Lipan Andreea-Alexandra
UBB FMI Software Engineering Master
9 March 2025

SonarQube analysis report on my bachelor thesis

My bachelor thesis application

It is an Android application, written in Kotlin. It implements an Android launcher and functionalities such as icons customisation, share screen with other app users, and remote control in app. Also uses speech to text API.

Sonarqube

SonarQube is a Code Quality Assurance tool that collects and analyzes source code, and provides reports for the code quality of your project. [1]

Results of using the tool & Interpretation

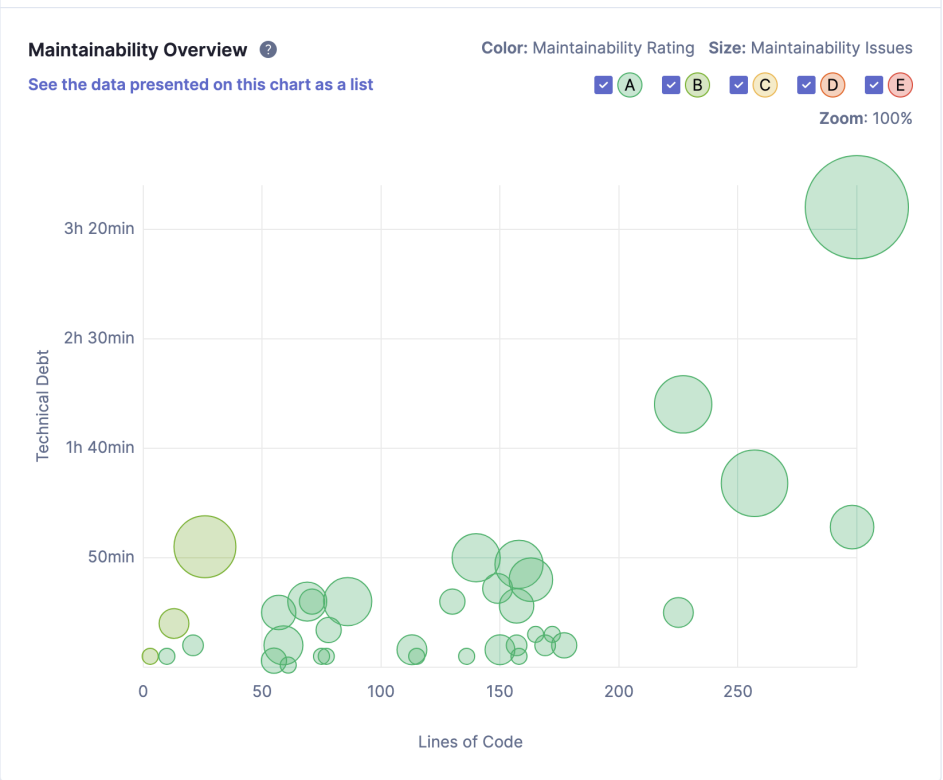
The tool has identified the following issues with the code:

Type	Issues identified	Interpretation
Clean Code	Hardcoded values (dependencies, or other constants)	I was not aware you can use variables inside dependencies
	No order in the dependency file	Agreed, the tool showed the way the file would look organise and it is indeed easier to read
	LOST of forgotten commented lines (15% of the total project)	This was definitely shocking. I was aware i had some leftover commented code but i didn't expect it to be 1000 lines worth of it
	Lots of unused imports and local variables	This is fair and i will be more careful about it;
	Use of deprecated code	It had a reason, it was the only version of code that was working at the time of writing the code
	Forgotten todos, and empty (unfinished) functions	

	Lots of leftover debug prints	
	Functions with way too many parameters	Had no idea 7 was the recommended maximum
	Really high complexity on some functions	Didn’t even realise how complex some functions were until the tool pointed them out
	Useless null checks	
Android specific issues	Exposing mutable flow types;	I had no idea these are bad practices. It is really nice that the tool could point out such problems. Great to learn
	Change from mutable to immutable for some vars for ease of readability;	
	Not implementing permissions on exported components;	
	Hardcoded dispatchers; views should not trigger coroutines;	
Security	Using usesCleartextTraffic enabled leads to sensitive data exposure	
	Didn't check if intents are received safely which has led to known vulnerabilities in the past [2]	
	Use of possibly dangerous permissions (such as RECORD_AUDIO, QUERY_ALL_PACKAGES, CAMERA)	These had a purpose, the app really needed these to function and the user was clearly informed of it

The bubble chart shows the classes that have issues that need to be resolved. It is really interesting to visualise, because I was aware of like 2 or 3 of them but the tool found soo many more. And they don't seem like they are really hard to fix problems, but i’d like to point out that there’s lots of them, so combined it would take quite a while to fix this app. All of this could’ve been reduced if the tool was integrated in the development process.

Lipan Andreea-Alexandra
UBB FMI Software Engineering Master
9 March 2025



Advantages of using this tool

I found some issues that I would've never thought would be problematic until I used this tool and read what it had to say. I could've seen the unused imports problems myself or the forgotten comments and prints, but i couldn't have seen all the android specific principles that i accidentally broke. Because I had no idea they existed.

Also I appreciate the explanations the tool gives and how it shows ways to fix things. I expected it to only point out problems but the fact that it gives solutions is incredibly useful especially on those issues i didn't know were issues.

Plus the tool was also pretty easy to set up compared to two others I have tried (and failed to make them function).

Disadvantages of using this tool

Not everything the tool points out is a problem, IS an actual problem. I definitely don't think such tools should be taken at face value.

The tool did point out some decent issues and even one android specific architectural problem, which are all valuable. But still, I already know this app has a pretty big architectural flaw (some of the classes are way too complex for what they try to do) which

makes maintenance for it really hard (it is the reason I have been dreading going back to fix this code). This analysis did not manage to point that out. It also offers no analysis as to how interconnected different modules are, which from what I have learned last semester at SSI, is one of the biggest problems when it comes to maintaining software long term.

Conclusions

I was skeptical at first because, yes of course I thought my project had a few issues but no big deal. It turns out it had lots of tiny issues which I didn't even know were problems. So I think using such tools at least once in the development process would help a lot. But combined with a tool that analyzes intermodular dependencies to really help make this app as easy to maintain as possible.

Bibliography

https://www.devopsschool.com/blog/what-is-sonarqube-and-how-it-works-an-overview-and-its-use-cases/#What_is_SonarQube
https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-1677