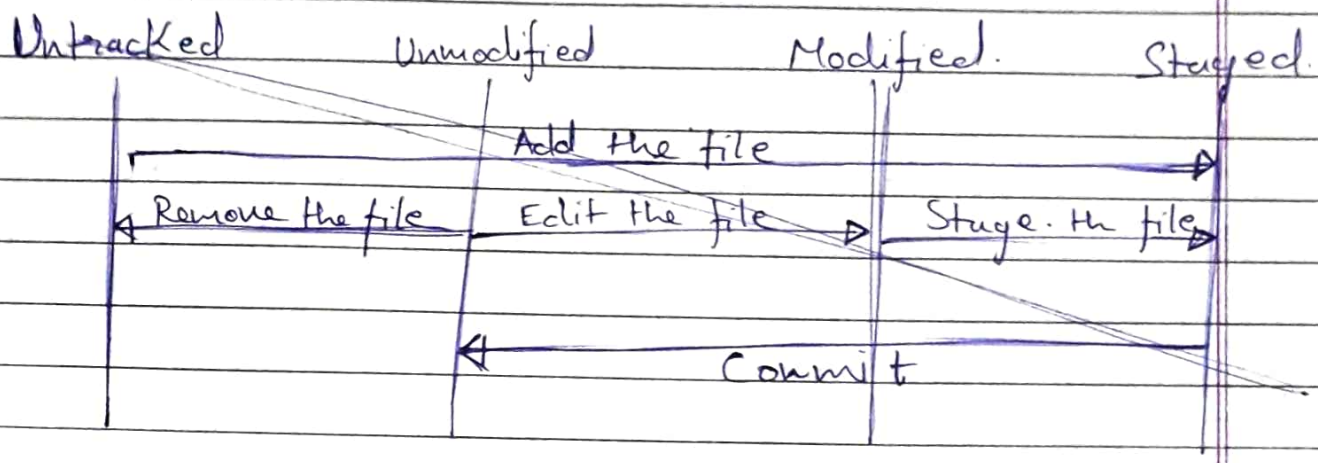
Git Workflow



* Untracked :- no relation of the Git with the file.

~~Staged is~~

Config. Git :-

$ git config --global user.name <username>

$ git config --global user.email <example@gmail.com>

VS code

~~Code~~

code . $ code .

Initialization.

to initialize git repo :-

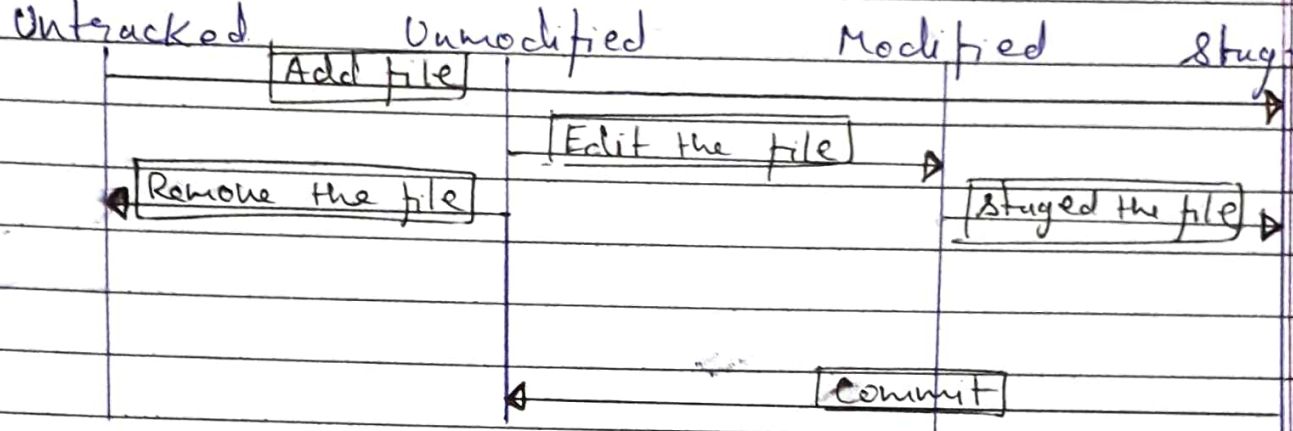
first open the directory in git bash then.

~~git init~~

git init $ git init

git status to check status of the file
$ git status

git add add to staging area ready to Commit.



$ git add <filename>

$ git commit

~~Aftt~~ After that name the commit
then **:wq**

$ touch <filename>
to create blank files.

$ git add -A
to add all files to unstaged present in the directory

$ git commit -m "anyname_name_the_edit"

$ git checkout <file name>

* to match with the last commit
* last commit ke baad ke saare changes revert karke files ko us hi position pe la deta h. Jaha vo tha uska pehle wale commit me thi.

$ git checkout -f

* to match all files in directory with the last commit.

$ git log.

to view the updates done to the file with Date and time and who did it.

$ git log -p -1 (or any no).

to view specific no of updats also shows what changes are done.

$ git diff

shows compares the working tree (committed files) to Modified files and highlights the differences or changes made in the file.

$ git diff --staged

Compares the staged files to last commit.

$ git commit -a -m "name"

to directly commit all files in directory skpping staging.

$ git rm <filename>
delete file from commit as well as staged.

$ git rm --cached <filename>

$ ls (shows all files present in directory)

$ git status -s
(shows in short)

ignore.
create a file using touch

touch .gitignore

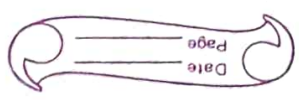
save it with the names of fillers which will by ignored by git every time while commit or push.

<foldername>/ add this to .gitignore for ignoring a folder.

Branch.

$ git branch <name-branch>
to create a branch.

$ git checkout feature1
~~swith~~
switch to feature1 (example branch.)



$ git merge feature 1
to merge feature1 in main Branch.

$ git checkout -b <branchname>
checkout and switch to a branch.

Github.

remote repos. repo present on github.

to add a remote repo.

$ git remote add origin <link_of_remote_repo>
example <remote-reponame>

# to view remote repo connected to git.
$ git remote.

# to view remote repo links.
$ git remote -v.

# Push.
$ git push <remote_repo_name> <branch>
$ git push -u <remote-repo-name> <branch>
# to change url in a existing remote repo
$ git remote set-url <remote-repo-name> <url>.