**Reflection:**

We had faced multiple issues at the beginning of code analysis; the project was big, the PHP language was new, and as well the code analysis. Based on the tools suggested by the professor, the task became easy and we also had to rely on a few online links for installation and setup purposes.

Initially, we were in a dilemma on how to approach code analysis as PHP is a new language for all the team members. Having said that we started to go through those tools which the professor has listed as part of the lecture and identified the tool which best suits for our PHP project. We found SonarQube supports PHP and started to dig into it, which almost took two hours for the setup and 9 hours to complete execution. Parallelly few of the team members were doing manual code reviews using issues reported in the actual Magento GitHub website based on our earlier understanding from misuse case and assurance case assignments. Since we were using SonarQube for the first time we had faced multiple issues while executing. We referred to their website and few YouTube links for installation, setup, and any language-specific rules. The first time when we executed it, even though the code analysis execution was successful in the backend, the frontend dashboard was throwing some errors and not reflecting the analysis report. Hence, we had to do a fresh setup once again from the beginning and run the execution again. Finally, we were able to see the issues on the dashboard this time. Since we used the community edition, we weren’t allowed to download the analysis report as it was supported only for the enterprise edition, and hence had to come up with a way to share the analysis report with team members. Tool analysis reflected around 4.5K issues, so we first analyzed major issues reported for security and explored more using CWE and CVE.

For manual analysis, as mentioned earlier since everyone is new to PHP and because the codebase was huge, we had to struggle a bit to analyze the code flow and the strategy was a bit vague. We later set up a proper code review strategy of going through issues reported on GitHub, identified related CWE’s, and CVE to explore more from the codebase, after the team’s check-in with Professor.

Additionally, we also explored a few more automated tools available online; Codacy and Sonar Cloud, to validate whether all tools were reporting similar issues. Codacy free version was not so good and the whole team had to rely on SonarQube analysis which wasn’t easy to share as mentioned earlier.

We learned multiple things from this project, as to what is code analysis and its significance; the varieties of automated code analysis tools and how to use those; and why manual code review is important. Professor’s instructions on manual and automated code analysis helped us to move forward with our assignment. Also, we understood together as a team, we can make uneasy tasks easy and we enjoyed working together. Our regular team meetings, planning and peer review on segregated tasks helped us to meet this milestone on the given timeline.