1. Version Control

2. Shell commands: pwd, ls, cd path, nano filename (read a file), echo "Review" > todo.txt - create new file,

echo "Review" >>todo.txt - edit existing file, git --version, ls -a(shows hidden files - .git), git add filename, git add . (all modified files and directories in current location), git commit -m "Review" - add a log message when committing, git status (which files are in the staging area and which files have changes that aren't in the staging area yet), git diff filename (comparing an unstaged file with the last commit), git diff -r HEAD filename (comparing a staged file with the last commit): -r - look at a particular revision of the file, HEAD=shortcut for most recent commit, git log - display all commits





git reset HEAD filename - unstage a single file

git reset HEAD - unstage all files

git checkout -- filename - undo changes to an unstaged file and revert it to its last commit state

git checkout . - undo changes to all unstaged files (. - refers to current directory)

git log -3 - shows the 3 most recent commits

git log -3 filename - commit history of a particular file

git log --since='Apr 2 2023'

git log --since='Apr 2 2023' --until='Apr 11 2023'

git clean -n - track unchanged files, git clean -f - delete unchanged files

3. Git Workflow - Modify a file, Save the draft, Commit the updated file, Repeat

4. Branches - systematically track multiple versions of files - avoid endless subdirectories, multiple users can work simultaneously, everything is tracked, minimize the risk of conflicting versions; One of the reasons Git is popular is its support for creating branches, which allows you to have multiple versions of your work, and lets you track each version systematically.

Each branch is like a parallel universe: changes you make in one branch do not affect other branches (until you merge them back together).

git branch

git checkout -b branchname

git checkout branchname

git diff branch1 branch2

git merge source destination

git commit -m "message"

git commit --amend -m "new message"

5. Creating repos - remote repo is the source of the truth for the project

git init reponame - creates a subdirectory in the diretory we were in when running the command

git clone reponame - clone locally

git clone URL - clone remote repo

git remote

git remote -v

fetching from a remote - git fetch remotename localname, synchronize content between the 2 repos: git merge remotename localname (remot is often ahead of local repos), fetch+merge=git pull remoterepo localrepo

git push remotename localname

6. GitHub - cloud-based hosting service



- repo will contain all files of a project and record past versions of files

- README files are like instruction manuals for repositories