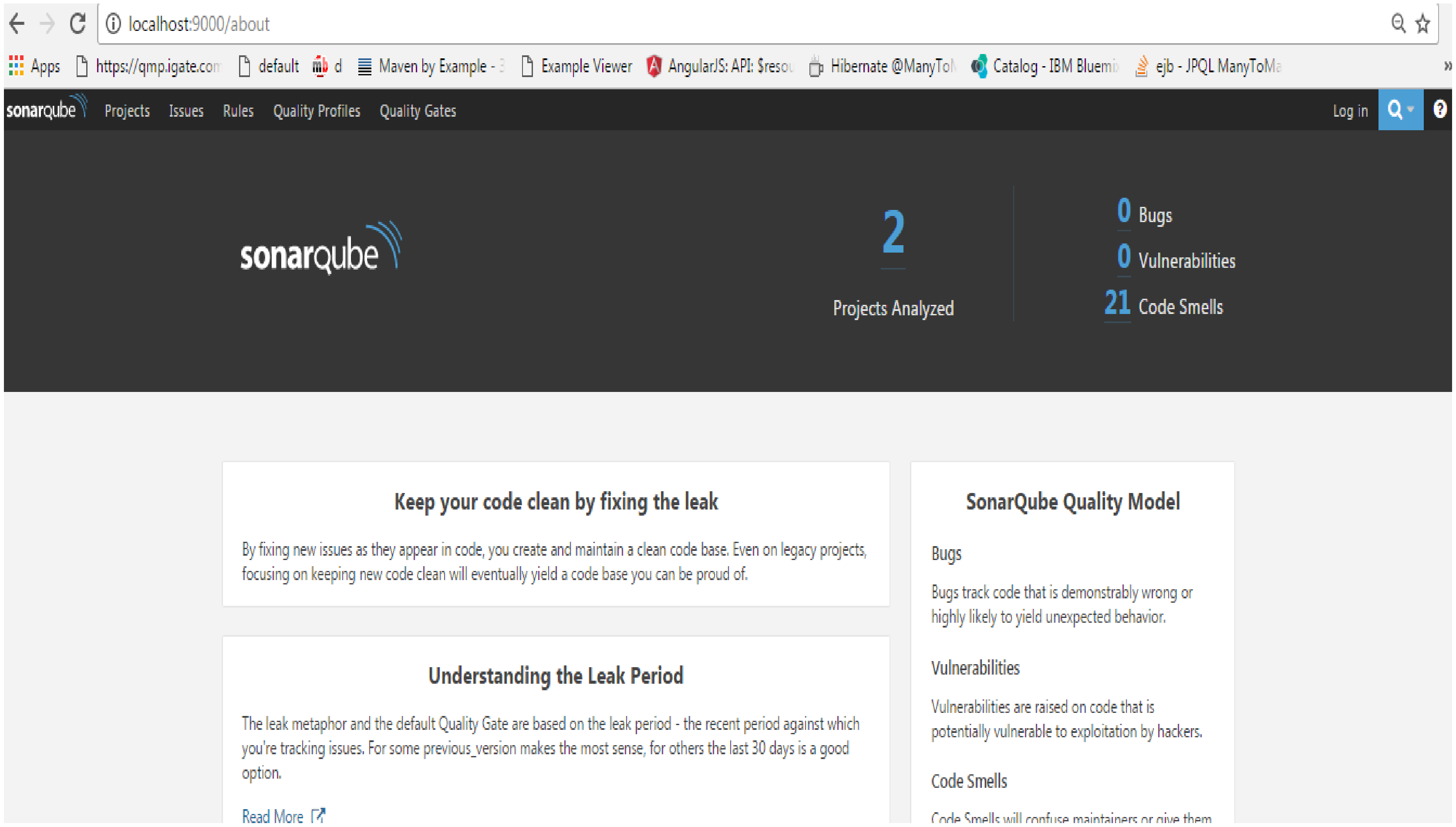
For stopping sonar server use – StopSonar.bat

Once sonar is started, the sonar dash board can be accessed by giving the following link in the browser

http://localhost:9000/

© 2017 Capgemini. All rights reserved.

5.1: introduction of Sonar
Sonar Installation



sonarqube

2 Projects Analyzed

0 Bugs
0 Vulnerabilities
21 Code Smells

Keep your code clean by fixing the leak

By fixing new issues as they appear in code, you create and maintain a clean code base. Even on legacy projects, focusing on keeping new code clean will eventually yield a code base you can be proud of.

Understanding the Leak Period

The leak metaphor and the default Quality Gate are based on the leak period - the recent period against which you're tracking issues. For some previous_version makes the most sense, for others the last 30 days is a good option.

Read More

SonarQube Quality Model

Bugs

Bugs track code that is demonstrably wrong or highly likely to yield unexpected behavior.

Vulnerabilities

Vulnerabilities are raised on code that is potentially vulnerable to exploitation by hackers.

Code Smells

Code Smells will confuse maintainers or give them

© 2017 Capgemini. All rights reserved.

5.2: Analyzing Java code with Sonar
Analyzing Java with Sonar

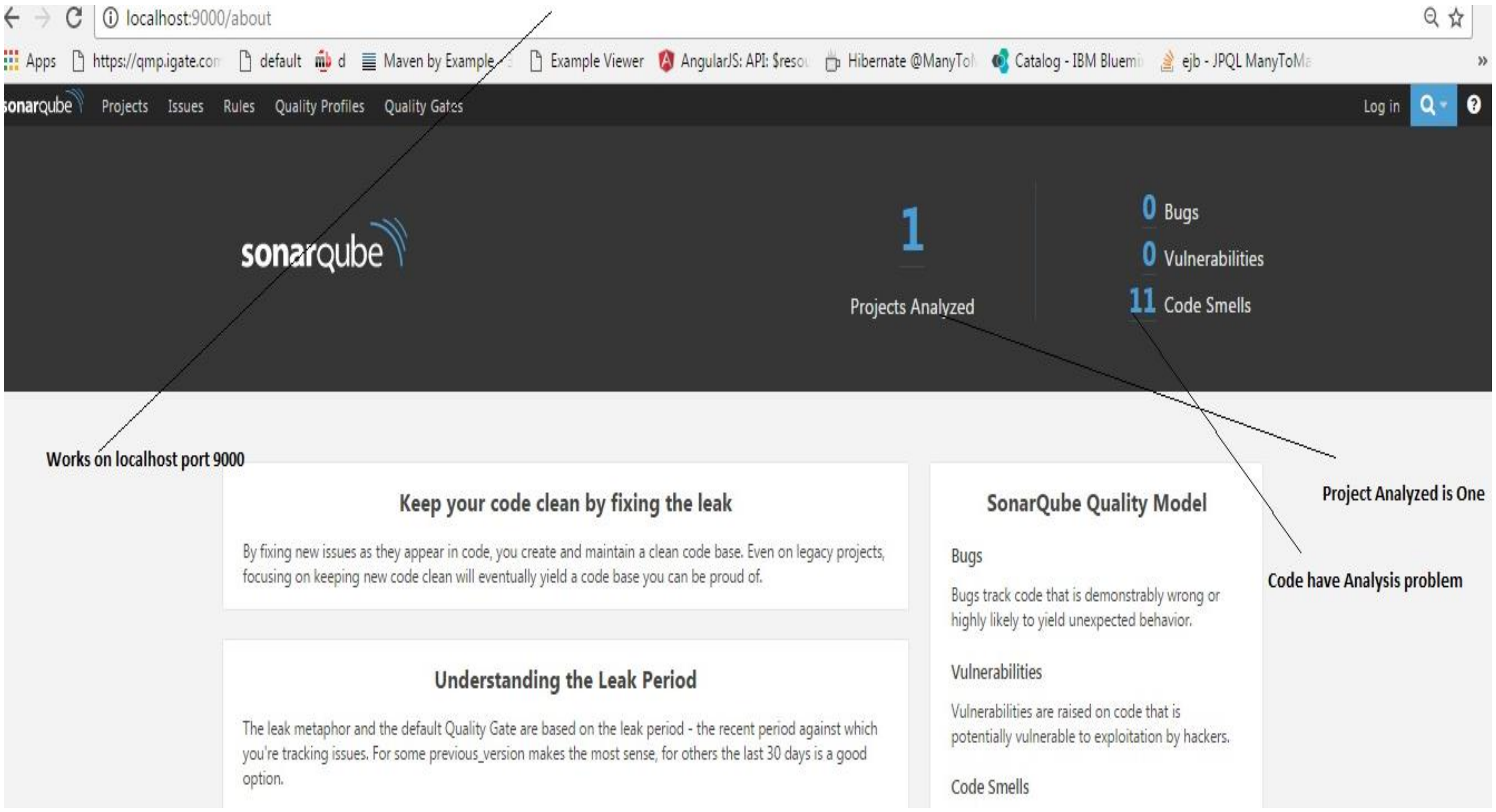
Integrating Java program with SonarQube

- Create a Java Project
- Add description of your project in sonar-scanner-x.xx->conf ->sonar-scanner. Properties
  - sonar.projectKey=JavaProject
  - sonar.projectName=JavaProject
  - sonar.projectVersion=1.0
  - sonar.sources=C:/DevOps/Training/JavaProject/src/com/cg/sonardemo
- Run Sonar server by using command **StartSonar.bat**
- Go to project folder & run command **sonar-scanner.bat**
- Open http://localhost:9000/ & we can see code is analyzing

© 2017 Capgemini. All rights reserved.

5.2: Analyzing Java code with Sonar
Analyzing Java with Sonar

First Project run on http://localhost:9000



Keep your code clean by fixing the leak

By fixing new issues as they appear in code, you create and maintain a clean code base. Even on legacy projects, focusing on keeping new code clean will eventually yield a code base you can be proud of.

Understanding the Leak Period

The leak metaphor and the default Quality Gate are based on the leak period - the recent period against which you're tracking issues. For some previous_version makes the most sense, for others the last 30 days is a good option.

SonarQube Quality Model

Bugs

Bugs track code that is demonstrably wrong or highly likely to yield unexpected behavior.

Vulnerabilities

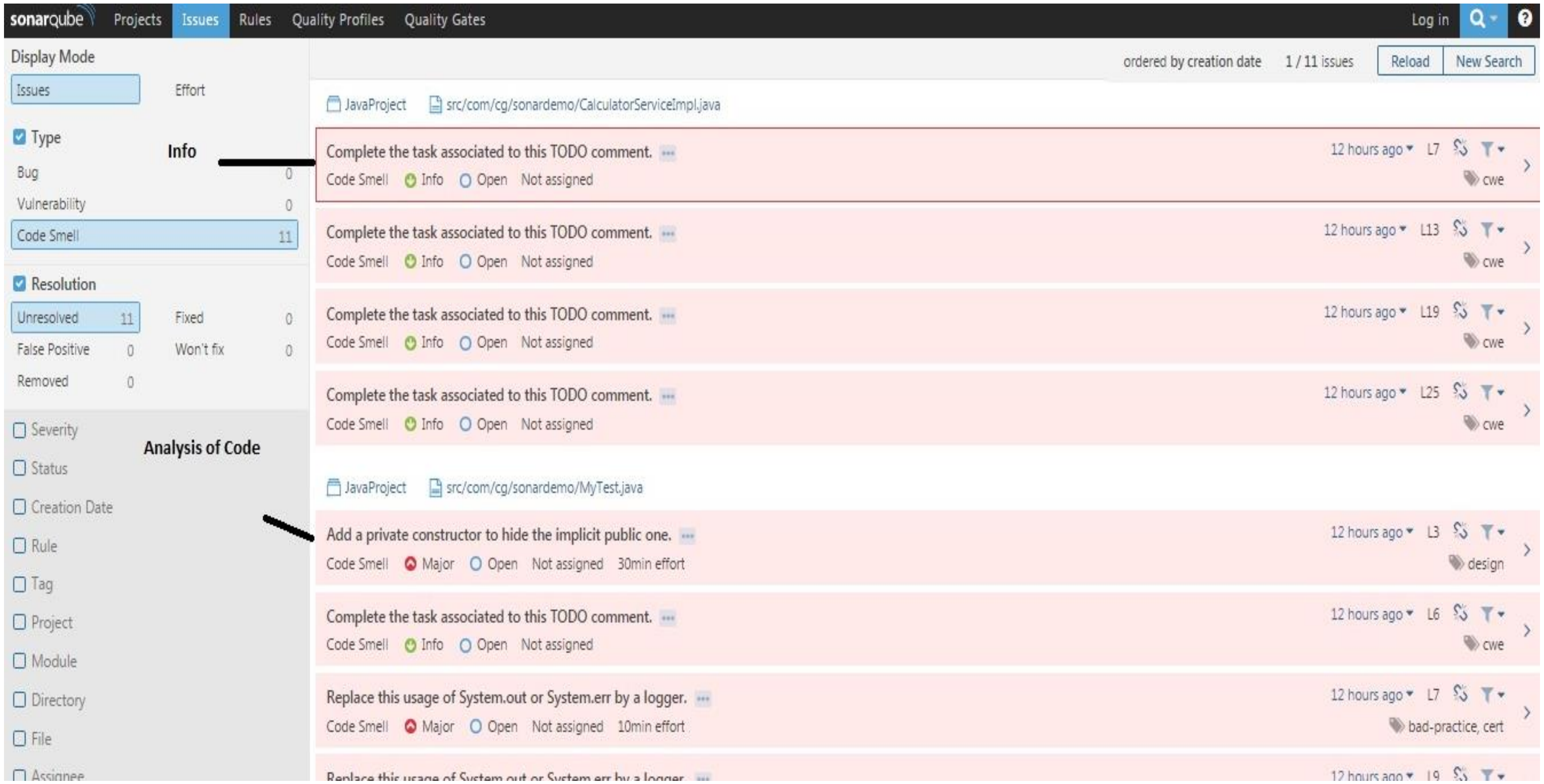
Vulnerabilities are raised on code that is potentially vulnerable to exploitation by hackers.

Code Smells

© 2017 Capgemini. All rights reserved.

5.2: Analyzing Java code with Sonar
Analyzing Java with Sonar

Select code smell after log in ,you will get all kind of major and minor problems



ordered by creation date 1 / 11 issues

JavaProject src/com/cg/sonardemo/CalculatorServiceImpl.java

Issue	Details	Age	Line	Tags
Complete the task associated to this TODO comment.	Code Smell · Info · Open · Not assigned	12 hours ago	L7	cwe
Complete the task associated to this TODO comment.	Code Smell · Info · Open · Not assigned	12 hours ago	L13	cwe
Complete the task associated to this TODO comment.	Code Smell · Info · Open · Not assigned	12 hours ago	L19	cwe
Complete the task associated to this TODO comment.	Code Smell · Info · Open · Not assigned	12 hours ago	L25	cwe

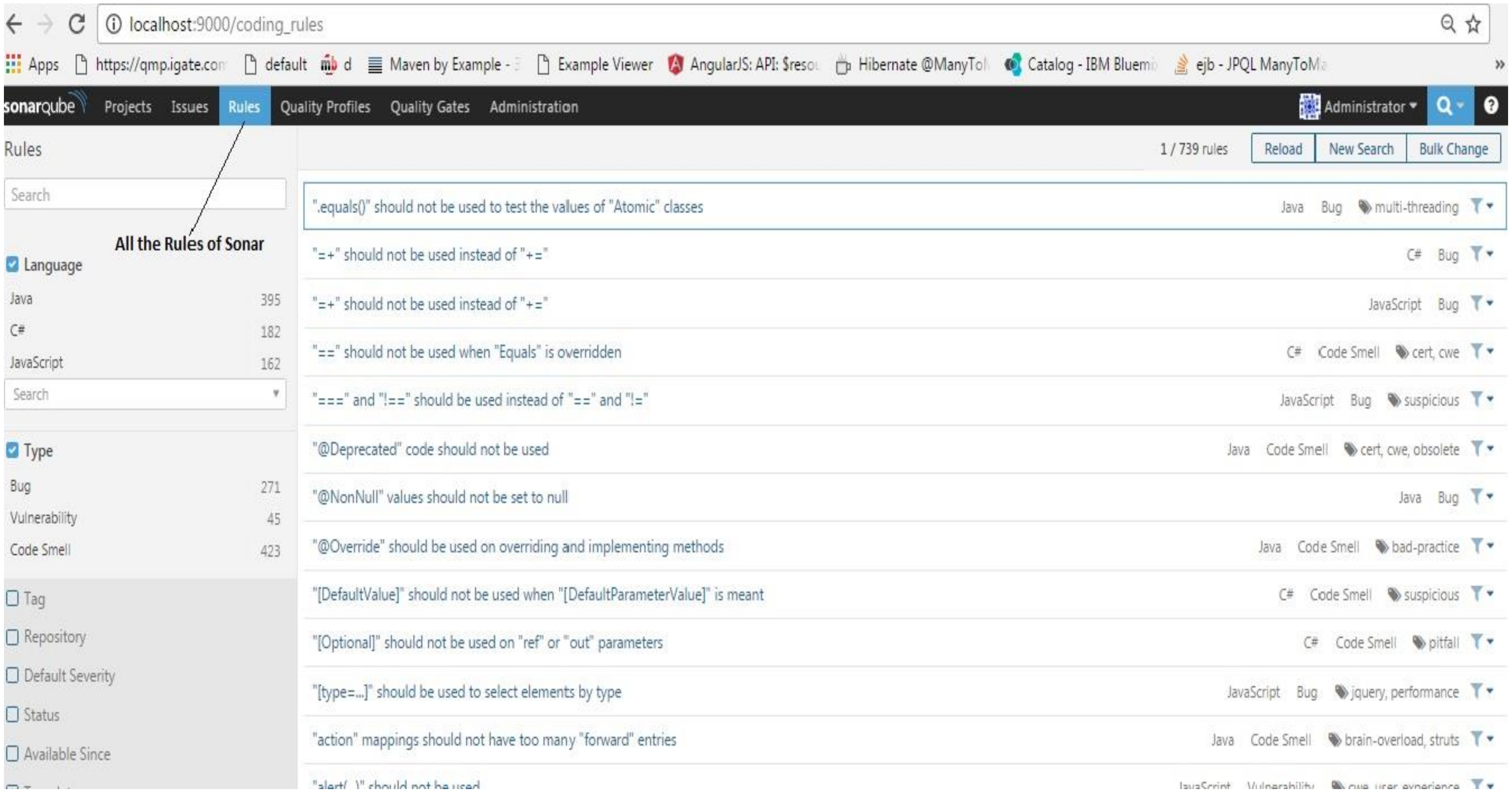
JavaProject src/com/cg/sonardemo/MyTest.java

Issue	Details	Age	Line	Tags
Add a private constructor to hide the implicit public one.	Code Smell · Major · Open · Not assigned · 30min effort	12 hours ago	L3	design
Complete the task associated to this TODO comment.	Code Smell · Info · Open · Not assigned	12 hours ago	L6	cwe
Replace this usage of System.out or System.err by a logger.	Code Smell · Major · Open · Not assigned · 10min effort	12 hours ago	L7	bad-practice, cert
Replace this usage of System.out or System.err by a logger.		12 hours ago	L9	

© 2017 Capgemini. All rights reserved.

5.2: Analyzing Java code with Sonar
Analyzing Java with Sonar

Rules to analyze Java Code



Language	
Java	395
C#	182
JavaScript	162
Type	
Bug	271
Vulnerability	45
Code Smell	423

Tag
Repository
Default Severity
Status
Available Since

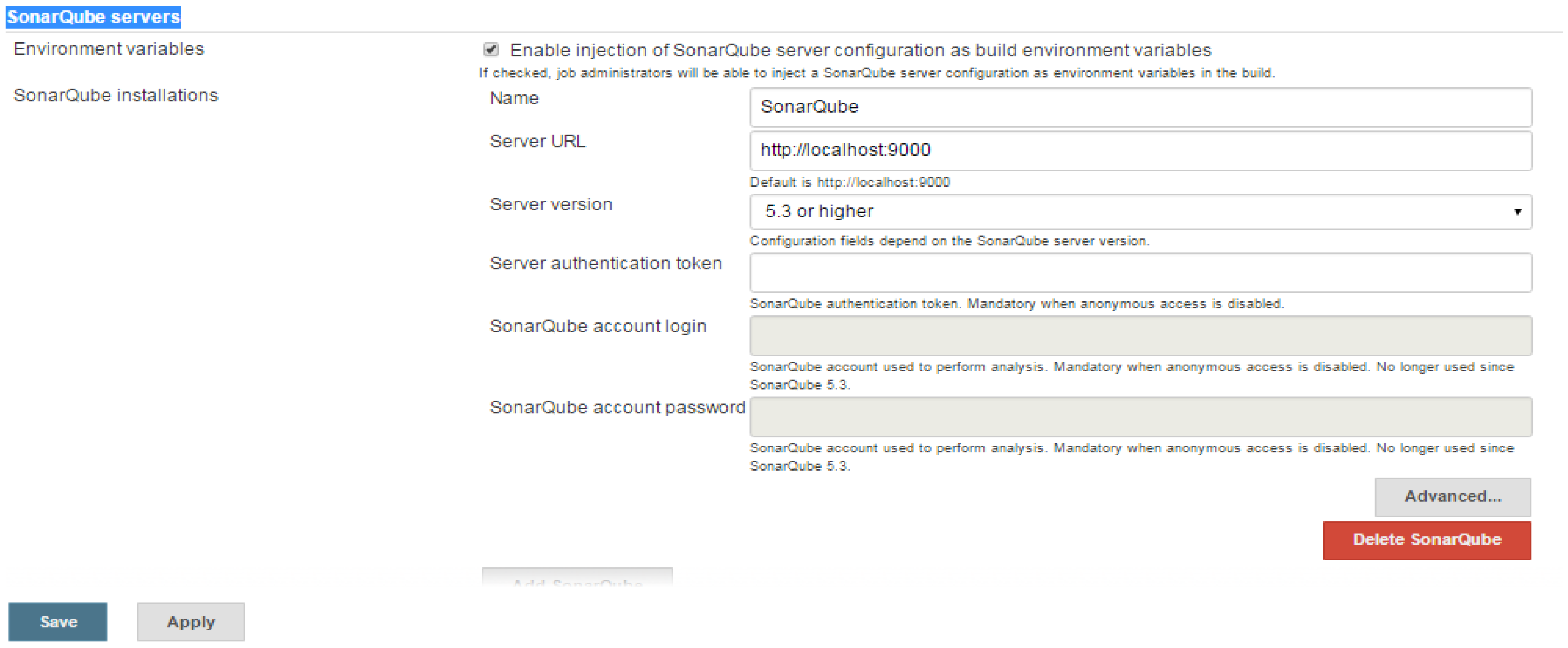
Rule	Language	Type	Tags
".equals()" should not be used to test the values of "Atomic" classes	Java	Bug	multi-threading
"=+" should not be used instead of "+="	C#	Bug	
"=+" should not be used instead of "+="	JavaScript	Bug	
"==" should not be used when "Equals" is overridden	C#	Code Smell	cert, cwe
"===" and "!==" should be used instead of "==" and "!="	JavaScript	Bug	suspicious
"@Deprecated" code should not be used	Java	Code Smell	cert, cwe, obsolete
"@NonNull" values should not be set to null	Java	Bug	
"@Override" should be used on overriding and implementing methods	Java	Code Smell	bad-practice
"[DefaultValue]" should not be used when "[DefaultParameterValue]" is meant	C#	Code Smell	suspicious
"[Optional]" should not be used on "ref" or "out" parameters	C#	Code Smell	pitfall
"[type=...]" should be used to select elements by type	JavaScript	Bug	jquery, performance
"action" mappings should not have too many "forward" entries	Java	Code Smell	brain-overload, struts

© 2017 Capgemini. All rights reserved.

5.3: Integrating Jenkin with Sonar
Sonar Jenkin Integration

Download SonarQube Plugin in Jenkins

Go to Manage Jenkin->Configure System->Go to SonarQube servers-> check on Enable injection of SonarQube->add Server name & server URL



SonarQube servers

Environment variables	✓ Enable injection of SonarQube server configuration as build environment variables If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.
SonarQube installations	
Name	SonarQube
Server URL	http://localhost:9000 Default is http://localhost:9000
Server version	5.3 or higher Configuration fields depend on the SonarQube server version.
Server authentication token	 SonarQube authentication token. Mandatory when anonymous access is disabled.
SonarQube account login	 SonarQube account used to perform analysis. Mandatory when anonymous access is disabled. No longer used since SonarQube 5.3.
SonarQube account password	 SonarQube account used to perform analysis. Mandatory when anonymous access is disabled. No longer used since SonarQube 5.3.

© 2017 Capgemini. All rights reserved.

5.4: Analyzing Maven code ,Jenkin with Sonar Sonar,Maven,Git & Jenkins Integration

Create New item->Enter item name->Select Maven Project->Ok

Give Git Repository link, in build environment check prepare sonarqube scanner environment

Give path of pom.xml of your project & then select post build action as sonarqube analysis with maven

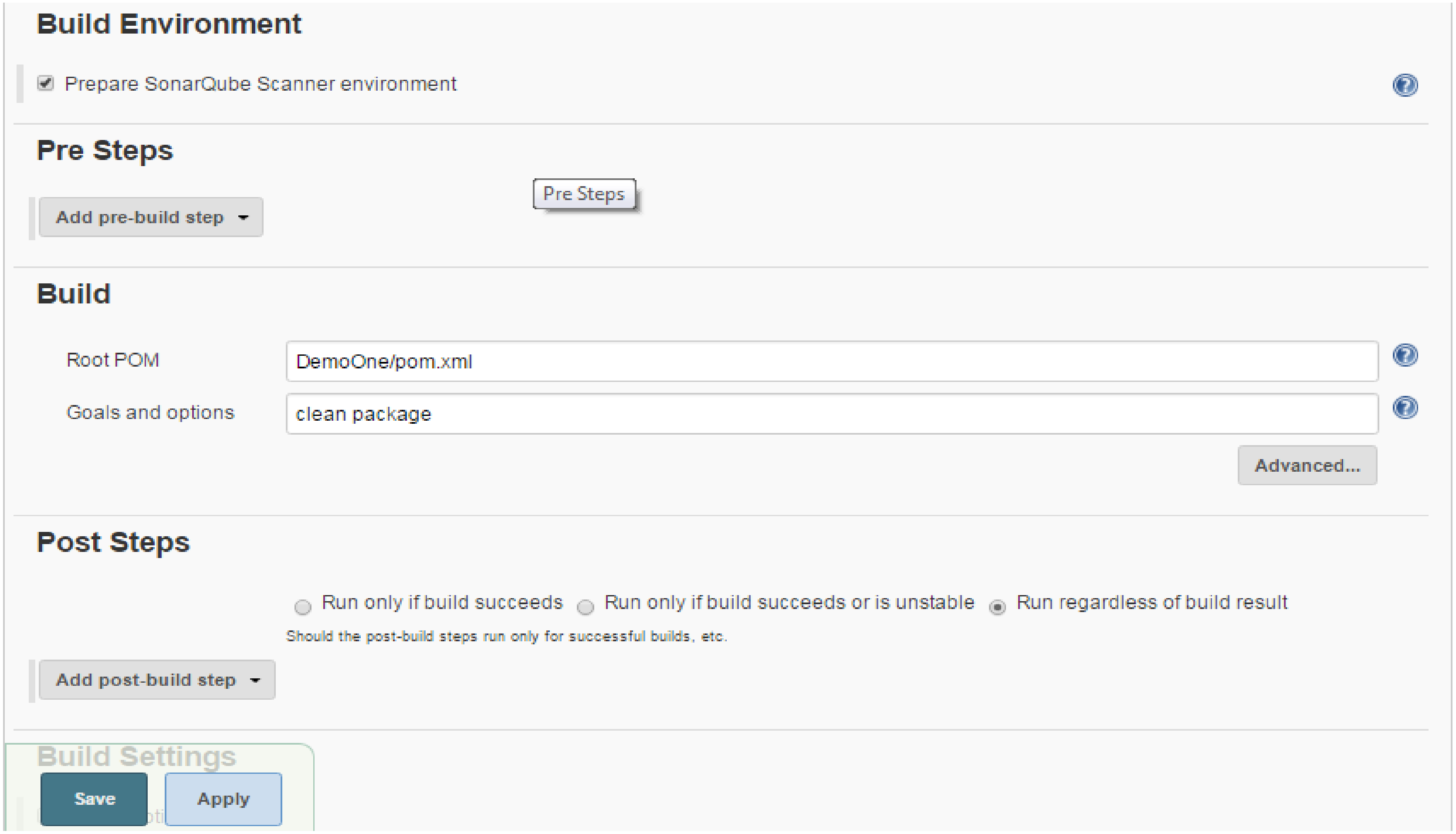
Then apply & Build now

We can see in console output build success and failure

Analyze in SonarQube

© 2017 Capgemini. All rights reserved.

5.4: Analyzing Maven code ,Jenkin with Sonar Sonar,Maven,Git & Jenkins Integration



Build Environment

[✓] Prepare SonarQube Scanner environment

Pre Steps

Build

Root POM: DemoOne/pom.xml

Goals and options: clean package

Post Steps

Run only if build succeeds   Run only if build succeeds or is unstable   [✓ Run regardless of build result]

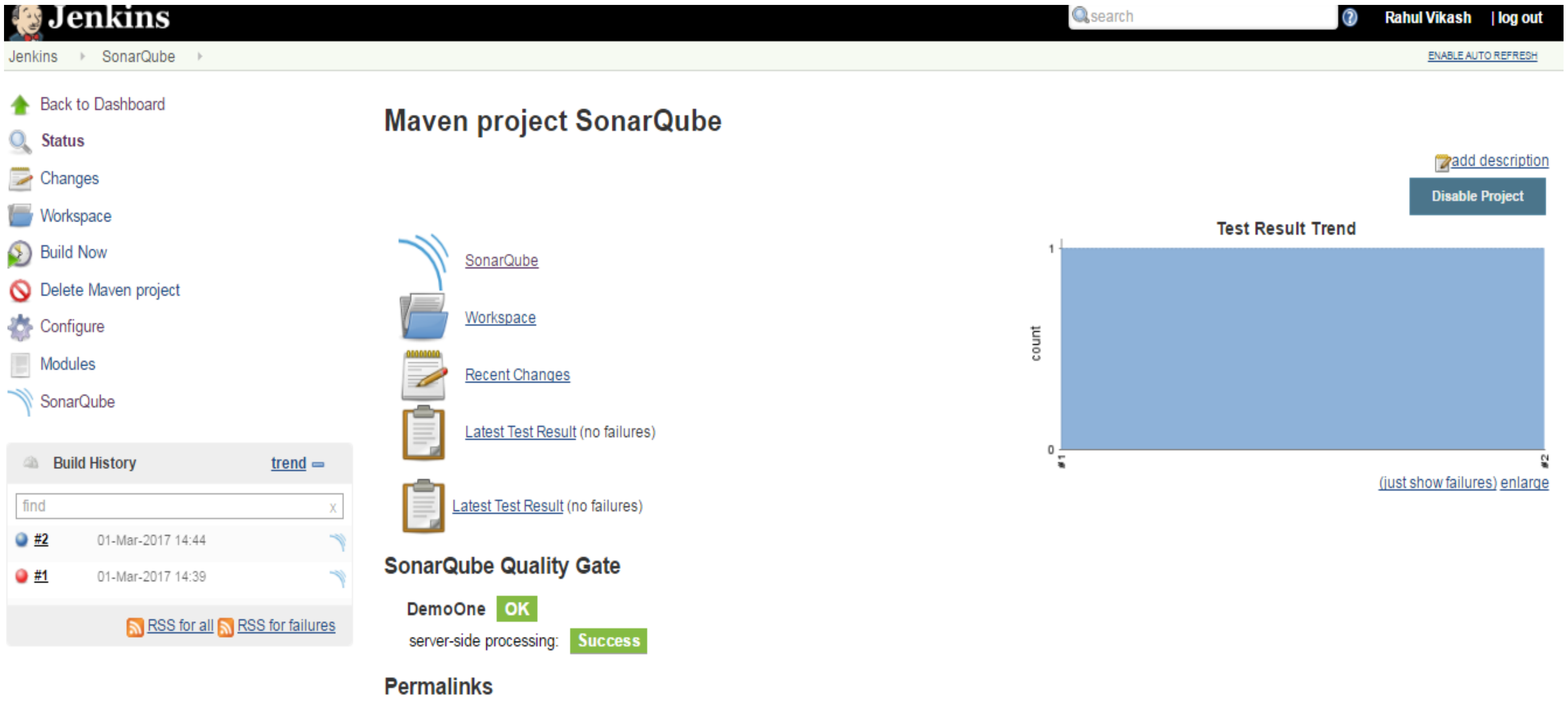
Should the post-build steps run only for successful builds, etc.

© 2017 Capgemini. All rights reserved.

5.4: Analyzing Maven code ,Jenkin with Sonar Sonar,Maven,Git & Jenkins Integration

After successful completion, sonarqube analysis can be checked.

- Click on sonarqube or Open http://localhost:9000/ & code analyzing is seen.


Jenkins | search | Rahul Vikash | log out

Jenkins › SonarQube › ENABLE AUTO REFRESH

- Back to Dashboard
- Status
- Changes
- Workspace
- Build Now
- Delete Maven project
- Configure
- Modules
- SonarQube

Build History trend

find

#2	01-Mar-2017 14:44
#1	01-Mar-2017 14:39

RSS for all RSS for failures

Maven project SonarQube

add description

Disable Project

- SonarQube
- Workspace
- Recent Changes
- Latest Test Result (no failures)
- Latest Test Result (no failures)

SonarQube Quality Gate

DemoOne OK

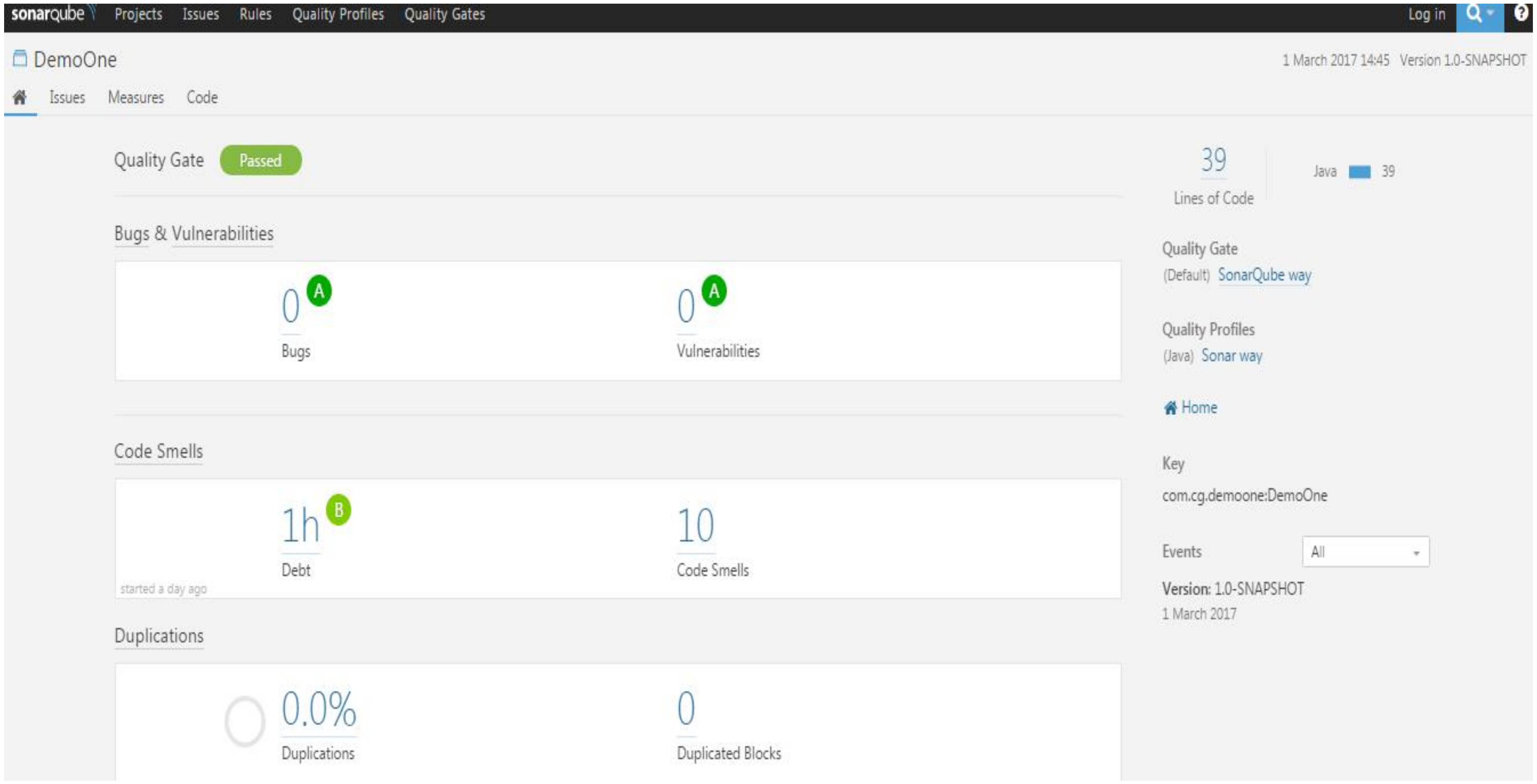
server-side processing: Success

Permalinks

© 2017 Capgemini. All rights reserved.

5.4: Analyzing Maven code ,Jenkin with Sonar Sonar,Maven,Git & Jenkins Integration

Clicking on SonarQube & analyzing the code



DemoOne

1 March 2017 14:45 Version 1.0-SNAPSHOT

Quality Gate Passed

Bugs & Vulnerabilities

0 A Bugs

0 A Vulnerabilities

Code Smells

1h B Debt

10 Code Smells

started a day ago

Duplications

0.0% Duplications

0 Duplicated Blocks

39 Lines of Code

Java 39

Quality Gate
(Default) SonarQube way

Quality Profiles
(Java) Sonar way

Home

Key
com.cg.demoone:DemoOne

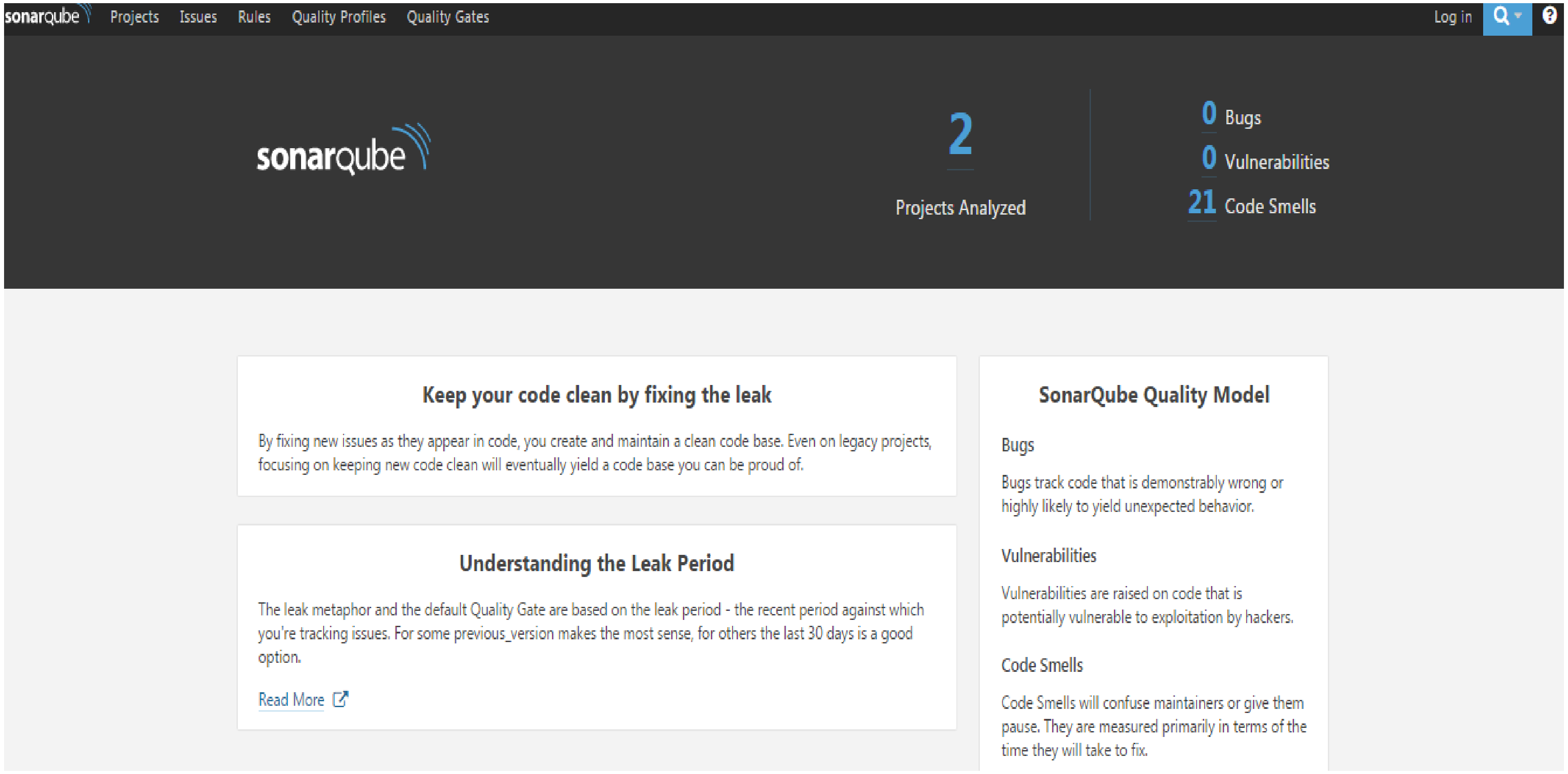
Events All

Version: 1.0-SNAPSHOT
1 March 2017

© 2017 Capgemini. All rights reserved.

5.4: Analyzing Maven code ,Jenkin with Sonar Sonar,Maven,Git & Jenkins Integration

Open http://localhost:9000/



Keep your code clean by fixing the leak

By fixing new issues as they appear in code, you create and maintain a clean code base. Even on legacy projects, focusing on keeping new code clean will eventually yield a code base you can be proud of.

Understanding the Leak Period

The leak metaphor and the default Quality Gate are based on the leak period - the recent period against which you're tracking issues. For some previous_version makes the most sense, for others the last 30 days is a good option.

Read More

SonarQube Quality Model

Bugs

Bugs track code that is demonstrably wrong or highly likely to yield unexpected behavior.

Vulnerabilities

Vulnerabilities are raised on code that is potentially vulnerable to exploitation by hackers.

Code Smells

Code Smells will confuse maintainers or give them pause. They are measured primarily in terms of the time they will take to fix.

© 2017 Capgemini. All rights reserved.

Demo

Analyze Java code with Sonar
Jenkins Maven Git integration & analyzing with sonar

© 2017 Capgemini. All rights reserved.

Lab

Lab 03

© 2017 Capgemini. All rights reserved.

Summary

Sonar is an open source platform used by development teams to manage source code quality. Sonar has been developed with a main objective in mind: make code quality management accessible to everyone with minimal effort.

Working with code analyzing tool with Maven Jenkins, Git

© 2017 Capgemini. All rights reserved.

Review Question

SonarQube platform is made of components, choose the correct one

- Database
- plugins
- Server
- All of above

________ plugin needs to be downloaded for Jenkins and sonar integration.

- ___________________ command is used to run Sonar software.

© 2017 Capgemini. All rights reserved.