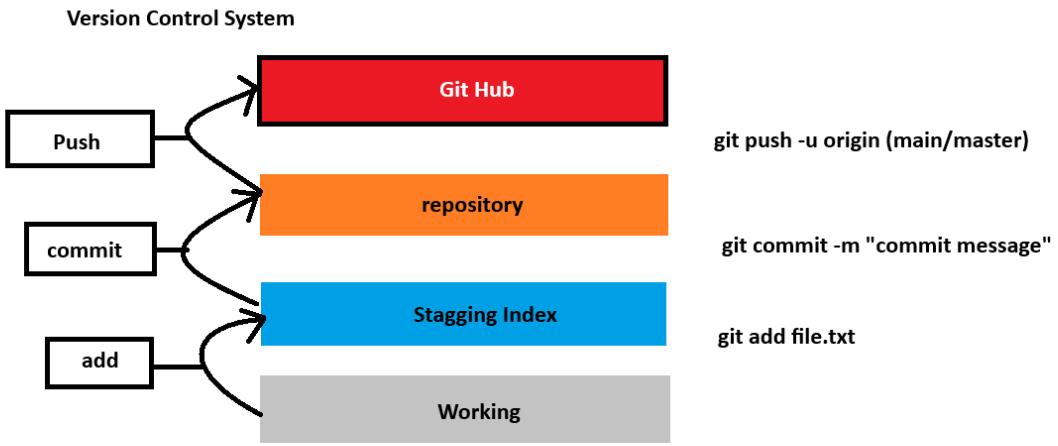
Version Control System

Git and Git Hub

- it is distributed Version Control System
- Tracks the changes in files and directories over time
- Helps to manage source code for software development

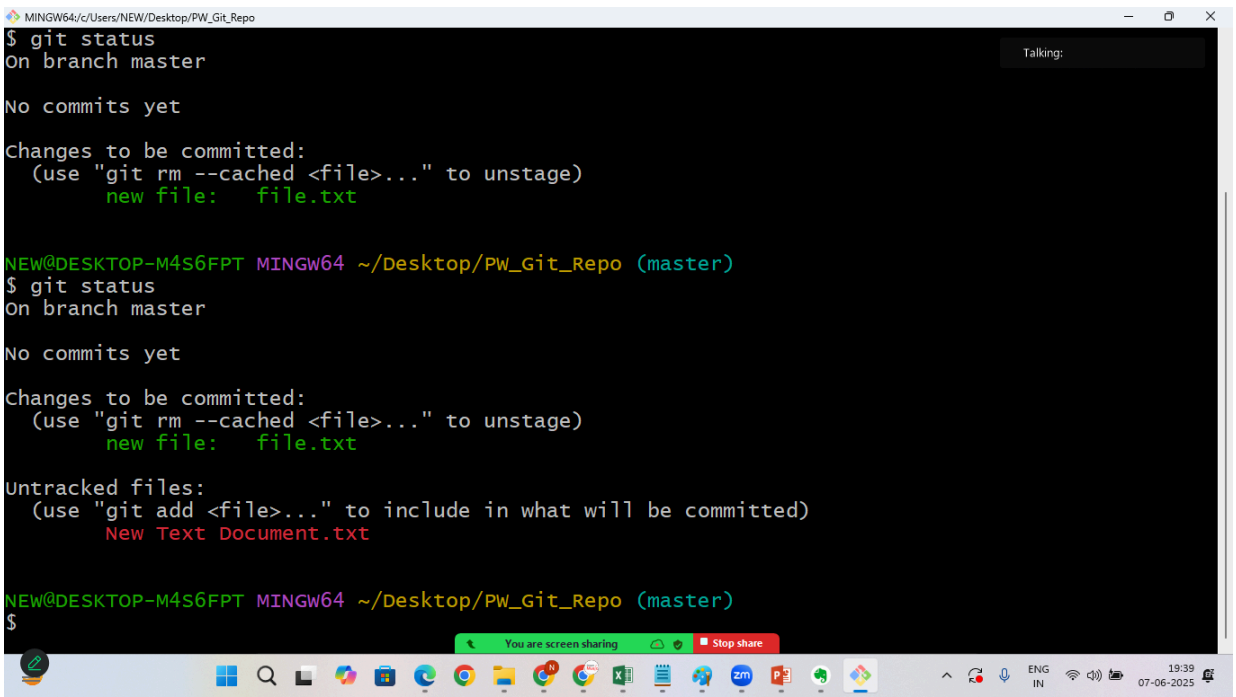
Git Terminology

1. Repository: A Space where code is Stored (Mainly open Source code).
2. commits: every commits creates a snapshot of a given file or directory that we can compare at any time



1. Download the GitBash from : https://git-scm.com/downloads
2. Login to the git hub and create the account, ensure you remember email,password and username for future reference
3. do the global configuration
   a. check the list of global users

    i. **git config --global --list**
  b. if the username and email is not coming you can set this up using below commands
    i. git config --global user.name "your username"
    ii. git config --global user.email "yourEmail@gmail.com"
4. create folder in your local repository
5. open the git bash in that folder
6. initialized and empty git repository :
   - **git init**
7. you will see .git folder in that
8. create one file named **file.txt**
   a. **file.txt**
9. add the file to the stagging are
   a. **git add <filename>**



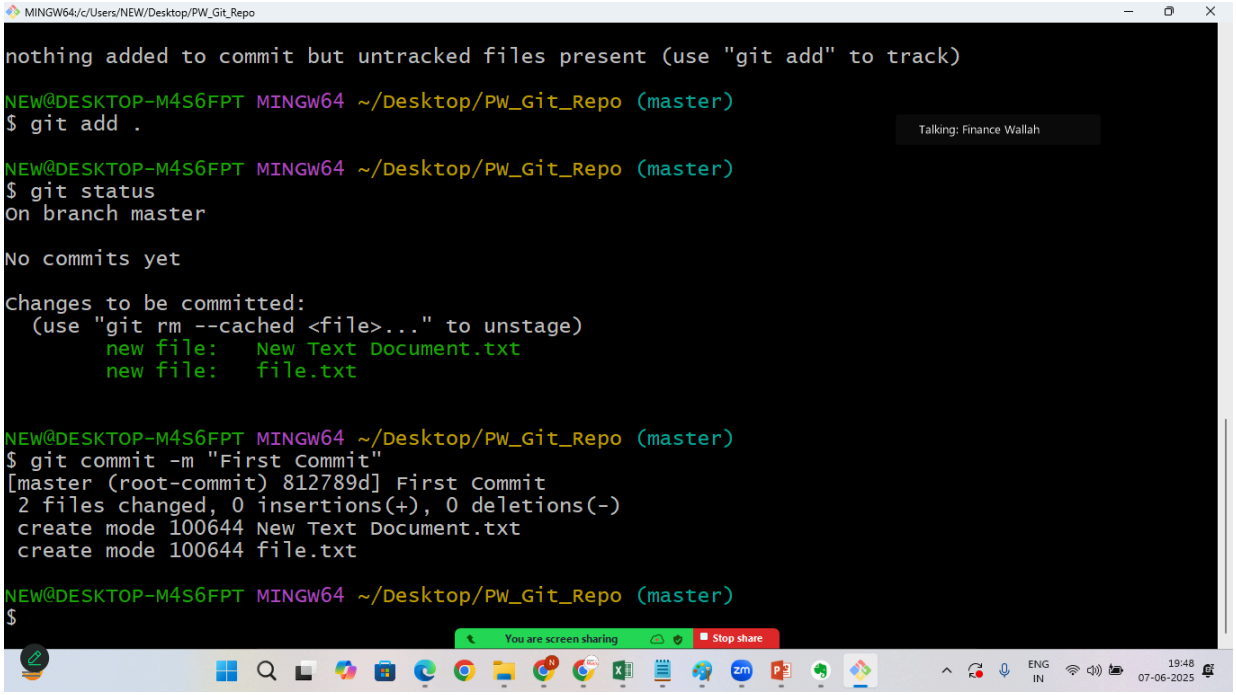
- The file shown in green colour is added to stagging where as the red colored files are considered as untracked files
- to revert back from the stagging are the command is
  - **git rm --cached file.txt**

```
MINGW64:/c/Users/NEW/Desktop/PW_Git_Repo
  (use "git rm --cached <file>..." to unstage)
        new file:   file.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        New Text Document.txt


NEW@DESKTOP-M4S6FPT MINGW64 ~/Desktop/PW_Git_Repo (master)
$ git rm --cached file.txt
rm 'file.txt'

NEW@DESKTOP-M4S6FPT MINGW64 ~/Desktop/PW_Git_Repo (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        New Text Document.txt
        file.txt

nothing added to commit but untracked files present (use "git add" to track)

NEW@DESKTOP-M4S6FPT MINGW64 ~/Desktop/PW_Git_Repo (master)
$
```

10. add all untracked files to the stagging are
    a. git add . (. represents all the files)

```
MINGW64:/c/Users/NEW/Desktop/PW_Git_Repo
No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        New Text Document.txt
        file.txt

nothing added to commit but untracked files present (use "git add" to track)

NEW@DESKTOP-M4S6FPT MINGW64 ~/Desktop/PW_Git_Repo (master)
$ git add .

NEW@DESKTOP-M4S6FPT MINGW64 ~/Desktop/PW_Git_Repo (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   New Text Document.txt
        new file:   file.txt


NEW@DESKTOP-M4S6FPT MINGW64 ~/Desktop/PW_Git_Repo (master)
$
```

11. check the status
    a. git status
12. do the commit to get the files ready for upload and create snapshot
    a. git commit -m "your message"



13. check the logs for stagging
    a. git log (full detailed log will be printed)
    b. git log --oneline (oneline log will be printed)
14. Create the same process again from adding the new files

Master:

- it is a default branch
- it is used by CI tools for build and deployment
- it is followed by other repositories

Branch:

- it is a light weight working directory
- it has a stagging Area
- it works without impacting the master branch

Head:

- it is a pointer to the latest commit of the working branch
- it is present on every repository
- it will point to the latest commits during branch switch

Remote Repository:

- it is git repository on a network outside the local machine
- it can have more than one remote repositories pointing from the local repository

- it can be managed and reference with short names.

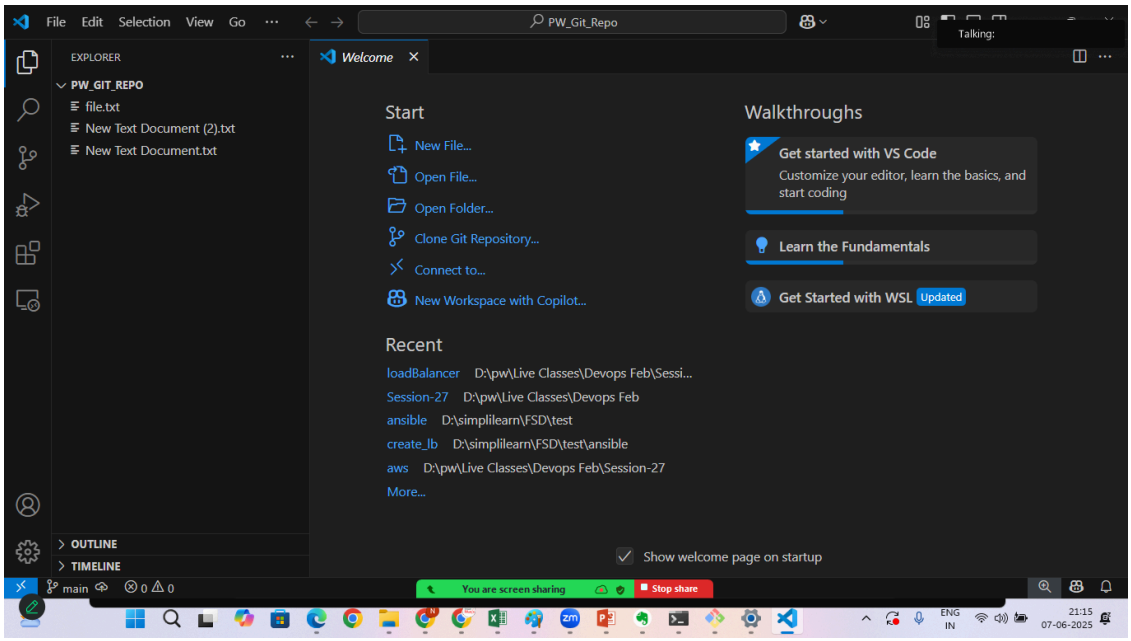
Push:

- it pushes changes from the local Repository to the remote repository
- it is performed after committing the changes to the local repository.
- it sync the changes with the local and remote repository

Git Difference :

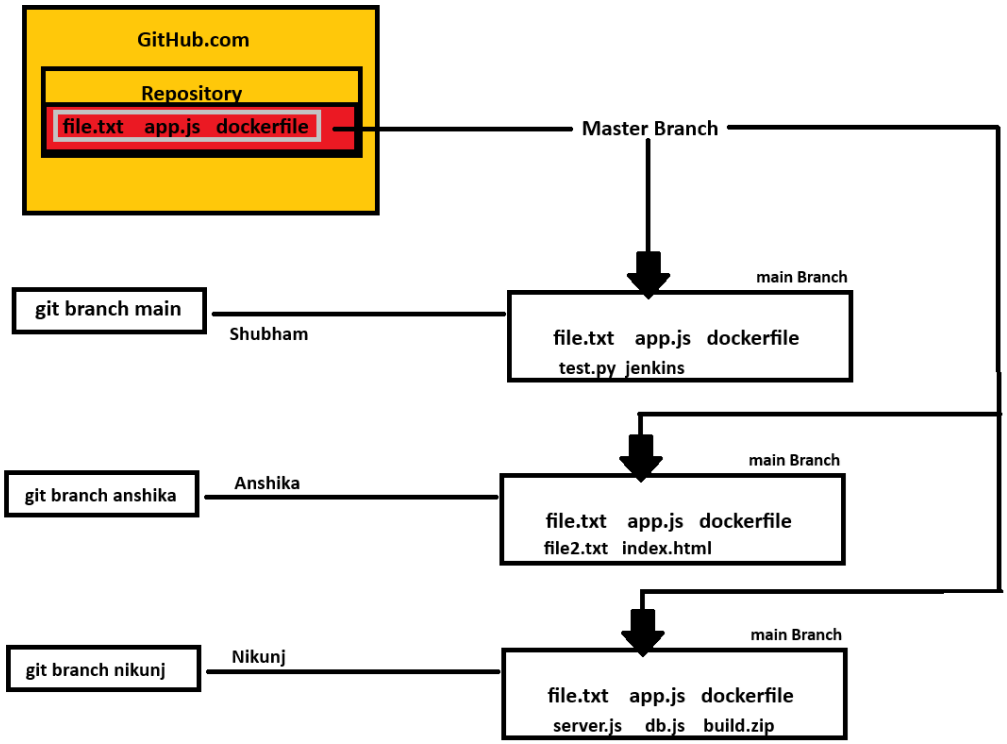
- git diff <first commit> <second commit>

CREATING A BRANCH IN A GIT HUB

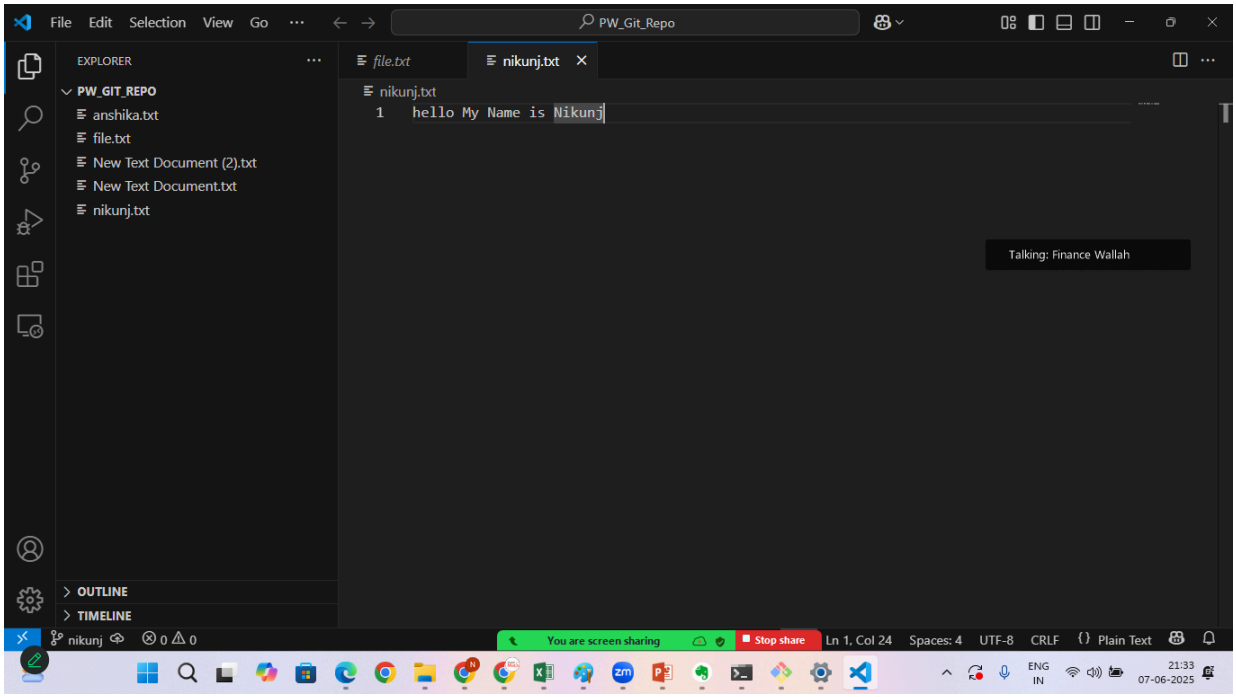
1. To get the available branch
   a. git branch
2. to create a new branch
   a. git branch <name of your branch>
3. to switch between a branches
   a. git checkout <name of the branch you want to switch>
   b. 

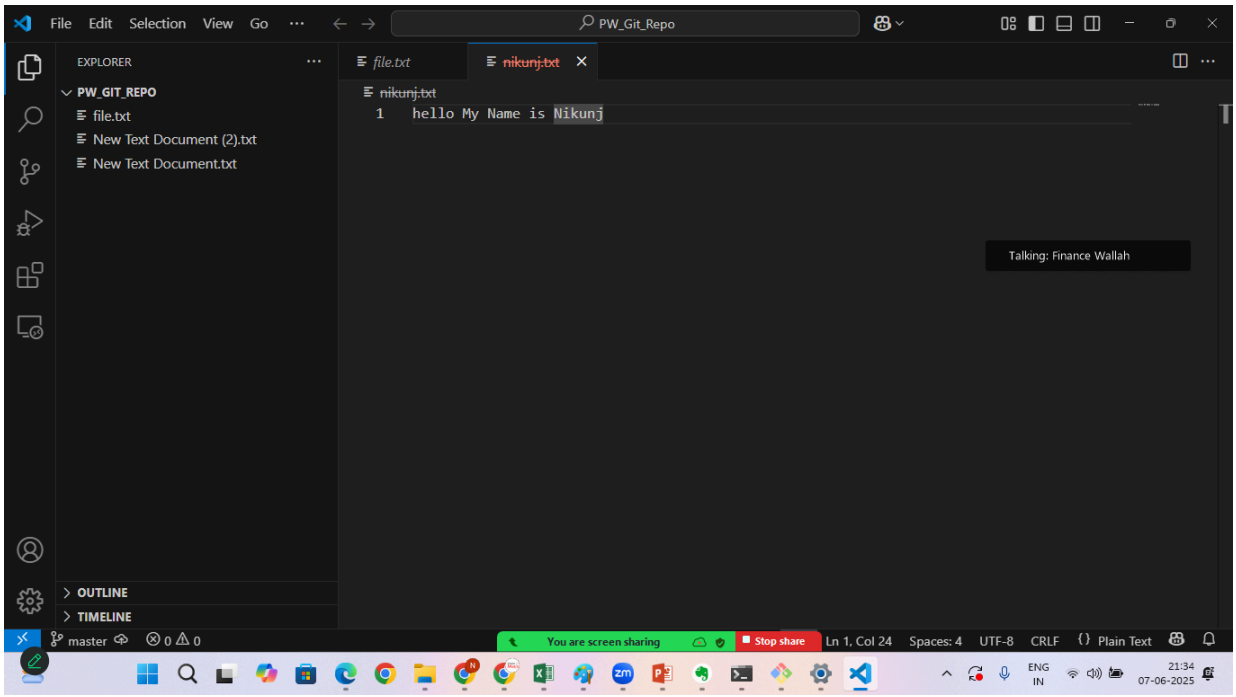
   c. when you create the new branch the content will be copied automatically to that branch

4.



5. create 3 -4 multiple branches and check the branch switching and files disappearing in different different branches via VS Code


File Edit Selection View Go
PW_Git_Repo
EXPLORER
PW_GIT_REPO
anshika.txt
file.txt
New Text Document (2).txt
New Text Document.txt
nikunj.txt
file.txt
nikunj.txt
nikunj.txt
1 hello My Name is Nikunj
Talking: Finance Wallah
OUTLINE
TIMELINE
nikunj
You are screen sharing
Stop share
Ln 1, Col 24
Spaces: 4
UTF-8
CRLF
Plain Text
ENG IN
21:33
07-06-2025


File Edit Selection View Go
PW_Git_Repo
EXPLORER
PW_GIT_REPO
file.txt
New Text Document (2).txt
New Text Document.txt
file.txt
nikunj.txt
nikunj.txt
1 hello My Name is Nikunj
Talking: Finance Wallah
OUTLINE
TIMELINE
master
You are screen sharing
Stop share
Ln 1, Col 24
Spaces: 4
UTF-8
CRLF
Plain Text
ENG IN
21:34
07-06-2025