Cody VanGosen

Southern New Hampshire University

CS-499 Computer Science Capstone

Professor Kraya

March 16, 2025

**2-1 Journal What Makes a Productive Code Review?**

Code review is a structured process in which developers systematically examine each other’s code to identify defects, improve readability, ensure adherence to best practices, and enhance security. This collaborative approach is widely recognized as an essential part of software development, as it promotes maintainability and reliability across projects. By integrating peer review into the development cycle, teams can proactively identify and resolve issues before they escalate into larger problems. According to SmartBear (n.d.), effective code reviews significantly reduce defects, leading to improved software quality and increased team efficiency. Additionally, they serve as a knowledge-sharing mechanism, allowing junior developers to learn from more experienced team members, while also fostering a culture of continuous improvement and accountability.

For computer science professionals, code review is a critical practice because it ensures consistency in codebase structure, minimizes technical debt, and enhances software security. The process aligns with industry standards, where organizations implement rigorous review systems to uphold software integrity and mitigate potential vulnerabilities (OWASP, n.d.). Beyond its technical benefits, code review encourages collaboration and cultivates a mindset of constructive feedback. By reviewing and refining code regularly, teams can ensure that their software meets functional requirements while also remaining scalable and maintainable.

Several best practices should be incorporated into any code review process to maximize its effectiveness. Reviews should occur frequently and early in the development cycle to prevent major rework later (Pluralsight, n.d.). Keeping code changes small improves readability and allows reviewers to focus on specific aspects rather than being overwhelmed by large commits. Maintaining a consistent coding style across the project ensures that all contributors adhere to established formatting and documentation guidelines, reducing ambiguity and improving long-term maintainability. Furthermore, leveraging both automated tools and manual reviews enhances accuracy, as automated tools can catch syntactical errors, while human reviewers focus on logic, structure, and security considerations. Reviews should not only identify issues but also provide actionable, constructive feedback that helps developers improve their coding approach.

Code reviews should take place before merging the new code into the main repository, ensuring that any identified defects are resolved prior to integration. Conducting reviews during pull requests allows teams to catch errors before they impact production environments, making the process both proactive and preventative. Industry leaders, such as Meta, have demonstrated that optimizing review time significantly enhances development efficiency without compromising code quality (Meta, 2022). By establishing a structured review workflow that integrates peer feedback and automated checks, organizations can streamline the process while maintaining a high standard of code quality.

For this project, I have chosen to use OBS Studio to record the code review. OBS is an open-source and highly flexible software that allows for seamless screen recording with audio narration. Its ability to capture high-resolution video while integrating additional annotations makes it an ideal tool for documenting and presenting the review process. The flexibility of OBS also ensures that any necessary edits or overlays can be incorporated without requiring extensive post-production.

Before recording, I will develop an outline to ensure that the review remains structured and aligned with assignment requirements. The review will be divided into three key areas: functionality and efficiency, readability and maintainability, and security and error handling. First, I will verify whether the code meets its intended functional requirements while also evaluating whether the implementation is optimized. This involves assessing algorithmic efficiency, avoiding redundant logic, and ensuring proper resource utilization. Next, I will examine the overall readability and maintainability of the code by checking for clear variable naming, appropriate commenting, and adherence to coding standards. Readable code significantly improves collaboration and reduces onboarding time for new developers (Codecademy, n.d.). Finally, I will focus on security and error handling by reviewing how the code manages exceptions and safeguards sensitive data. Common vulnerabilities such as insufficient input validation or poor exception handling will be flagged and addressed.

To maintain consistency throughout the review, I will cross-reference my analysis with the code review checklist provided in the course materials. By structuring the review according to these guidelines, I can ensure that all key aspects are thoroughly examined while also adhering to best practices. OBS Studio will be used to record both the screen and voiceover, ensuring a comprehensive and professional presentation. The goal of this review is not only to identify areas for improvement but also to reinforce industry standards and refine my approach to evaluating code. Through this structured methodology, the code review process becomes an integral part of software development, fostering both individual growth and collective team success.

**References**

Codecademy. (n.d.). *How to review someone else’s code: Tips and best practices.* <https://www.codecademy.com/resources/blog/code-review-best-practices/>

Meta. (2022). *Move faster, wait less: Improving code review time at Meta.* <https://engineering.fb.com/2022/11/16/culture/meta-code-review-time-improving/>

OWASP. (n.d.). *OWASP code review guide.* <https://owasp.org/www-project-code-review-guide/>

Pluralsight. (n.d.). *Code review checklist: 7 steps to level up your review process.* <https://www.pluralsight.com/blog/software-development/code-review-checklist>

SmartBear. (n.d.). *Best practices for peer code review.* <https://smartbear.com/learn/code-review/best-practices-for-peer-code-review/>