20+ Questions to Ask for a Successful Code Audit

Revelry

Code reviews / audits (also known as codebase audits, technology assessments, risk assessments, and peer reviews) are methodical evaluations of software code designed to:

- Identify bugs and logic issues
- Improve code quality
- Help developers learn the source code
- Spread knowledge across a team or organization
- Ensure a product and/or business is well-positioned to grow and scale

Code reviews / audits are critical to preventing unstable code from being shared with the world or shipped to customers (gasp!), and should be a part of any development team's workflow.

At Revelry, our team of software engineers regularly conducts code reviews / audits for partners. **This is our guide for helping ensure review / audit success, including 20-plus questions we ask and answer as we work through the process each and every time.**

Revelry

How to Approach a Code Review

Preparation

1. What is the goal of your code audit?
There can be several reasons you may want to do a code review / audit on a software product.

If you are an **investor**, a code audit can help you understand the technology risk and potential of a company. Things to look for include:

- Technical debt
- Evidence of poor development practices
- Third-party service exposure
- Security vulnerabilities

If you are a **business executive**, a code audit can help identify strengths and weaknesses in your tech stack and development process, highlighting opportunities to improve security and efficiency, as an example.

If you are an **engineering leader** about to take over the maintenance of an existing codebase, a code review / audit is the first step to set your team up for success.

2. How do you get access to the code?
If you're asking the existing development team to meet and discuss the codebase, do this as soon as possible, preferably before you devote time to digging through code, documentation, etc. without context. It may seem strange to meet with the people whose code you are auditing... you may feel like not meeting with them keeps you "impartial"... however, in most cases, you will be better positioned to guide the audit if you learn about the code, the history of its development, future plans, strength of the existing team, etc.

In additional, when a dev team knows they are being audited, they may assume they've done something wrong. Meeting with them early provides an opportunity to explain the goals of the review.

3. Are there any other documents you need to see? Yes - and the more the better. Project documentation can vary from project to project and team to team, so it's helpful to ask project / product managers and other key contributors what tools and content repositories exist. For example: Does the team document its stand-ups (daily or weekly meetings) in a Google shared drive or a GitHub ticket? Is there a project-specific Slack channel? Where can things like architecture diagrams, product roadmap, etc. be found?

The Audit

The Code

Once you've had your conversations with the existing dev team and sifted through project files / documentation, it's time to dig into the codebase. We recommend doing so with the following questions in mind and, as you're reviewing, taking notes you can use to create an audit report.

4. What is the code like? Read it with a critical eye. Is it well organized? Does it follow the standard conventions of the language in question?

5. What are the code comments like? Review all code comments and keep an eye out for unresolved TODOs and FIXMEs.

6. What is the documentation like? Read it and ask how it can be improved. Does it clearly outline the purpose of this codebase and how to get set up to contribute to it?

7. When you check the test suite and measure test coverage, what do you find?

The Tech Stack

In evaluating the tech stack, be sure to ask:

8. Is the product using the right

programming language for this task? If not, why? What language should be considered instead? Why?

9. Is the product using the right framework for this task? If not, why? What framework should be considered instead? Why?

10. Is the product using the right programming database for this task? If not, why? What database should be considered instead?

Security

As you conduct your review / audit with an eye for security / vulnerabilities, ask:

11. Are there any dependencies that put the product at risk from third-parties? (i.e. what happens if the dependent service stops being maintained / changes their license / changes pricing, etc)

12. Are there any dependencies that could have license issues? (i.e. GPL libraries being used in a way that spreads GPL requirements to the total codebase)

13. Are there any dependencies that could cause security issues? (e.g. npm audit and similar)

14. Are there any other security issues?

Development Process

15. Is the team following strong coding conventions? (This Wikipedia page explains coding conventions and their importance.)

The Code Review / Audit Report

Writing an easy-to-digest and comprehensive report is key to any good code audit. These are the sections Revelry includes in our partner reports:

16. What is the executive summary? The executive summary provides a very short (1-2 paragraphs or a bulleted list) summary of key findings and recommendations that will be detailed in your report.

17. What was the scope of the investigation? Explain what you did and did not do in the code review / audit. What did you look for? What techniques did you use? What wasn't investigated? (e.g. "We didn't investigate product-market fit in this analysis" or "This is not a full PCI/DSS compliance review".)

18. What were the findings? What did you observe? What data was gathered?

19. What are the key risks? In your opinion, what are the biggest risks facing the codebase / company / team? These should tie into your findings (#18), because they should be based on things that you investigated (e.g. Don't say "I think this product is going to flop" unless you did a business analysis or product-market fit on it).

20. What are the recommendations for the code? What steps need to be taken in response to your findings?

Conclusion

While code reviews / audits can be time consuming (especially if you've never done one or running at it without an established process), they are essential to your software product and company's success - especially if you're planning to grow and scale.

At Revelry, we've been honing our technology review / audit process for years and would welcome the opportunity to help you with yours; let's connect.

p.s. Learn more about code reviews in our blog post on common code review pitfalls

About Revelry

Established in 2012, Revelry is a New Orleans-based digital innovation company that specializes in creating custom software solutions that help its partners across a variety of verticals (from healthcare and logistics to finance and start-up) succeed and scale. To learn more about Revelry's strategic development services, or our team of passionate product managers, design engineers, and back-end developers, visit revelry.co.

Revelry