ASSIGNMENT 4 (FINAL)
REFACTORING & WRAP UP

CSE2115 Software Engineering Methods
Team 90

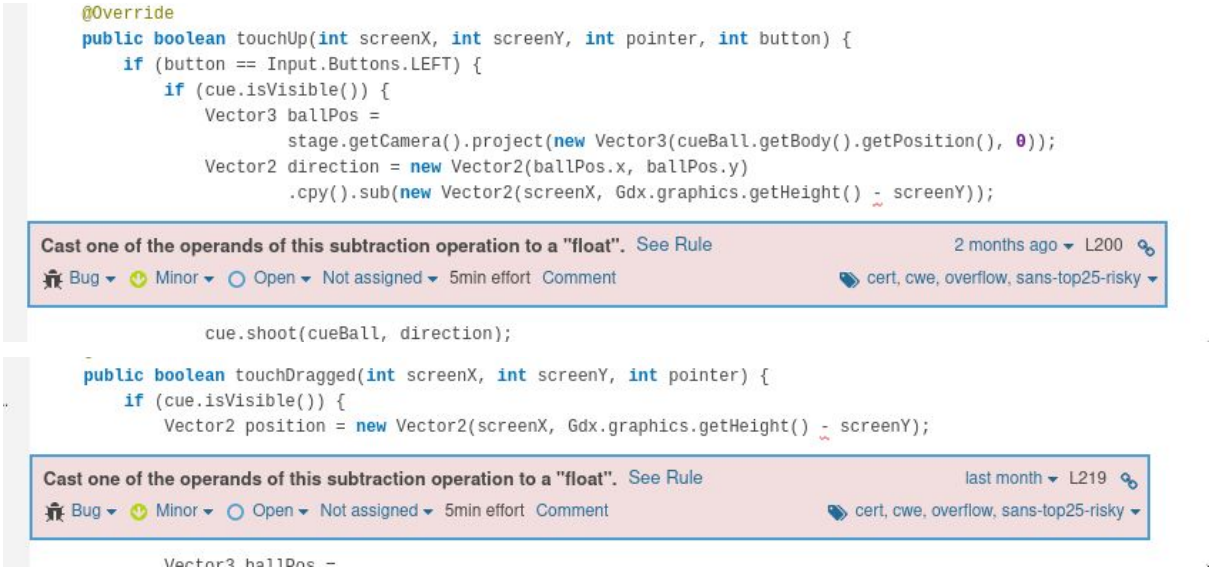
Made by:

Victor Roest
Tim Numan
Maikel Houbaer
Jesse Nieland
Eduard Filip

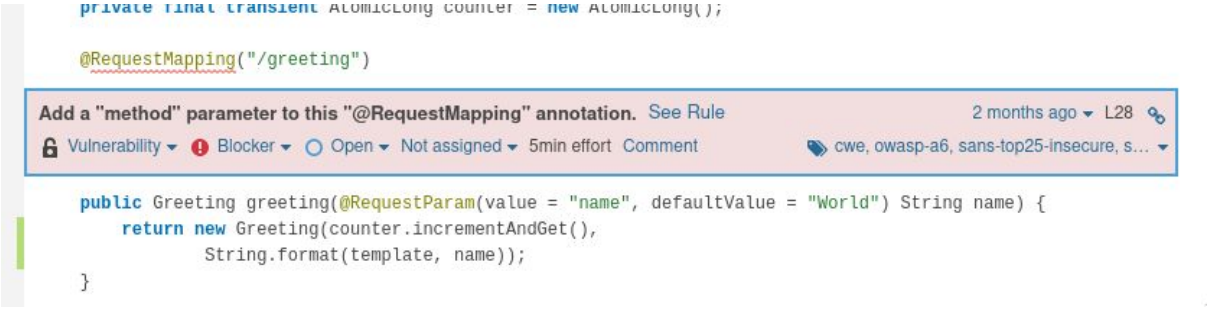
What and why?

We evaluated two code metric tools: MetricsReloaded and Sonarqube, we ended up choosing Sonarqube over MetricsReloaded because Sonarqube gave much more verbose and useful information, such as tips on how to solve and graphs to show our progress. All screenshots used in the rest of this document are therefore screenshots from the Sonarqube webgui.

Method level



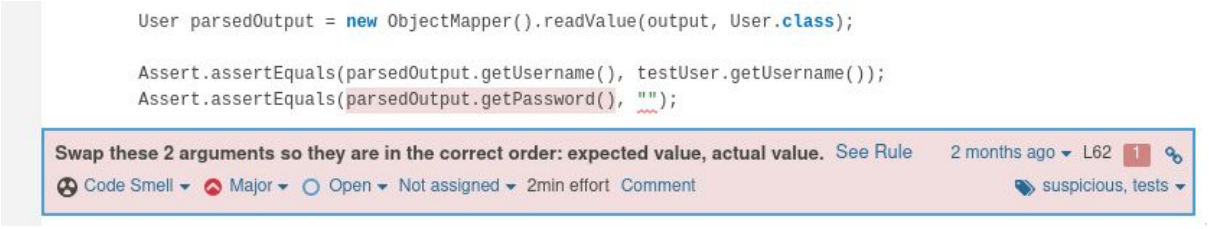
2 Times a possible overflow: Fixed by casting to float.



“RequestMapping” is unnecessarily vague and poses a security risk, changed to “GetMapping” to fix.

Changed access to a static as suggested.

The “Integer” specifier was just unneeded, removed to fix the complaint.



In our tests we sometimes specified the order of variables in the asserts in the wrong order: the expected value should be first, followed by the returned value.
Sonarqube didn’t flag all of them, as seen in the screenshot above the first assert also has the order of variables in the wrong order, this made us check and fix all asserts (using some regex).

Class level

Fixed by making them private, static and final (where applicable) and providing getters and setters where needed, also renamed each static variable to UPPERCASE.

Unnecessary public static variables: fixed by making them private and providing getters and setters where needed.

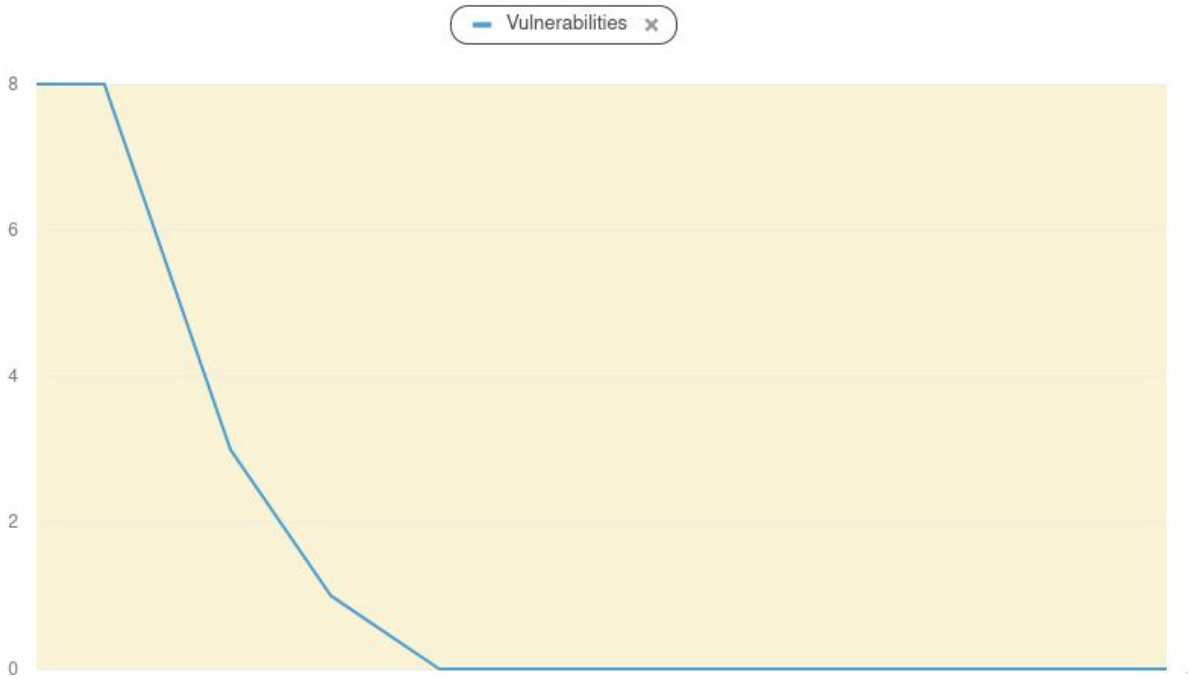
Static final variable not matching the conventional naming scheme, changed to match.

Unnecessary overrides, fixed by removing them.

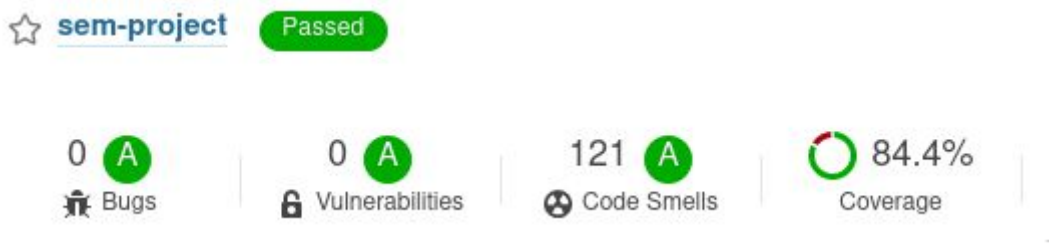
The analysis didn't detect this completely correctly, this should actually be a *final* variable, of which the naming scheme does match. Changed the variable to final to fix.

Some graphs showing our improvements

This shows the amount of code smells over time, left is before the refactor and to the right is after.



This shows the amount of vulnerabilities over time, we fixed all the reported ones as can be seen in the graph.



Sonarqube also gave us an “A” in all categories with which we are really satisfied.

Remarks

Sonarqube reported that we added a lot of unnecessary “transient” specifiers, however we chose to remove that rule. This is because otherwise PMD would complain, in fact PMD actually ‘forced’ us to use all those “transient’ specifiers. It was simply not allowed to have variables that didn’t have either static or transient before them. Because PMD is one of the tools we had to use according to the template project/project description, we decided to following is rule in this case and suppressed Sonarqube’s.