chat gpt.

learning github to upload projects.

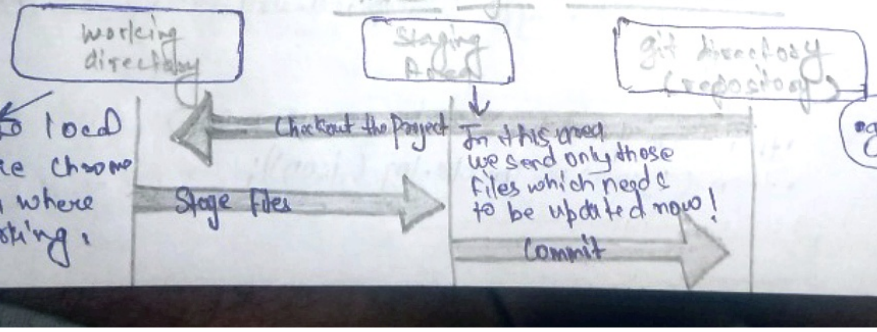
[Github] Tutorial (code with harry)

Git is a version control system (Distributed).
(1) It easily recovers files
(2) who introduced an issue and why?
(3) Rollback to previously working state.

History :

- Local VCS → keep tracks in local system (Hard disk)
- Centralised VCS → keep tracks in Server.
- Distributed VCS → can mirror complete repository. (have complete backup in all system)

Three stage Architecture



This is the local Area like chrome & explorer where dev. is working.

In this area we send only those files which need to be updated now!

Local operations

.git File contains all the versions and help to retrieve the data.

→ Set your identity

✓ $ git config --global user.name " " → set your username
✓ $ git config --global user.email " " → set your email.

→ Command to initialise repository :-

$ git init. (you need to select the path where your project is to start initialisation)

→ Command to check the status :-

$ git status

→ Command to sent files into staging area :-

$ git add --a (This will send all files into staging area)

→ Command to make a snapshot

$ git commit -m " Message ".

→ command to check all the commits :-

$ git log

→ Command to add particular file in staging area.

$ git add file name.

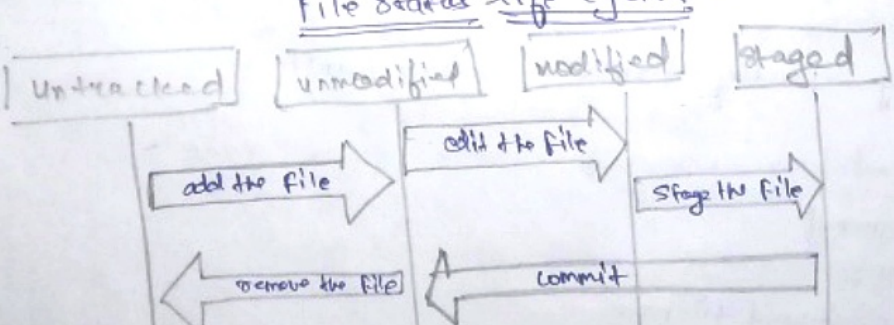
→ To delet complete git repository use :-

$ rm -rf .git

→ Command to clone git repository :-

$ git clone url, "New name" (if wanted)
space

File status life cycle :-



→ Command to make git ignore file

$ touch gitignore → whatever file you will add in this will get ignore by git and remain untracked.

→ blank folders are ignored bydefault

→ command to ignore particular extension file - write dir/file ext. name. (Inside git ignore)

→ command to ignore dir dir/ write inside git ignore.

→ command to compare working area and staging area

$ git diff

→ command to compare commits of staging area. (old & New)

$ git diff --staged

→ command to skip staging area

$ git commit -a -m "Message".

→ command to delete a file :-

$ git rm filename.extension.

→ command to rename a file :

$ git mv oldfilename Newfilename.

→ command to untrack a file

$ git rm --cached File name.

→ command to view commits with changes :-

$ git log -p

→ command to view commits in short and brief

$ git log --stat

→ command to view all the commits in one line

$ git log --pretty = oneline

in short view ⇒ $ git log --pretty = short

to view author ⇒ $ git log --pretty = full.

→ command to view commits of particular time / day / week / year

$ git log --since = quantity.day / week / year etc.

→ commands to view hash along with author name :-

$ git log --pretty = format : "%h -- %an"

abbreviated hash

author name

→ command to amend previous commit

$git commit -amend

→ command to unstage any file.

$ git restore --staged filename.

→ command to match the previous commit and undo recent changes,

$git Checkout -- Filename.

→ command to undo all the changes and match previous commit

$ git checkout -f.

→ command to add files on github

$ git push -u origin master.

→ command to set remote repository and origin

```
$ git remote add origin git@github.com:Username/Repository.git.
```

→ command to set remote repository url after setting ssh keys

```
$ git remote set-url origin git@github.com: Username/Repository.git.
```

→ command to set the alias for the git commands

```
$ git config --global alias.st status
                            ↓     ↓
                   alias name   command to be aliased.
```

→ command to restore changes in the current working directory

```
$ git restore <filename>
```

→ command to make new branch

```
$ git checkout -b branchname.
```

→ command to get back to the master branch

```
$ git checkout master.
```

→ command to view all the branches

```
$ git branch
```

→ command to switch between branches

```
$ git checkout branchname
```

→ command to merge developer branch into main branch

```
$ git merge Branchname (but switch to main branch before)
```

→ command to show branches with all last commits

```
$ git branch -v
```

→ command to view all merged branches & non merge branches

```
$ git branch --merged
$ git branch --no-merged
```

→ command to delete branch

```
$ git branch -d develop (It will give error message if the branch is unmerged)
             -D (It will not give error message even if it is unmerged.
```

→ command to make branch on profile (Github)

```
$ git push origin bugfix → branchname
```

→ command to delete branch from profile (Github)

```
$ git push -d origin branchname
```