Static Code Analysis Exercise

For this exercise, you will run SonarQube to analyze your Java project code and SonarScanner to analyze your TypeScript code. There is a deliverable due by end of class and another for Sprint 4.

With the generated reports, you will put together a plan to address issues that were flagged in your Java or TypeScript code.

Note: Make SonarQube is **running**, which you should have verified in the setup document.

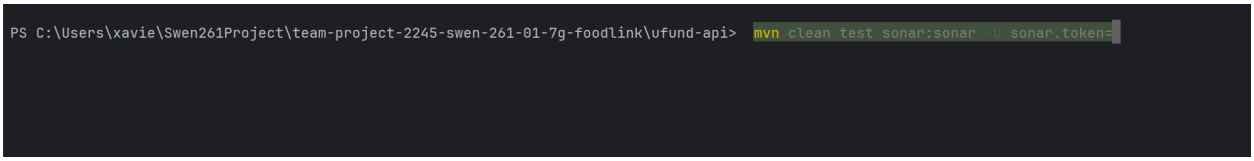
Analyzing code with SonarQube

We will use SonarQube to analyze our Java code

1. Go to your project directory where you would run maven to build and test your code and enter the following placeholder incomplete command but **don't run it yet**:

   ```
   mvn clean test sonar:sonar -D sonar.token=
   ```

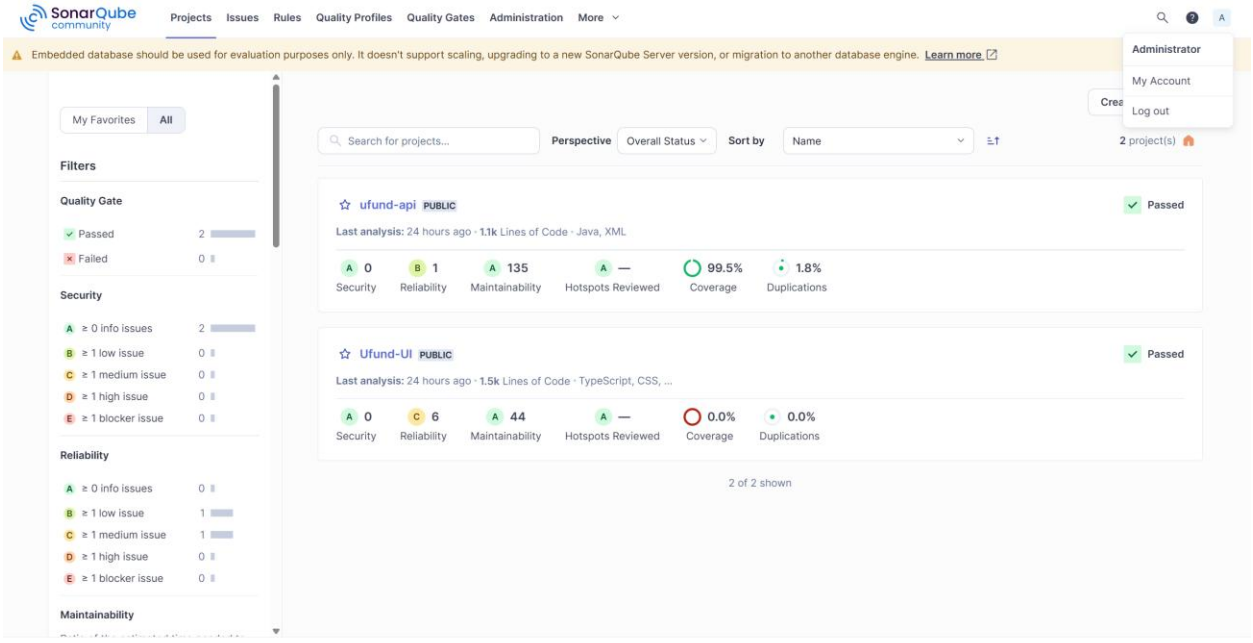
   Should look something like this:

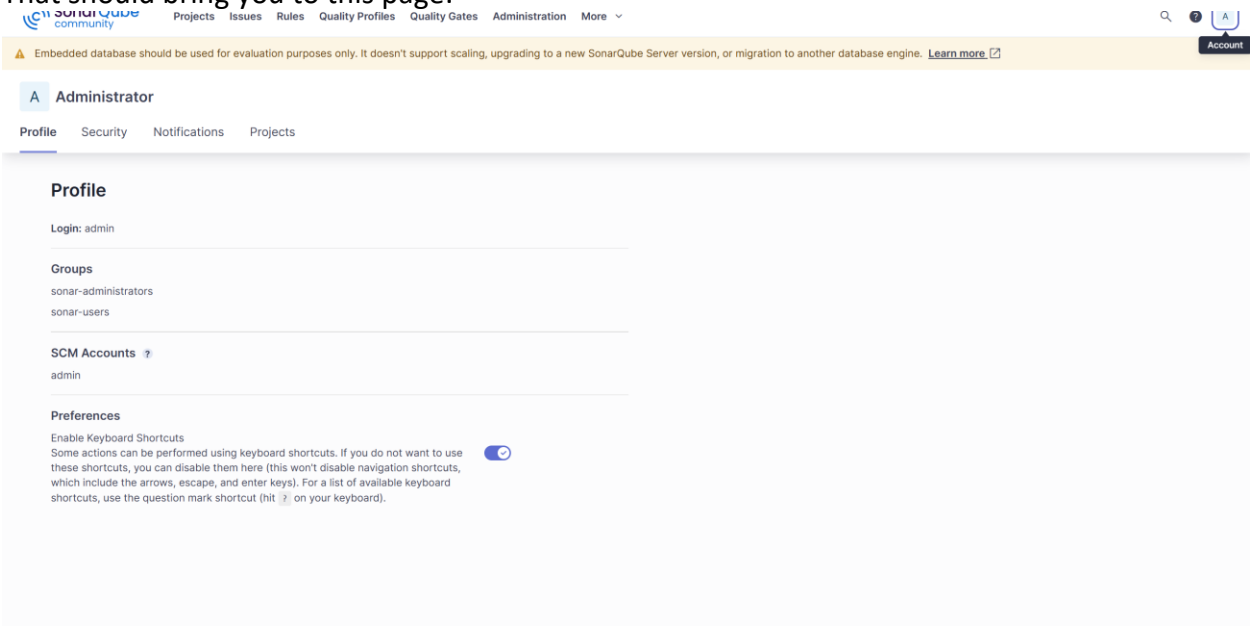
   NOTE: in some literature you may see the DEPRECATED **DO NOT USE:**

   ```
   mvn clean test sonar:sonar -D sonar.login=admin -D sonar.password=admin password
   ```

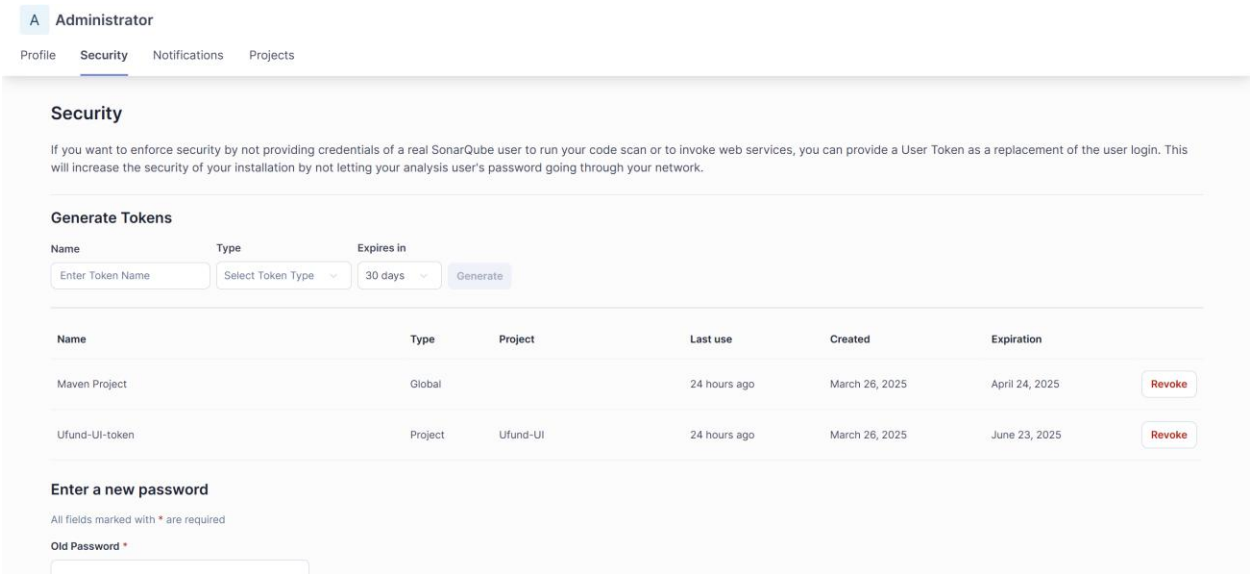
2. Instead you have to first MAKE a token by going to http://localhost:9000

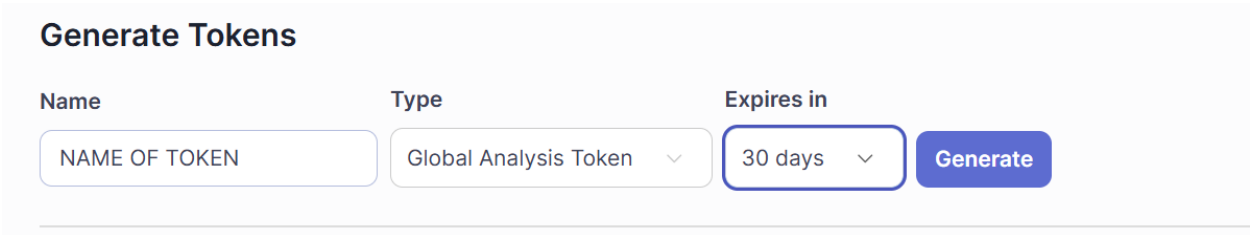
3. Click on the little **A** in the top right, and click the “My Account option”
4. That should bring you to this page:



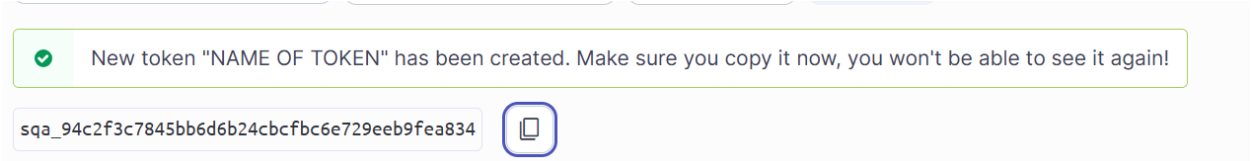
5. Click on the “Security” tab. **NOTE**: there probably won’t be any token initially



6. Fill in the token parameters in the Generate Tokens section. Select “Global” type of token. Then click on “Generate” as shown below:

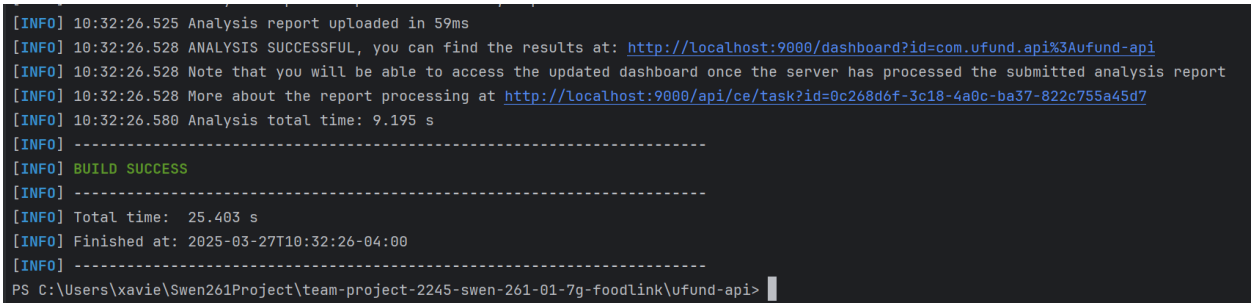


7. This should produce a new token as such:

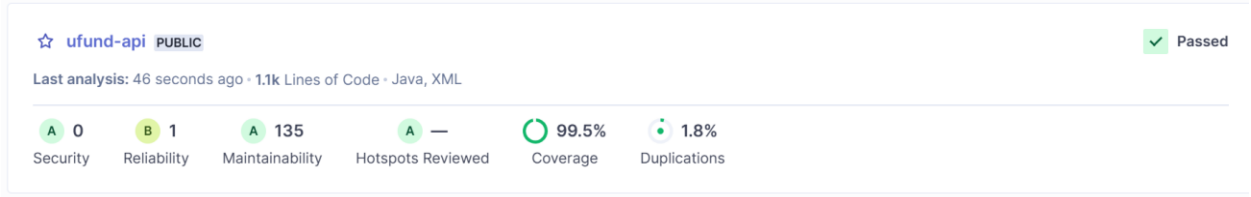


8. **COPY THE TOKEN** as you otherwise won't be able to recover it in the future
9. **Return to your terminal** and enter **YOUR TOKEN** at the end of the command and run it!

10. This will execute all unit tests followed by SonarQube. You should see something similar to below:



11. Go back to http://localhost:9000 and you should see your SonarQube report for project called Ufund-Api

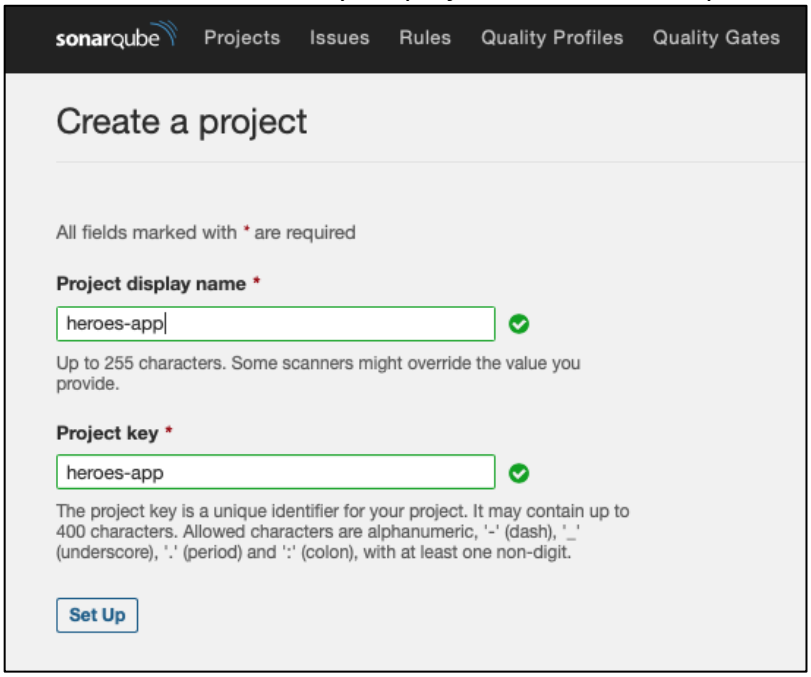


12. It might take a while to analyze it the **first time**, **so be patient** if docker doesn't show you the project immediately

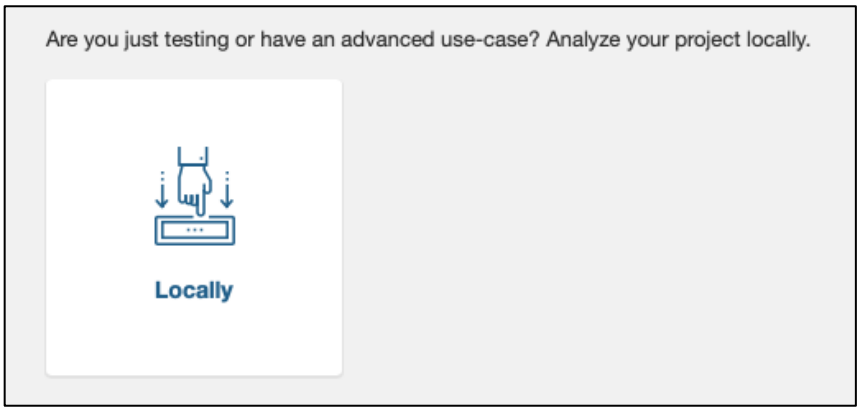
Running SonarScanner

Next, we will run SonarScanner to analyze our TypeScript code

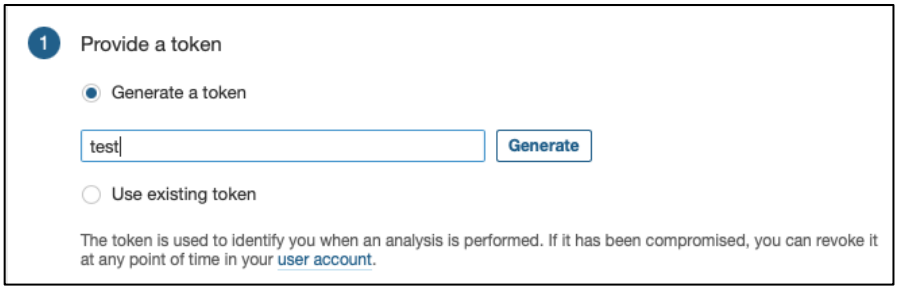
1. In SonarQube, click the “Create Project” button in the upper right corner and select “Manually”
2. Enter a name for your project and click “Set Up”



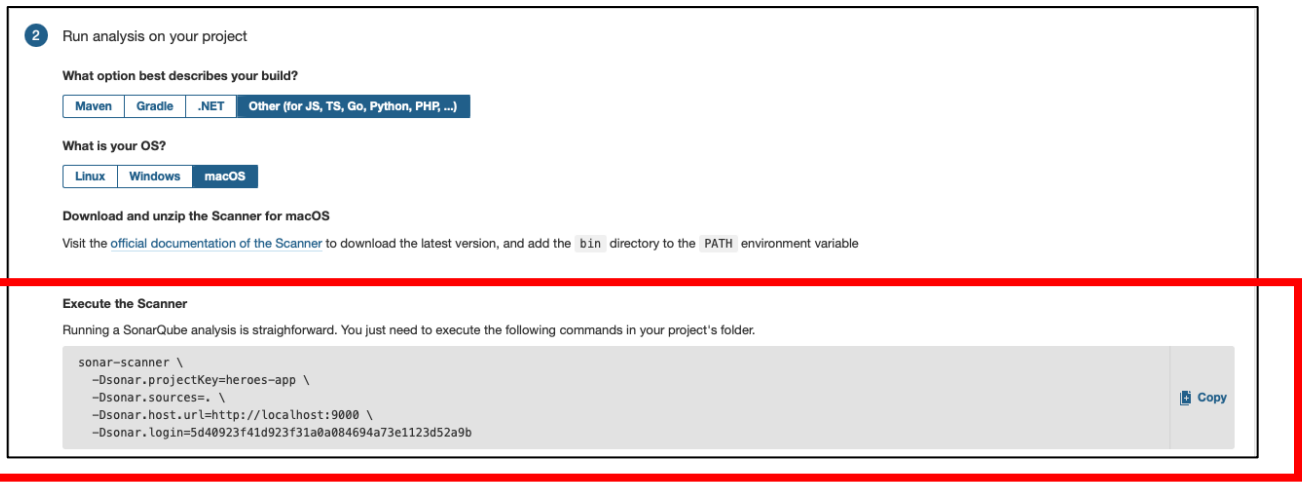
13. Select “Locally”



14. For **#1 Provide a Token**, Enter any name for your token and click “Generate”



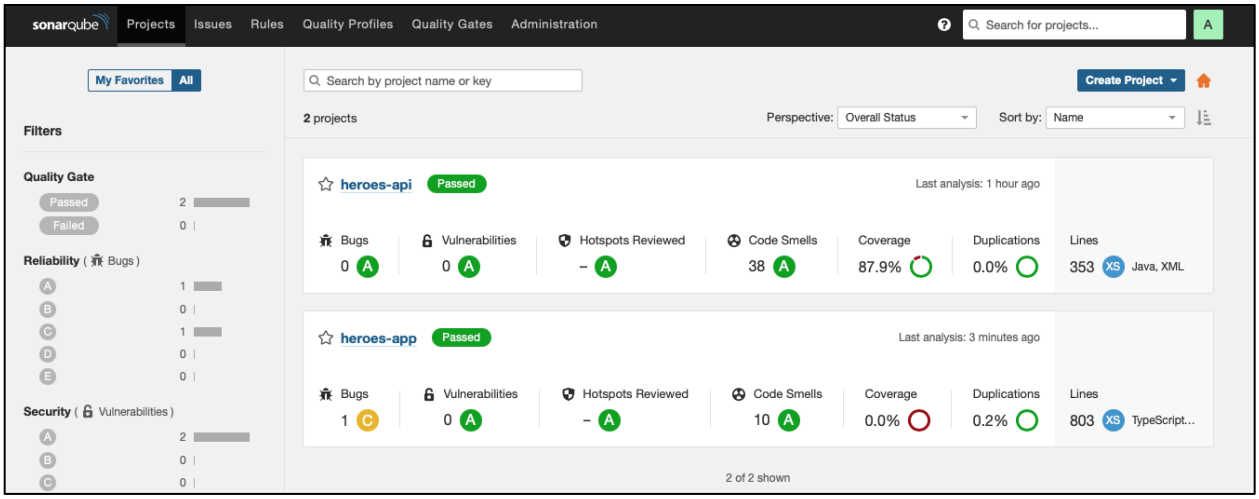
15. Click “Continue”
16. For **#2 Run analysis on your project**, select Other (for JS, TS, Go, Python, PHP, ...)
17. Select your OS
18. You should see something similar to below:



19. Copy the command under “Execute the Scanner” and go to the directory where your TypeScript code resides and paste this to run. You should see something similar to below:

```
INFO: CPD Executor 9 files had no CPD blocks
INFO: CPD Executor Calculating CPD for 24 files
INFO: CPD Executor CPD calculation finished (done) | time=19ms
INFO: Analysis report generated in 93ms, dir size=1.2 MB
INFO: Analysis report compressed in 197ms, zip size=317.4 kB
INFO: Analysis report uploaded in 181ms
INFO: ANALYSIS SUCCESSFUL, you can browse http://localhost:9000/dashboard?id=heroes-app
INFO: Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
INFO: More about the report processing at http://localhost:9000/api/ce/task?id=AX3J217VfDrwyKT3wXYQ
INFO: Analysis total time: 42.204 s
INFO: ------------------------------------------------------------------------
INFO: EXECUTION SUCCESS
INFO: ------------------------------------------------------------------------
INFO: Total time: 43.154s
INFO: Final Memory: 14M/57M
INFO: ------------------------------------------------------------------------
```

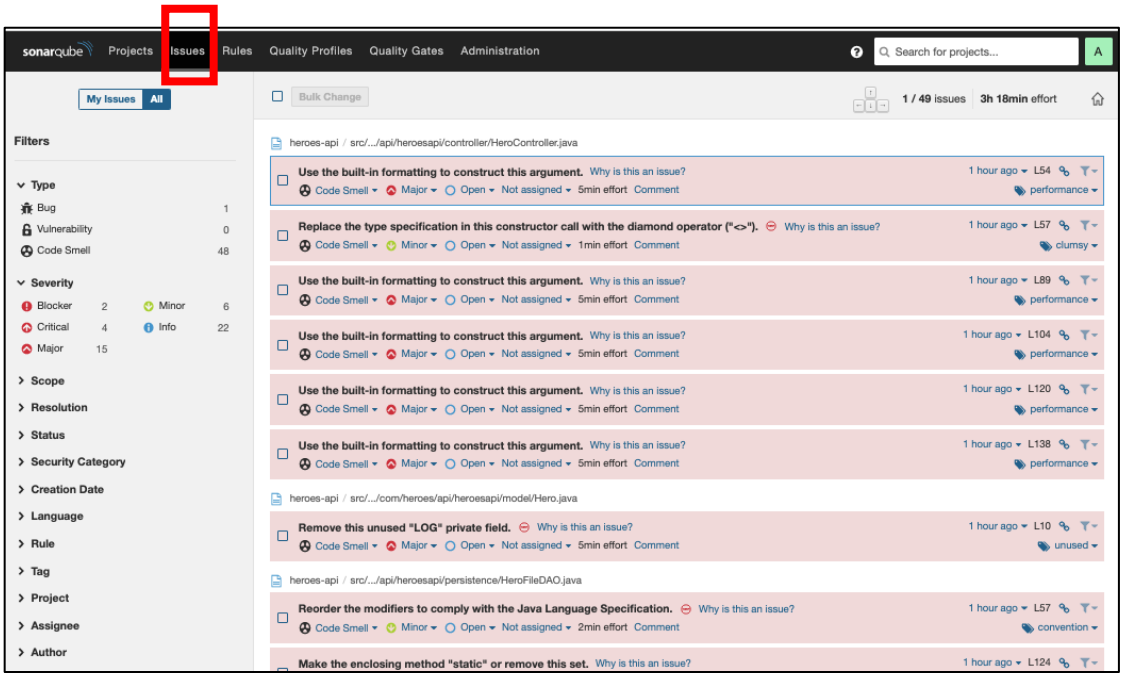
20. Go back to the SonarQube web page and you should see both your projects listed.



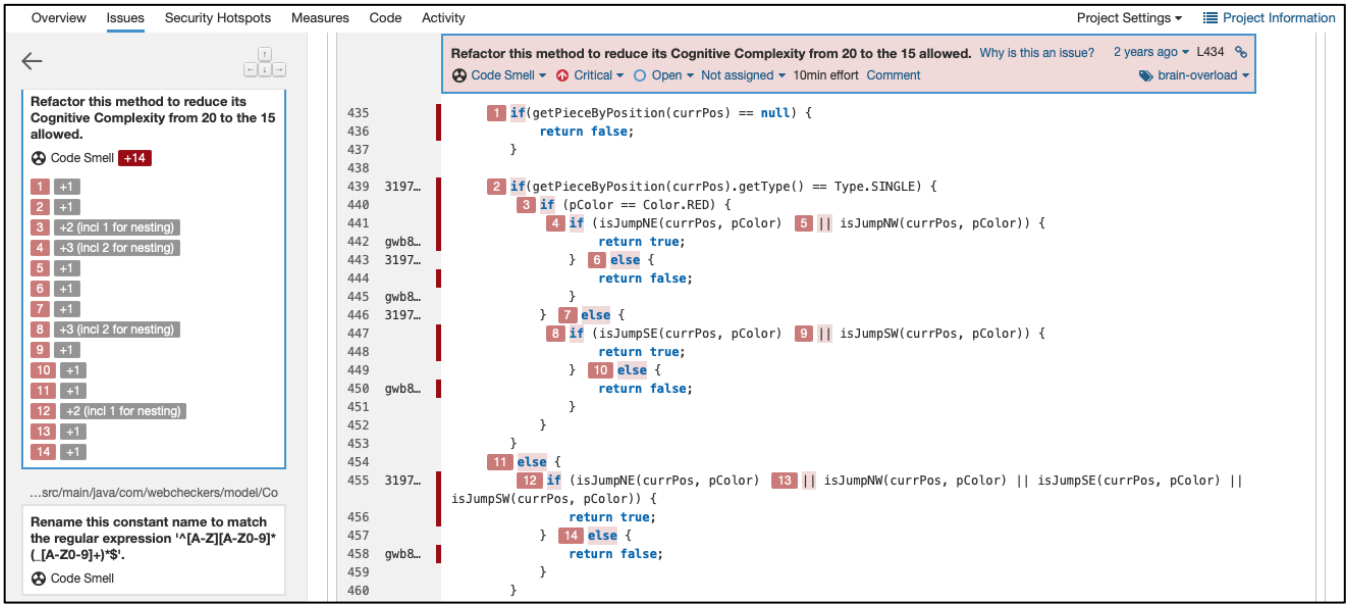
Take a screen shot of the projects (similar to above) and deposit it in the *Static Code Analysis - individual* in the myCourses Assignments by the date shown on your section's schedule.

Explore and Analyze Your Reports

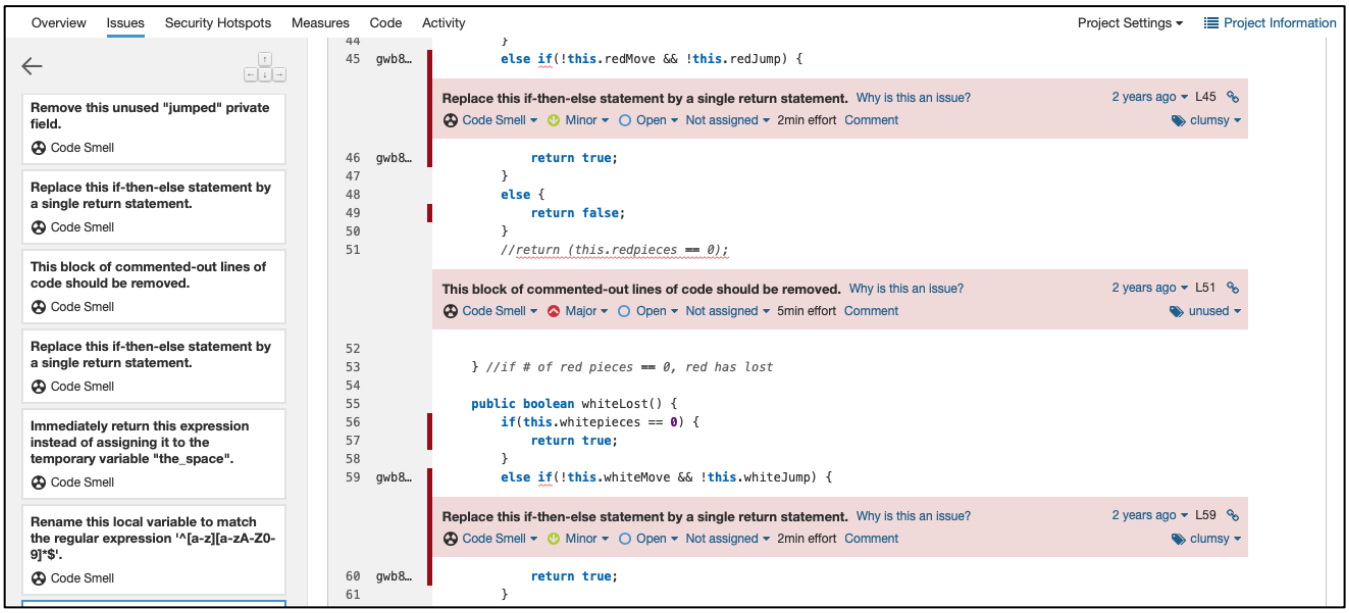
Every report will be different based on what the code analysis identified. Clicking on “Issues” (top of screen – see below) will list all the issues across your projects, which can then be categorized by severity (e.g. major, critical, blocker) and type (e.g. bug, vulnerability).



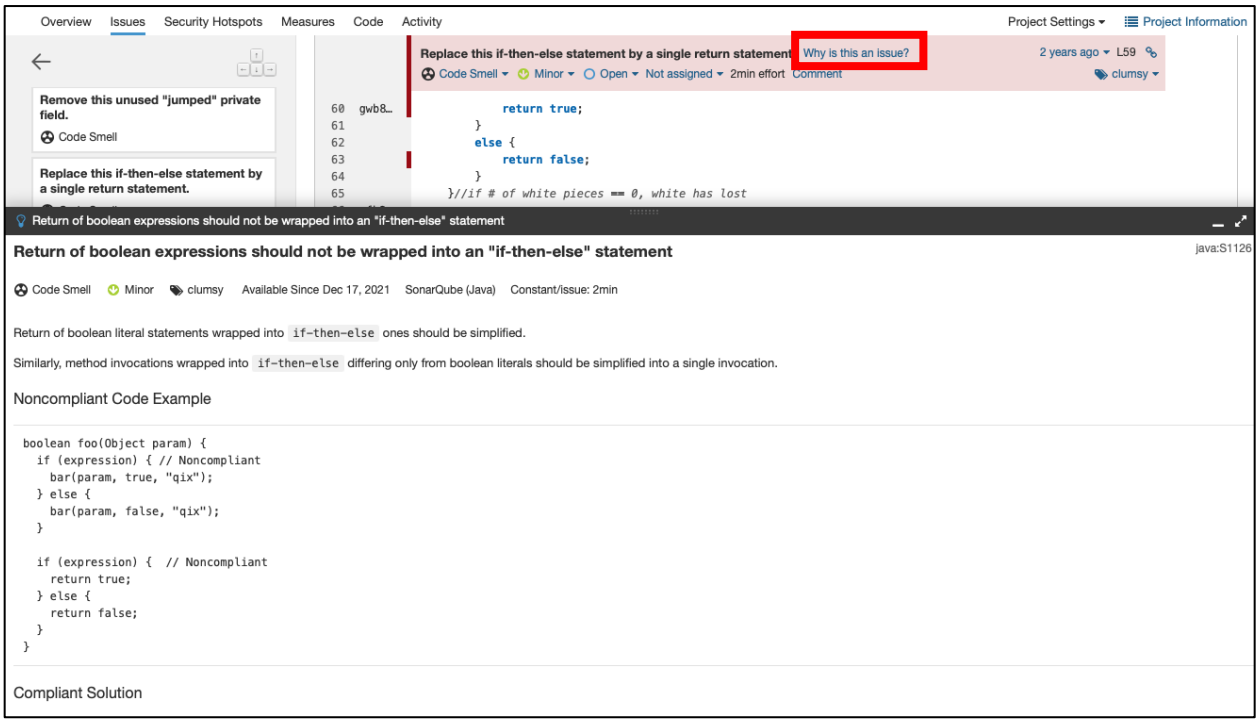
Depending on the complexity of your code, certain metrics like **Cognitive Complexity** might get triggered, requiring attention for potentially refactoring:



Other “code smells” may get flagged due to issues with readability or other factors. This may indicate a problem or possibly a false flag.



If a particular issue is not clear, click the “Why is this an issue?”, which will provide a description of the issue including code examples of a non-compliant issue and a compliant solution.



Project Sprint 4 Deliverable

Identify 3-4 areas within your code that have been flagged by SonarQube and provide your analysis and recommendations. Include any relevant screenshot(s) with each area. This will be part of your final design documentation in your Sprint 4 submission. Be sure to include at least one from both reports (Java and TypeScript).

Stopping SonarQube

Refer to the setup document to stop SonarQube.