Joseph Marek

11/10/2024

Part 1: Code Review and Its Importance

1. What is code review?

Code review is the process of systematically examining code written by another programmer, typically as part of a collaborative software development process. The aim is to ensure that the code is efficient, clean, secure, and maintainable, as well as to identify potential bugs or security vulnerabilities. Code reviews can be performed manually by peers or automated using specific tools to ensure adherence to coding standards and best practices.

2. Why is code review an important practice for computer science professionals?

Code review is an essential practice for computer science professionals for several reasons:

- Quality Assurance: It helps identify bugs or errors in the code early in the development cycle, which can prevent issues from escalating in later stages.

- Knowledge Sharing: Code reviews foster collaboration and knowledge exchange among team members. It gives developers the opportunity to learn from each other’s code, improving overall team competency.

- Improved Code Quality: By having multiple eyes on the code, the likelihood of missing design flaws or performance issues is reduced, resulting in better software quality.

- Maintainability and Scalability: Code reviews ensure that code is written with future maintainability and scalability in mind, adhering to best practices and design patterns that improve long-term usability.

3. What are some code review best practices that are crucial to include in a code review? Include when a code review should occur in the development process with a rationale as to why.

Some key code review best practices include:

- Review Early and Often: Code reviews should occur at regular intervals, ideally after a developer has completed a small task or feature. This prevents large code changes from accumulating, making the review process more manageable.

- Focus on Code Quality, Not Just Syntax: It’s essential to evaluate the structure, design, and logic of the code, rather than just correcting minor syntax issues.

- Provide Constructive Feedback: The review should be collaborative and focused on improvement. Feedback should be respectful, specific, and actionable, rather than critical.

- Limit the Scope of Reviews: To keep reviews manageable, they should focus on small, specific sections of code. Large chunks of code are harder to review thoroughly and more likely to be skipped over.

- Automate Where Possible: Automated tools can catch common issues such as syntax errors, security vulnerabilities, or style guide violations. This allows reviewers to focus on more complex aspects of the code.

- When Should Code Reviews Occur? Code reviews should occur regularly throughout the development cycle, ideally after the completion of a feature or a task but before the code is merged into the main branch or released. This helps catch issues early, reduces bugs, and ensures quality code before integration.

Part 2: Approach to Writing the Code Review

Since I am not comfortable recording a video, I have chosen to write my code review as a detailed, structured text document. This method allows me to focus entirely on content, ensuring that I can deliver a thorough, clear, and well-organized review without worrying about technical aspects of recording. Here’s my approach to writing and organizing the review:

1. Software Chosen to Record the Code Review

I plan to use a text document to write my code review. A written document will allow me to carefully organize my thoughts and present my analysis clearly. I can focus on the quality of the content and structure without worrying about video production details, such as recording audio, editing the video, or adding visual aids.

The text document will allow for easy navigation, enabling me to break the review into clearly defined sections. I can use headings and subheadings for each category of review, making it easy to follow.

2. Approach to Creating an Outline for the Code Review

To ensure my code review is comprehensive and aligned with the rubric and the code review checklist, I will organize my review into clear sections. The three categories in the rubric are: Software Design and Engineering, Algorithms and Data Structures, and Databases.