Web+CloudCoopOnboardingNotes

tl;dr

The content and links provided were very helpful in giving an overview of the tools used by the team. The hands-on activities were beneficial, and adding a more extensive project could enhance the learning experience. The order and content of the chapters could be slightly rearranged as follows (topics covered are in order):

- Chapter 1: Linux, WSL, Git
- Chapter 2: HTML, CSS, JS
- Chapter 3: TypeScript, React, Next.js, (GraphQL, Ant Design)
- Chapter 4: Node.js, (Apollo, SQL)
- Chapter 5: Docker
- Other Technologies: Python, and others

Chapter 1

This chapter served as a solid introduction to Linux, specifically WSL. For someone with limited knowledge of Linux, this chapter would bring them up to speed on what Linux is and how to use basic commands.

The tutorials provided are excellent. Creating our own tutorials would be redundant given the vast resources available online. The guides are very helpful and offer great foundational learning. Including links for further exploration and documentation could be beneficial for new coops who want to delve deeper.

The hands-on practice activity at the end of the chapter was useful. However, it could be more in-depth with additional steps or by incorporating a more comprehensive Linux introduction practice activity from the internet. For example: Hack The Box Linux Learning.

Chapter 2

This section's usefulness depends on the coop's prior knowledge. For those with experience in HTML, CSS, and JavaScript, this chapter may be less critical. However, for someone new to these technologies, it would be quite valuable.

The links provided are helpful and offer a good background on these tools. The hands-on practice activity is also useful, and I appreciate its open-ended nature. I personally find working in a sandbox environment beneficial when learning new tools or languages, so the activity's flexibility was advantageous.

Chapter 3

More emphasis on Git would be beneficial. I have encountered many instances where team members lacked Git knowledge, which delayed project progress. Given Git's importance in preventing issues, a bit more coverage would be helpful.

The hands-on practice was once again very useful. I had already completed many LearnGit modules in a previous class, and this experience was a major factor in my Git proficiency.

Chapter 4

This chapter was where I spent most of my learning time. While I quickly completed Chapters 1-3, I dedicated more time to Chapters 4 and 5.

I came into this coop confident in standard client-side JavaScript and Node/Express, but with no prior experience in TypeScript or Next.js, so I was learning these tools from scratch.

The documentation links were very useful and will likely be valuable throughout the semester. However, the "cheat sheets" for TypeScript felt a bit overwhelming due to the dense information.

I found it beneficial to create my own practice project to apply the new concepts. My goal was to build a basic app that collects data from a server and displays it to the client. Although the app was not very presentable, it served as a proof of concept and a reference for various tools.

It might be a good idea for coops to undertake a similar project—a basic and open-ended one that allows them to apply what they have learned. My project lacked organization, and with more guidance, I could have developed something more substantial.

Chapter 5

Like Chapter 4, I had minimal prior experience with Docker. The guides provided were very helpful and gave me a good starting point for further research into containers. However, even after going through the guides and experimenting on my own, I still don’t feel fully comfortable working with containers in production.

Given that other coops may have similar experiences, and likely little prior exposure to containerization, additional guidance could be helpful. A simple project, such as running a Node.js or Next.js server in a container, would be beneficial and is something I managed to accomplish on my own.

Other Technologies

The section on other technologies is useful but could be better integrated into the relevant chapters.

GraphQL, Apollo, Ant Design, and SQL could be moved to Chapter 4. To avoid overwhelming coops, consider combining Chapters 1 and 3, as they both cover peripheral tools (Git and WSL) compared to the client/server focus of the other chapters. Chapter 4 could be split into two sections: client-side tools and server-side tools. Chapter 4.1 could cover TypeScript, React, Next.js, with additional reading on GraphQL and Ant Design. Chapter 4.2 could focus on Node.js, with additional reading on Apollo and SQL.