Secure Coding Tools: An Analysis

A Report by

Mark Kardash

SDEV360: Secure Software Engineering

Professor David Gabello

30 November 2023

University of Maryland Global Campus

Security is, without doubt, one of the most important components in software development. Attackers have countless ways of infiltrating an application, which is why equally countless defenses are needed. One of the most effective ways to safeguard a digital product from attacks is to secure the very code used to write it. According to Dev (2019), “Secure coding is the process of developing code in a way that ensures security and eliminates vulnerabilities.” (What is Secure Coding?, para.1). By detecting and eliminating vulnerabilities in the code, developers ensure safety of a program “from within”, thereby reducing the number of protection measures that may need to be taken for other, non-code components. To quickly and efficiently handle code vulnerabilities, a variety of useful secure coding tools have been developed. They simplify the work of programmers by doing some of the security tasks for them. These tools can either be language-specific, or multi-language (Dev, 2019). They also can be either free-to-use or require payment. Their tasks include anything from code review to automation to application scanning (Dev, 2019). In this report, a brief overview of several secure software tools will be conducted, with one of them being selected for a more in-depth analysis.

After some research, several notable and useful security tools were found, all of them multi-language, and each having its own approach to software. For example, Code Warrior is a tool working through a user’s web browser, suitable for MacOS, Linux, and BSD. It uses the HTTP protocol to scan an application for issues, along with Transport Layer Security (TLS) (Dev, 2019). TLS is a protocol for encrypting incoming and outgoing data to keep communication in applications safe (Jane, 2023). When examining an application, Code Warrior detects vulnerabilities by following a set of built-in rules that cover important secure coding standards. It also gives users the ability to create custom rules. Languages supported by Code Warrior include JavaScript, Java, Ruby, C, C#, and PHP (Dev, 2019).

Another valuable coding security tool is DependencyTrack. Its uniqueness lies in its ability to detect third-party components in an application. This allows developers to spot any malware or malicious pieces of code that may have been injected by hackers. Another advantage of this tool is that a programmer can use it on both the applications that they created themselves, and those that they may simply be using (Dev, 2019). It has many different features for things like LDAP and Active Directory integration, centralized tracking, and webhook notifications. It can analyze vulnerabilities and foreign components to determine their impact on an application. DependencyTrack is also suitable for integration with several databases, such as Sonatype OSS Index, the National Vulnerability Database (NVD), and VulnDB (Dev, 2019).

Some secure code tools detect vulnerabilities through testing of a program. For example, Checkmarx is a platform on which teams could perform security and application security testing to find software bugs, while also performing decomposition analysis to cover every possible corner of the program (Zelleke, 2022). To gain more experience in keeping code secure, developers can use Embold. This tool does not so much detect threats as it educates programmers on how to do so. Embold offers visual demonstrations of weaknesses to inform them of best practices for securing their code (Zelleke, 2022).

Finally, SonarQube is a great tool to measure and enhance the quality of a program by exposing weaknesses in its source code (Dev, 2019). This report will analyze SonarQube in more detail, demonstrating its functionality through a real code sample.

**SonarQube: A Pros and Cons Analysis**

SonarQube is an open-source tool that measures the quality and safety of a user’s code (Dev, 2019). It was developed by Swiss software company SonarSource, to uphold their mission of offering developers “Clean Code” that can be used without complications (SonarSource-About, n.d.). SonarQube is completely automated, and can be used with over 30 programming languages. The tool works by integrating into the workflow of a program, and inspects its code, finding any threats or vulnerabilities (SonarQube, n.d.). It then performs a rank of all issues found, based on how severe they are. Among the wide variety of features that SonarQube possesses is the option to view a project timeline, which helps developers track their progress without getting lost. Other features include the ability to analyze pull requests, and code branch tracking (Dev, 2019).

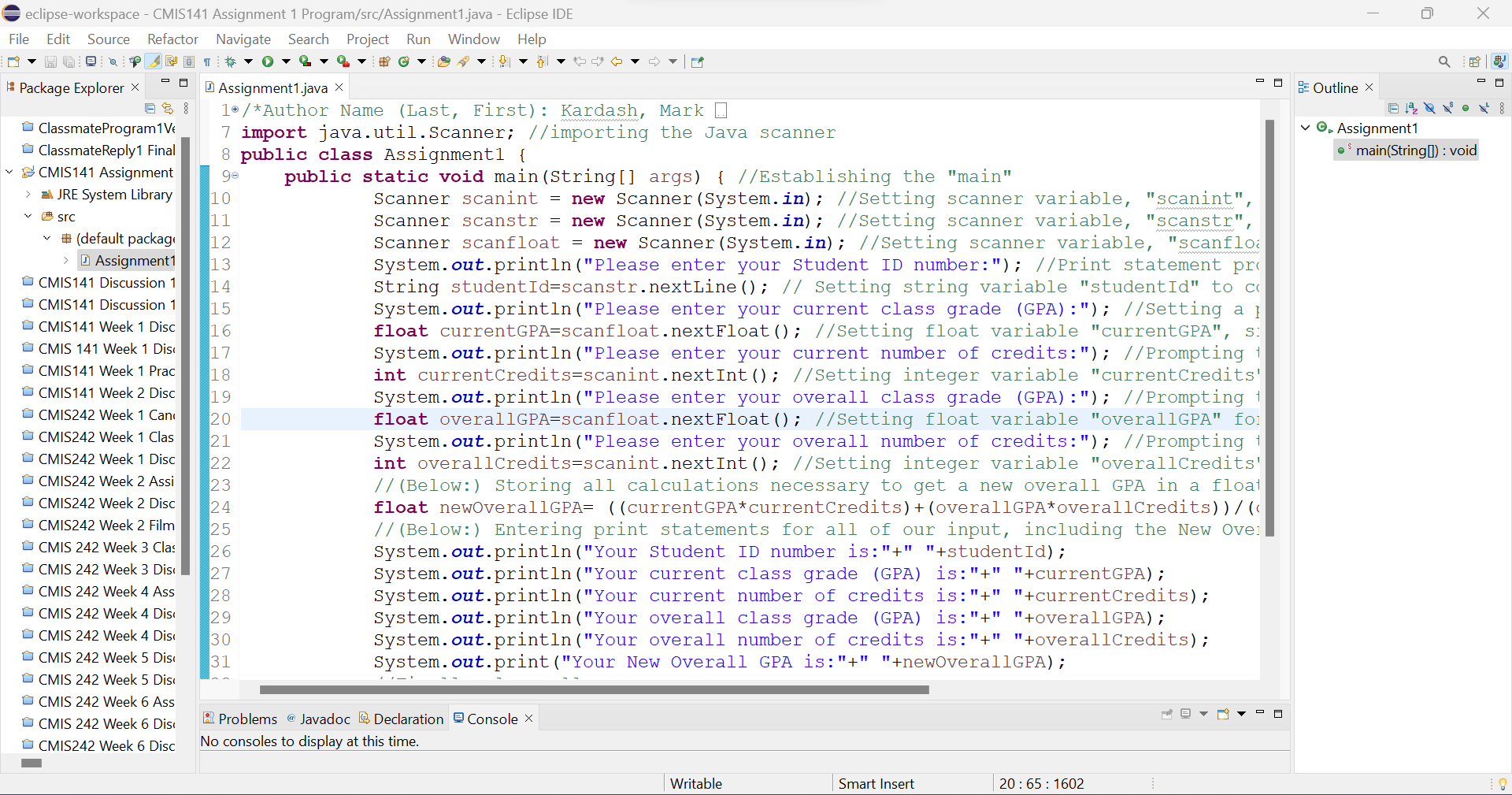
There are many advantages to using SonarQube in software development. In addition to its already mentioned capability of being used with different computer languages, and open availability, it can also be integrated into DevOps platforms, which makes the DevOps process more automated and easier. Continuous integration tools, such as Jenkins, also allow for the use of SonarQube, so that issues are detected and repaired while the application build is still in progress. SonarQube’s simple and interactive user interface also makes it a great tool for beginning developers (Dev, 2019). This is helped by the fact that it offers a description of the issues detected, and then suggests tools to fix it (Malhotra, 2020). This ensures that a bug or threat will not be misinterpreted, avoiding any further complications. Its effectiveness in error detection increases team productivity, as it can be used to eliminate any redundant or duplicate code. By providing valuable feedback and quality repair measures, SonarQube educates developers on their craft, giving them a standard to aim for, and enhancing their skills (FoxUTech, 2018). By defining requirements, providing services for vendor support, and allowing for management of code quality, SonarQube can scale with a company’s business needs, providing for a future of innovation (FoxUtech, 2018).

Although the advantages of SonarQube outweigh its disadvantages, the tool has a few weaknesses that cannot be ignored. These are mostly related to delays in setup and startup. For example, when used on the software system Kubernetes, SonarQube may take much longer to start if a special feature called “persistency” is disabled. This is because certain features of Kubernetes need time to become fully operational, and certain features must be adjusted to match the user’s system (SonarQube: Deploy SonarQube on Kubernetes, n.d.). Overall, the tool’s behavior and startup time generally depend on the systems or platforms it is being used on. Developers will have to approach the problem individually, based on each system’s specifics, minimizing the production delay it could potentially cause.

**SonarQube: A Hands-On Demonstration of Functionality**

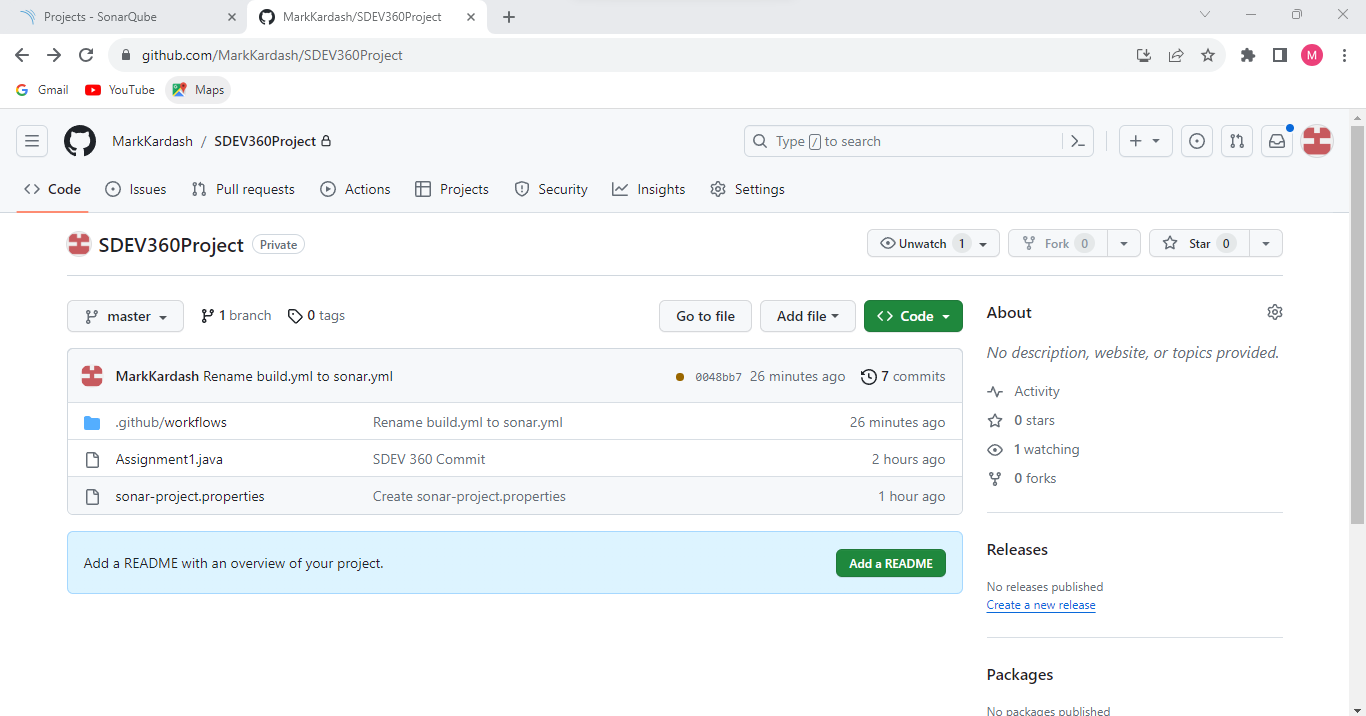
While positive descriptions and recommendations certainly benefit any product, the best way to check the effectiveness of a tool like SonarQube is to test it through a real-world example. For this report, a sample of code in the Java programming language has been obtained from the CMIS 242 course, previously taken by the author, which will be tested for vulnerabilities using SonarQube. The results will be analyzed and discussed.

To test out SonarQube, the code from the following program for calculating overall student GPA, minus the comments, will be inserted into the app:

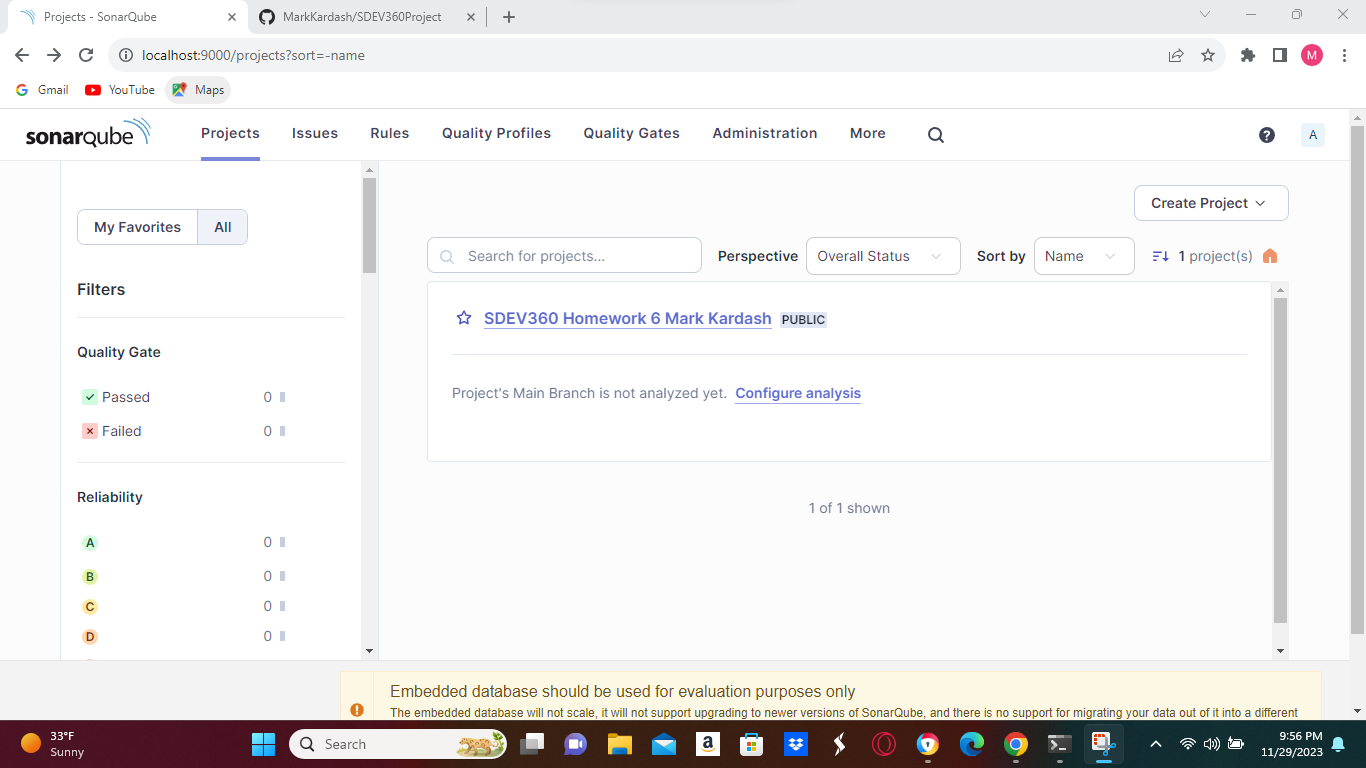




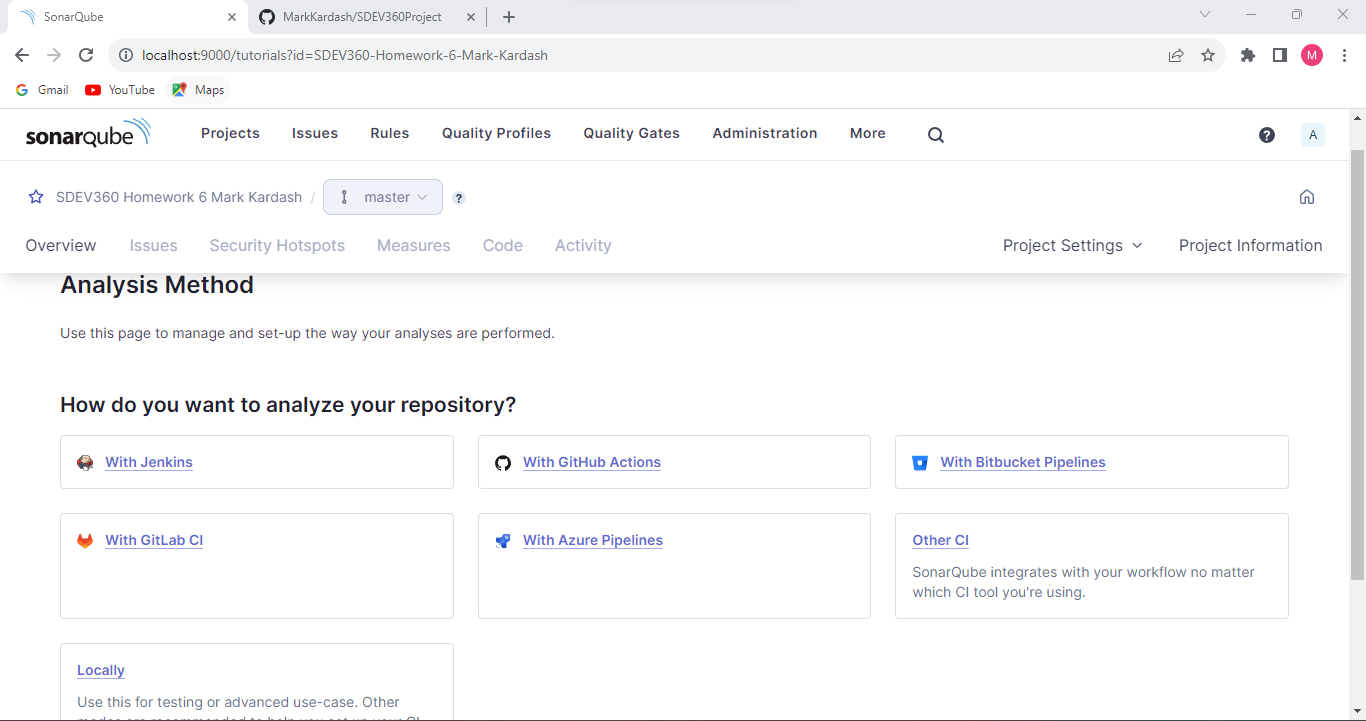
For the present situation, the analysis will be done from GitHub. A GitHub account and repository were created, and the project was pushed into it. Following a set of instructions from SonarQube (accessed on localhost:9000, two files: a “.yml” file and a “sonar\_project.properties” file were created. This was done to connect SonarQube with GitHub for the coming analysis. Screenshots of the process are shown below:



**Figure 1:** The files created in GitHub for connection with SonarQube

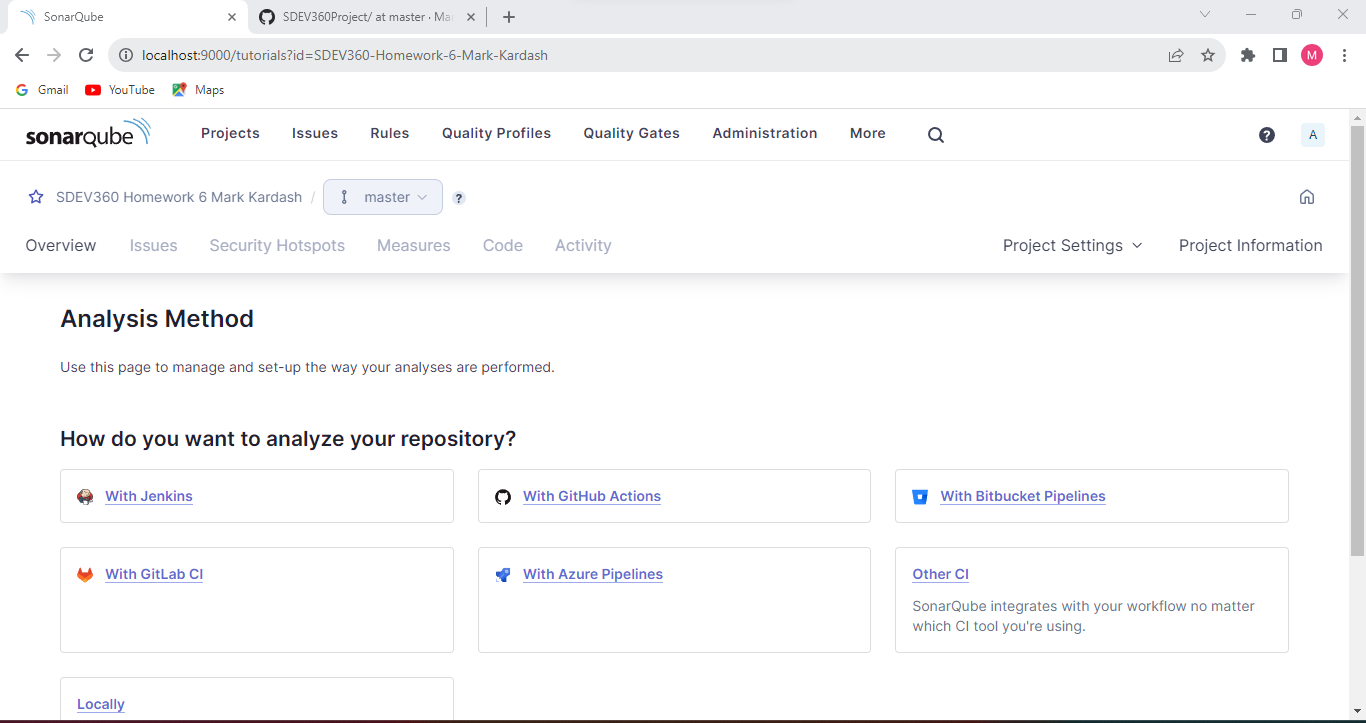


**Figure 2:** The Project to be used, “SDEV360 Homework 6 Mark Kardash”, located by SonarQube after establishing connection to GitHub repository.



**Figure 3:** User is given the option of analyzing the code via different ways.

From the above request, “GitHub Actions” was selected. To trigger the analysis, it was necessary to “push” the project into the repository on our GitHub Master Branch. This can be done after making any changes to the repository. A new file called “Test File” was created with the experimental purpose of making a change to create a new analysis. After multiple failed attempts, an analysis still did not occur, with the “Issues” and “Measures” buttons, meant to show any bugs and proposed fixes, respectively, remaining disabled.



**Figure 4:** As can be seen from the above screenshot, the “Issues”, “Security Hotspots”, “Measures”, and other sections that were supposed to show analysis results were disabled.

Security tools are great allies in detecting threats and vulnerabilities in applications. Despite failing at demonstrating a hands-on example of SonarQube use, much was learned about the tool and others like it, giving good knowledge for future projects.

References

*7 open-source tools for securing your code*. (2019, November 15). DEV Community. <https://dev.to/eddiesegal/8-open-source-tools-for-securing-your-code-4gm5#:~:text=7%20Open-Source%20Tools%20for%20Secure%20Coding%201%20Juice,...%206%20Metasploit%20Framework%20...%207%20SonarQube%20>

*About - Sonar and SonarSource*. (2021, May 15). Clean Code: Writing Clear, Readable, Understandable & Reliable Quality Code | Sonar. <https://www.sonarsource.com/company/about/>

*Deploy SonarQube on Kubernetes*. (2021, May 15). SonarQube 10.3. <https://docs.sonarsource.com/sonarqube/latest/setup-and-upgrade/deploy-on-kubernetes/deploy-sonarqube-on-kubernetes/#known-limitations>

Jane, R. P. (2023, January 16). *What is TLS and how does it work?* TechRadar. <https://www.techradar.com/features/what-is-tls-and-how-does-it-work>

Malhotra, S. (2023, July 7). *Why SonarQube: An introduction to SonarQube and its benefits*. DevOps Blog - VegaStack. Retrieved November 30, 2023, from <https://vegastack.com/blog/why-sonarqube-an-introduction-to-sonarqube-and-its-benefits/>

Simplilearn. (2022, August 30). *Top 30 SonarQube interview questions and answers for 2024*. Simplilearn.com. <https://www.simplilearn.com/sonarqube-interview-questions-answers-article>

*Sonar*. (n.d.). CBInsights. Retrieved November 30, 2023, from <https://www.cbinsights.com/company/sonarsource>

*SonarQube 10.1*. (n.d.). SonarQube 10.1. <https://docs.sonarsource.com/sonarqube/latest/>

*What are the benefits of SonarQube*. (2018, May 14). FoxuTech. <https://foxutech.com/benefits-of-sonarqube/>

Zelleke, L. (2022, November 10). *The 7 best secure code training tools for 2023 (Paid & free)*. Comparitech. <https://tmp.comparitech.com/net-admin/best-secure-code-training-tools/#:~:text=The%20seven%20best%20Secure%20Code%20Training%20Tools%201,Checkmarx%20scan%20history%20showing%20vulnerabilities%20and%20issues%20>