Kyle Geddes

10/29/2023

Probability and Applied Statistics

Github Assignment

Git workflow provides an organized, and efficient way to manage, collaborate on, and document projects that you are working on. Git supports command prompt integration on windows for an easy to understand process. Alternatively, most modern IDE’s allow for git integration and streamline the process even more. For example in my IDE of choice, Visual Studio Code, there is a tab dedicated to git, and allows for easy commits, pulls, merges, and branch separation. Once you wire your IDE to your GitHub account, you’ll always have it there ready to be used. If neither of these options suit you, you can always download the GitHub desktop client use the GUI presented to manage your repositories.

For starters, a repository is the working directory you are in. For example, on my GitHub I have a small number of repositories that I commit to and work in weekly. They allow for a simple, safe, and secure way to store your files, organize programs, and build projects with a team. Each repository should be equipped with a *readme.md* file which explains the purpose of the repository, and outlines each file within.

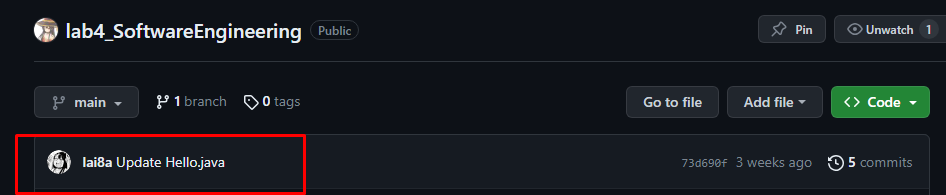
Each update you make to a repository is called a commit, and it’s pretty simple to understand. Commits can be made to side branches, in the case of local work, or the main branch when it’s time to update the entire repository. In a team, you’ll always have people working at their own pace and on different areas of a project – and git allows for more flexibility when working with others. Pushes are like commits, but they are to a local branch, and not updated in the main repository. Think about it like this: you push a change, realize it doesn’t work – and instead of ruining everything for everyone else, you can simply pull from the main repository again and undo the local changes.

When your coworkers or partners make a commit to a project, in order to receive the update on your local machine, you must perform a pull. A pull is the opposite of a commit – where instead of updating the repository, the repository updates the files on your local machine. This is especially useful to avoid conflicts, overriding, and losing data. Each time you begin work on a project that you haven’t been working on that day, it is always best to perform a pull, to make sure you’re on the latest version. A merge occurs when there is a source repository, and each member of the group may be working on their own repositories. This is useful in the case a large team, where each sub division is using their own repository, and the project lead is the only one allowed to merge onto the main repository.

Sometimes, problems arise – one such problem is called a merge conflict. This is when someone within the group attempts to commit outdated files to the main repository without pulling first. Git workflow has a counter to this built into it, so there is no need to worry about overriding something. If you get a merge conflict error, simply make the appropriate pull and commit your changes as you normally would.

Overall, git and git workflow make workplace collaboration more bearable than it otherwise would be. All team members can work at their own pace and make commits when they have requirements completed.

**Committing to another person’s repo:** 

**Another person committing to my repo:** 

**IDE Wired to GitHub:**

