**Assignment 1 – The Cathedral and The Bazaar**

23102009 Park Shinhyung

**What is Cathedral and Bazaar?**

When we think of a cathedral, what image comes to mind? Both externally and internally, a cathedral is considered a result of meticulous design and systematic construction. In this book, the traditional software development process is compared to a cathedral from a developer’s perspective. Like a cathedral, this approach follows a strict blueprint, executing a series of steps carefully, but it has the downside of being difficult to incorporate external feedback. In contrast, the bazaar is a place where many people lively engage in buying and selling. From the viewpoint of software development, this represents an environment where various individuals solve problems from their own perspectives, collaborating and interacting along the way. Unlike the cathedral, the bazaar offers an open and cooperative setting where anyone can provide input on a program and gain inspiration, which is a significant advantage.

In this book, open-source software development is symbolized by the bazaar. In open-source communities, many developers collaborate and contribute, leading to better outcomes. Problems are solved through collective intelligence, and each contribution improves the overall quality of the project—something that is difficult to achieve with the cathedral-like traditional methods.

As a university student, I believe it is essential to experience and understand both the cathedral and bazaar approaches to development. However, the future of development is increasingly moving towards more collaborative and open models. Many companies are also valuing these collaborative skills. The various interactions that take place in the bazaar will become crucial abilities for developers, and the ability to solve problems in such environments will only grow in importance.

**19 good practices (lessons) presented in the book, Which 3 would you like to highlight?**

1. *Every good work of software starts by scratching a developer's personal itch.*

This statement left a strong impression on me. Although it refers specifically to software developers, I believe this principle is crucial no matter what kind of developer I become. Let me share my own experience. As a frontend developer, I once worked on a team project. Since I didn’t have much development experience, I relied heavily on my teammates, and I had to study and work hard to contribute. As a result, I developed a deep attachment to the project. Throughout the process of planning and developing the service, we encountered numerous problems. However, as we worked through those challenges, I could feel not only my own growth but also the growth of the team and the service. Although our team didn’t win any awards, we became closer during the process, and we still meet regularly. This experience taught me how fulfilling it is to be deeply passionate and engaged in something. That’s why I agree with the idea that developers should start with a genuine love for programming.

*9. Smart data structures and dumb code works a lot better than the other way around.*  
As I gained more experience in various projects, I began to have doubts. With the rapid development of generative AI, anyone can write code, which led me to question whether my own code was truly efficient. While it’s relatively easy to create a service that simply works according to the design and plan, creating a well-designed service is a completely different challenge. Through this struggle, I realized the truth behind the lesson in this statement. My complex code often caused confusion during debugging, and I learned that simple code combined with well-structured data could solve problems much more effectively. This realization has reshaped my approach to development.

*19. Provided the development coordinator has a communications medium at least as good as the Internet, and knows how to lead without coercion, many heads are inevitably better than one.*Before reading this essay, I thought of open source as merely “public code.” However, this project showed me how a collective effort, with many eyes reviewing the code and improving it from various perspectives, leads to efficient solutions for complex problems. Looking back, my own development process has always been similar. I’ve relied on the experiences of others, learned from online resources, attended university lectures, and worked with teammates on projects. There’s hardly anything I’ve accomplished entirely on my own. I’ve come to realize that developers are not individuals who solve everything by themselves, but rather individuals who learn, collaborate, and grow together. The open-source project is not just about the use or concept of open code it provides a valuable lesson for me and many other developers. That’s why I chose this as the final topic to emphasize.