GIT

Git is a distributed version control system that tracks changes in any set of computer files, usually used for coordinating work among programmers collaboratively developing source code during software development.

Inorder to keep track of our commits i.e to check who has made certain changes we save our credentials in git by using below command:

>git config --global user.name Paras

>git config --global user.email [paras.juneja6021@gmail.com](mailto:paras.juneja6021@gmail.com)

To verify if the above commands executed successfully

>git config --global user.name

>git config --global user.email

In order to open code in VS code

We may use the below command after entering into that particular folder

>code .

There are 2 ways to access the repository:

1. Cloning – it is used to download code from the repo(mostly through the internet)
2. Init – it is used when the codebase needs to be initiated.

In order to initialize empty git repository, use this command(into the folder for which repository needs to be created.)

* git init

To view hidden folder we use this command

* ls -lart

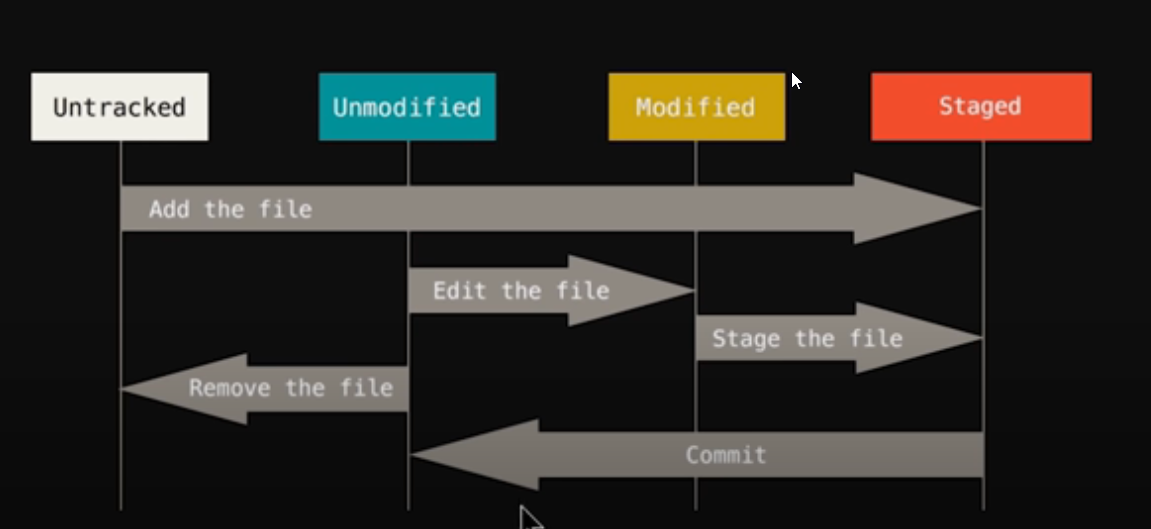
This command tells about the untracked changes made into the git folder(i.e whether any changes has been made into that folder like file creation, deletion, or any changes into that file).

* git status

**States of File**

There are 4 states of a file:

1. Untracked – These files are not tracked by git. Programmer does not want to track some kind of file i.e log file, binary file.
2. Staged – Staging area contains the file which needs to be committed.
3. Unmodified – When we commit any file which is staged, then the file is recorded by git and it is now unmodified state.
4. Modified – If any unmodified file is modified and moved to staging area, then it can be committed again.

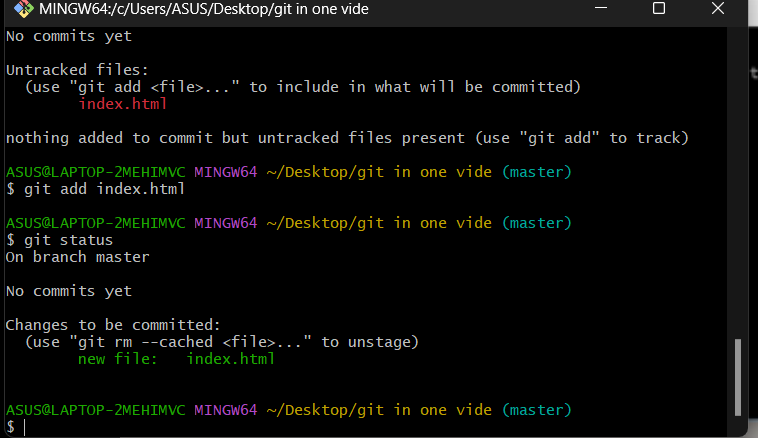


This command is used to move file to staging area

* git add <file\_name>

To move all the files to staging area

* git add -A



In order to commit changes initially for git repo, we use the below command

* git commit

This will open vim editor which we use to enter initial commit message.

An alternative way to commit

* git commit -m “commit message”

use this -m flag in order to avoid using vim editor.

* git commit -a -m “message”(to commit files directly without moving to staging area)

To create a blank file, use this command

* touch file\_name

To get the previous version of the file(in case any irrelevant changes are made on local)

* git checkout <file\_name>
* git checkout -f(to match all the files with previous commit)

To check all the commit

* git log
* git log -p -1(to check previous commit with details -1 can be changed with any number of commit)

To see the difference between the local and working tree provided file is not moved to the staging area(i.e changes that are present on local and not yet committed)

* git diff

If file is moved to staging area, then the above command will not show any difference.

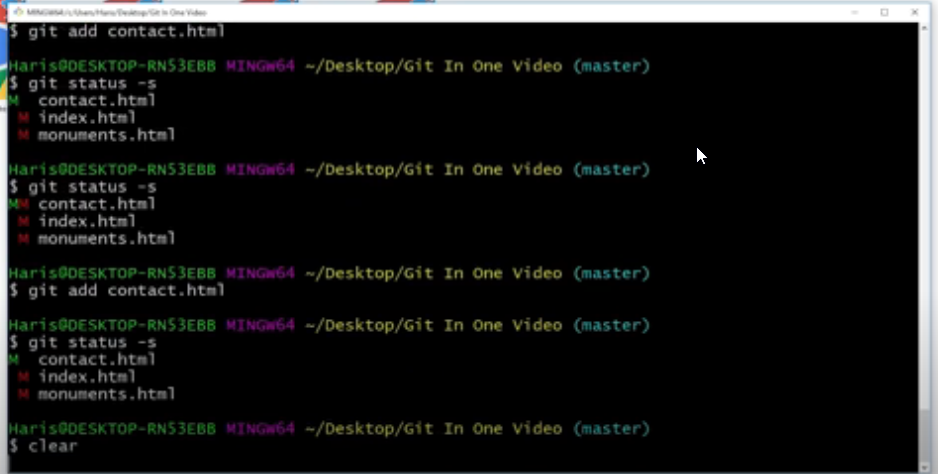
To see the difference between the staging area and the last commit

* git diff –staged

if you have committed any file by mistake and it needs to be removed

* git rm file\_name(to delete the file from working directory and git staging area)
* git rm --cached file-name(to delete the file from staging area and move it to untracked file).

There are two types of modified version as displayed below:



Green M denotes that these changes are added to the staging area.

Red M denotes that these changes are present in the working directory/ untracked changes.

In order to ignore files, we create a new file with the name .gitignore and mention the file names which needs to be ignored.

To ignore all the files with the same extension mention \*.extension\_name in .gitignore file.

To create a branch with git

* git branch branch\_name

in order to move to that branch

* git checkout branch\_name

To merge 2 branches, first, go to the master branch and use the below command

* git merge branch\_name

To check current branch(in which you are currently working)

* git branch

To create and move into a particular branch

* git checkout -b branch\_name

To push the local repository to a remote repository, we will make the connection between them.

* git remote add origin repository\_name

The url(repository\_name) can now be called with name origin.

To check remote connection with repo

* git remote

To check in which branch code fetched or pushed

* git remote -v

We may use the below steps to commit code to github

**…or create a new repository on the command line**

echo "# technical\_notes" >> README.md

git init

git add README.md

git commit -m "first commit"

git branch -M main

git remote add origin https://github.com/parasJune/technical\_notes.git

git push -u origin main

**…or push an existing repository from the command line**

git remote add origin https://github.com/parasJune/technical\_notes.git

git branch -M main

git push -u origin main

To push code

* git push

To get a clone of repository

* git clone repository\_link