E-ISSN : 2540 - 8984

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Volume 07, Nomor 03, September 2022 : 728–734

IMPLEMENTING CONTINUOUS CODE QUALITY FOR CODE QUALITY DEVELOPMENT IN THE SCRUM TEAM

Ilham Izzul Hadyan[1)], Dana Sulistyo Kusumo[2)], Jati Hiliamsyah Husen[3)]
[1, 2, 3)]School of Computing, Informatics, Telkom University
Jl. Telekomunikasi No. 1, Bandung, Indonesia
e-mail: ilhamizzul@student.telkomuniversity.ac.id[1)], danakusumo@telkomuniversity.ac.id[2)],
jatihusen@telkomuniversity.ac.id[3)]

ABSTRAK

Terdapat tantangan dan kekurangan yang akan dihadapi dalam implementasi metode Scrum. Salah satu permasalahan ada dalam Code Quality karena tim hanya memiliki waktu yang singkat dalam setiap sprint. Hal ini menyebabkan tim tidak bisa melakukan code review secara menyeluruh, mengakibatkan tim harus menambah beban dan tekanan kerja mereka. Code review diperlukan untuk mengidentifikasi kecacatan dalam kode yang ditulis sebelum masuk kedalam inti proyek. Tetapi sifat rumit, memakan waktu dan tenaga dalam code review menyebabkan penghambatan dalam mengadopsi praktik code review dengan benar. Untuk mempermudah dan mempercepat code review, peneliti akan mencoba untuk mengimplementasikan Continuous Code Quality (CCQ) dengan proses Scrum dengan melakukan eksperimen secara empiris. Dengan meneliti bagaimana efek dalam kecepatan proses dan mengetahui sudut pandang dari tim pengembang. Berdasarkan eksperimen yang dilakukan, sistem automasi CCQ membantu tim pengembang dalam mempercepat proses code review dan menjaga code quality tetap bagus. pengaruh terbesar dalam kecepatan proses tetap berpaku dalam kompleksitas sistem yang dibangun dalam sprint yang berjalan. Walau dengan adanya sistem automasi code review, masih diperlukan pengecekan manual untuk memastikan tidak ada kesalahan yang tidak bisa dideteksi oleh sistem automasi.

***Kata Kunci**: Automasi, Agile, Code Review, Code Quality*

ABSTRACT

There are challenges and drawbacks that will be faced in implementing a Scrum method. One of the problems is in Code Quality because the team only has a short time in each sprint. This causes the team to not be able to conduct a thorough code review, resulting in the team having to increase their workload and pressure. Code review is necessary to identify defects in the code before it goes into the core of the project. But the cumbersome, time-consuming and labor-intensive nature of code review causes a barrier in adopting code review practices properly. To simplify and speed up code review, the researcher will try to implement Continuous Code Quality (CCQ) with the Scrum process by conducting empirical experiments. By examining the effect on the speed of the process and knowing the point of view of the development team. Based on the experiments conducted, the CCQ automation system helps the development team in speeding up the code review process and maintaining good code quality. The biggest influence on the speed of the process remains in the complexity of the system built in a running sprint. Even with an automated code review system, a manual code review process is still needed to ensure there are no errors that cannot be detected by the automated system.

***Keywords**: Automation, Agile, Code Review, Code Quality*

I. INTRODUCTION

Implementation of the Scrum method has many advantages in teams[1]. Some of them are improving communication between team members, improving product quality, and being adaptable to changing requirements as the software development process progresses[2]. However, there are challenges and drawbacks that will be faced in the implementation of the Scrum method[2][3]. One of the issues is in the Code Quality[3]. In the Scrum process, teams only have a short time in each sprint because Scrum demands speed in the process of each sprint[4]. This prevented the team from conducting a detailed code review, resulting in additional workload and pressure on the team[3]. Therefore, the Code Quality of the developed product decreases and produces critical problems, such as the program is not easy to maintain[3].

Code review is needed to identify defects in the code written before it goes into the core of the project[5]. But the complicated, time-consuming and labor-intensive nature of code review causes a barrier in adopting code review practices properly[5][6]. To simplify and speed up code review, Carmine, et, al.,[7] outlined the Continuous Code Quality (CCQ) method to address code quality related issues in general and speed up the code review process. In the study conducted by Carmine, he explained how CCQ works in general. CCQ provides analysis results related to the code quality of each build performed. From the results of the research conducted, it was noted that many practitioners rarely inspect code quality. And projects that use automation processes such as Continuous Integration, it is noted that not many practitioners conduct code inspections in every build process carried out.

E-ISSN : 2540 - 8984

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Volume 07, Nomor 03, September 2022 : 728–734

Vipin[6] explained that code review can improve code quality but at the same time the code review process requires significant human labor. Vipin created Review Bot by combining several static analysis tools to perform code review automatically and conducting empirical studies. Based on the study conducted, integrating static analysis tools with the code review process can improve the quality of code review by finding issues in the source code automatically.

Reinhold et al[8] describe the Code Quality Monitoring Method (CQMM) which is similar to the CCQ method with the difference being in the monitoring process. CQMM is a systematic approach to assessing and improving the code quality of software products in the ongoing development project process. CQMM is extended with the idea to measure, evaluate and improve code quality continuously by automating repetitive processes.

The implementation of Scrum according to the Scrum guide[8] does not indicate any CCQ practices. Based on that, this research will focus on adding CCQ practices to the Scrum process and see the effect of CCQ implementation on Scrum and velocity in each sprint along the software development process by using planning poker as a way to collect story points. The research will be conducted empirically through a case study experiment in the software development of the Clinical Information System to answer the following Research Question:

RQ1: What effect does associating Continuous Code Quality practices with Scrum have on the velocity of each sprint?

RQ2: How does combining Continuous Code Quality practices with Scrum affect the performance of the development team?

II. RESEARCH METHOD

The research will be conducted empirically by raising a case study of creating a clinic information system using the Scrum and CCQ methods. The creation of a clinical information system was raised based on a case study in research conducted by Topan in the Sam Ratulagi Air Force Hospital[9]. The research will begin by conducting 2 phases of system development and continue with interviews with the development team regarding their experiences during the experiment.

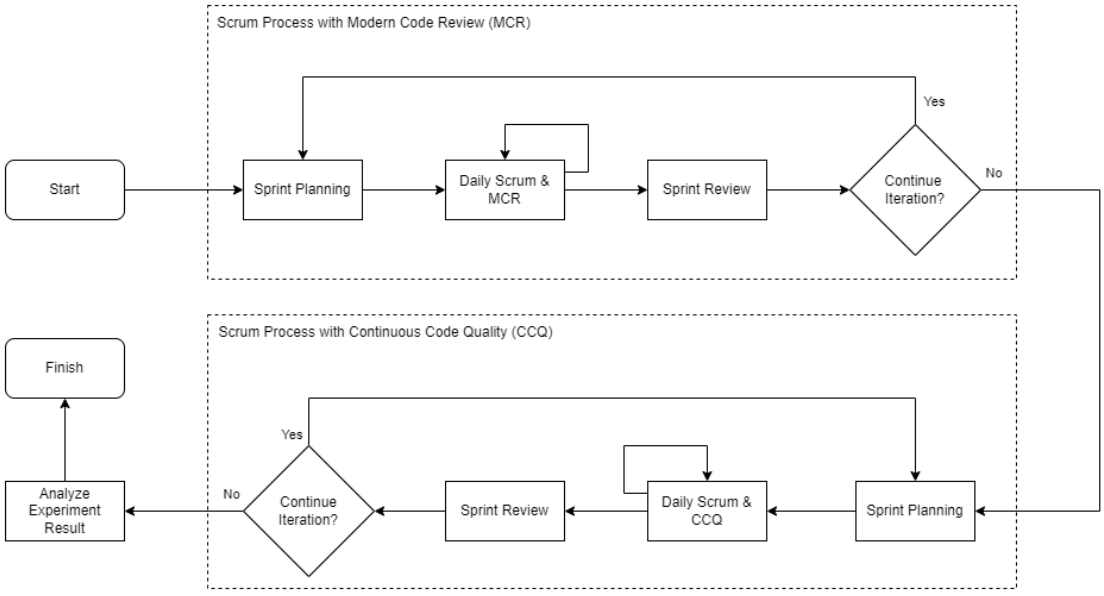

Fig. 1. Research Method Workflow

For the first phase, system development will be carried out using the scrum method in general[10] and will try to implement Modern Code Review(MCR). MCR is a lightweight process that involves few formal rules, a tendency to add support tools, and strives for more efficient and less time-consuming reviews [5][11]. The first phase will be conducted for 4 sprints with each sprint running for 1 week. This needs to be done so that the perception of the development team has the same point of view and experience in conducting code review and the scrum process. MCR will be done asynchronously when the development team submits a pull request and will be checked by another development team[11], so that modern code review can be done at any time during the sprint[5]. The results of the code review will be revised in the next sprint if the current sprint time has been finished.

E-ISSN : 2540 - 8984

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Volume 07, Nomor 03, September 2022 : 728–734

The second phase will develop the system using Scrum and replace the modern code review process with the CCQ automation process. CCQ is a continuous code inspection process by performing static/dynamic code analysis on every build performed, as a way to ensure code quality[12]. The second phase will be conducted for 4 sprints with each sprint running for 1 week. CCQ will replace the code review process with an automated system that will run with certain rules. The concept of CCQ implementation in general is mapped in fig. 4. below[7]:

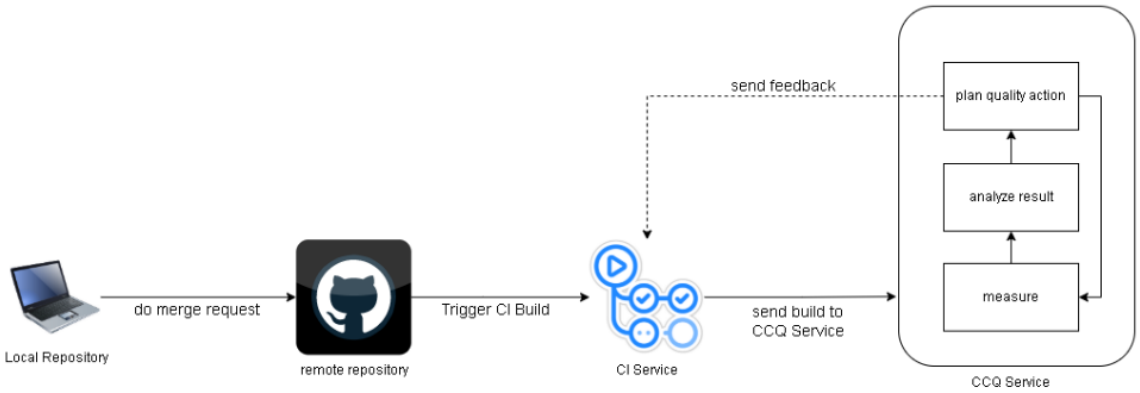

Fig. 2. CCQ Automation System Flow

CCQ has the same concept as practical Continuous Integration (CI), so to implement CCQ, a development pipeline consisting of a repository, CI build server, and CCQ service is needed[7]. To run CCQ, the development team will make a merge request to the remote repository. The remote repository must be connected to the CI service to perform automatic builds. In CI service, special configurations can be made to connect with CCQ service to perform code review. The CCQ service will continue the process of the CI service by measuring, analyzing, and determining whether the received code meets the desired quality criteria. The good or bad results analyzed in the CCQ service will be stored as historical data and will be continued back into the CI service which produces the status of the build performed.

As the experiment phase continues, story points will be recorded in each sprint planning using planning poker to see the effect of code review on the scrum process using velocity charts[13][14][15]. The determination of the story points score in the planning poker process is determined by considering all the efforts made by the development team[15], which is the effort in conducting code reviews and building applications during the sprint. Planning poker and story points are not a part of the Scrum framework[13]. Scrum framework only provides rules and freedom in estimating[13].

After the experiments have been completed, interviews were conducted with two development teams with the aim of knowing the response and perspective of the experiments that have been carried out by the development team. The interview questions directed the development team to give their opinions regarding the comparison of speed and effort made during the experiment.

III. Result & Discussion

A. Velocity Chart

Quantitative data was obtained from the results of estimating story points during the experiment using planning poker and formed into a velocity chart. The results showed significant fluctuations in the middle of the sprint due to under-estimation, unfulfilled commitments and hidden complexities[13][16]. The development team has committed to a task and during the sprint process it is discovered that the work at sprint 4 to sprint 5 is too simple and at sprint 6 to sprint 7 the work is too complex.

E-ISSN : 2540 - 8984

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Volume 07, Nomor 03, September 2022 : 728–734

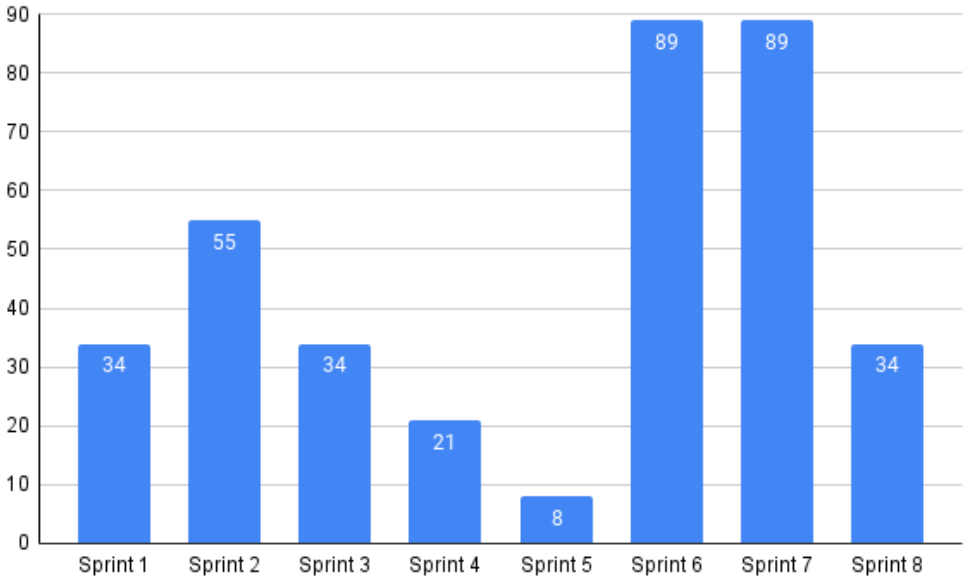

Fig.3. Story Point Results in Velocity Chart

The overly simple work in sprint 4 and sprint 5 occurred due to an inappropriate estimate. The development team has committed to a sprint by assuming that the work is within the capabilities of the development team. After the development and actual story point data collection, there was a significant decrease according to the graph listed in Fig. 5 because the tasks performed were the development of features that were relevant or almost the same as the features that had been built in the previous sprint, resulting in the development team completing the work tasks faster and long before the sprint review activities were carried out. Whereas in sprint 6 to sprint 7 there was a significant increase in story points due to overly complex work. The development team is faced with working on new features that have never been created before, so when taking story points there is an increase in story points and work that has not been completed in sprint 6 will be continued in the next sprint which results in sprint 7 experiencing an increase in story points. To handle these problems, a scrum master is needed to be more descriptive, more active in communication with the team, and can direct the project more precisely and correctly.

.

RQ1: What effect does associating Continuous Code Quality practices with Scrum have on the velocity of each sprint?

Based on the results of quantitative data obtained from the experimentation process, RQ1 can be concluded that the biggest influence on the speed of the process in a sprint remains in the complexity of the work performed. The CCQ process still affects the speed of the code analysis process before the system is released into production. This is proven in the data comparison between Table. I and Table. II which shows the comparison of the time required in the code review process carried out by Modern Code Review and using Continuous Code Quality (CCQ).

TABLE I
MODERN CODE REVIEW TIMES

Repository	**Time Needed to Modern Code Review**			
	Sprint 1	**Sprint 2**	**Sprint 3**	**Sprint 4**
Backend	± 5 Minutes	± 5 Minutes	± 8 Minutes	± 15 Minutes
Frontend	± 30 Minutes	± 15 Minutes	± 10 Minutes	± 15 Minutes

TABLE II
CONTINUOUS CODE QUALITY TIMES

Repository	**Time Needed to Continuous Code Quality**			
	Sprint 5	**Sprint 6**	**Sprint 7**	**Sprint 8**
Backend	1 Minutes 20 Seconds	1 Minutes 4 Seconds	1 Minutes 25 Seconds	1 Minutes 12 Seconds
Frontend	1 Minutes 32 Seconds	2 Minutes 53 Seconds	1 Minutes 42 Seconds	1 Minutes 40 Seconds

E-ISSN : 2540 - 8984

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Volume 07, Nomor 03, September 2022 : 728–734

B. Interview

Before the interview was conducted, the researcher designed 5 interview questions to find out the point of view of the development team during the experiment.

TABLE III
INTERVIEW QUESTION

Question Code	Question Asked
Q1	What are the factors that make the work in a sprint hard?
Q2a	How is the velocity and effort in the modern code review process running in sprints 1-4? Was there an increase/decrease in code quality management?
Q2b	How does the modern code review process affect the effort put into a sprint in general? Is there an increase or decrease in terms of effort?
Q3a	How do you feel about the speed and effort of the automated code review process with Continuous Code Quality that runs in sprints 5-8? Was there an increase/decrease in code quality management?
Q3b	How much does the automated code review process affect the effort put into a sprint in general? Is there an increase or decrease in terms of effort?

The interview questions will be used in interviews with two development teams who have the backgrounds and tasks shown in Table. IV.

TABLE IV
NAME & ROLE OF THE DEVELOPMENT TEAM IN THE EXPERIMENT

Code	Name	Major	Jobdesk
P1	Ananda Affan Fatahilla	Computer Science	Back-end Developer
P2	Reva Doni Aprilio	Software Engineering	Front-end Developer

Interviews were conducted individually with an estimated time of 1 hour. The interviews were recorded with the permission of the development team and will be used for analysis in the research results. The results of the video recording will be transcribed in text form and will be analyzed inductively to clarify the results and analysis of the research[17].

TABLE V
RESULT OF CODE ANALYSIS WITH INDUCTIVE METHOD

Themes	Code	Quote
Efforts in Code Review	Effort MCR	in the context of code review, the task is heavy but not too heavy (P1.Q1)
		checking thoroughly is not an easy activity (P1.Q1)
		It is directly proportional to the effort required because exploration is still needed to conduct code review as well as checking line by line carefully during code review (P2.Q2a).
	Effort CCQ	So the effort put into code review decreases. (P1.Q3a)
		There is a significant reduction in effort, eliminating the need for manual checking and making the process much faster. (P2.Q3a)
		there is a decrease in code review effort, because it has been assisted by an automated system (P2.Q3b)
		there is a decrease in the effort to do code review (P1.Q3b)
The velocity of Code Review	MCR Velocity	Regarding the speed in code review, it is not so long because I am used to and know the writing style of P2 (P1.Q2a)
		Regarding speed, it cannot be compared with automatic code review, because it requires manual line-by-line checking and knowledge of what P1 has created, with an average code review process time between 15-30 minutes while exploring the language used. (P2.Q2a)
	CCQ Velocity	there is a significant increase in speed because we have used the system to check code smell, vulnerability, etc. (P1.Q3a)
		There is a significant reduction in effort, eliminating the need for manual checking and making the process much faster. (P2.Q3a)
Sprint effort in general		There is a general increase in effort at the beginning of the sprint, because it requires effort to relearn how the FE framework used works. (P1.Q2b)

E-ISSN : 2540 - 8984

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Volume 07, Nomor 03, September 2022 : 728–734

	Effort Sprint with Modern Code Review	Then, the effort began to decrease because they already knew and were accustomed to how the framework worked so that only a general review was carried out without having to learn again. (P1.Q2b) there was a general increase in effort at the beginning of the sprint, then there was a decrease in effort because the intensity of exploration to help code review began to decline (P2.Q2b).
	Effort Sprint with Continuous Code Quality	There is an increase in effort because there is a new feature development that has never been made. So that learning is needed to help review and provide solutions to P2 (P1.Q3b) there is a decrease in the effort to do code review but there is an increase in general effort due to research or exploratory factors (P1.Q3b) There is a decrease in code review effort, because it has been assisted by the automation system. (P2.Q3b) But there is an increase in effort in the development process effort, due to the factor of developing new features that have never been made. (P2.Q3b)
	Exploration Effort	Also, the biggest effort is the effort to learn something new. (P1.Q1) Throughout the project, the toughest thing usually happens when determining the method that should be used to create a feature, so it can be said to be exploratory. (P2.Q1) It requires manual line-by-line checking and knowledge of what P1 has created (target review), with an average code review process time between 15-30 minutes while exploring the language used. (P2.Q2a) there is a decrease in the effort to do code review but there is an increase in general effort due to research or exploratory factors (P1.Q3b)
Quality in code improves	Code Quality	With code review, P2 experienced quality improvement in its code structure. (P1.Q2a) With the code review, the quality of the code that I write becomes better. (P2.Q2a) There is always an increase in code quality management. (P1.Q3a) There is an increase in code quality management, as the system automatically shows and suggests improvements. (P2.Q3a)

RQ2: How does combining Continuous Code Quality practices with Scrum affect the performance of the development team?

Based on the results of qualitative data and analysis conducted in Table. V, it is found that the effort in sprints in general is more likely to be explorative of something new. Knowledge and experience in the tools used by the development team to be reviewed affect the effort in conducting code reviews. The more familiar the reviewer is with the language and tools used by other development teams, the easier it is for the reviewer to do a manual code review, and vice versa. However, with the automation of code review, the burden on the development team to conduct code review becomes lighter and the team can focus more on the system development process.

"Even with an automated code review system, there is still a need for manual checking to ensure there are no errors that cannot be detected by the automated system. (P1)"

Using an automated system like CCQ will not be enough to cover all the flaws that can be found in the code structure. Such as code efficiency, code consistency, and code components that should be reusable cannot be analyzed automatically so manual review is required.

IV. Conclusion And Future Work

In this research, an experiment has been conducted to try to use the CCQ automation method into the scrum process to determine the effect from the perspective of the development team. Based on the results of quantitative and qualitative data, CCQ has a positive response from the development team. With a system that can perform detailed checks, the development team can focus more on system development. However, a manual code review process is still needed to check the code structure such as checking code efficiency, code consistency, and code consistency.

By automating the process using CCQ, the code review process becomes faster than the manual code review process. But in the context of overall speed, the speed of the process within a sprint is still dependent and fixed on the complexity of the work done in that sprint.

This research was conducted with a development team consisting only of students. So that for further research it is hoped that this research can be carried out with a team of developers who are more professional in their fields.

Also, it is hoped that further research can maintain the balance of the velocity chart so that it is more stable and stagnant.

REFERENCES

[1] W. Mahmood, N. Usmani, M. Ali, and S. Farooqui, "Benefits to organizations after migrating to Scrum," *Proc. 29th Int. Bus. Inf. Manag. Assoc. Conf. - Educ. Excell. Innov. Manag. through Vis. 2020 From Reg. Dev. Sustain. to Glob. Econ. Growth*, no. May, pp. 3815–3828, 2017.

[2] M. Mahalakshmi and M. Sundararajan, "Traditional SDLC Vs Scrum Methodology – A Comparative Study," *Int. J. Emerg. Technol. Adv. Eng.*, vol. 3, no. 6, pp. 2–6, 2013.

[3] R. Akif and H. Majeed, "Issues and Challenges in Scrum Implementation," *Int. J. Sci. Eng. Res.*, vol. 3, no. 8, pp. 1–4, 2012, [Online]. Available: http://www.ijser.org/researchpaper%5CIssues-and-Challenges-in-Scrum-Implementation.pdf.

[4] P. Adi, "Scrum Method Implementation in a Software Development Project Management," *Int. J. Adv. Comput. Sci. Appl.*, vol. 6, no. 9, pp. 198–204, 2015, doi: 10.14569/ijacsa.2015.060927.

[5] M. Beller, A. Bacchelli, A. Zaidman, and E. Juergens, "Modern code reviews in open-source projects: Which problems do they fix?," *11th Work. Conf. Min. Softw. Repos. MSR 2014 - Proc.*, pp. 202–211, 2014, doi: 10.1145/2597073.2597082.

[6] V. Balachandran, "Reducing human effort and improving quality in peer code reviews using automatic static analysis and reviewer recommendation," *Proc. - Int. Conf. Softw. Eng.*, pp. 931–940, 2013, doi: 10.1109/ICSE.2013.6606642.

[7] C. Vassallo, A. Bacchelli, F. Palomba, and H. C. Gall, "Continuous code quality: Are we (really) doing that?," *ASE 2018 - Proc. 33rd ACM/IEEE Int. Conf. Autom. Softw. Eng.*, pp. 790–795, 2018, doi: 10.1145/3238147.3240729.

[8] R. Plösch, H. Gruber, C. Körner, and M. Saft, "A method for continuous code quality management using static analysis," *Proc. - 7th Int. Conf. Qual. Inf. Commun. Technol. QUATIC 2010*, pp. 370–375, 2010, doi: 10.1109/QUATIC.2010.68.

[9] M. Topan and X. B. N. Najoan, "Perancangan Sistem Informasi Manajemen Rumah sakit berbasis web," *J. Tek. Inform.*, vol. 6, no. 1, pp. 1–6, 2015, doi: 10.35793/jti.6.1.2015.9968.

[10] K. Schwaber and J. Sutherland, "Scrum Guide V7," no. November, pp. 133–152, 2015.

[11] C. Sadowski, E. Söderberg, L. Church, M. Sipko, and A. Bacchelli, "Modern code review: A case study at google," *Proc. - Int. Conf. Softw. Eng.*, no. May 2018, pp. 181–190, 2018, doi: 10.1145/3183519.3183525.

[12] "Code Quality and Code Security | SonarQube." https://www.sonarqube.org/ (accessed Oct. 31, 2021).

[13] R. K. Mallidi and M. Sharma, "Study on Agile Story Point Estimation Techniques and Challenges," *Int. J. Comput. Appl.*, vol. 174, no. 13, pp. 9–14, 2021, doi: 10.5120/ijca2021921014.

[14] V. Mahnič and T. Hovelja, "On using planning poker for estimating user stories," *J. Syst. Softw.*, vol. 85, no. 9, pp. 2086–2095, 2012, doi: 10.1016/j.jss.2012.04.005.

[15] A. Raza, M. Tayyab, D. Shahid, D. Abdullah, and D. Muhammad, "Impact of Story Point Estimation on Product using Metrics in Scrum Development Process," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 4, 2017, doi: 10.14569/ijacsa.2017.080452.

[16] F. Albero Pomar, J. A. Calvo-Manzano, E. Caballero, and M. Arcilla-Cobián, "Understanding sprint velocity fluctuations for improved project plans with Scrum: a case study," *J. Softw. Evol. Process*, vol. 26, no. 9, pp. 776–783, 2014, doi: 10.1002/smr.1661.

[17] M. Skjott Linneberg and S. Korsgaard, "Coding qualitative data: a synthesis guiding the novice," *Qual. Res. J.*, vol. 19, no. 3, pp. 259–270, 2019, doi: 10.1108/QRJ-12-2018-0012.