Initializing makes the repo.

Date: _____

- then he can push them to server.

- a folder in d drive → GitDemo commands
* (have to make this a repo of git) local repo using cmds

d drive → Git Demo
on Git Demo → git init (a repo is made)

- add a text file in GitDemo folder → first write something in the file → save and close.
- can check status using git status
on master branch.
→ for tracking file → add it
git add first.txt

git wants track all files in a folder
↳ git add .

→ open file & make changes in it

→ now our file is modified & in order to commit → first add (for staging)
git add. ↳ sending to staging area

(it was an untracked file → to track we had added it → its modified → then we modified → its in modified state → for staging → add again)

add → track untracked file
→ to stage file

whenever we'll make changes & before committing → add.

now commit → git commit -m "updated first"

- make 2 files → second & third &
check status
↳ now we've 2 untracked files

Date: _____

→ If we wanna track only one file (second) not third
→ make a file .gitignore
→ open .gitigore → add in there name of the file you wanna ignore → third.txt.
→ now third.txt will be ignored & status would show second.txt & .gitignore

↳ track these 2 → git add . (should have committed)

If wanna make branch here → git branch testing

→ now switch to this branch → (switch) git checkout testing

~~to~~ (now we're in our testing branch)

- commit files → git commit -m "updated second and git ignore"

→ make a file → newfeature.txt & write something
→ add & commit
↳ "updated new feature"

→ switch to master branch now
→ in Git Demo folder → only first.txt & third.txt
→ (gitignore & second.txt disappeared bec they weren't committed)
→ newfeature.txt disappears as it was in testing branch

→ now wanna merge changes of testing to master branch.

in cmd → git merge testing

(All files will appear in Git Demo)

Date: _____

(This all was done in Local repo)

Local repo → one person
when in team → GitHub

On GitHub

using cmd

→ make a new repo → Demo

a repository of Demo made in nawalkashif

→ want to bring this repo in our local machine
* ↳ CLONE
↳ copy link from code

→ make a folder in d drive → Second demo

→ in cmd → d drive → Second demo

→ in second demo → initialise git init (a repo is made)

→ now we wanna bring our remote repository (Demo) over here

↳ git clone url
→ now this Demo repo will be cloned in second demo

→ now in second demo → Demo → make a text file.txt & write something in it

→ in d drive → second demo → Demo
↳ git add first.txt
→ git commit -m "updated first.txt" Date:"

- (we are in main branch)

★ sending changes to GitHub account
↳ PUSH THEM

★ d drive → second demo → Demo
git push (changes made locally will be pushed to github's account.

→ open your account & refresh → will be able to see first.txt in Demo repo (online)

★ If we want to push a locally made repo in our account (GitDemo)

↳ go to your account → make a repo → git-demo
↳ don't initialise it
↳ will open a window → copy link from there

→ In cmd → d drive → GitDemo
→ In GitDemo
↳ git remote add git-demo url
name of repo

→ then do git remote -v
↳ will show locations for push & fetch
↳ it will let you know where will push & fetch be happening
also the name of online repo

→ now → ~~git push -u~~ git push online repo name branch name
~~↓ connects local repo with remote~~
gitdemo

→ open account & see
in git-demo → will have all changes of GitDemo
Date:
[local repo published in remote repo]

pull req is generated for where we want to take changes from & its shown on repo we took our changes to

forking
→ like stealing something
→ repo of tcs iba → we forked into our account
→ the repo will be forked to your account

* → (If wanna take nawalkaohit/ICI → local)
↳ git clone url

→ now whatever changes we'll make in this repo in local machine ICI → no changes will be there in online ICI

* ↳ If we wanna push changes of local ICI → nawalkaohit ICI

↳ will have to connect

* → git remote -v
(will tell from repo its connected)
* → git remote add url → will connect
online repo name
* → git push
① (↳ all changes from local ICI → nawalkaohit ICI)

* (If we want to take changes we are making in our account (nawalkaohit/ICI) to from where we forked (tcs-iba))

↳ will have to generate pull request

→ New pull request

↳ Base repo → from where we are sending our changes

Date: _____

Head repo → from where are the changes going

Base → tcsiba

head → nawalkaushik/ICI

→ select the one needed & create pull request

→ if wanna changes made by sir in our acc

→ go to ICI

→ create pull req

head → tcsiba/ICI

base → nawalkaushik/ICI

All changes made in tcsiba will come in my repo

→ want to changes from nawalkaushik/ICI → local ICI

↳ pull

↳ d drive → ICI → git pull

(All this can be done using GitHub Desktop as well)

↳ want to push a file from local (cloned) repo to online

→ select repo → add file → commit to main →

① pull origin → push

- any desk

for help
`git --help`

Git Bash

Date: _____

- git window
- to go to the desired folder
→ cd ___ / enter

↳ if repository folder in d: drive
cd d:
cd Repository/

* create a folder where you want to keep your repository

→ otherwise, always write git before giving any command.

In repository folder → git bash

→ to initialise → • git init

- git repository will be initialised.
- a folder .git created in repository

- a new word document named demo → create it in repository folder in windows!!
- open this file and write ' This is my first git Repository '

to add file in git bash → • git add demo.docx

now this becomes a tracked file by git

• now we have to commit these changes to the branch (master)

→ • git commit → for comments.

a dialogue box opens → close it and come to git bash again (open git bash again.

↳ perform all the steps again to reach your repository folder.

↳ all those changes made prev will hold → have to commit those.

→ • git commit -m "This is my first commit"

↳ gts a comment → can write anything

↳ when you revert back, you'll know what changes you made.

master
↳ branch

master → main branch

↳ when we'll branch it → original work remains → an exact copy of it will be created where we can make changes without affecting anything in main branch.

Date: _____

- create a branch in git bash
  - git branch firstBranch
    (name of branch)

github

-- → used to specify switches

→ help is a switch also a command.

→ to see your branches
  - git branch --list

- to go to the branch you created
  git switch firstBranch
  switched to first Branch

now come to windows → repository folder → open to demo file
(always make sure to save changes and close the file when you have done work on your file)

- in demo doc → write "This is my first Branch"
  ↳ *This command is being written in your first Branch.
  - close the file then.

↳ now in git bash → in first branch
  ~~↳ go to master~~
  ↳ add your file again.
    git add demo.docx

→ to update what'll be committed.
for file
- Add
- commit

→ now commit
  git commit -m "second commit"

→ now switch to master
  git switch master

↳ come back to your repo folder & open word file
  ↳ what you wrote in first Branch (This is my first Branch) won't be available.

- can switch to first branch to see the data you wrote prev is there but its not available in main branch.

Date: _____

- Pull & push → if copy of repository is on another machine.

→ now we want to merge the branches

On master

↳

```
git add demo.docx
git commit -m "just"  ☆
git merge firstBranch.
```

↳ check your word file → whatever you wrote in first branch will now be available in master (main branch)

→ when we make branches, make changes in them and merge to main → delete those branches.
~~those~~.

how to delete branch.

On master → git branch --delete firstBranch

first branch will be deleted.

- GitHub member. → search two

firstbranch
myfirstbranch

If command is written without --