2: What is Git

- need to practise for it to be helpful, otherwise it is clunky, like terminal.
- Git is a tool that allows developers to track versions of their code over time.
- in big companies there are many developers that are working with one code base with hundreds of files and thousands of lines. It is important to know what has changed and when, who changed it. It is good to compare changes together from one dev to another and to undo changes.
- using Git, you can make a snapshot of the current state of the code whenever you want to, not just one file, but maybe 100 or 1000 files.
- Git is essential when collaborating with other devs to ensure that the previous snapshots of the code can be revisited if necessary.
- lets say you make 1000's of changes but you want to go back to what you did before, git hub allows you to store all of that easily, so that you can go back(this is the snapshot).
- you do not need to use github to use git, but github is useful.

3,4:Git init and status,Git add and commit

- You have to select folders that you want git to track(basically allows you to set check points).

command (git init): it is used to initialize a repository**.** it is a good idea to use a different command first called **git status:** this checks the status of git in the file.

- you do not want to initialize a git repository(repo) inside of another git repository, git will get confused.
- the git file is hidden (use ls -a to see).
- to delete, you can just use rm -rf .git
- can make small check points rather than a bulk checkpoint that includes every single change.
- if you do git status in a file with content in it, it will show untracked files.

there are three conceptual areas:

1. **working directory:** just where you normally work, like the springboard folder.
    - As files are edited, git sees that the files are being modified but their changes are not recorded.

2. **staging area:**
    - git add is used to save your modified files so that they will be included in the next commit.
    - when you git add a file and use git status, it will show that file is being tracked, but not yet committed.
    - if you move a modified file to the staging area, you are preparing it to commit that state of the file at that point.

git commit(still in 2)

- once you are satisfied with the work of a file in you staging area, you can commit to the local repository
- include a commit message that summarizes the changes that were made.
- the message is connected to the commit with the -m flag.
- ex. git commit -m ‘summary’
- you should not make a commit every single time you make a new file.
- to commit an empty file use git commit -m ‘message’
- **commit log:** will show the commits that have been made.
- if you type git status after modifying the file, the file will show back up but this time, the modified file, whereas the un modified file is in commit log as it was before modification.
- if you have multiple files ready to be committed, use **git add .** instead of **git add index.html**
- when you git add at a certain point and change it after, the added file will still be of that previous one(before the change). When you type git status it will show that the file was modified, but the original git add is still intact, so that when you commit, it will commit the original git add unless you git add the modified file.
- **git diff:** if you modify a file and use this command, it will display what the modifications were. It will show what was added and removed