Journal Entry 1 – 1/22/2025

Chapters 1 through 5

As I read through Chapter 1, I was reminded of all the difficulties I encountered while attempting to collaborate with others back in the early 1990’s. One of the key takeaways for me in the first chapter was the various options for saving work and collaboration and that git represents a safer and more efficient means of doing so. I can clearly remember collecting multiple diskettes and later USB drives and having to meet with other members of my project at odd hours. One of the difficult things about using git is that it is so far removed from what I first learned. This makes it hard to come up with analogies for what it is doing which is a habit I have formed in my learning. Also, some of the terminology is very new to me, specifically concepts like repositories, staging, working areas, and committing. Something that makes the concepts especially difficult is the fact that at least in the beginning, the Git Bash interface we are using, doesn’t have a visual representation of “where” things are. I am hoping perhaps there is something with more of a graphical representation as we proceed.

Chapter 2 was fairly straightforward to follow. I actually installed git on both a PC and a Mac. I have been more of a Mac user for the past ten years, but I recently purchased a PC for use in a IT Support class I am taking. The most troubling aspect of chapter 2 was that there were many details of installation that of necessity were glossed over. My experience working “under the hood” of a PC and a Mac is very limited, but I am not one that likes to just select options because I am told to do so. I like to understand every aspect of a process, but at this level in my education, there are too many aspects of PC and Mac operating systems that I have yet to been fully exposed to. One specific complaint I have about the text is that it was stated that we could default our Git Bash to use “master” or “main” as our original branch and that it “didn’t really matter” but GitHub uses “main” by default so in the early stages when I was relying on code lines from GitHub to set up my repository, I was having problems because I was using Master with one program and Main with another.

The processes we learned in Chapter 3 were somewhat challenging for me because I have had limited exposure to using terminal commands and at times it was tempting to do as much as I could in Windows with respect to file manipulation. With respect to the actual concepts of git, I feel the most important things to keep in mind are that we modify our files in the working directory, then we add them to staging area, then we commit them which is just creating a “snapshot” of the project. These concepts are very new to me as a method of saving work and collaborating. As I stated earlier, I try to learn by analogy, but I am trying to be cautious because this is very different from my initial experience when swapping physical media was one of the primary means of collaboration and backup, and it was common to have multiple distinct versions of a project saved in various places in their entirety.

Chapter 4 focused on the .gitignore feature. It was actually interesting to see commands put into place and the effect they had on which files were tracked. The most important part of the chapter from my perspective was the use of what I am told are called “wildcard” characters to allow multiple files and directories to be ignored with one line or to create very specific parameters if needed. This chapter also included the beginnings of seeing the history of our changes and even seeing what some of the changes were. It also introduced keyboard commands to allow navigation among the various commits.

Chapter 5 provided a helpful recap of the three possible states a file can be in: modified, staged, or committed. As stated, these concepts are very new to me, so repetition is extremely helpful. The chapter also tried to emphasize the fact the way git “saves” is different from my previous experience. An important concept is the idea of the “snapshot” that occurs during a commit. The text says that “the state of the entire project is saved, not just small changes.” It also clarifies that changed files are saved for the snapshot, while files that are unchanged are only referenced for the snapshot. This concept is still very difficult to grasp, and it will take time to understand the finer points of it. The last key takeaways for me with respect to Chapter 5 are perhaps the most useful processes which are the checkout, revert, and reset features. These were very difficult processes for me to implement perhaps because I have been using Windows and MacOS for so long that I need a special representation of what I am working on. Like my difficulties utilizing terminal commands in chapter 3, if I am working on a certain branch, or moving backwards or forwards through a series of commits, I want a visual representation of where I am “at”. As I have said before, I hope that once we understand the git basics, there are more sophisticated interfaces available.