GIT

1. What is git?

=>Git is version control system.

=> keep track of changes made to our code using special database called repositories.

1. Why Git?

=>keep track of our code

=>revert changes if needed

=>who made the changes

=>open source platform

=>mūltiple users can collaborate and work on same project by tracking changes.

1. How to access git?

=>modern IDE (vs code)

=> terminal/command prompt

=>GUI (gitkraken,sourcetree)

1. Install git
2. Git configuration?

=>we have to do 4 configuration

Username (git config --global user.name "name")

Email (git config --global user.email yashpatidar040@gmail.com)

Default editor ( git config --global core.editor "code --wait")

End of line ( git config --global core.autocrlf true/input) windows/mac

6. How to use?

=>Go to desired directory using cd command

=>Various commands

* git init : it will make a hidden repository of name .git
* git add filename : put the file into staging area.
* git commit -m : final commit (-m message to be given is optional)
* git status : shows the status of git

### We can skip staging area by directly commiting ( but not recommended)

For it

* git commit –am “message”

for deletion of file we have to remove it from directory as well as stag area.

* rm filename 🡪 it removes file from directory.
* git add filename 🡪 remove file from stag area.
* git commit 🡪 commit

git gives feature to directly delete files from both(dir and stag area) by using command

* git rm filename

to rename file

* mv file1.txt main.js ( rename file.txt to main.js)
* now we have to stag both the unstagged changes

git add file1.txt

git add main.js

git gives special command to rename file directly

* git mv file1.txt main.js

Skip file

Lets say we have a logs folder which have files we want to skip.

We use special file which has no extension .gitignore

Command :- echo logs/ > .gitignore

###if we have committed a dir then adding that directory to .gitignore will not work . it will keep track of it, so we have to remove file from only stagged area by using command

* git rm --cached filename (if recursively then add –r also)(it will remove file from staging area)
* now commit and git will not keep track of file.

Short status

Command:- git status –s

Display Two column LHS is staged area and RHS is filename..

Viewing staged and unstaged changes

git diff –staged (shows the exact line of code where changes are done in staging area)

in large files it display information in chunks with header .

Example) @@ -1,3 +1,5@@

-1,3 means in old copy starting from line 1 ,3 lines are extracted and shown.

+1,5 means in new copy starting from line 1 , 5 lines are extracted and shown.

git diff =>shows unstagged changes , shows changes in our working directory that are not staged.

We can use diff tools also such as

* p4Merge
* VScode

For using vscode as diff tool we need to configure it

* git config --global diff.tool vscode
* git config --global difftool.vscode.cmd "code --wait --diff $LOCAL $REMOTE"(if not added in config file than go to git config file using git config --global -e and add manually.)
* git difftool(for changes in directory)
* git difftool –staged(staged changes).

Commit history

git log => shows detail history of commits.

git log --oneline => history of commit in short(latest commit first)

git log --oneline --reverse => initial commit first

git show uniqueidno/HEAD~1(shows detail about all the differences in file)

git show uniqueidno/HEAD~1 : path to file (shows final version what actually stored)

###

git show uniqueidno/HEAD~1 will show only differences but to see files and directories in commit we use

git ls-tree uniqueidno/HEAD~1

Unstaging Files

git restore --staged filename

Discarding local changes

git restore filename(to discard local changes , basically git gives us previous version which is in staged area but for new file git do not find previous version so we have to clean it using clean command)

git clean –f(forcefully otherwise will get fatal error)

restoring file to previous version

if you have commited changes but want previous version

so use command

git restore --source=HEAD~1 filepath (HEAD~1 basically tells previous version that is commited)