

Git Tutorial :

Git is version control system it means it- Track and Manages changes in code and documents.
E.g: website ,desktop applin{we continuously make changes in web and application code}
Lets have simple example : prviously we was having windows 7 suppose it is of 2 Gb and now we have extended version that is windows 11 (10 GB) so we dont want prious windows 7 to be there in our pc so that case will require 12 GB so git help to mange the chages so that it will require only 10 Gb space.

Suppose we have 8GB space and An AI chatbot initially it contain F1,F2,data.csv as project.zip of 2GB and lets say it takes 2GB now we made some changes F1->F3 now we have saved it with projectn.zip again we have saved it with projectnm.zip in that case it will take 6 GB space so to avoid and manage changes in code we use git.

Advantages of Version Control System :

- 1)Easy File Recovery : suppose in project we have deleted two files and made new version and we want that files back then it is possible with the help of git.
- 2)Rollback to previous working stage : suppose someone made changes in working web commit and now it stop working so we can rollback to previous working stage.

VCS History :

Local VCS : db was used to track pros : can track file and rollback cons: if u lose hd then everything lost.

Centralized VCS :To avoid pb of local vcs(hd) centralized vcs is made.developer decided to maintain server and started pull files and push to centralized server.

Git works in distributed manner so we can avoid issue of centralized server as in centralized that if we lost connection we are unable to access our data but in distributed system wherever user are connected to server as they have full control or access of repository of project so everyone have backup backup of project.

In distributed vcs everyone have backup of project so if anyone lost it then it can be pulled by server.

github : is website that help to host git repository.->github,gitleab,bitbucket.

git features: fast ,snapshot captured ->if we made any chage it will capture snapshot
git folder contain history of project and we can extract perticular version from it -->if we want latestest version of project f1,f2,f3 it can give us or if we want files of project where chages made 3 year ago it can provide us the same so it will remove f1,f2,f3 from working directory and provide us f1,f2,f3 by pulling it from repository.

git is hidden folder that contain whole project history

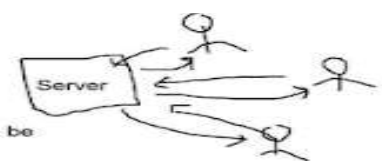
2)almost every operation is local then we push to the server to server(github). we can do whatever work on our pc and then we can push it to server.

3)remote means outside the local with the help of network we push project

4)git has integrity means it calculate checksum of current project and when it get reciver then it verify weather its checksum is same .if not means some third person or hacker made changes in file.

f1.mp4 -->checksum : 321h sent by harry and rohan download it and he will calculate checksum if it is same means no hacker or third person made chages in file

5)it generally adds data developed by linux



secure

GIT Bash : its command line tool through which we can run git commands on specific directory .it is helpful when we switch to mac or linux and it talkies commands in linux form.

Cd command we can use to switch to the folder can use git commands like push ,pull commit and clone for that folder

Name and email so that user from global can get to know who has made changes and they can contact.

```
Omkar@LAPTOP-04G033UL MINGW64 /d/GIT
$ git
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
          [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
          [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
          [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
          [--config-env=<name>=<envvar>] <command> [<args>]

These are common Git commands used in various situations:


start a working area (see also: git help tutorial)
  clone      Clone a repository into a new directory
  init       Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
  add        Add file contents to the index
  mv         Move or rename a file, a directory, or a symlink
  restore    Restore working tree files
  rm         Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
  bisect     Use binary search to find the commit that introduced a bug
  diff       Show changes between commits, commit and working tree, etc
  grep       Print lines matching a pattern
  log        Show commit logs
  show       Show various types of objects
  status     Show the working tree status

grow, mark and tweak your common history
  branch     List, create, or delete branches
  commit     Record changes to the repository
  merge      Join two or more development histories together
  rebase     Reapply commits on top of another base tip
  reset      Reset current HEAD to the specified state
  switch     Switch branches
  tag        Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
  fetch      Download objects and refs from another repository
  pull       Fetch from and integrate with another repository or a local branch
  push       Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
See 'git help git' for an overview of the system.
```

pwd :to get present working directory

/d/GIT

If omfolder is present inside GIT so we can write cd om

/d/GIT/om

If we write : /c

will go in c directory (bash always takes linux commands and its advantage is to go in any folder

-this help to run commands for that specific directory we can make any number of commit,rollback for that directory)

```

$ git help git
See 'git help git' for an overview of the system.

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT
$ git status
fatal: not a git repository (or any of the parent directories): .git

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT
$ pwd
/d/GIT

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT
$ cd om

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT/om
$

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT
$ git config --global user.name "Omkar"

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT
$ git config --global user.email "omkarlokhande425@gmail.com"

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT
$ git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/etc/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
user.name=Omkar
user.email=omkarlokhande425@gmail.com

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT
$ git config user.name
Omkar

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT
$ git config user.email
omkarlokhande425@gmail.com

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT
$

```

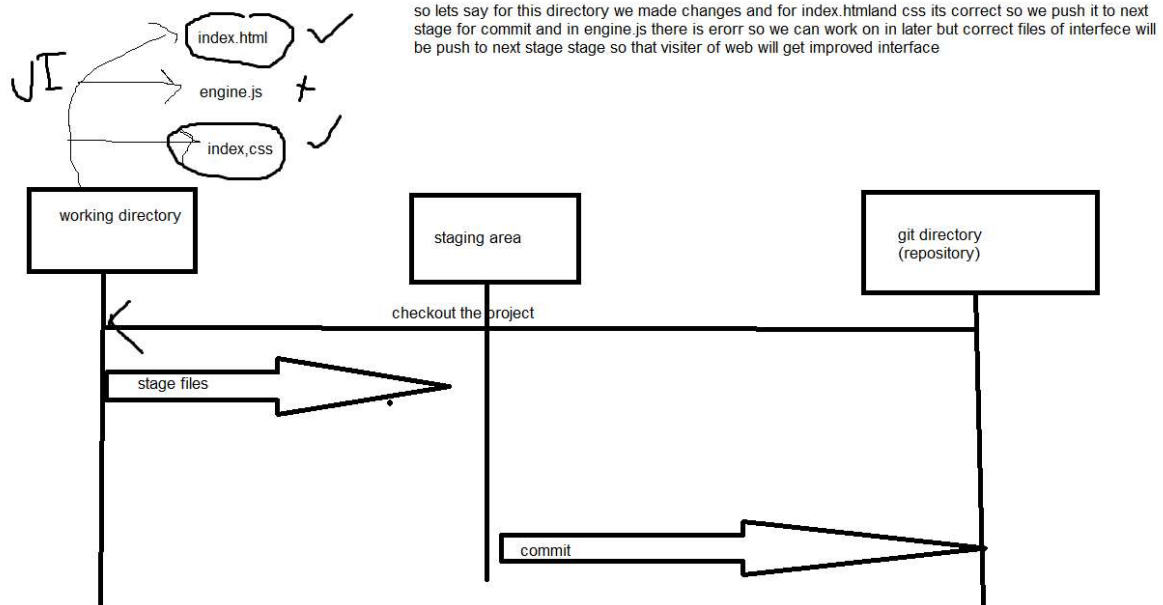
Git Workflow :

Suppose we have directory UI to improve where we have- index.html
 -static folder /temp
 -engine.js

Now we have made changes and created version v1 and again lets we are working on it and due to some error its not working properly so back to working version v1 and made commit with commit msg like working C1[version]

Suppose we want to improve UI for this we changed index file ,temp file but suppose we made changes in engine.js and it cause error but index,html changes are correct .so will wish

index.html to be kept in new snapshot (version) and engine.js to be in old snapshot [we don't want to introduce in new commit] so for this three stage archi is introduced.



For linux user we can set editor using command: with the help of this editors we can open any file in editors and work on it.

git config --global code.editor emacs

git config --global code.editor vim

Master : main branch

Git Commands :

git status : used to check status of git repository .if .git is not present in folder then it show it is not git repository.

git init : we can made that folder git repository by using this command.[convert folder into git repository]

git add --a : all files are added to staging area in one time.

git commit -m "msg" :for commit the repository .means record changes .

git log :get list of commit done so far.


```

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT
$ git status
fatal: not a git repository (or any of the parent directories): .git

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT
$ git init
Initialized empty Git repository in D:/GIT/.git/

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        db.accdb
        first.txt
        myxcel.xlsx

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

```

```

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git commit -m "Initial Commit"
[master (root-commit) 5ef7601] Initial Commit
3 files changed, 1 insertion(+)
create mode 100644 db.accdb
create mode 100644 first.txt
create mode 100644 myxcel.xlsx

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git status
On branch master
nothing to commit, working tree clean

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git log
commit 5ef7601220b2e2ff71c3ee8e89a14fbda4e39057 (HEAD -> master)
Author: Omkar <omkarlokhande425@gmail.com>
Date: Thu Jul 13 15:02:54 2023 +0530

    Initial Commit

```

Suppose we make change in first.txt file then it is tracked .

```

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   first.txt

no changes added to commit (use "git add" and/or "git commit -a")

```

We have db file we add some data to it it check log now it show both db and first.txt files are modified .

Now we want to commit only first.txt so will write command as -

```
Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git add first.txt

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   first.txt

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   db.accdb
```

It shows that we allow to stage first.txt for commit.

```
Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git commit -m "chnge first.txt and added better design"
[master 2455e48] chnged first.txt and added better design
 1 file changed, 2 insertions(+), 1 deletion(-)

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   db.accdb

no changes added to commit (use "git add" and/or "git commit -a")
```

Db.accdb will go in next commmit

```

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   db.accdb

no changes added to commit (use "git add" and/or "git commit -a")

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git add db.accdb

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   db.accdb

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        calc/

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git commit -m "second commit"
[master ffbcc21] second commit
1 file changed, 0 insertions(+), 0 deletions(-)

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        calc/

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

Omkar@LAPTOP-04G033UL MINGW64 /d/GIT (master)
$ git log
commit ffbcc216c1c5252800c3d8d7826cfd6cc3894274 (HEAD -> master)
Author: Omkar <omkarlokhande425@gmail.com>
Date: Thu Jul 13 15:34:20 2023 +0530

    second commit

commit 2455e48e8f12914365e76fe1325e7ce48f046df4
Author: Omkar <omkarlokhande425@gmail.com>
Date: Thu Jul 13 15:25:47 2023 +0530

    chnged first.txt and added better design

commit 5ef7601220b2e2ff71c3ee8e89a14fbda4e39057
Author: Omkar <omkarlokhande425@gmail.com>
Date: Thu Jul 13 15:02:54 2023 +0530

    Initial Commit

```