Daily Work of a Software Engineer

by Devmountain Tutorials

WHAT'S COVERED

In this lesson, you will learn how to describe the common daily activities of a software engineer. Specifically, this lesson will cover:

1. Collaboration with the Team

The code that I'm responsible for includes the servers and databases mentioned in the last section as well as applications that control the logic for how things work.

In the last iteration, I created a recommendation engine to help customers view the service providers based on their reviews and the reputation of the service provider. I create integrations between different parts of our technology stack but also integrations with third parties. When Camilla was creating the iOS app, I helped create an integration to send notifications to our users through Apple's Push Notification service.

In addition to the code I write, I'm responsible for collaborating with the dev team. My coworkers told you about some of the regularly scheduled meetings we have.

I'm going to show you more about engineering collaboration that happens. You might think that engineers spend a lot of time working alone based on stereotypes, but that isn't the case for most engineers. We spend a significant amount of our time in meetings and collaborating. Collaborating with your peers feels more like conversations and less like traditional meetings because they happen as needed.

The first collaboration point is an architecture planning discussion. This is where the developers and engineers get together to diagram or whiteboard, how different systems will integrate. We identified dependencies, requirements, and possible solutions. We focus on creating a solution that is flexible so it can change as needed, scaleable so it can grow to support a large number of users without significant cost, modular so parts of the code can be reused in other places, and reliable so we can count on it working. We plan who is going to complete the different development tasks and make sure that we've documented the design.

Architecture Planning Meeting

---

2. Code Reviews

Code reviews are the most common way engineers and developers collaborate. A code review is a process where your peers review the code you've written before it is accepted and merged into the main code repository. This allows us to check for mistakes with a fresh pair of eyes and ensure we have the same understanding of the requirements. Because everyone has their own way of writing code, I need to make it presentable and readable, so it is easy to understand.

Feedback from code reviews can be contentious if you aren't self and socially aware. Critiques can be painful, especially if you haven't built a productive relationship with your peers. When I receive criticism for my code, I try to understand where they are coming from and look for ways to improve my skills. Reviewing my teammate's code allows me the opportunity to share my experience and expertise with them as well. This creates a great culture of learning.

When engineers find a problem particularly tricky to solve, we like to leverage pair programming, which is a collaborative technique where two individuals program on one computer. This allows us to talk through the smallest details and complexities together. One person types while the other person describes what the code should be doing at a high level. Then, they switch roles. It can be awkward at first to vocalize the thoughts you have when coding solo but practicing explaining complicated ideas clearly and concisely improves your communication skill and helps you write better code.

Another way we collaborate is with written documentation. Our development team keeps a wiki to capture important decisions, requirements, architecture designs, standards, and best practices. Everyone is responsible for contributing and maintaining the wiki.

When writing code in the wiki, you can notate or comment right in-line with the code to explain why you built something a specific way. There have been times when I've forgotten details of my code, so documentation isn't just for everyone else–it helps me be efficient as well.

Other documentation we maintain includes release notes and FAQs for the support team. This helps them understand the expected behavior and how to troubleshoot simple issues that may come up. When an issue happens that our support team can't solve, they escalate it to our development team. I'll walk you through that process more in the next section.

SUMMARY

This lesson discussed the daily work of a software engineer, with a focus on **collaboration with the team** and **code reviews**. The engineer is responsible for creating integrations between different parts of the technology stack, as well as collaborating with the development team. The first collaboration point is an architecture planning discussion, where the engineers diagram and whiteboard how different systems will integrate, identify dependencies, requirements, and possible solutions. Code reviews are the most common way engineers collaborate, allowing them to check for mistakes with a fresh pair of eyes and ensure the same understanding of requirements. Feedback can be painful, but it creates a culture of learning. Pair programming, written documentation, release notes, and FAQs for the support team are also used to collaborate effectively.