Adam Donner

Web 430

Assignment 6.3

October 16, 2019

The premises of DevOps is improving the software development lifecycle, the speed that the software is delivered to the customers, and the quality of that software. There are tools and techniques that allow these things to be achieved.

In DevOps, the practice of requiring engineers to get peer reviews of their changes before deployment is called code review. It is equally as crucial in changes made to the application as it is to changes made to servers, networking, and databases. The goal of code review is to find errors by having peers carefully review and scrutinize any changes. Code review helps to improve the overall quality of any change, allows for cross-training, peer learning, and skill improvement. Code reviews should be performed before the code being put into the trunk branch, before changes could have impacts on the team level or even the global level (Kim, Debois, Willis, & Allspaw, 2017).

It is essential to have a well-defined process and guidelines for code reviews that are set for the reviewers and the ones being reviewed. Perquisites for creating pull requests should be defined. The one reviewing the code needed to understand their responsibilities. Before the code is handed off to be reviewed, there needs to be excellent communication with the reviewer, mall pull requests should be made, changes to the code should not be made during the code review, and any feedback should be responded to. The responsibilities of the reviewer also need to be defined. The reviewer needs to be aware of the task description and requirements, make sure that they completely understand the code, evaluate all the architecture tradeoffs and keep comments to critical comments, optional comments, and positive comments (Elovic, 2019).

Several different approaches can be taken when it comes to code review. One of the approaches is the email thread. When a piece of code is ready to be reviewed, the file is sent to the appropriate reviewers via email. Pair programming is an approach where two developers work on the code together and check each other’s work as they go. The over-the-shoulder approach is the easiest and most intuitive way to perform peer review. The simplest and easiest approach is through software-based code review tools (Huston, n.d.)

To ensure that there is efficiency and effectiveness of code reviews, there are some best practices that should be followed. Developers should know what to look for in code reviews. They should look for things such as structure, style, logic, performance, design, readability, and functionality. Some of these checks can be done with automated checked, while others require a human to evaluate. When the code is evaluated critically and questions or kept in mind, the code will be checked for the right things, which will, in turn, reduce the amount of time needed for testing. It is critical to build and test before doing a manual code review. This is ideally done after the code has passed any necessary tests. Automated checks will cut down on euros and save time in the code review process. The code should not be reviewed for longer than one hour. It is best to perform the code reviews often, and in a short session, there is better attention to detail. There should be no more than four hundred lines of code checked at a time. This makes for a more effective review and ensures higher quality in the codebase. Reviewers should give constructive feedback rather than critical feedback (Bellairs, 2018).

Running code reviews can be a challenge. For code reviews to be effective, some things should be kept in mind. The goals and expectations of the code review should be clearly communicated. This saves everyone involved time when the reviews know what to look for. Everyone needs to be involved in the code review process. When everyone is involved in the process, there is improved collaboration and connection between programmers. A positive culture is vital to have an effective code review process. This helps to ensure that programmers appreciate rather than hate code reviews. Using automated tools can help to save time in the review process. This frees up time that allows the reviewers to be able to focus on the things that automation cannot find (Bellairs, 2018). There are tools such as Helix QAC and Helix Swarm that can also ensure that the code review is effective.

Even though code reviews can be difficult and time-consuming in the software development process, there are dangers if there is not a code review process. Not doing code reviews can result in inconsistent design and implementation. This can also affect the maintainability and longevity of the software. The absence of code reviews can increase mistakes and increase the problems that these mistakes have. Project quality can also be affected when there is no code review. Code performance and efficiency suffer when there are no reviews. New techniques and technologies are also not shared among different levels of engineers that allow knowledge and experience to be passed along ( Zhang, 2019).

The code review process is an essential step in DevOps. It helps to ensure that code is effective and efficient.

References

Bellairs, R. (2018, December 4). 9 Code Review Best Practices. Retrieved October 15, 2019, from <https://www.perforce.com/blog/qac/9-code-review-best-practices>.

Elovic, A. (2019, January 28). Code Review - The Ultimate Guide. Retrieved October 15, 2019, from https://www.freecodecamp.org/news/code-review-the-ultimate-guide-aa45c358bbf5/.

Huston, T. (n.d.). What is Code Review? Retrieved October 15, 2019, from <https://smartbear.com/learn/code-review/what-is-code-review/>.

Kim, G., Debois, P., Willis, J., Humble, J., & Allspaw, J. (2017). The DevOps handbook: how to create world-class agility, reliability, and security in technology organizations. Portland, OR: IT Revolution Press, LLC.

the-ultimate-guide-aa45c358bbf5/.

Zhang, L. (2019, September 30). Why the Code Review Process Is Critical for Developers. Retrieved October 15, 2019, from <https://www.brightspot.com/blog/developer-life-5-reasons-why-the-code-review-process-is-critical-for-developers>.