What makes a good pull request?

3 November 2026 · 4 min read · blogs.iictp.org/what-makes-a-good-pull-request

A good pull request is small, has one purpose, explains why the change is needed, and is easy to review and to undo. It respects the reviewer's time, which is the quickest route to faster, better feedback.

TL;DR The short version

- A good pull request is small, has one purpose, explains why, and is easy to review and to undo.
- Its description says what changed, why, how it was tested and where to look closely; read your own diff and run the tests first.
- Reviewers comment on the code, not the person, because review is a shared safety net, not a test of the author.

Read online, listen to it, or change the text size: https://blogs.iictp.org/what-makes-a-good-pull-request

Why does size matter?

Reviewers find fewer problems in large changes because they cannot hold all of it in mind. A change that does one thing can be understood, tested and, if needed, reverted on its own. If a pull request mixes a refactor with a new feature, split it.

What should the description say?

What changed, why it changed, how you tested it, and anything the reviewer should look at closely. Link the ticket or issue. A reviewer who understands the intent can judge whether the code achieves it.

What should the author check before asking for review?

- Read your own diff first, as a reviewer would.
- Run the tests and linters locally.
- Remove debug output, commented-out code and unrelated edits.
- Make sure no keys, passwords or personal data are in the change.

How should a reviewer give feedback?

Comment on the code, not the person. Separate must-fix issues from suggestions, explain the reason for a request, and ask a question when unsure. Approve when the change is good enough to improve the codebase, not only when it is perfect.

How does this connect to blameless culture?

Review is a shared safety net, not a test of the author. Teams that treat feedback as collaboration catch more defects and learn faster, in line with the blameless approach described in what is a blameless postmortem.

Related credential: Associate Technical Practitioner. Explore the exam facts and syllabus.

Sources and further reading

1. Google. Engineering Practices: Code Review Developer Guide. google.github.io/eng-practices.
2. Chacon, S. and Straub, B. Pro Git (2nd ed.). git-scm.com/book.

Published by IICTP, an independent technical credential body, on 3 November 2026. The current version of this post is at https://blogs.iictp.org/what-makes-a-good-pull-request. More guides: blogs.iictp.org.