**CS763 Lab 1 Static Application Security Testing**

Yourname

**Instructions**

* Please document your findings by providing both text descriptions and screenshot(s) of the execution results in the lab report.
* Please name your report as CS763\_<Last Name><First Name>\_Lab1. It can be either a PDF or Word document.
* Please provide your feedback in the “Add comments” section when submitting your lab report. Thanks!

## **Brief description (purpose and overview)**

SAST tools are designed to conduct automated inspection on a given codebase and detect potential defects without actually executing the code. There are quite a number of commercial and open-source tools designed to inspect code and report found issues. In this lab, students will use two of them, SpotBugs and SonarQube to find bugs in a given Java program(s).

The purpose of this lab is to help students get to know static analysis tools, so that they can start to use these tools to improve the quality of their software projects.

## **Learning Objectives**

After finishing this lab, students shall be able to:

1. Understand basic concepts about static code analysis tools.
2. Use a static code analysis tool to analyze the code.
3. Understand some Java code bugs.

## **Prerequisite knowledge**

* Have very basic knowledge of Java
* Be comfortable using an IDE.

## **Lab Setup Requirements**

* Install Java JRE (11) and JDK (11)
* Install an IDE such as Intellij IDEA (<https://www.jetbrains.com/idea/download/>).
* Install the Spotbugs plugin for IDEA.
* Install SonarQube Community Edition (<https://www.sonarqube.org/downloads/>)
* Program under test: <https://github.com/CSPF-Founder/JavaVulnerableLab/> (A vulnerable web application developed by Cyber Security and Privacy Foundation (www.cysecurity.org)
* Programs of your choice.

**Resources**

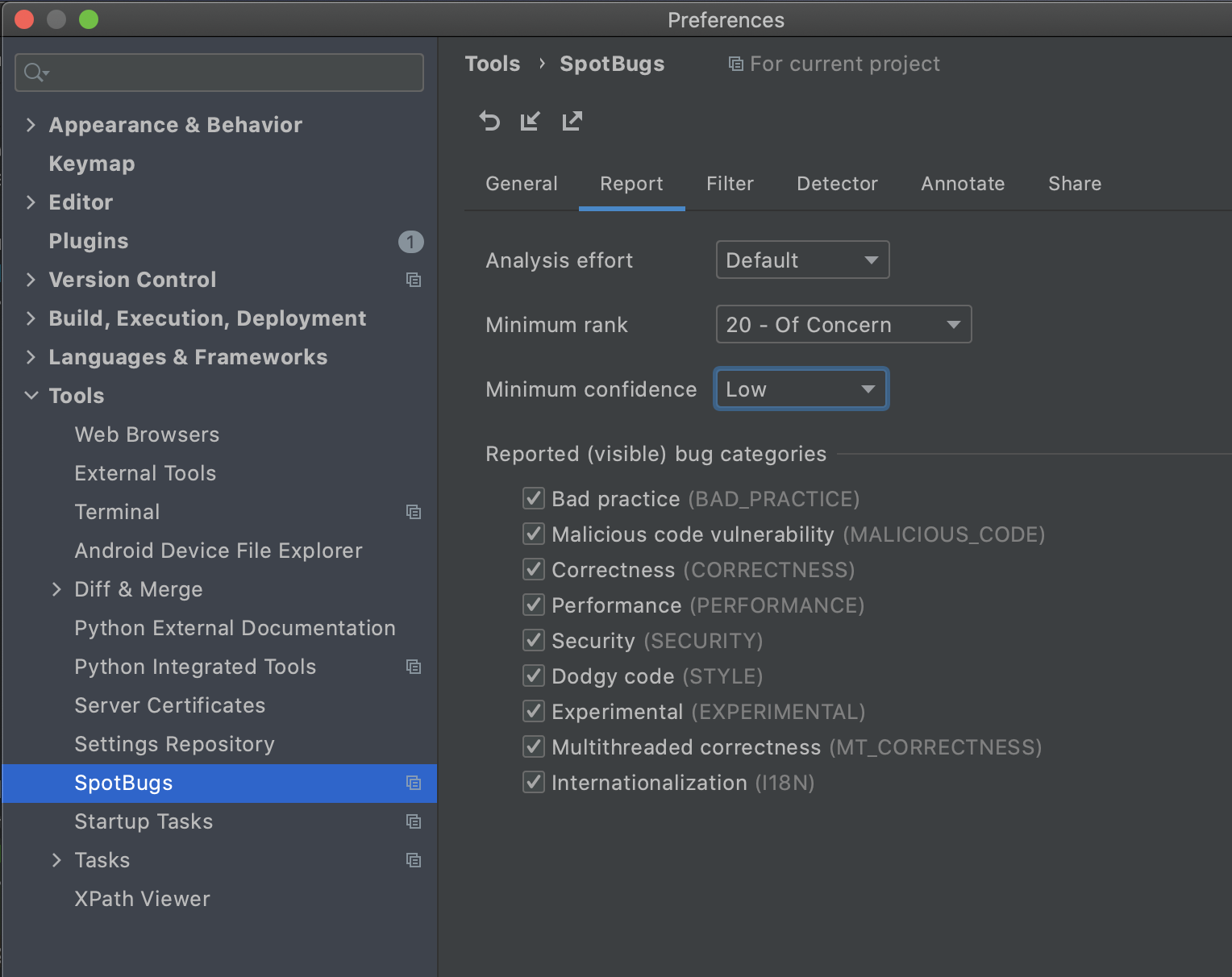
FindBugs is an open source static code analysis tool. It is also integrated into an industrial security tool Fortify SCA. You can find more information about FindBugs at <http://findbugs.sourceforge.net/> . You can find a talk by Prof. Bill Pugh, one of its creators at <https://www.youtube.com/watch?v=GgK20Yv9QRk>. SpotBugs is the spiritual successor of [FindBugs](http://findbugs.sourceforge.net/), carrying on from the point where it left off with support of its community. SpotBugs requires JRE (or JDK) 1.8.0 or later to run. However, it can analyze programs compiled for any version of Java, from 1.0 to 1.9.

SonarQube (<https://www.sonarqube.org/>) is another static code analysis tool not only for Java, but many other programming languages. It is an open-source platform developed by SonarSource for continuous code inspection to detect bugs, code smells, and security vulnerabilities in 20+ programming languages. It provides fully automated analysis and integration with Maven, Ant, Gradle, MSBuild and continuous integration (CI) tools such as Jekins, Hudson. The community edition is completely free, which we will use in this lab.

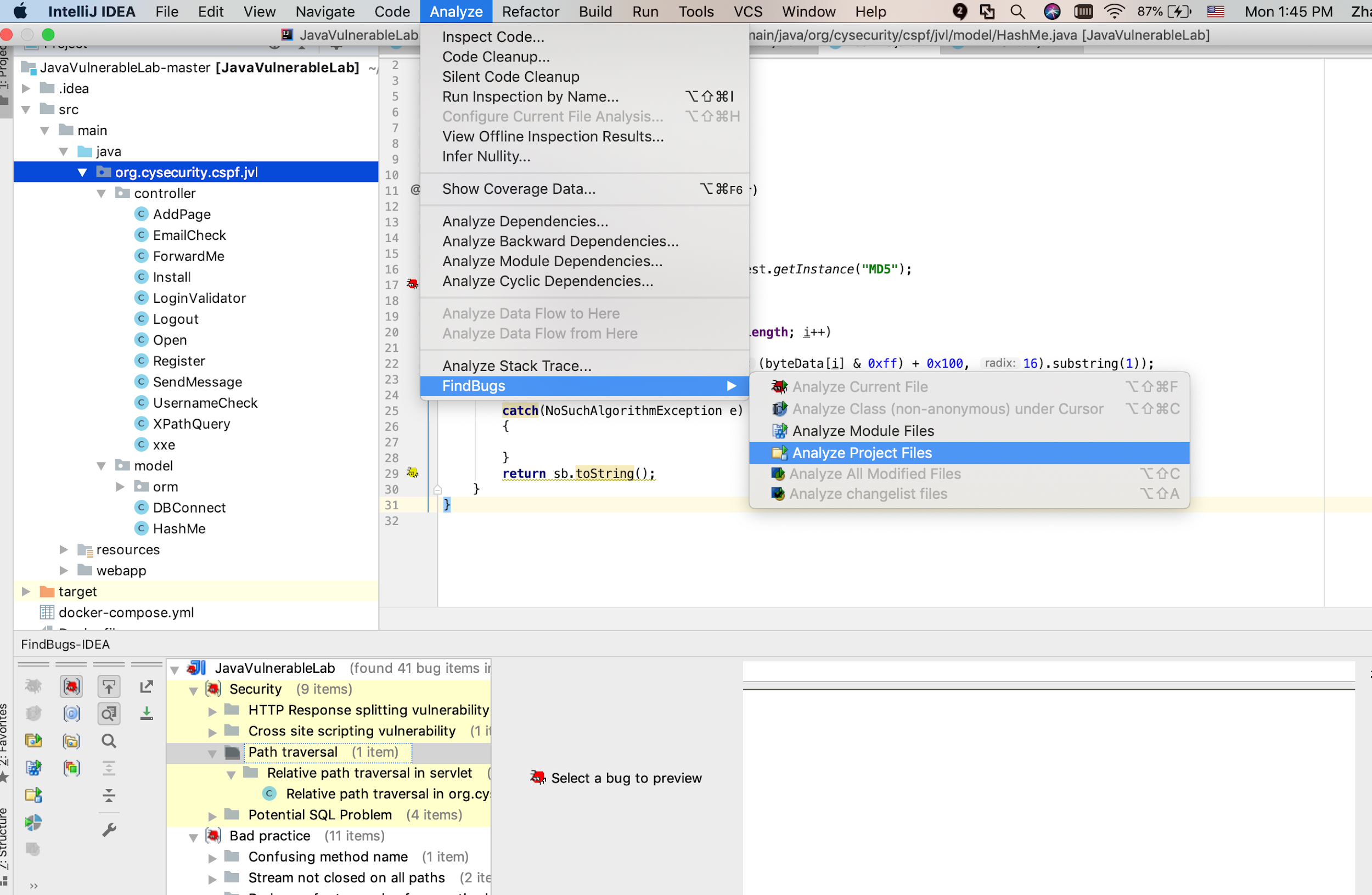
We will use SpotBugs and SonarQube to find bugs in a vulnerable web application developed by CSPF. This web application is written in Java and has a number of vulnerabilities. In this lab, you are NOT required to fully understand the application and run the application. However, you can definitely install the application and run it either in a docker or virtual machine and check the vulnerability through dynamic analysis or testing.

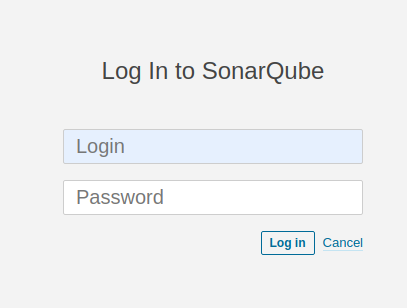
## **Detailed instructions**

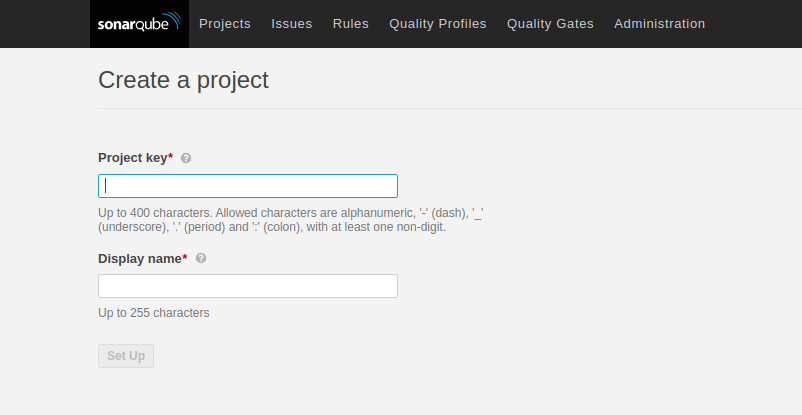
1. Assume you have installed the IDEA, you can install the plugin as the following:
   1. Open the Preferences dialog and then go to **Plugins**.
   2. Click the **Install** JetBrains **plugin**.
   3. In the dialog that opens, search for *spotbugs*.
   4. Press **Install**.
   5. Click OK in the Settings dialog and restart **IntelliJ** IDEA.
2. Config spotbugs
   1. Open the Preferences dialog and then go to Tools -> SpotBugs. You can set reported bug categories at the Report tab. For example you can select “Maximal” for the “Analysis Effort” and “Low” for the “Minimum confidence to report” to report all possible bugs. You shall also enable all of the bug categories. You can also suppress some types of bugs and set filters to reduce false positives. Go to the Detector tab, you will find more details of each detector.

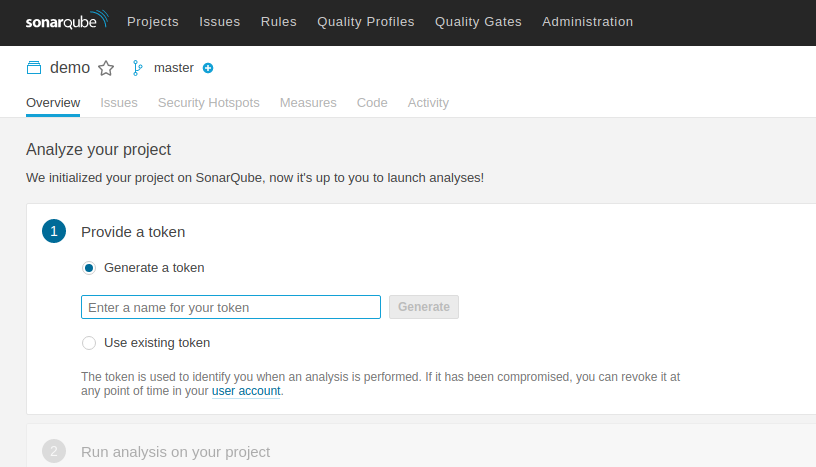


* 1. Download or fork the project from the github: <https://github.com/CSPF-Founder/JavaVulnerableLab/>. Then import the project into the IDE (In Intellij IDEA, click File->New->Project from Existing Sources…, and then choose the right fold to import) .
  2. Run SpotBugs to find all bugs in the project. In Intellij IDEA, click menu item Analyze->SpotBugs->Analyze Project Files. Then you will see a window showing bugs found and their detailed information.

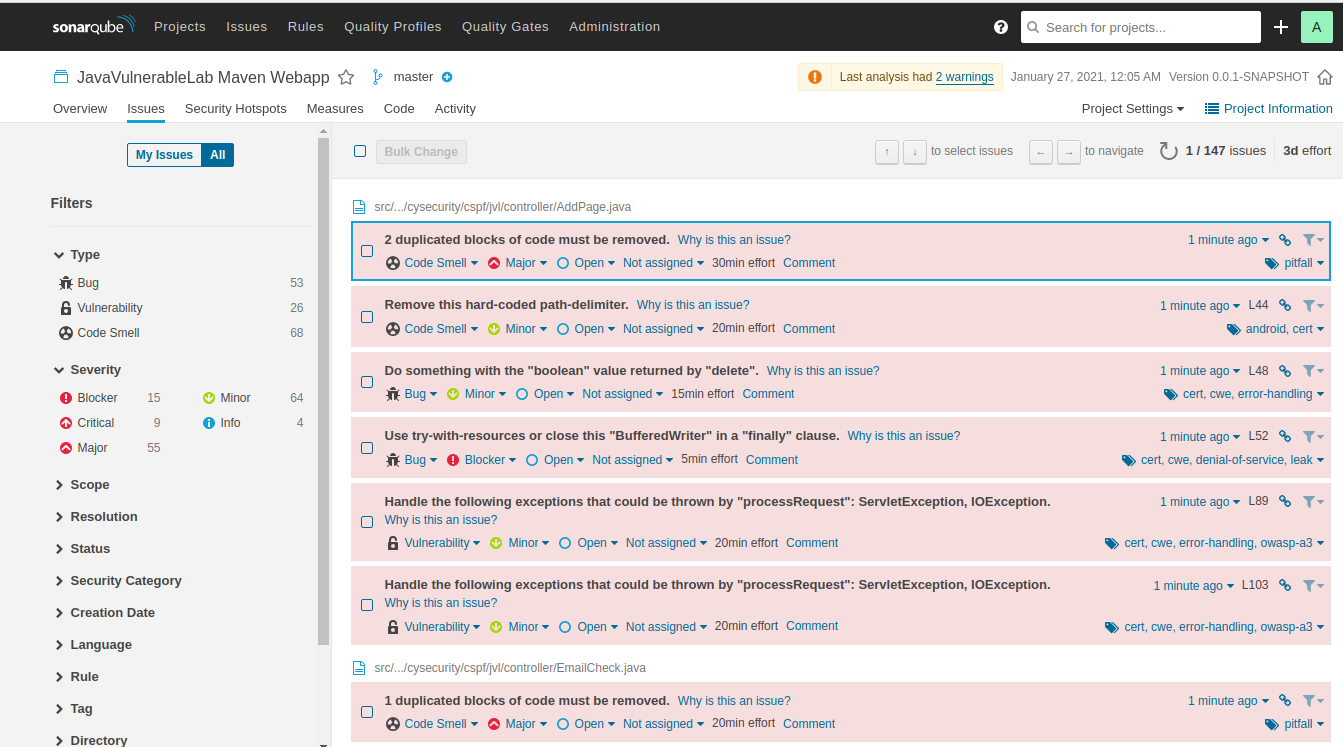
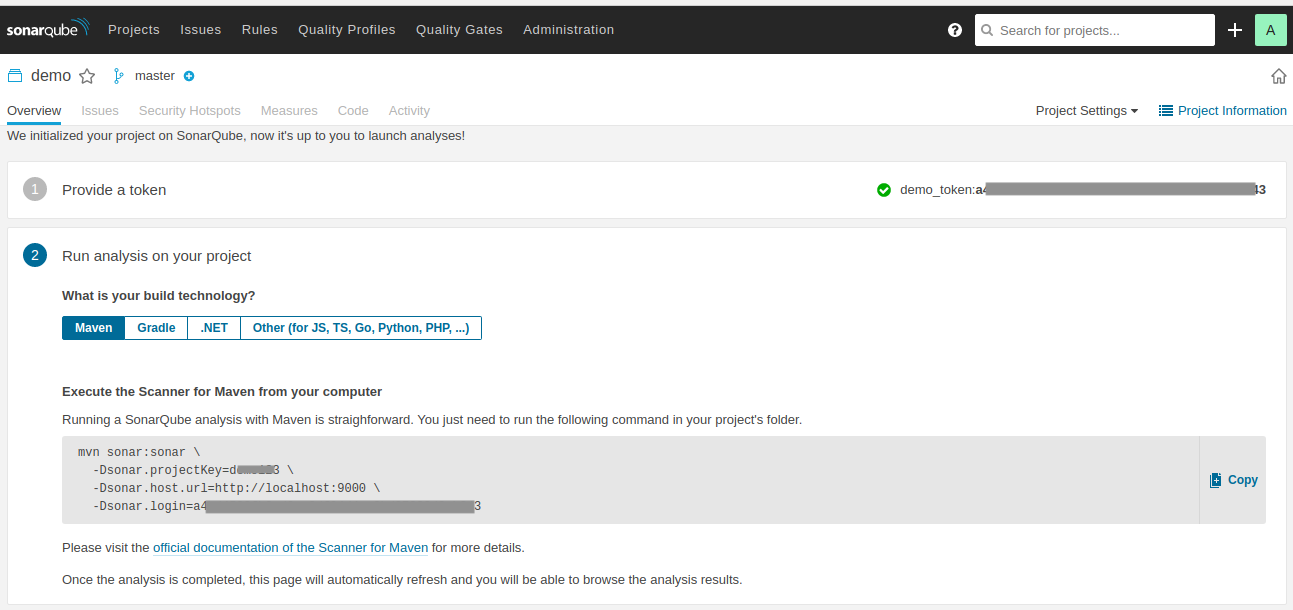


1. Assume you have already downloaded the SnoarQube Community Edition. Unzip the downloaded zip file and start the SonarQube Server based on the instructions at <https://docs.sonarqube.org/latest/setup/get-started-2-minutes/>. Make sure that you have JRE 11 and JDK 11 installed. Once your instance is up and running, Log in to [http://localhost:9000](http://localhost:9000/) using System Administrator credentials (login: admin, password: admin). You will be promoted to change your default password after you login.
2. Then you can create a project. You will need to specify a project key and display name. Afterwards, you need to provide a token. You can generate a token or use a previous token. Here we will generate a new token and provide a token name.





1. Now you can run analysis on your project. You need to choose the build technology. If your project uses Java with Maven or Gradle, you can scan the project directly by running some commands. The scanner is automatically downloaded. If you use other technology, you need to download the scanner first and then execute the scanner. In this example, the project that we analyze uses Java and Maven. So we can simply run the command to download and execute the scanner. If there are some errors, you may want to run “mvn clean” first. You can find more details at <https://docs.sonarqube.org/latest/analysis/scan/sonarscanner-for-maven/>. After the scanning is complete, you can see the report on the Sonarqube page.



## 

## **Questions**

1. (20 points) How many bugs are identified by SpotBugs? What are the categories of these bugs? What are security bugs identified? Modify the configuration to only report the scary bugs. How many are they? Can you find any false positive?
2. (20 points) How many bugs are identified by SnoarQube? What are the categories of these bugs? What are security bugs identified? How many critical security bugs? Can you find any false positive?
3. (10 points) Compare the results by SpotBugs and SnoarQube and state your findings.
4. (40 points) Choose 5 bugs from any report and explain them in more detail. Do you know how to fix them? You may need to do additional research on them. (Optional: If you are familiar with Java and web development, you may review the code manually, and see if you can find any bugs that are not identified by Spotbugs or SnoarQube?)
5. (Extra 10 points) If you have done any project before, use SpotBugs or SonarQube to examine one of your projects and report your findings.

## **Deliverables**

Please submit the lab report in a single document named *CS763\_yourusername\_Lab2*. Please submit a word or PDF document. The lab report should include:

1. Title, author
2. Table of Contents
3. The detailed steps and results using text descriptions and screenshots that answer the above questions and demonstrate your lab progress.
4. A summary of your own reflection on the lab exercise, such as:
   1. What is the purpose of the lab in your own words?
   2. What did you learn? Did you achieve the objectives?
   3. Was this lab hard or easy? Are the lab instructions clear?
   4. What do you think about the tools used? What worked? What didn’t? Are there other better alternatives?
   5. Any other feedback?