**CS-499 Module Two Journal**

Andrew Emilio DiStefano

SNHU

Professor Neil Kalinowski

May 14, 2025

**Part 1:**

1. **What is code review?**

“A code review is a peer review of code that helps developers ensure or improve the code quality before they merge and ship it” (GitLab Inc, 2025). Code reviews performed on software engineering projects are meant to improve code, to find and prevent existing or potential bugs, to check for code clarity and consistency with standards, and to improve the programming skills of those involved (George, n. d.). This code review will help me to identify and address any issues (bugs, vulnerabilities, inefficient methods, etc) in my own code from previous projects while documenting these changes in a professional manner to facilitate further review.

1. **Why is it an important practice for computer science professionals?**

The practice of performing code review serves to ensure the production of high-quality code while adding an essential layer of preliminal testing before the unit testing process takes place (Unadkat, 2023). The early discovery of bugs and the sharing of knowledge among team members (GitLab Inc, 2025) helps to ensure high code quality through communication and collaboration. Different team members will likely have their own specializations and areas of strength as developers and engineers. For example, perhaps some team members will be more effective at implementing secure coding practices while others have more experience optimizing for memory management. Collaboration between these diverse professionals enables the review of code from a variety of different vantage points, enhancing a team’s ability to identify bugs and vulnerabilities before unit testing takes place. Adhering to industry standards and best practices enhances code quality, and documenting the ways in which the project adheres to them increases reputability while fostering confidence and trust from potential clients and investors. I do not have a lot of experience performing these types of reviews, and I very much look forward to this opportunity to further develop this important skill to add to my software engineering arsenal.

1. **What are some code review best practices that you read about in the resources that are crucial to include in a code review? Include when a code review should occur in the development process with a rationale as to why.**

“Crawling code…[and]...searching for flaws” (Conklin et al., 2017) are two of the most fundamental of the best practices described in OWASP’s 2017 *OWASP Code Review Guide*. Crawling code involves “scanning a code base of the review target and interface entry points, looking for key code pointers wherein[sic] possible security vulnerability might reside” (Conklin et al., 2017). Searching for flaws involves “perform[ing] an intense code review and search[ing] for all places where user input through a HTTP request could possibly make its way into the HTML output” (Conklin et al., 2017). Having team members engage in this practice can reveal vulnerabilities which could yield devastating results if not corrected before releasing a product.

On the subject of avoiding attacks perpetrated through the entry of user input, “always check[ing] data” (Conklin et al., 2017) is another practice which is crucial to include in a code review. “When data is transmitted from the server to the client, untrusted data must be properly encoded and the HTTP response” (Conklin et al., 2017). This could of course result in unauthorized access to potentially sensitive data via injected attacks involving malicious code entering the system. For example, input code could include characters which could be interpreted as MongoDB commands or SQL queries rather than user input strings. Failure to address this type of vulnerability could enable malicious actors to penetrate the system using common attack methods such as buffer overflow and SQL injection attacks. Ensuring that the system has robust data validation procedures for untrusted data is an excellent way to mitigate against these potential attack vectors.

“Look[ing] for additional places where that same class of vulnerability is present” (Conklin et al., 2017) is a best practice which can greatly improve the efficiency of the code review process. Existing bugs are caught more easily when they are specifically being looked for during the code review process. Individual software engineers and developers often make the same mistakes repetitively (SmartBear, 2025), and noting these types of error trends is a great way to address potential issues efficiently while highlighting key areas for growth among team members.

Another best practice which stands out to me is the “foster[ing] a positive code review culture” (SmartBear, 2025). During my career as a recruiter for software engineering companies, I have seen how important the framing of business practices is to the overall cohesion of a team that is being expected to engage in them. Practices which some team members may consider unnecessary without the proper context could result in significant benefit to the organization. Companies should initiate conversations about the importance of code review and incorporate it into company culture as an opportunity for collaborative team-building rather than a form of busy-work meant only to satisfy company bureaucracy. Fully understanding why code review is so conducive to achieving organizational and personal goals will ensure full-scale psychological investment in code review rather than a begrudged compliance resulting in ineffective review.

“Embrac[ing] the subconscious implications of peer review” (SmartBear, 2025) is another best practice of high importance. The ability for high achievers to show off their skills to the rest of the team is an effective positive motivator for these gifted and/or highly dedicated professionals to continue putting in one hundred percent effort while knowing that their dedication will not go unseen. This could also single out certain team members for consideration when higher-ranking positions become available within the company. Additionally, the use of the potential for shame here is a powerful negative motivator which will almost certainly cause engineers and developers to think twice before pushing haphazard code riddled with obvious issues.   
 Code review should occur at the end of the development phase right before the testing phase in each iteration of the software development process. The testing phase typically begins with unit testing (GitLab Inc, 2025), and performing a code review at this phase is a great way for obvious errors to be caught before unit testing. This keeps the unit test results minimal and less populated with warnings originating from obvious issues such as the inclusion of unused variables or the incorrect use of data types.

**Part 2:**

1. **What software have you chosen to use to record your code review?**

Windows is still by far the most common operating system for desktop computers (Sherif, 2025). Since the most popular applications and tools for specific use-cases such as screen and video recording differ depending on the operating system being used, there is a popular sentiment online that many Linux software options are inferior to solutions only available on Windows or Mac. As someone who has been using Ubuntu as my main general-purpose operating system for the past few years, I completely disagree with this sentiment. It is my opinion that, as long as the user is willing to adapt to alternative tools in certain situations, tools and applications made to run on Linux can be much more powerful if installed and used correctly on a Linux system.   
 I have chosen to use a screen recording software called Kazam to record my code review. Kazam is a popular screen recording choice for Linux users. Released under the GNU Public License (Watkins, 2016), Kazam is a free and open-source screen recording option for Linux desktop users. I will be running Kazam from my Linux laptop, which is how I have always created screen recording presentations in the past. Kazam is simple and effective, with a minimalistic interface that is a refreshing deviation from the user experience offered through many applications commonly used on Windows systems. In my experience, Windows applications are often extremely bloated with extra features which will never be used by the vast majority of users. These highly convoluted interfaces packed with buttons and functions make one feel as though one is in the cockpit of a spaceship rather than in front of an interface meant for the simple purpose of screen recording. The use of Kazam for screen and voice recording will assure that the process of recording my screen and voice for the production of this code review is simple and straightforward while taking up only a small portion of my available system memory. Having avoided the learning curve associated with the installation and use of unnecessarily bloated software for screen recording, I will be able to focus all of my energy and attention on addressing the stated outcomes for this course.

1. **Describe your approach to creating an outline or writing a script for your code review for each of the three categories that you will be reviewing based on the rubric as well as the code review checklist.**

My experience giving presentations and engaging in public speaking has taught me that I perform much better when I have an outline rather than an actual script meant to be strictly adhered to or read word-for-word. I believe that a conversational presentation style matches my communication and personality type well, and that my best presentation will involve real-time wording choices using an outline that guides me through the presentation while assuring that no important information is left out.

My approach will involve making a list of the stated requirements and course outcomes described in the rubric and the code review checklist, and then placing each of them under one or more of the stated categories. These three categories would ideally be covered in order as they were listed in our instructions, but since I will be using the same project for the first and third category, I will first cover Category 1 and Category 3 before covering the changes made in Category 2. This will ensure presentational consistency and avoid a jarring back-and-forth from the first project to the second project, and then back to the first project again.

When this outline is complete I will print out a paper version of the outline. This way I can use it for reference without having it occupy precious screen space while I record my presentation. I believe that this approach will result in a high-quality code review which is comprehensive, focused, and easy to listen to. I am very much looking forward to developing this code review and to improving the quality of previous projects based on what I have learned in the Computer Science program at SNHU!

**Sources:**

Conklin, L., Curiel, J., Hidalgo, M. Keary, E., Mennouchi, I. A., Naderi, A.,

Pantelides, C. & Robinson. (2017). *Owasp Code Review Guide*.

Owasp.

<https://owasp.org/www-project-code-review-guide/>

George, C. (n. d.). *CS-499 Code Review Example* [Video]. SHNU Connect.

[https://snhu-media.snhu.edu/files/course\_repository/undergraduate/cs/](https://snhu-media.snhu.edu/files/course_repository/undergraduate/cs/cs499/cs499_code_review_checklist.pdf)

[cs499/cs499\_code\_review\_checklist.pdf](https://snhu-media.snhu.edu/files/course_repository/undergraduate/cs/cs499/cs499_code_review_checklist.pdf)

GitLab Inc. (2025). *What is a Code Review?*

<https://about.gitlab.com/topics/version-control/what-is-code-review/>

Sherif, A. (2025, March 25). *Monthly market share held by Windows*

*operating system for desktop PCs worldwide from January*

*2017 to March 2025, by version*. Statista.

[https://www.statista.com/statistics/993868/worldwide-windows-](https://www.statista.com/statistics/993868/worldwide-windows-operating-system-market-share/)

[operating-system-market-share/](https://www.statista.com/statistics/993868/worldwide-windows-operating-system-market-share/)

SmartBear. (2025). *Best Practices for Code Review.*

[https://smartbear.com/learn/code-review/best-practices-for-peer-](https://smartbear.com/learn/code-review/best-practices-for-peer-code-review/)

[code-review/](https://smartbear.com/learn/code-review/best-practices-for-peer-code-review/)

Unadkat, J. (2023, April 23). *Understanding Code Review and its Benefits*.

BrowserStack.

<https://www.browserstack.com/guide/code-review-benefits>

Watkins, D. (2016, October 16). *2 free desktop recording tools to try:*

*SimpleScreenRecorder and Kazam*. [Opensource.com](http://opensource.com).

[https://opensource.com/education/16/10/simplescreenrecorder-and](https://opensource.com/education/16/10/simplescreenrecorder-and-kazam)

[-kazam](https://opensource.com/education/16/10/simplescreenrecorder-and-kazam)