Code Reviews

CSE 403 Software Engineering

Winter 2026

Pull Request vs. Code Review

Pull Request: a collaboration feature provided by version control system hosts (e.g., GitHub) for proposing merging code changes

Code Review: a constructive review of a fellow developer's code

Pull request UIs provide support for code reviews.

A code review sign-off from another team member is often required before a developer is permitted to merge a pull request.

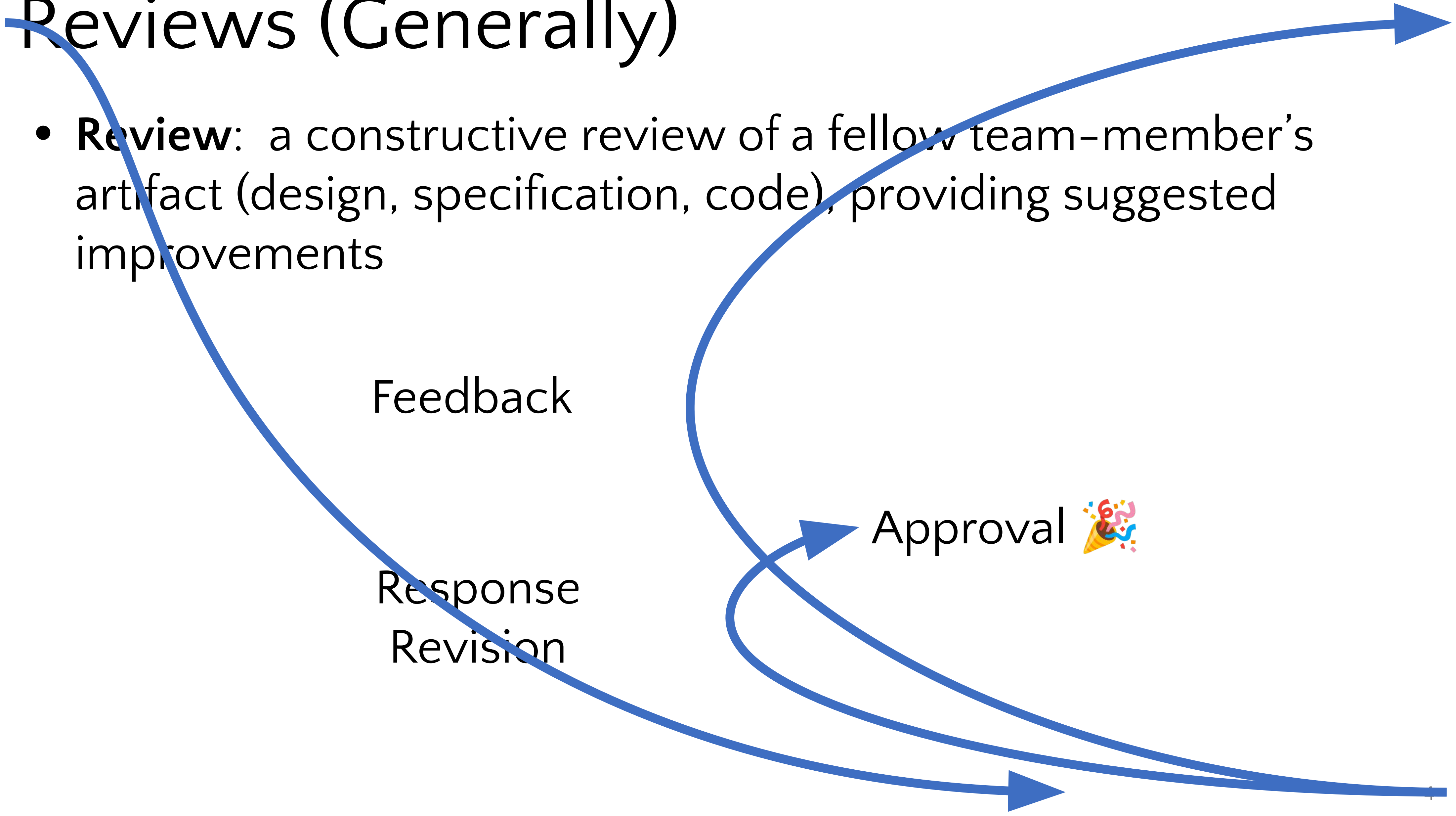
Today’s outline

Code Reviews

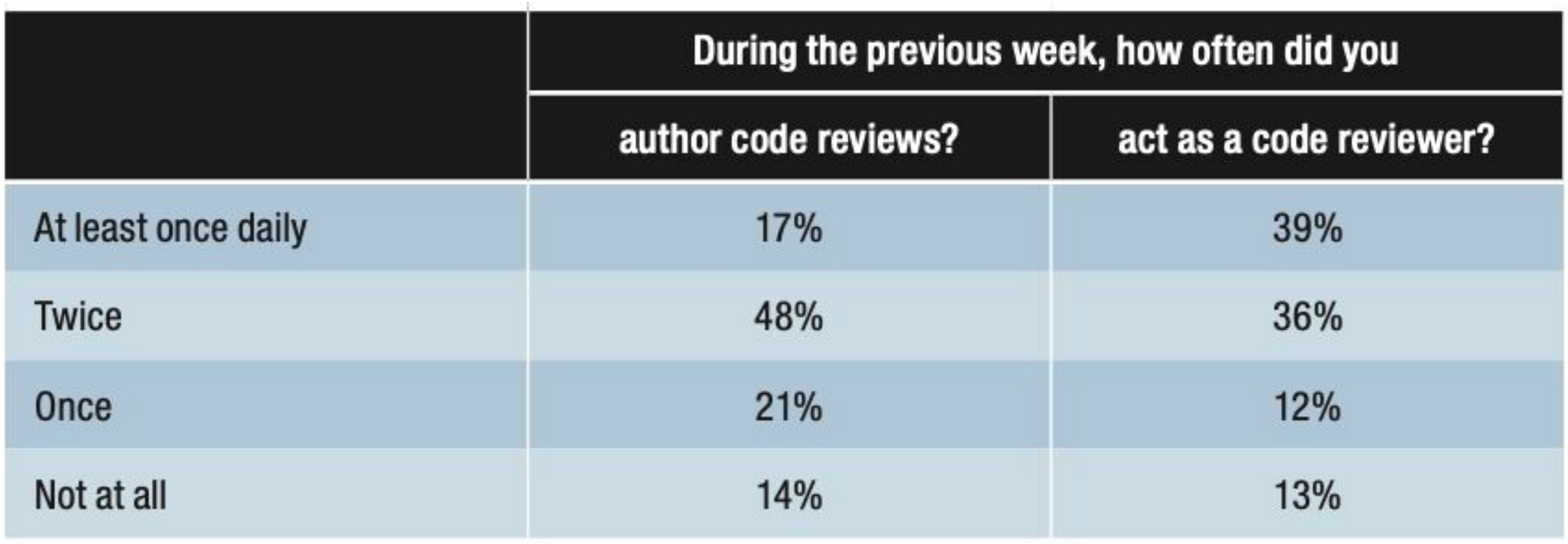
- What are they
- Why are they important
- What to consider when we do them
- Project requirements
- GitHub pull request UI
- Let’s practice

Reviews (Generally)

- **Review**: a constructive review of a fellow team-member's artifact (design, specification, code), providing suggested improvements



Code Reviews are a Big Part of SE



	During the previous week, how often did you	
	author code reviews?	act as a code reviewer?
At least once daily	17%	39%
Twice	48%	36%
Once	21%	12%
Not at all	14%	13%

* Survey of 911 Microsoft developers

Code Reviewing in the Trenches: Challenges and Best Practices
MacLeod et al.
IEEE Software 2018

Why code review?

Didn’t we already test?

Let's look at the data

- Average defect detection rates
  - Unit testing: 25%
  - Integration testing: 45%
  - **Design and code inspections: 55% and 60%** **<<<<<<<<!!**
- 11 programs developed by the same group of people
  - No reviews: average 4.5 errors per 100 LOC
  - **With reviews: average 0.82 errors per 100 LOC** **<<<<<<<<!!**
- After AT&T introduced reviews
  - **14% increase in productivity and a 90% decrease in defects** **<<<<<!!**

(Steve McConnell's Code Complete)

Motivations for Code Review

1. Improve code
2. Find defects
3. Transfer knowledge
4. Explore alternative solutions
5. Improve the development process
6. Avoid breaking builds
7. Increase team awareness
8. Share code ownership
9. Team assessment

Code Review is about much more than finding bugs or style issues!

Code Reviewing in the Trenches: Challenges and Best Practices
MacLeod et al.
IEEE Software 2018

What to consider with a code review
Best practices

LGTMeow

“Looks Good to Me”

Attribution: an engineer that probably doesn’t want to do code review, or a quick stamp of approval after a thorough code review

Agree on a plan with your team

- What has to be reviewed
  - Check out and run code
- Who participates
- Where:
  - Async
  - In-person meeting
- When:
  - What's the expected turn around time

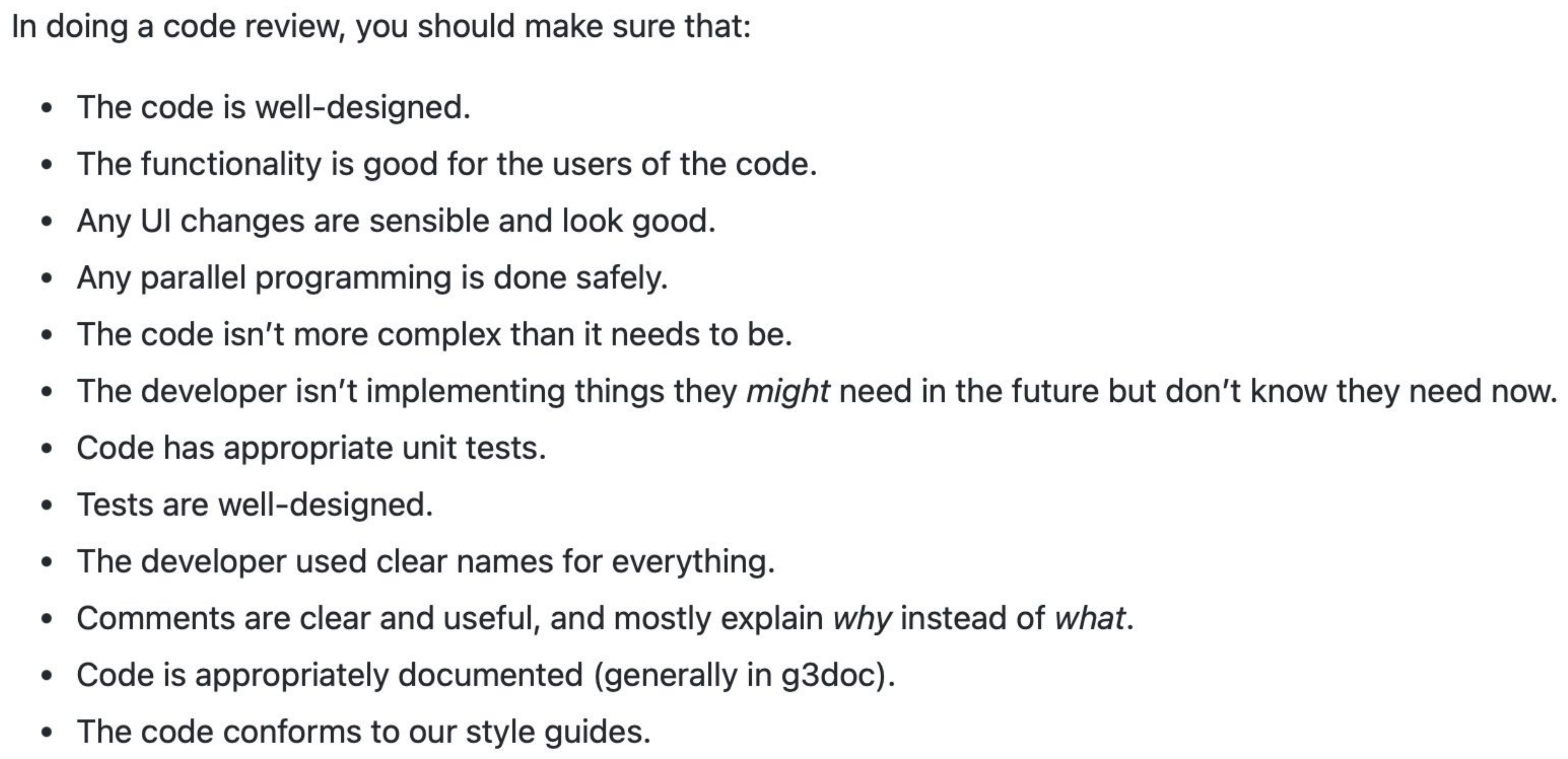
A Google Guideline

"Make sure to review **every line** of code you've been asked to review, look at the **context**, make sure you're **improving code health**, and compliment developers on **good things** that they do."

https://google.github.io/eng-practices/review/reviewer/looking-for.html

Leverage Review Checklists

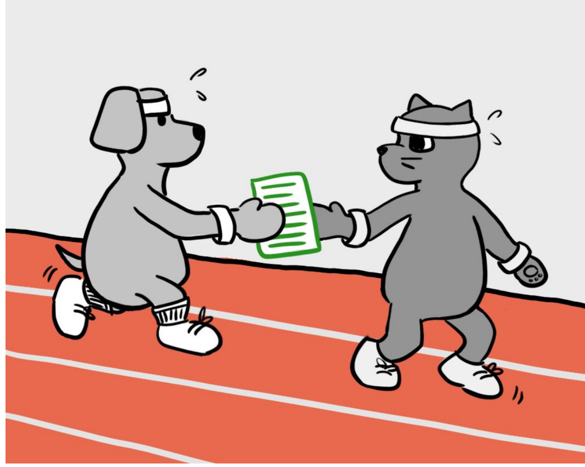
See:
https://google.github.io/eng-practices/review/reviewer/looking-for.html



In doing a code review, you should make sure that:

- The code is well-designed.
- The functionality is good for the users of the code.
- Any UI changes are sensible and look good.
- Any parallel programming is done safely.
- The code isn't more complex than it needs to be.
- The developer isn't implementing things they *might* need in the future but don't know they need now.
- Code has appropriate unit tests.
- Tests are well-designed.
- The developer used clear names for everything.
- Comments are clear and useful, and mostly explain *why* instead of *what*.
- Code is appropriately documented (generally in g3doc).
- The code conforms to our style guides.

Be a Human as You Do Your Review



1. Settle style arguments with a style guide
2. Let computers do the boring parts: linters/formatters/CI/AI(?)
3. Give code examples (build trust)
4. Never say "you" (focus on the code, not the coder!); "we" = team ownership
5. Requests and questions, not commands and criticism … frame it as an in-person conversation
6. Offer sincere praise
7. Incremental improvements instead of perfection
8. Handle stalemates proactively

See: https://mtlynch.io/human-code-reviews-1/

Your Job as the Code Reviewee

- Make your changes easy to review
  - Write small pull requests
  - Provide context and guidance
    - Descriptive title
    - Clear pull request body. Include:
      - The purpose of the pull request
      - An overview of what changed
      - Links to any additional context (e.g., tracking issues, previous conversations)
  - Review your own pull request first
- Be responsive to comments/suggestions

See:
https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/getting-started/helping-others-review-your-changes

Project Code Review Requirements

Expectations:

- Your team is expected to:
  - Have **at least one** student perform a code review on each pull request to main
  - All students must perform their fair share of code reviews
  - Show evidence of meaningful code reviews, such as leaving comments on code – "LGTM" type reviews will **not** receive full credit
- As a reminder:
  - Your team may use AI tools for writing project-related code in this course (see Syllabus for AI policy) which includes the use of an AI code review agent, *if* the use is agreed on and documented
  - If an AI code review agent is used, the student using the generated code review is responsible for updating the review to ensure it is correct and complete

GitHub Pull Request Interface

Practice
Let’s do some code reviews

Code Review In-Class Exercise

Today's Activity:

- Work with a partner (join a pull request group in Canvas)
  - You may work alone
- Follow the instructions on the handout
- Due tonight (Fri) 11:59PM
  - 1 group member submit your files
  - Each group member individually complete the survey

Code Review in Software Engineering Education Study

- Contribute to UW Research!
- Complete the exercise and survey
- Participation in the research is voluntary
  - You are required to complete the exercise and survey for course credit, but you can choose to have your data excluded from the study
- Contact in case of questions or concerns
  - Contact about the study: Hannah Potter (Lead Researcher), UW: hkpotter@cs.washington.edu
  - UW Human Subjects Division: hsdinfo@uw.edu or 206-543-0098