**CSE 5382 - Secure Programming**

**Assignment 1**

Submitted by: Biswa Ranjan Nanda (1001558251)

**Part 1:**

**Manual Analysis**

1. Improper indentation of the code inside the loops and functions.
2. In line number 22, static variable declaration i.e. PORT, using such hard-coded value could be a threat or a vulnerability to the program.
3. Static objects should be used once, whereas in the program “dServerSocket” is declared as static in line 26, it is being initialized again in line 29.
4. FileReader object fr is not closed in this serveFile method, which is another issue in this program.
5. No try catch block inside main method can be considered as a poor programming practice.

**Tool Choices/Versions**

Below are the tools mentioned with their version numbers that will be used in this assignment:

* IntelliJ IDEA IDE 2018.2.3 ultimate Edition- <https://www.jetbrains.com/idea>
* FindBugs- IDEA Plugin 1.0.1 (includes FindBugs 3.0.1 & Findbugs Security Plugin)  - <https://plugins.jetbrains.com/plugin/3847-findbugs-idea>
* PMD Plugin 1.8.8  - <https://plugins.jetbrains.com/plugin/1137-pmdplugin>

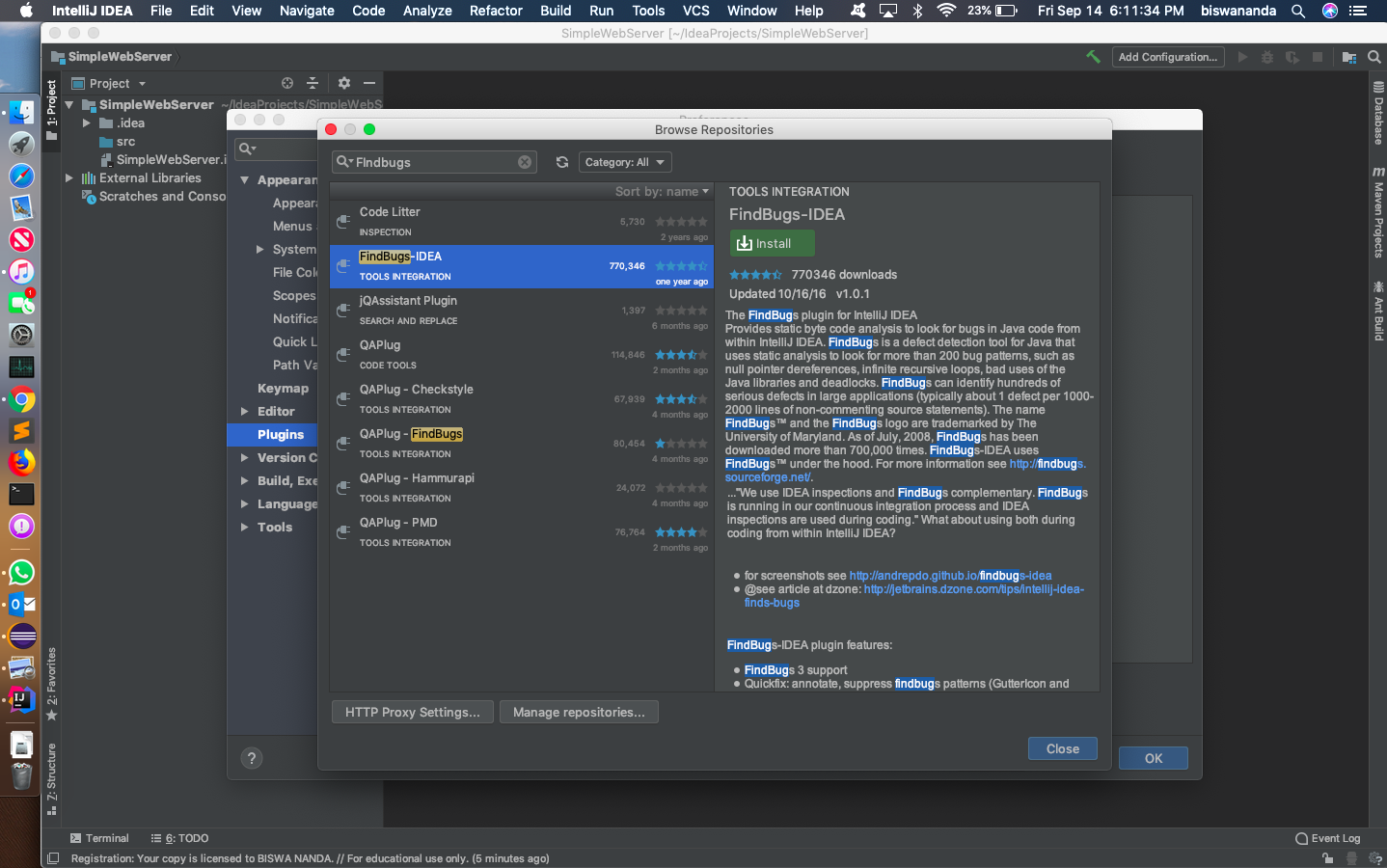
**Tool Invocation Process**

1. Using IntelliJ, we set up the project and the screenshots for all the steps are mentioned below:

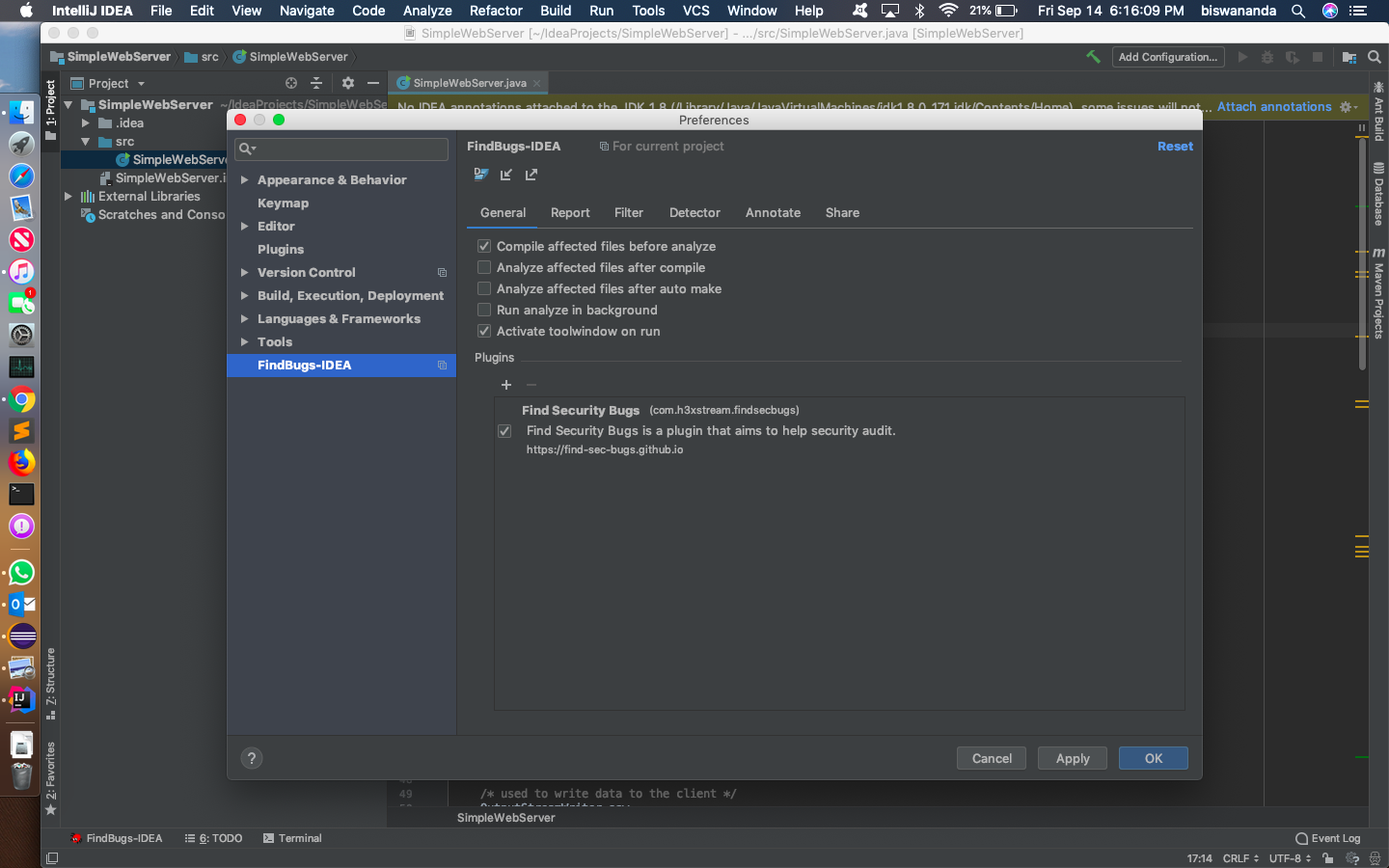
* Steps
  + IntelliJ IDEA->Preferences->Plugins Install FindBugs-IDEA or download & install from disk. Install PMD Plugin or download & install from disk. Restart every time, after installing each plugin.
  + IntelliJ IDEA->Preferences->Plugins FindBugs-IDEA Click ‘+’ Add Find Security Bugs plugin.

1. Steps performed to setup the project

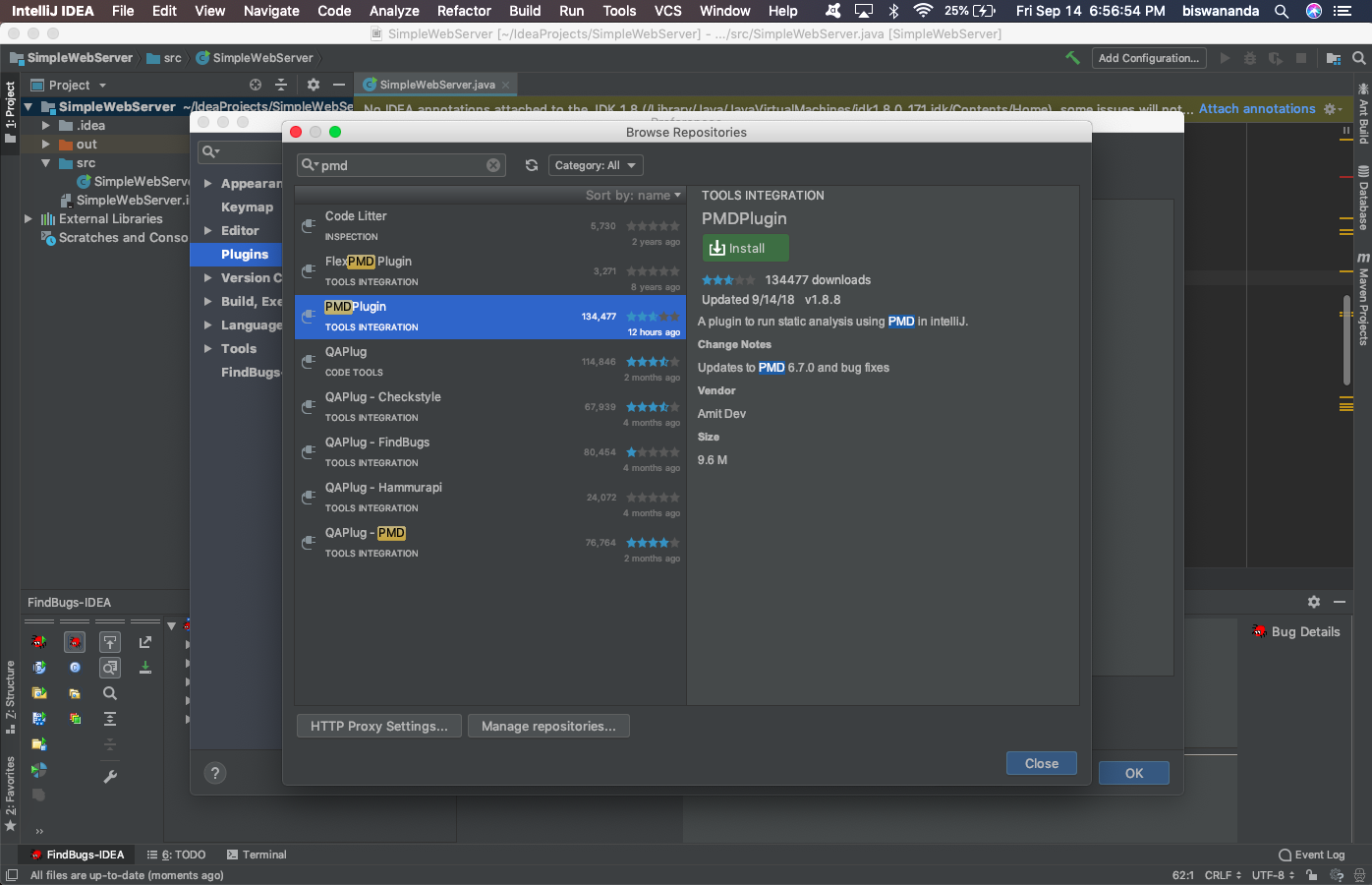
**2.1**



**2.2**



**2.3**



**Comparison/Contrast Tools**

1. Does the tool analyze source or binary as input?

- **FindBugs**

FindBugs uses static analysis to inspect Java bytecode for occurrences of bug patterns. Static analysis means that FindBugs can find bugs by simply inspecting a program's code, executing the program is not necessary. FindBugs works by analyzing Java bytecode (compiled class files), so you don't even need the program's source code to use it.

- **PMD**

PMD, an extensible cross-language static code analyzer. It finds common programming flaws like unused variables, empty catch blocks, unnecessary object creation, and so forth. Additionally, it includes CPD, the copy-paste-detector. CPD finds duplicated code. Analyses AST (Abstract Syntax Tree) generated by JavaCC.

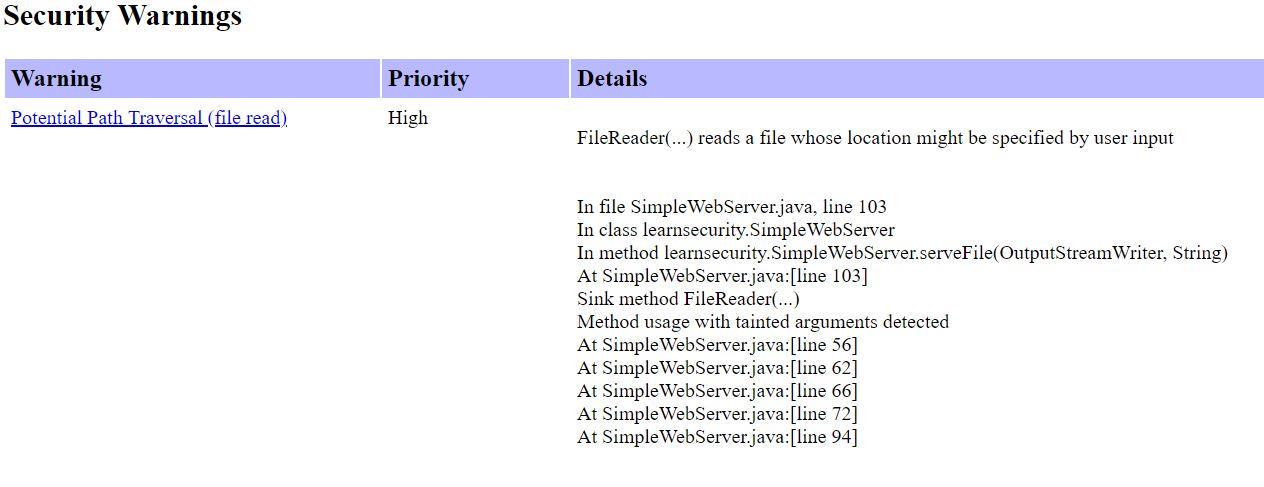
PMD scans source code, analyses AST (Abstract Syntax Tree) generated by JavaCC in Java and other languages and looks for potential problems like:

1. \* What category of tools is it?

* Bug Finding
* Security Review

1. Show an example (if one exists) of a finding that is reported by one tool and not others.

* Below is the screen shot from the report of Finbugs, Security warning shown by FindBugs, but not by PMD.



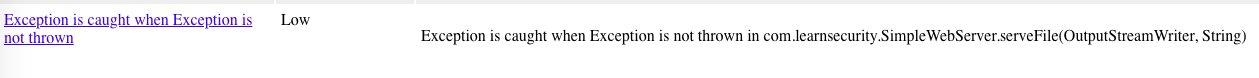
1. Show an example (if one exists) of a finding reported by multiple tools.

* Exception finding is reported by both PMD & FindBugs respectively.

**From PMD Report**

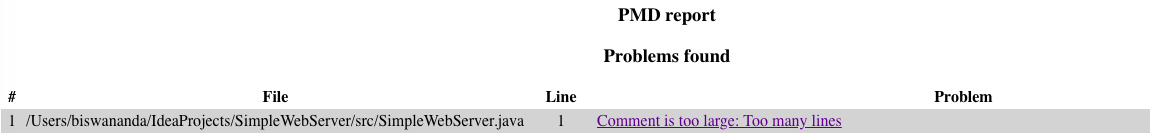


**From Findbug Report**

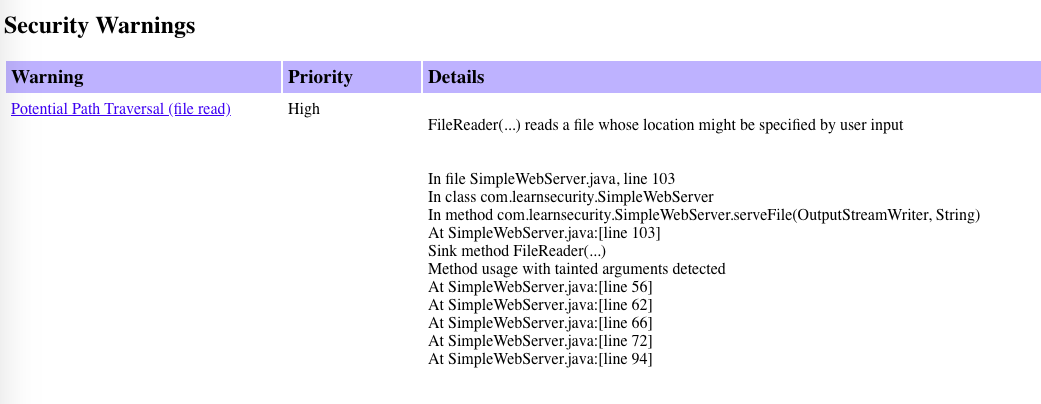


1. For the known flaw in the code used, document which tools reported it (true negative) and which tools did not (false positive).

* PMD reports comment is too large. False positive.
* PMD reports true negative. FindBugs reports it, though.

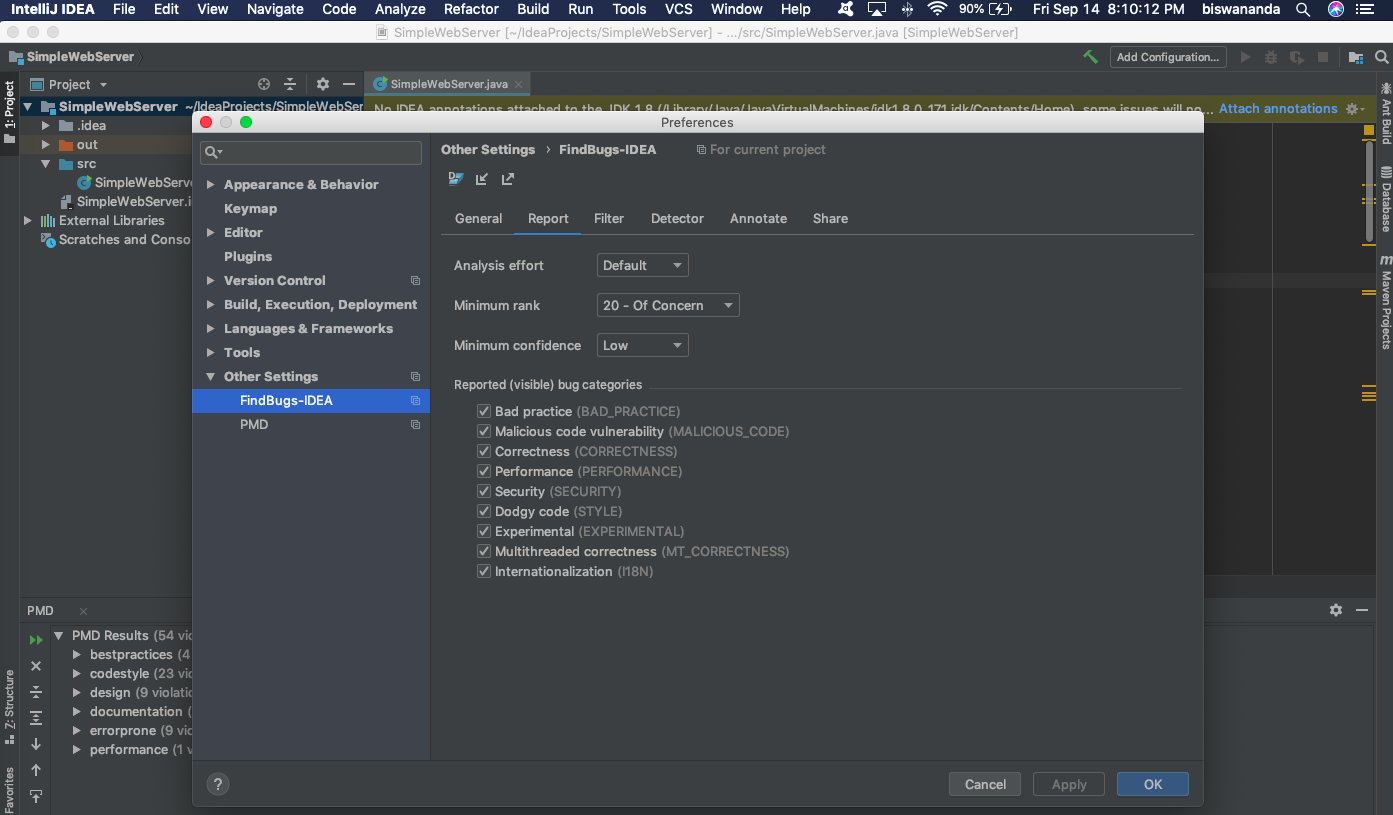


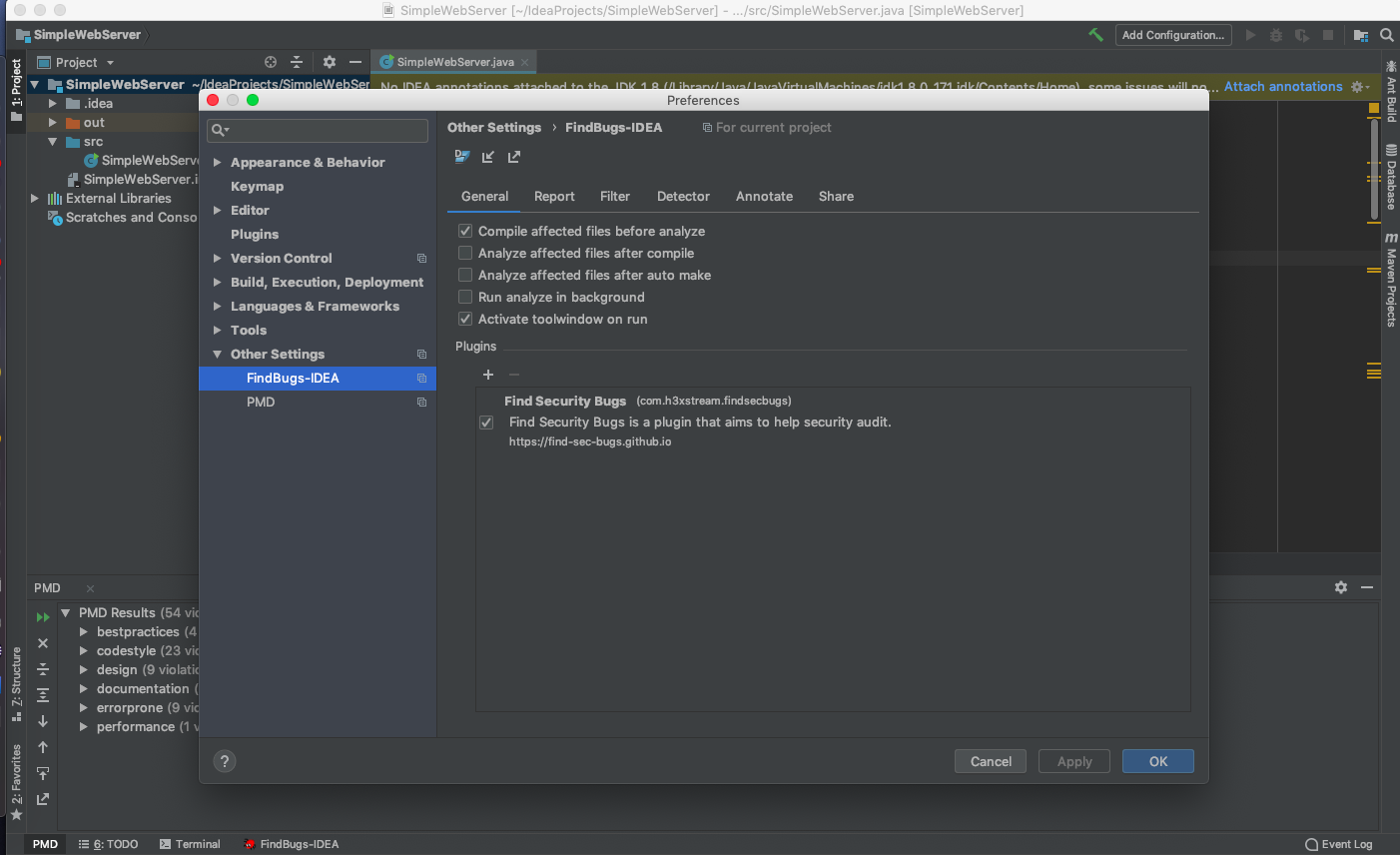
Findbug report mentioned below:



**Results**

* Reports have been generated and attached separately.
* Security rules enabled. (as mentioned previously)
* Turned on most aggressive mode.  (Screenshot mentioned below)





**Findbugs**

****

**PMD:**



**Part 2**

**Code Provided**

The code used for the part 2 of this project

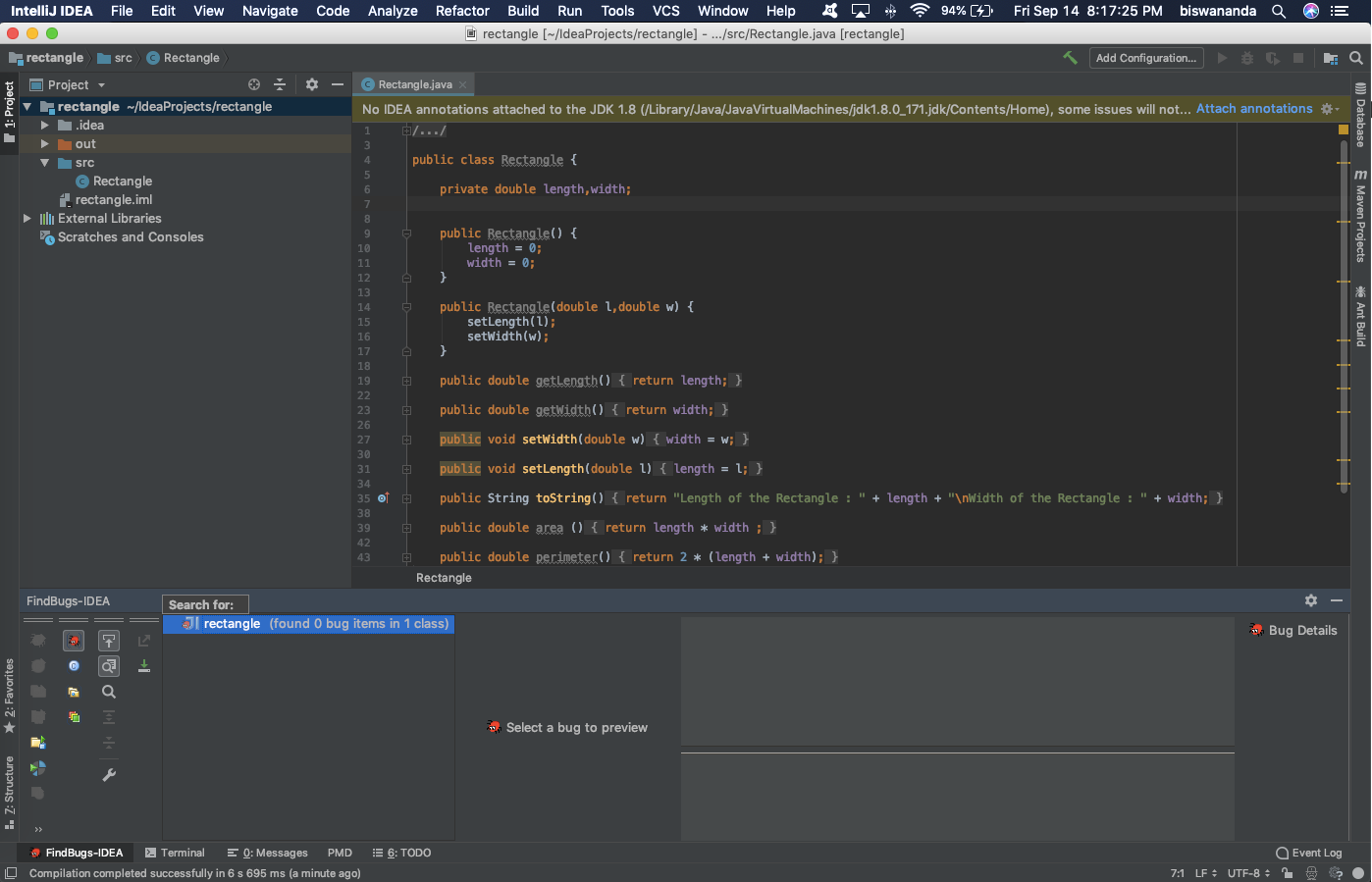


**Results:**

Below are the results returned by Findbugs and PMD tools:

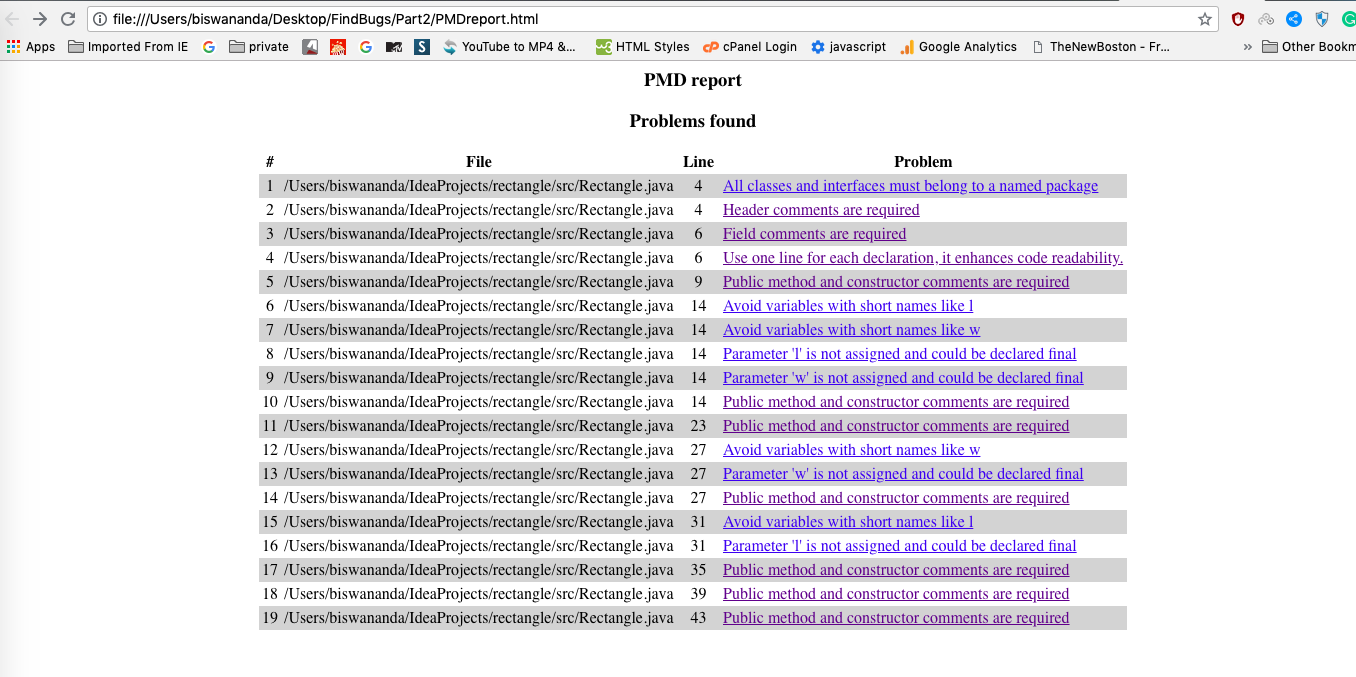
FindBugs tool Report

No bugs were found by FindBugs tool.



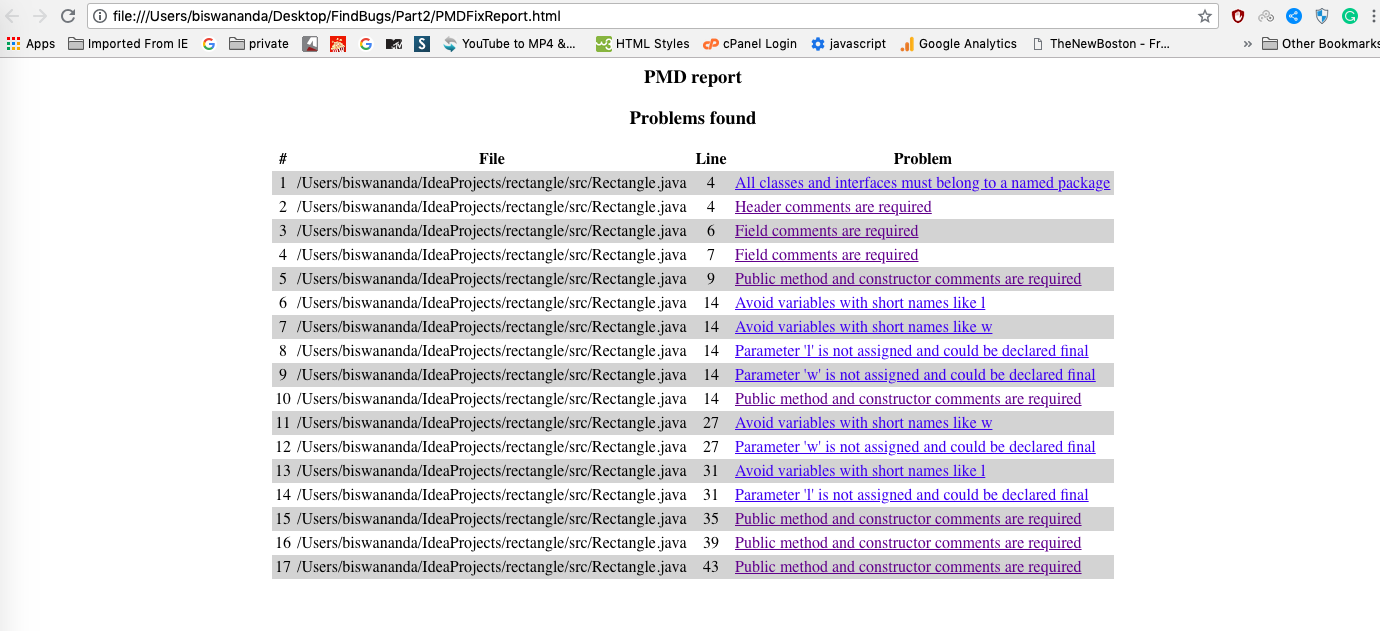
**PMD Report**

Report generated by PMD tool



**Fixes:**

* There is no bugs found by findBugs tool to fix.
* The bugs found by PMD in line 6 are fixed and the screenshot is mentioned below:



**Details of the bug fix:**

Bug -> **private** **double** length,width;

Fix to the bug:

**private** **double** length;

**private** **double** width; -> *declared in separate line*