Tanaka Makuvaza

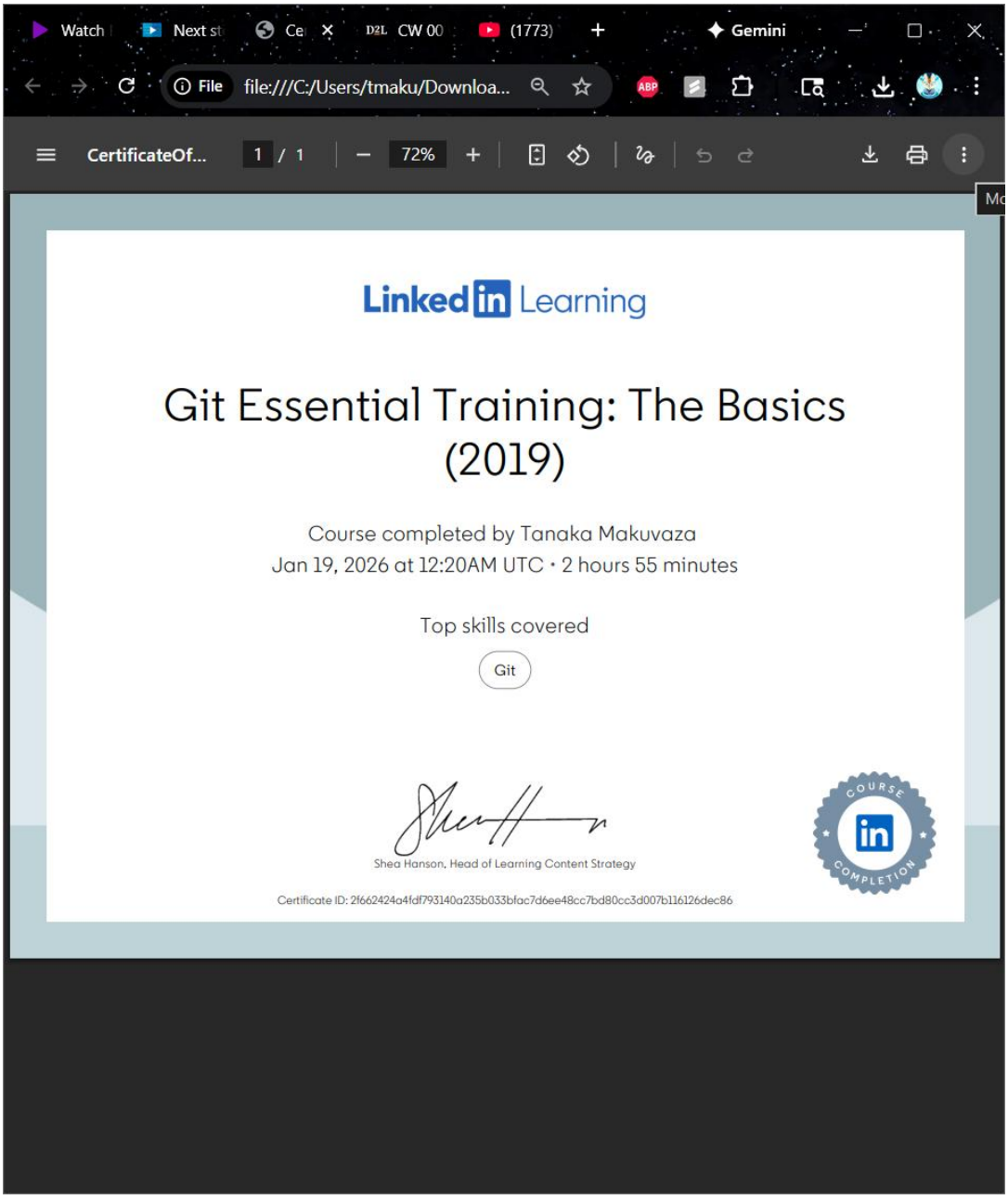
002252191

01/18/26

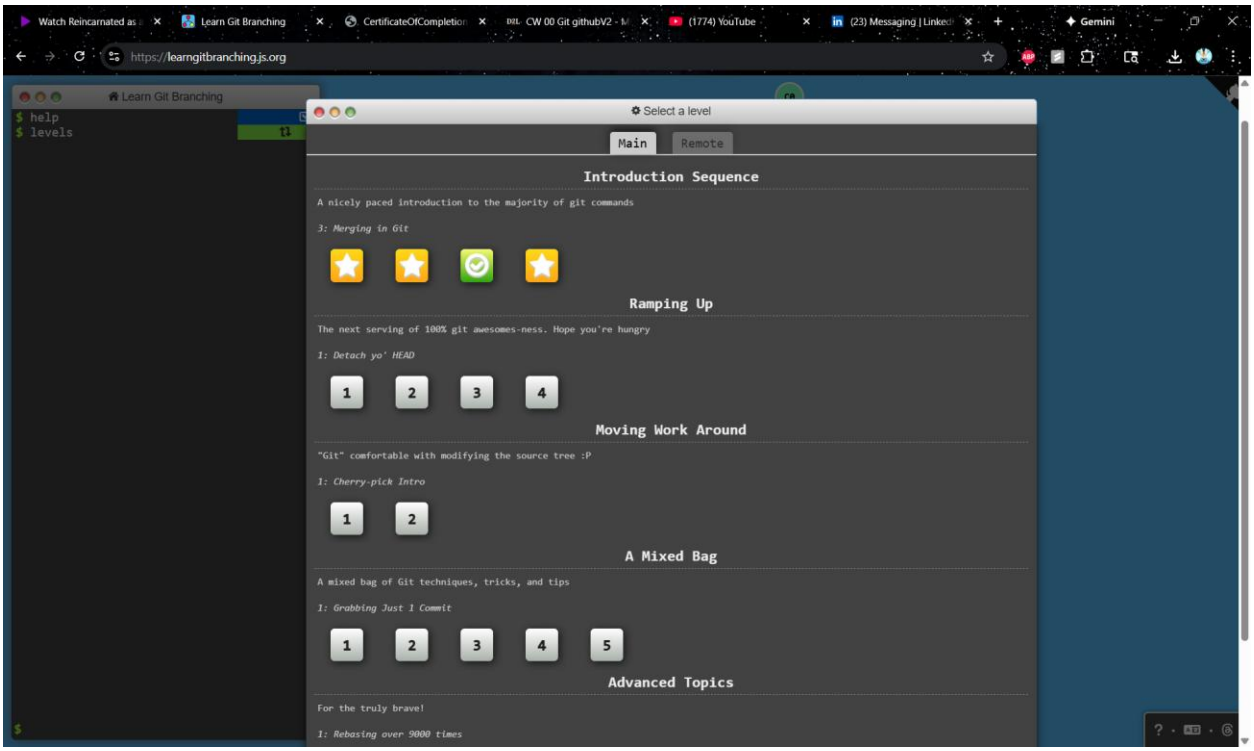
MAD 6360 MW 5:30

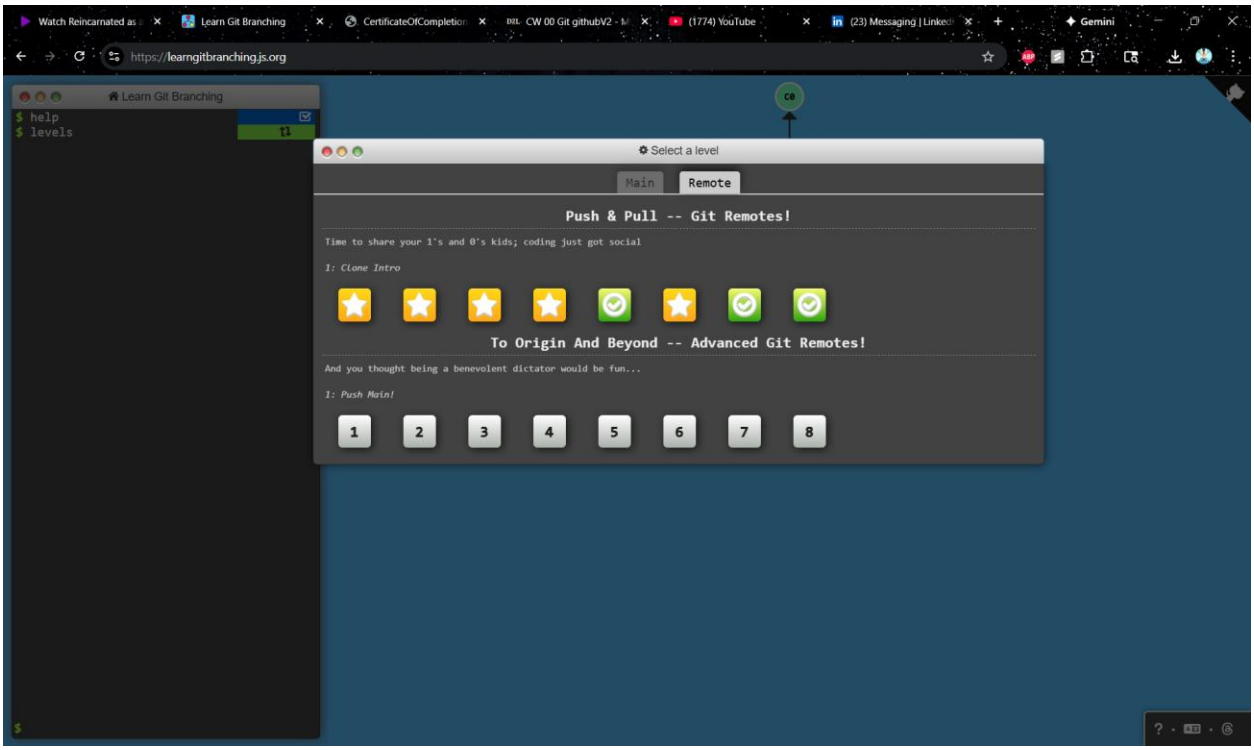
GitHub Assignment 1

1-3.Git Essesntial

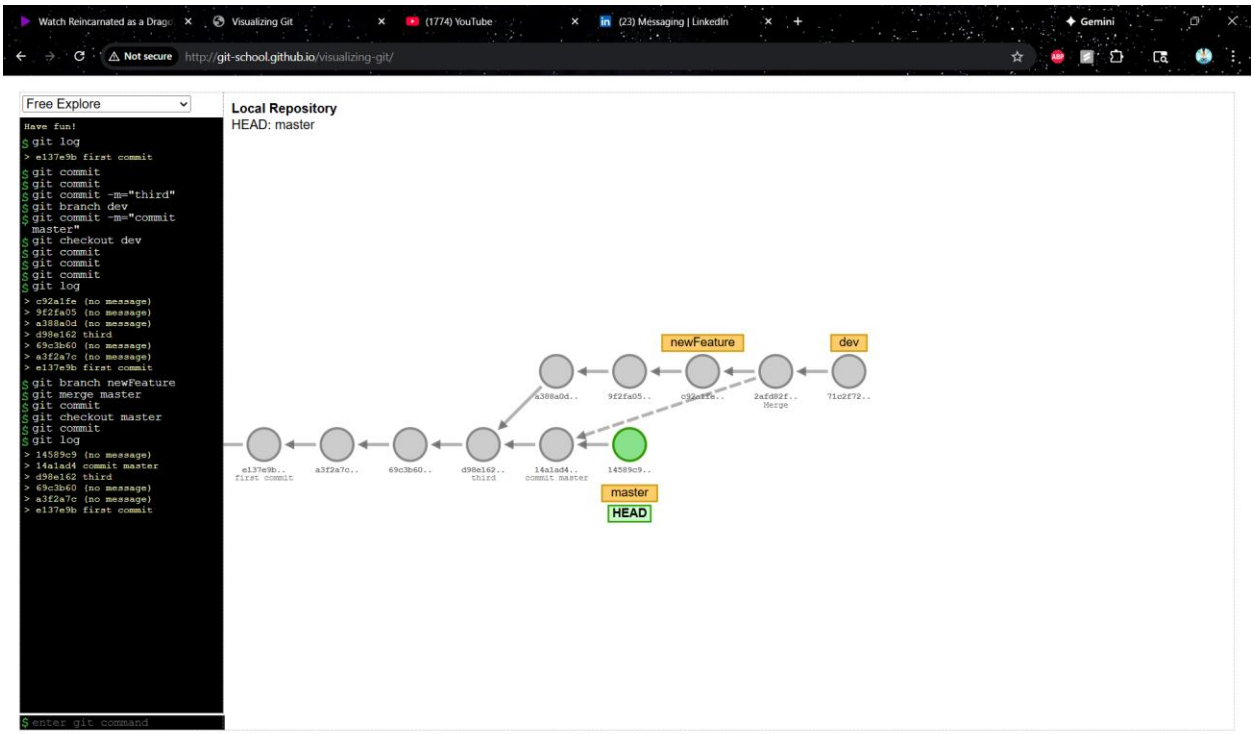


4.Learning Git Branching





5.Visualizing Git



6. Questions

A) What is a .gitignore file, and why is it important in a Git project? Provide examples of files or directories you might include in it.

A .gitignore file is a file that defines rules for how git tracks files for source control, and any files or folders included in this file will be ignored or not tracked for changes by git. You would keep files that you don't wan't to track changes of like log directories because they are constantly changing, or large packages or compressed files that you can't really track what's going on inside of, or compiled source code, operating system generated files, or user uploaded assets are examples of files you would include in it.

B) Which Git command allows you to compare the differences between two commits? Explain how you would use it with commit hashes.

Git diff. git diff <old_commit_hash> <new_commit_hash>

C) What is the HEAD pointer in Git? Describe its role in navigating commit history and how it changes during operations like checkout, commit, or reset.

the HEAD pointer in Git is a reference to the **current commit** you are viewing or working on. It essentially tells Git, This is where I am right now. When you look at your code in an editor, you are looking at the version of the files pointed to by HEAD. When you make a new commit, Git looks at HEAD to determine what the *parent* of that new commit should be. The new commit is added directly after wherever HEAD is pointing.When a new Commit B is created (with A as its parent), and HEAD moves to point to Commit B. If you run git checkout main, HEAD points to the branch reference main. As main updates, HEAD updates with it. If you run git checkout <commit-hash>, HEAD points directly to that specific commit rather than a branch name. The git reset command moves the HEAD pointer backward (or to a specific commit) to undo changes.

D) Which Git command can you use to discard changes made to a file in your working directory? Explain what happens to the file after running the command.
git checkout -- <filename>. Git replaces the files in your folder with the files from the main branch and moves the HEAD pointer to main.

E) What Git command can you use to revert a specific commit? How does this differ from git reset?
git revert <commit_hash>. Instead of deleting the commit from history, git revert figures out how to undo the changes introduced by that specific commit and creates a **new commit** with the inverse changes. git revert preserves history, while git reset alters it.**After git revert C:** A -> B -> C -> D *(Commit D contains the reverse of C. History moves forward.)*
After git reset --hard B: A -> B *(Commit C is gone. History moved backward.)*

7.Github Username

https://github.com/Tmaku18