**Gian Liu | Professor Hoy | GitHub Assignment | September 16, 2024**

**Git**

Git is a distributed version control software that allows developers to track changes in code without overwriting previous versions. Every developer working with a Git repository has a copy of that entire repository, enabling easy collaboration.

Git stores changes in SHA hashes, which work by compressing text files. That makes Git a very good version control system (VCS) for software programming, but not so good for binary files like images or videos.

**Git Workflow**

The main branch is usually called main. To work on another branch, creating a new branch off of main is required. The recommended branch naming structure should be based on the function or feature. Branches can be created by anyone for specific tasks, and multiple people can collaborate on a single branch. The new branch will be based on the current branch you are checked out to (where “HEAD” is pointing). Branches can be created from other branches, tags, or any commit.

**Commits**

To start a commit, use the command *git add [file]*. To make the commit, use the command *git commit –m “descriptive commit message”*. This process allows the developer to track modifications and maintain a clear history of the project’s development.

**Pushes**

After making the commit, the changes are only local to the person who made the commit. To let others see work and begin collaboration, use the command *git push*.

If pushing from a branch for the first time that is created locally, use the command *git push –u origin [branch name]*. This command tells Git to push the current branch, and create a branch on the remote that matches it with the same name. Additionally, it creates a relationship with that branch so that *git push* can be used in the future. By default, *git push* pushes the branch that the user is currently on.

If there has been a new commit on the branch on the *remote*, the push may not be able to go through. Use *git pull* to incorporate the changes on the remote to the local branch, resolve conflicts and finish the merge to push again.

**Pulls**

When pushing a branch for the first time, a pull request should be created. A pull request compares two branches, typically the *main* branch, and the feature branch. Pull requests allow for integrated testing and peer code reviews.

**Merges**

Merges integrate the feature branch into the other branch (most typically the *main* branch). Once the branch is updated, the pull request will be closed. After merging, it is advisable to delete the feature branch, as it is no longer needed. Branches are lightweight, and can be easily recreated from the latest commit on the main branch. A pull request can be closed without merging.

**Merge Conflicts**

Merge conflicts handling allows simultaneous edits by multiple developers. When working on a branch, developers can make necessary changes while knowing their work is secure.

**Repositories**

A repository is a storage location for code, files, and each file’s revision history. Repositories can have multiple collaborators and can either be public or private.