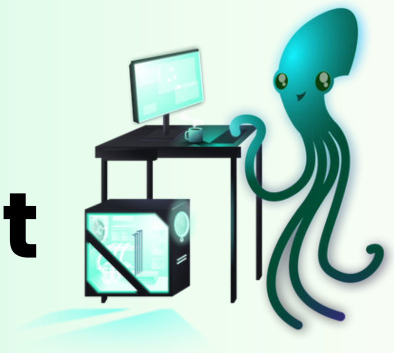
GitKraken

Code Review Checklist



✓ **Read the PR description and context**

Before diving into the code, make sure you understand why the change is happening. Check the issue, ticket, or task it's related to, and get the full picture of what's being solved.

✓ **Look at the overall diff**

Get a sense of the scale of the changes. Are they within a reasonable size? Does the structure of the diff make sense, or do you see any red flags at a high level?

✓ **Review the logic, flow, and structure**

Does the code follow project architecture patterns? Is the solution correct, and could it be simplified? Are there any edge cases? Make sure both the core logic and overall structure are sound before diving into minor details.

✓ **Ensure readability and maintainability**

Good code should be easy to understand. Are variable names clear? Is the function structure logical and easy to follow? Ensure future developers (including yourself) can jump into this code later without confusion.

✓ **Check for test coverage**

Has the developer added tests where needed? Do the tests cover the main functionality and edge cases? If tests are missing or insufficient, flag this early on.

✓ **Look for unnecessary complexity**

Is there any code that could be simplified or removed? Look for over-engineering or logic that could be broken down into smaller, reusable pieces.

✓ **Check for performance considerations**

Make sure the code doesn't introduce unnecessary performance hits. Are there areas where memory usage or speed could become an issue, especially with large data sets or frequent calls?

✓ **Leave actionable, concise feedback**

Be direct and clear with your feedback. Instead of just pointing out what's wrong, offer suggestions or ask clarifying questions. Aim to help the author improve their code without overwhelming them with vague comments.