Git & Github

- Use of Git ?

Git is version controlled system.
It helps to keep track of files and changes made in a project when you switch from one version to newer version of project. And you can revert to any version when required.
it is Distributed VCS. (not centralised)
↓
everyone has local copy.

Github :- website which host git repository

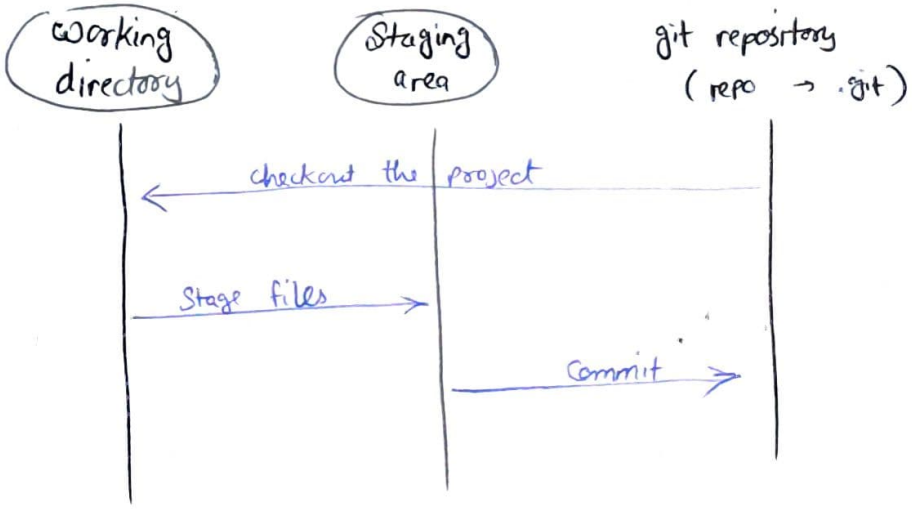
- shift + right click
  & open git bash (terminal) in required folder.

Configuration

1 time work

```
git config --global user.name "PALASH - BAJPAI"
git config --global user.email "palashbajpai45@gmail.com"
git config --list    # see configuration
```

- Git 3 stage architecture



Git in project

initialize project as git
git init

check status
git status

add files (all) to staging area
git add --a (a for all) / git add .

git commit with message
git commit -m "Initial commit"

---

see all commits
git log (q → to go out), git log --oneline

now let's made a change in file first.txt.
git add first.txt (added to staging area)

commit new change
git commit -m "changes in first.txt, added better design)

remove folder as git repo.
rm -rf .git

Cloning a git Project

- goto github repo, click on code & you will find HTTPS link, copy it.

git clone HTTPS link (will download folder in your device)

or

git clone link, P1 → to save with another name.

- gitignore (Ignoring files in git)

eg some files like error.log are generated themselves. & let we want to ignore all such files.

1) let ignore error.log file

touch .gitignore (make .gitignore file)

open this file & type error.log there.

add & commit gitignore.

- if wanna ignore all files with extension .log

so

inside gitignore → *.log

- Compare working directory & staging area

Show changes made

git diff

- Compare staging area with previous commit

git diff --staged

- git delete file from repo

1) can do manually, delete from folder

then add & commit

2) git rm third.txt (delete file & add to staging area)

now commit change

Same rename file

git mv first.txt first_renamed.txt

& commit change.

- Remove file from staging area

git rm --cached third.txt

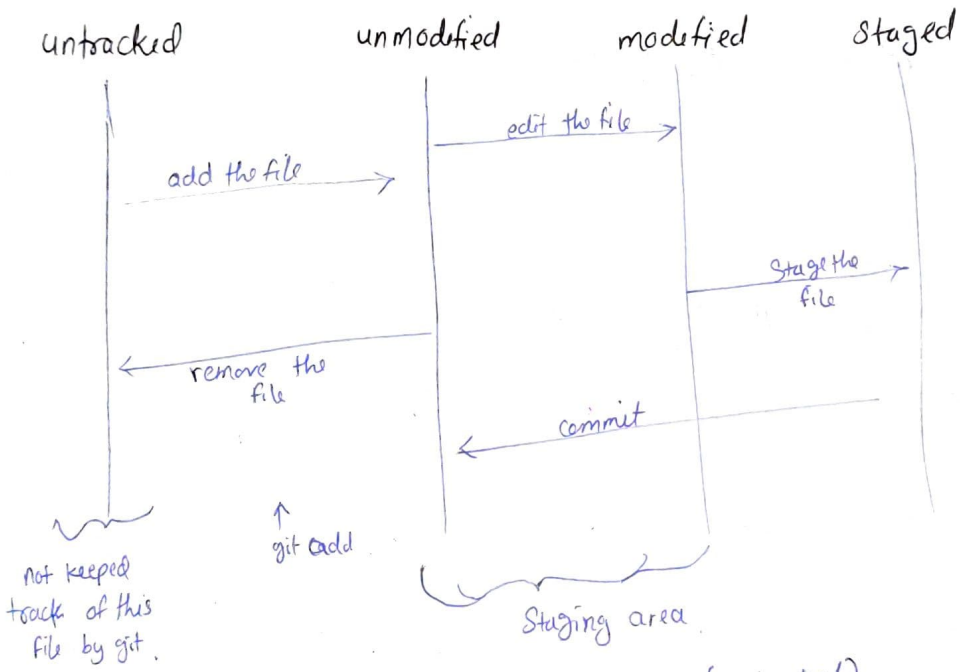
(goes to untracked)

Basic Linux command required

1) ls — list files
2) pwd — present working directory
3) cd ... — change directory
4) touch error.txt — create file named error.txt

file status Life cycle

we have .git folder hidden which keeps version control info



let [a.txt b.txt] (untracked)

↳ git add --a → unmodified: a.txt b.txt

↓ made change in a.txt

new git status

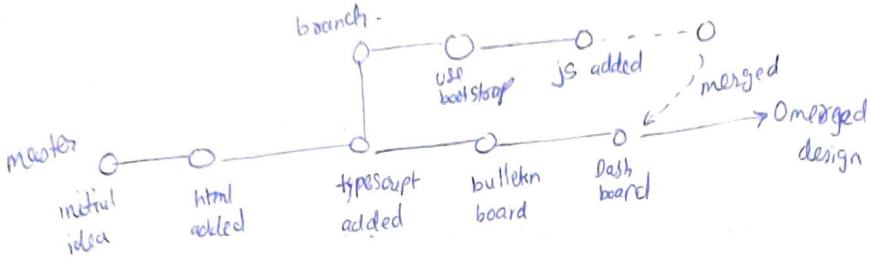
new file : a.txt
new file : b.txt
modified : a.txt
} modified

to consider change ← git add a.txt ← git status
new file : a.txt
new file : b.txt

Staged

↳ commit → git commit -m "..."

Creating & switching branches



- can even add branch to branch
- can make any number of branch.

- make new branch develop & start working on it

git checkout -b develop) make change.

now git add .

git commit "beautified structure"

→ if do git checkout master → or main (see from git branch)

folder changes to original one earlier befor changes made in branch

- git branch

know all branches

Working with github (github = host git file)

1) Make Repository with name
after this a document type is shown

- if already had git file, open bash in that folder
& copy command git remote add origin git@github ------.git

pushed → a file

2)nd git branch -M main

3rd) git push -u origin main (enter PALASH-BAJPAI)
& password (with capital & numbers)

→ for a first time you need to add SSH key to github
See in net how to add it

Consider changes of file in github too

1) git status (check changes)
2) git add .
3) git commit -m "add this .txt"
4) git push -u origin master

(changes made in github)

Git (summary)

1) git init (initialize project as git)
2) git status (check status)
3) git add --a (add all files to staging area)
4) git commit -m "added Readme.txt" (git commit with message)

5) git log (see all commits)
   gd log --oneline

6) git clone HTTPS Link (clone a repo)
   SSH link

7) git checkout -b develop (create & move to develop branch)
8) git branch (view all branches)
9) git push origin develop (push to develop branch)
10) git checkout develop (switch branch to develop)

11) git merge develop (merge develop branch to main)
Now in github
goto Pull requests → New pull request → Compare main & develop
→ View pull request → merge pull request.

12) * before pushing on any branch
git pull (to update local directory with directory of github)
& then push. (avoid using -f)
we lost old progress here
git pull origin develop
develop is branch name.