Gregoria Ramirez

July 6, 2025

CS 499 Module 2-1

“What makes a good code review”

Dr. Troy Hawk

Code review is a peer review process where someone else looks over the code you wrote to check for mistakes, suggest improvements, and make sure it follows the team’s agreed standards. It is like proofreading, but for code. Having another person review your code can help catch errors you might have missed and may even lead to better or easier ways to write the same thing. This process helps ensure the code stays organized, easy to understand, and dependable.

One of the main benefits of code review is that it helps catch mistakes early, before the code is shared or causes problems. This saves time and avoids bigger issues later. It encourages teamwork, where developers can learn from each other, share ideas, and improve their skills. Code reviews help spot security risks and make sure everyone follows the same style.

According to GitLab, a review should happen after the code is written but before it is added to the main project, while the code is still fresh in the developer’s mind (GitLab, n.d.-b). It is recommended to only review small blocks of code at a time, usually no more than a few hundred lines (SmartBear, n.d.). Focusing on small chunks makes it much easier to spot mistakes, reduces the chance of reviewers feeling overwhelmed, and leads to more precise feedback that helps improve the code faster.

For my code review I have chosen CamStudio to record my code review because it lets me record for 30 minutes or more without cutting me off or adding a watermark. Some of the other free programs I looked at only let you record for up to fifteen minutes, which would not be enough time to finish my full walkthrough.

For my code review outline, I plan to make a simple outline that covers an intro, and the three main areas: software design, algorithms and data structures, and databases. Under each section, I will write down what my current code does and what I want to improve and close with what I plan to see after enhancements. I will gather all my information first and then I will build a script that is easy to follow based on the rubric and code review checklist. I will check to make sure my Microphone is working and test first for voice clarity. For example:

1. For the software design part, I will make sure my code is organized, uses a consistent style, and does not have any repeated sections that can be cleaned up.
2. For algorithms and data structures, I will check that my variable names are easy to understand and that my loops and logic are written clearly and work the way they should.
3. For the database section, I plan to make sure my connection is secure and that the queries are written in a clean and reusable way.

I will make sure I have the correct documents to present on the screen, speak clearly and explain my code as if I am training someone at work and hope for the best.

References

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

GitLab. (n.d.). What is a code review? about.gitlab.com. <https://about.gitlab.com/topics/version-control/what-is-code-review/>

Lazar, M., & Lazar, M. (2024, September 20). *Why code reviews are essential for software projects’ success*. HyperSense Blog. <https://hypersense-software.com/blog/2024/08/14/importance-of-code-reviews-in-software-development/>

Unadkat, J. (2023, April 23). Benefits of code review: Every team must know | BrowserStack. BrowserStack. <https://www.browserstack.com/guide/code-review-benefits>