git, command line

week 2

Komal Kaur

what is git

- version control and tracking changes
- work with others remotely
- interact with terminal/command line and GitHub
  - need to know shell/unix commands to do this

example - shell commands

- i want to make a folder (repository) of notes for my classes

command	`pwd`	`cd`	`ls`	`mkdir`	`rm`	`mv`
meaning	print working directory	change directory	list	make directory	remove	move
what it does	tells you which directory you’re in	change to a different or home directory	list all items in current directory	create a new directory	delete files or directories	move or rename files and directories
common flags	`−L`: logical path `−P`: physical path	`−`: previous directory `~`: home directory `..`: go up one directory level	`−a`: lists all files `−l`: long format	`−p`: create parent directories as needed `−v`: verbose	`−r`: recursive (for directories) `−f`: force	`−i`: prompt before overwrite `−f`: force

live demo - shell commands

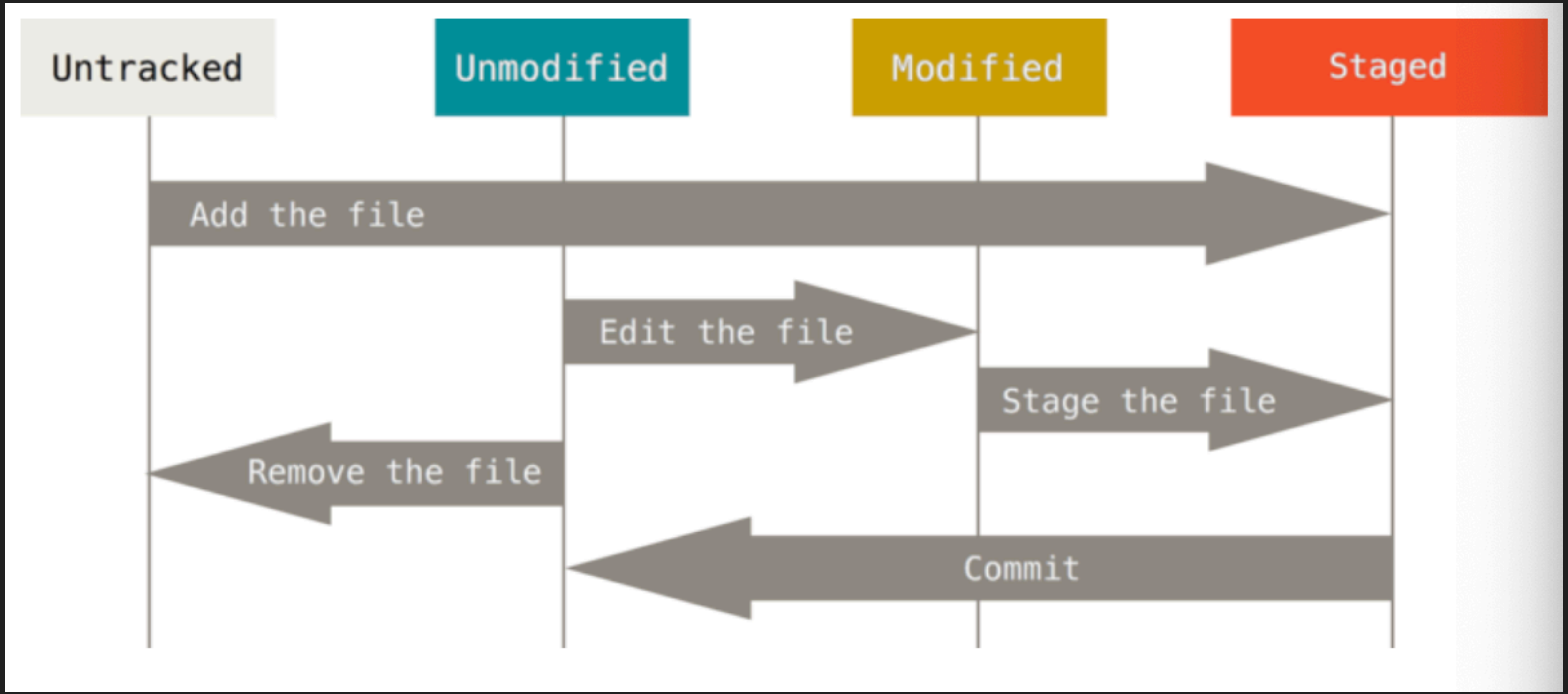
- not able to view this live? code is in the GitHub under week2 -> live-shell-demo

now what?

- we have our first version of our lecture notes done
- we want to save it somewhere and be able to track changes in the future and maybe even collaborate with others……sounds like git!!

git intro

- files are stored in repositories
- to stage a file for commit, use `git add`
- commit your changes with `git commit`
- push your commits to the cloud with `git push`



git setup

- open a command line
- set the name and email for git (create an account on GitHub if you don’t have one):
  - `git config -—global user.name “<your name here>”`
  - `git config -—global user.email “<your email here>”`
- set default branch name and merge strategy
  - `git config -—global init.defaultBranch main`
  - `git config --global pull.rebase false`
- set default text editor (i will support and use Vim but feel free to use others)
  - `git config --global core.editor "vim --nofork"`

git setup cont

- authenticate, SSH keys
  - create SSH key to allow your computer to push and pull
    - `ssh-keygen -t rsa -b 4096 -C “your email here”`
      - enter a password if you’ll remember it, otherwise hit enter
    - check if it worked with `ls -al ~/.ssh`
    - `eval "$(ssh-agent -s)”`
    - `ssh-add ~/.ssh/id_rsa`
    - `cat ~/.ssh/id_rsa`
      - copy the public part of the key
  - add key to GitHub
    - go to settings -> SSH and GPG keys -> new SSH key

rate my commit messages

message	good or bad
done	
?	
it doesn’t work	
add [REDACTED] file	
hold on	

message	good or bad
implement [REDACTED] feature	
attempt to [REDACTED] again	
plsplspls	
update [REDACTED].java for [REDACTED]	
add lab notebook	

rate my commit messages

message	good or bad
done	bad: done with what??
?	bad: ? ??
it doesn’t work	bad: what doesn’t work?
add [REDACTED] file	good
hold on	bad: what??

message	good or bad
implement [REDACTED] feature	good: detailed, specific
attempt to [REDACTED] again	good: specific goal
plsplspls	bad
update [REDACTED].java for [REDACTED]	good: specific goal
add lab notebook	good

practice

- clone the python bootcamp repo so you can have the files on hand
- let’s discuss how to do the following:
    - make a folder for this bootcamp in a location you want
    - clone the course repo from github (use `git clone [insert link]`)

git commands to know

- `git init`: create a repo
- `git add`: stage current version to be saved
- `git commit`: saves staged files
- `git push`: push committed changes to cloud
- `git pull`: pull current form of repo from cloud
- `git log`: list all commits
- `git show`: look at the specifics of a commit
- `git restore`: revert back to the versions of files in a commit but does not affect repo

references

- https://fa24.datastructur.es/resources/guides/git/#b-local-repositories-narrative-introduction
- https://www.w3schools.com/git/
- https://git-scm.com/book/en/v2/Getting-Started-What-is-Git?