SonaQube(previously known as Sonar) is an open source platform for Continuous Inspection of code quality. It is written in java and supported for 25+ languages such as Java, C/C++, C#, PHP, Flex, Groovy, JavaScript, Python, PL/SQL, COBOL etc it is also used for Android Development.

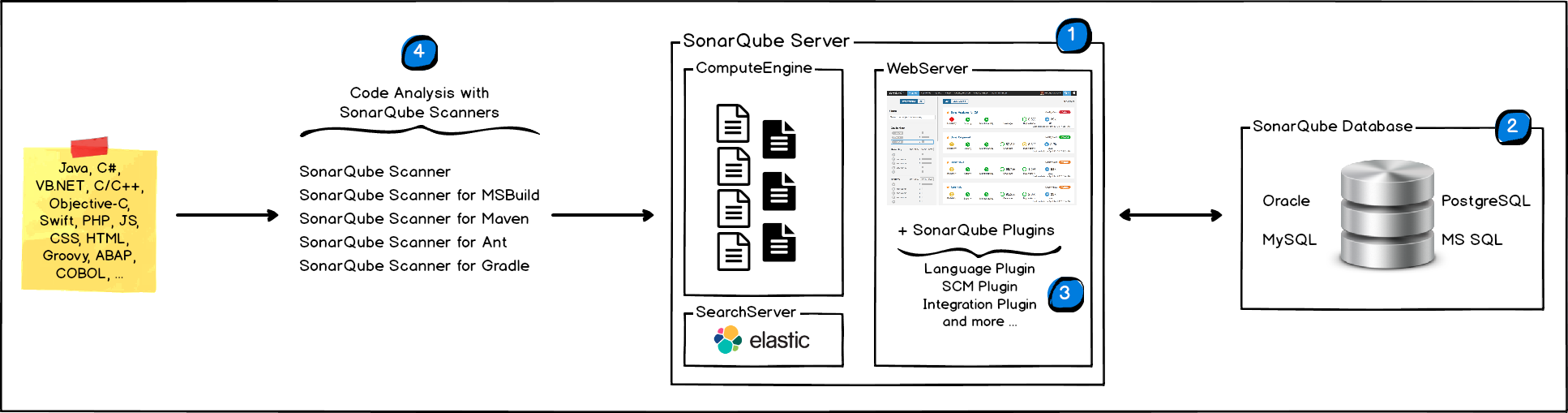
It helps for various tasks and provide reports on duplicated code, coding standards, unit tests, code coverage, complex code, potential bugs, comments and design and architecture.

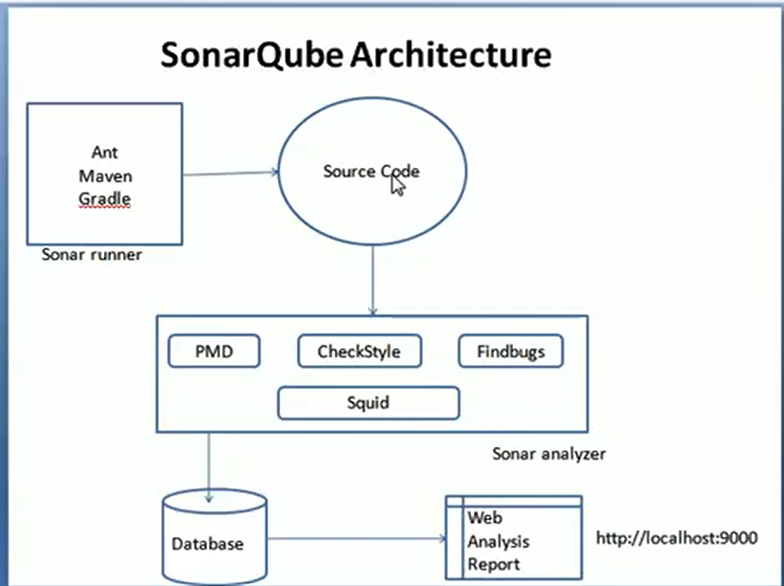
SonarQube is internally using PMD, Find-bugs, Check-Style etc. You can add additionally plugins according to your requirement

**Architecture**

The SonarQube Platform is made of 4 components:

1. One **SonarQube Server** starting 3 main processes:
   * a **Web Server** for developers, managers to browse quality snapshots and configure the SonarQube instance
   * a **Search Server**based on Elasticsearch to back searches from the UI
   * a **Compute Engine Server** in charge of processing code analysis reports and saving them in the SonarQube Database
2. One **SonarQube Database**to store:
   * the configuration of the SonarQube instance (security, plugins settings, etc.)
   * the quality snapshots of projects, views, etc.
3. Multiple **SonarQube Plugins** installed on the server, possibly including language, SCM, integration, authentication, and governance plugins
4. One or more**SonarQube Scanners** running on your Build / Continuous Integration Servers to analyze projects





**Installation the Sonar server on windows**

1. Unzip - let's say in "C:\sonarqube" or "/etc/sonarqube", the SonarQube distribution once it's downloaded. ([Download Page](http://www.sonarsource.org/downloads/))
2. Start the SonarQube server:

|  |
| --- |
| # On Windows, execute:  C:\sonarqube\bin\windows-x86-xx\StartSonar.bat |

1. Log in to [http://localhost:9000](http://localhost:9000/) with System Administrator credentials (admin/admin).

**Installation SonarQube Scanner**

1. Expand the [downloaded](https://sonarsource.bintray.com/Distribution/sonar-scanner-cli/sonar-scanner-cli-3.0.3.778-windows.zip) file into the directory of your choice. We'll refer to it as *<install\_directory>* in the next steps.
2. Update the global settings to point to your SonarQube server by editing *<install\_directory>/conf/sonar-scanner.properties*:

|  |
| --- |
| #----- Default SonarQube server  #sonar.host.url=<http://localhost:9000> |

1. Add the *<install\_directory>/bin* directory to your path(windows environment variable).
2. You can verify your installation by opening a new shell and executing the command sonar-scanner -h (on Windows platform the command is sonar-scanner.bat -h) . You should get output like this:

|  |
| --- |
| usage: sonar-scanner [options]    Options:   -D,--define <arg>     Define property   -h,--help             Display help information   -v,--version          Display version information   -X,--debug            Produce execution debug output |

If you need more debug information you can add the sonar.verbose property by adding the command line parameter -Dsonar.verbose=true.

# Usage of SonarQube Scanner

1. Create a configuration file in the root directory of the project: sonar-project.properties

# must be unique in a given SonarQube instance

**sonar.projectKey=my:project**

# this is the name and version displayed in the SonarQube UI. Was mandatory prior to SonarQube 6.1.

**sonar.projectName=My project**

**sonar.projectVersion=1.0**

# Path is relative to the sonar-project.properties file. Replace "\" by "/" on Windows.

# This property is optional if sonar.modules is set.

**sonar.sources=.**

# Encoding of the source code. Default is default system encoding

#**sonar.sourceEncoding=UTF-8**

Example:

**sonar.projectKey=sareaOverbookings**

**sonar.projectName=sareaOverbookings**

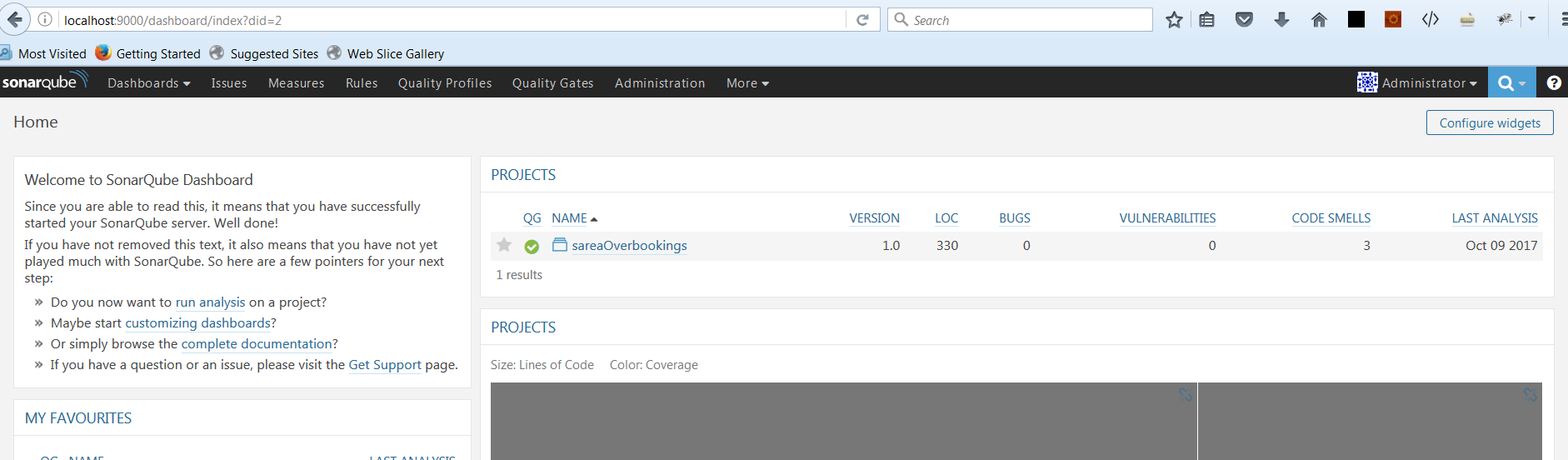
**sonar.projectVersion=1.0**

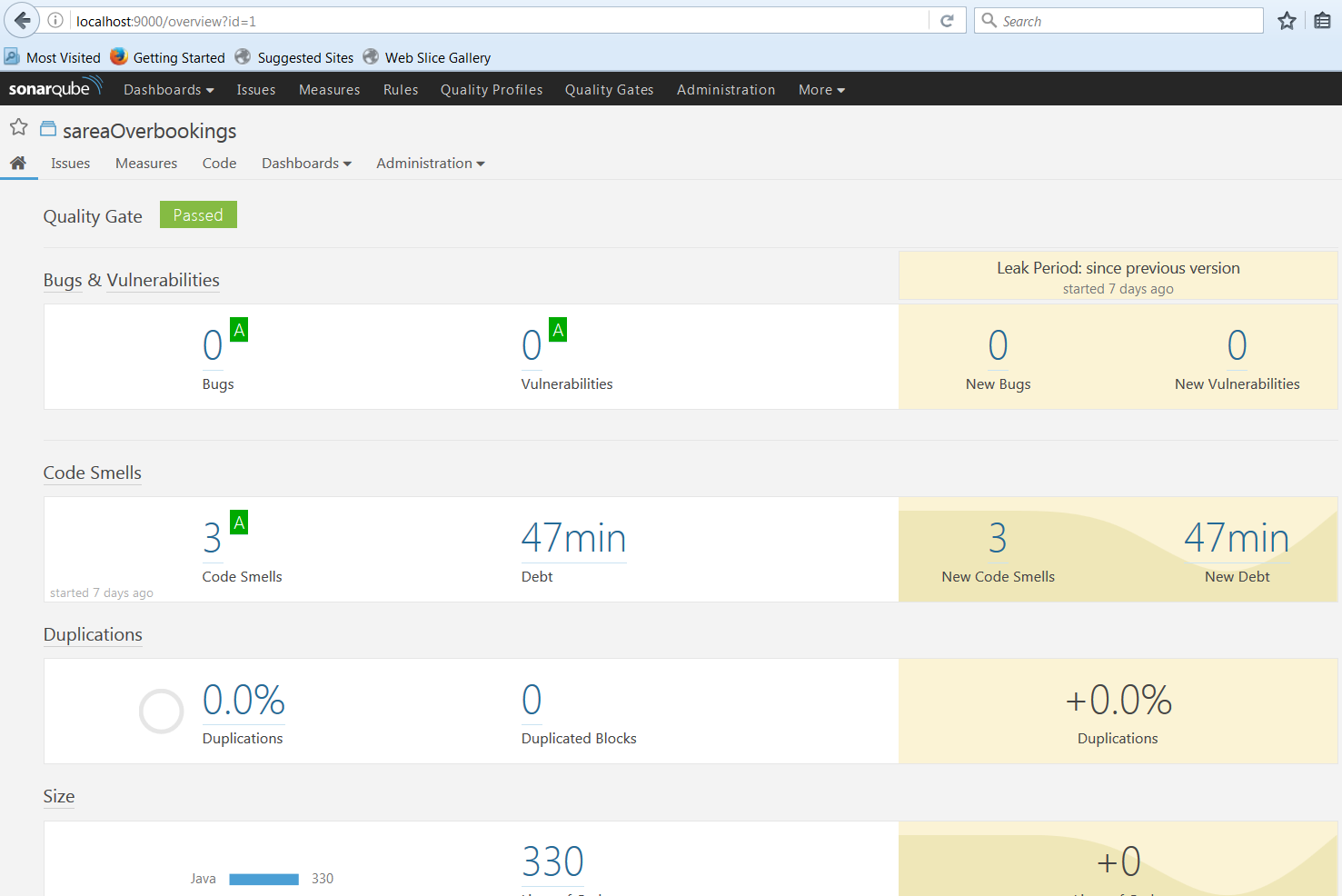
**sonar.sources=C:/sachin/ATT Application/CTI\_New\_repo/cti/src/sareaOverbookings/src/main/java/com/att/pmoss/cti**

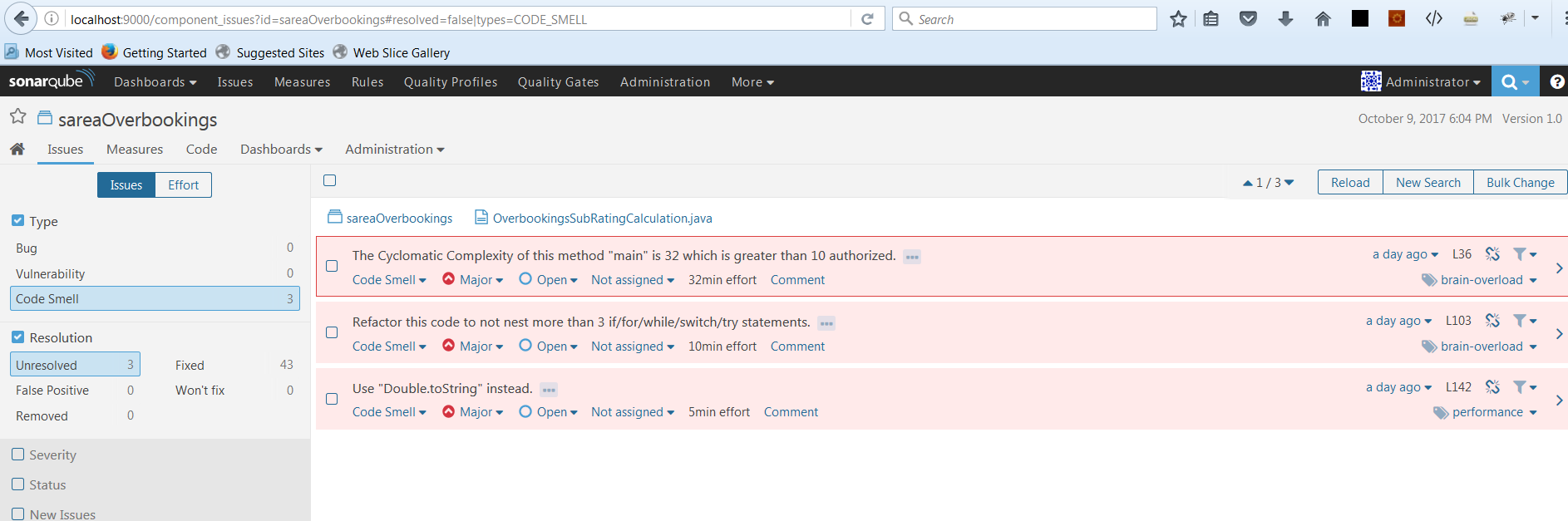
1. Run the following command from the project base directory to launch the analysis:

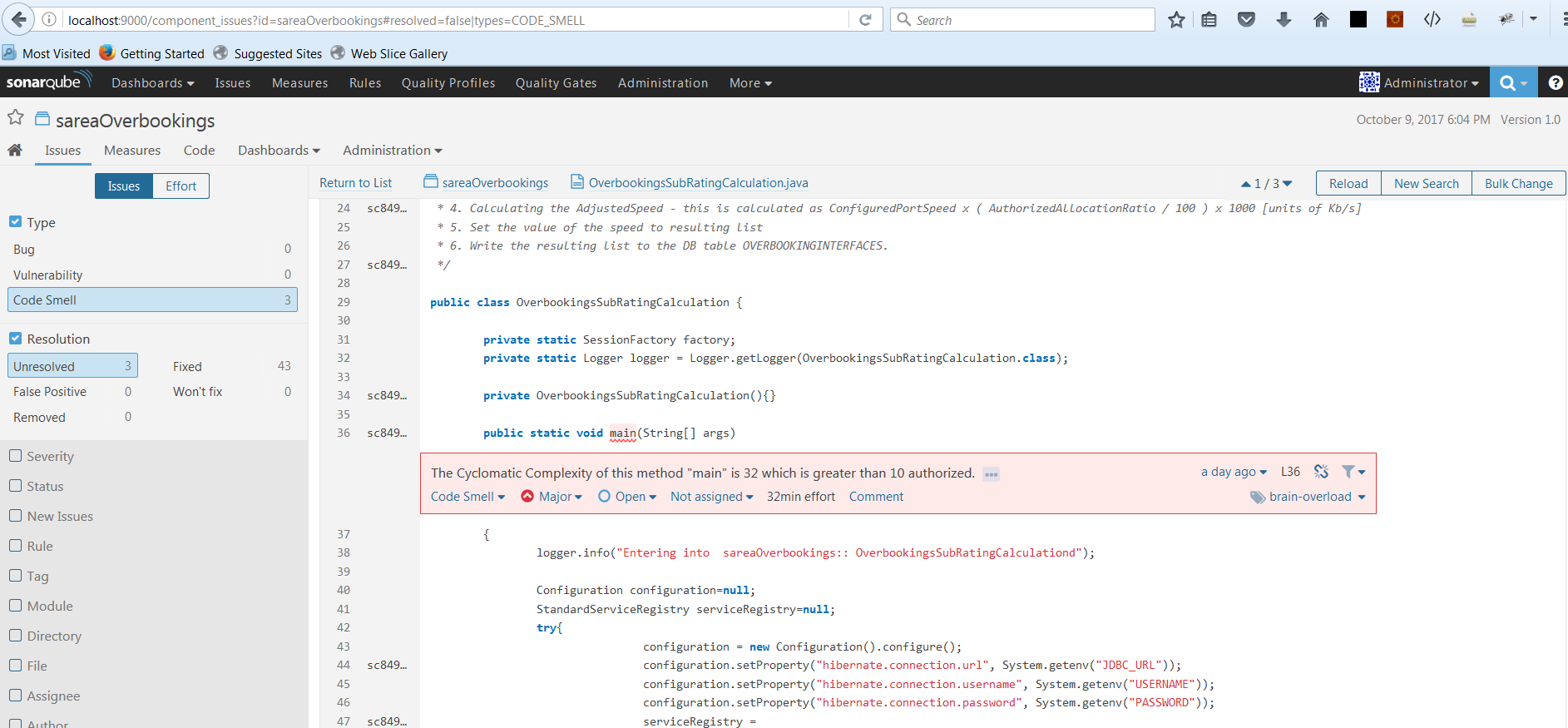
sonar-scanner.bat

1. Finally, report will be available in <http://localhost:9000>









# Integration

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

1. Developers code in their IDEs and use [SonarLint](http://www.sonarlint.org/) to run local analysis.
2. Developers push their code into their favourite SCM : git, SVN, TFVC, ...
3. The Continuous Integration Server triggers an automatic build, and the execution of the SonarQube Scanner required to run the SonarQube analysis.
4. The analysis report is sent to the SonarQube Server for processing.
5. SonarQube Server processes and stores the analysis report results in the SonarQube Database, and displays the results in the UI.
6. Developers review, comment, challenge their Issues to manage and reduce their Technical Debt through the SonarQube UI.
7. Managers receive Reports from the analysis.  
   Ops use APIs to automate configuration and extract data from SonarQube.  
   Ops use JMX to monitor SonarQube Server.

# References:

# <https://docs.sonarqube.org/display/SONAR/Documentation>

# <https://docs.sonarqube.org/display/SCAN/Analyzing+with+SonarQube+Scanner>

# <http://marketplace.eclipse.org/marketplace-client-intro?mpc_install=2568658>

# <https://www.sonarqube.org/downloads/>