**How to install git**

sudo apt-get install git

sudo add-apt-repository ppa:git-core/ppa

sudo update

sudo apt install git

**After that make a github account**

git config --global user.name "skysumitkumar"

git config --global user.email "sky3081999@gmail.com"

git config –list

**Then download github desktop**

sudo apt upgrade

sudo apt install wget apt-transport-https gnupg2 software-properties-common

wget https://github.com/shiftkey/desktop/releases/download/release-3.1.7-linux1/GitHubDesktop-linux-3.1.7-linux1.deb

sudo apt install -f ./GitHubDesktop-linux-3.1.7-linux1.deb

**Now commands in git**

**git init**->in any folder when you enter this command it convert that folder to as a git repository(.git hidden file)(ls -al to see the pointer in that file)

**git clone**->it is use to clone any repo of any person on our computer for this we use(git clone urlof repo newfoldername(jis name se aap us repo ko save karna chate h)/

**git diff**->To find what content we change in this file

**git log**->It give firstly top commit that you add then all other commits

**git log -n(eg. 1,2,3)** ->To see no of commit to that you want to see

**git log -p** -> It give top down commits with content that you change

**git log** -oneline ->It give all commits id with name of commits

**git log - -stat** ->It give filenames that we commits

**git show shawId(Id of commit)**->It give what commit changes in that perticular shawid

**git add filename**-> add file to track

New file->add file to track

Old file-> staging index(commit karne se phale)

**git add .** ->add all modified files to track(this is not recommended)

**git commit -m “file\_comment”** ->to commit our file and give it commit comment

**git restor filename**->restor file to latest commit

**.gitignore-**>In this file we give all file names that we want git should not track these file(eg \***.**txt**,**assets**/**images**/**\*.png )

**git branch-**>list all branches we have

**git branch name(name of new branch that you want to make)-**>to make a new branch

**git checkout branchname-**>to switch diff branch

**git checkout -b branchname** ->to make a new branch and also shift to that branch

**git commit -am “commit”** ->to add and commit our file

**git merge branchname-**>to merge branch to mainbranch

**git branch -d branchname** ->to delete branch

**Note->**when we merge 2 branch git confuse which i take and which i abort to solve this problem hame khud dekhna hoga ki hame file me keya chyea or keya nahi or phir use save kar dena h or git ko batane ke lyea ki hame confucion resolve kar deya h ham **git add** **filename** command ka use karange

**git tag -a tagname shawId -m “comment”** ->to tag a file or we can say it is our betaversion to test program

**git tag -d tagname** -> to delete tag

**git pull** -> it is use to update local repository to remote repository

**Note->**when we update our local repository and use git pull command to get updated file from remote than we have probem because git confuse to wich file he take our which not so for to solve this issue we use **git stash** command when we use this command our local file content is removed from file and save in some stash area to see the list of git stash we use **git stash list** after the git stash command when we use git pull it work it pull remote file in our local file but we also want our previous local file changes so to unstash our changes we use **git stash apply** then git also confuse which content it save or which he removed so than we update our file and remove confusion to told git i resolve issue we use **git add filename**

**git stash**->when we use this command our current file content is save in some stash area and removed from our current file

**git stash list** ->from this command we see the list of stash files

**git stash apply-**>when we use this command our stash file is merge in our original file than it create a issue to which content we save or which we removed than we update our file and resolve this issue and to told git i resolve the issue we use **git add** command

**git push -u origin master** -> push in master branch

**git revert showid** ->to revert our initial commit

1.Press Esc

2. :wq then press enter

**git reset - -soft showid** ->To move head to any commit and remove upper ommits

here modification will show as staged in staging index

**git reset - -mixed showid** ->when we use this head move to new commit but our local changes see as modification in files

**git reset --hard showid ->** when we use this head move to new commit and local changes are discarded

**git diff HEAD~**

**git commit --ammend ->**1.press i for edit

2.then press enter

3. :wq then press enter

**Note->**for push file in remote

step 1. git remote add origin <https://github.com/skysumitkumar/repository.git->>with this we map local branch to github remote branch

step 2. git config --global user.name “skysumitkumar”

step 3. git config --global user.email “[sky3081999@gmail.com](mailto:sky3081999@gmail.com)”

step 4. git push -u origin master

4.1 ask for username->username

4.2ask for password ->token no

**For GUI VERSION**

1.git desktop

2.sublime merge

**git rm --cached filename:-**to remove the file

**First time push data to github**

git config --global user.name "skysumitkumar"

git config --global user.email [sky3081999@gmail.com](mailto:sky3081999@gmail.com)

git remote orign

git init

git add .

git commit -m "image\_gallery"

git remote add origin https:///github.com/username/

git branch -m main

git push -u origin main

git fetch origin

git pull origin main

git pull origin main --allow-unrelated-histories

**Step-by-step Vim Instructions**

1. If you are editing the commit message:
   * Make your changes.
   * Press Esc to switch to command mode.
   * Type :wq and press Enter.
2. If you are keeping the default message:
   * Press Esc to switch to command mode.

git add .\README.md

git commit -m "image\_gallery"

git push origin main

**When some large file push than**

1.download this:- <https://adoptopenjdk.net/>

2.download this:- <https://repo1.maven.org/maven2/com/madgag/bfg/1.13.0/bfg-1.13.0.jar>

3.go to your git repository:- eg: cd D:\Study\Du\_Mca

4. **Move the JAR file to an accessible location:**

* For example, move it to D:\Downloads.

5. java -jar D:\Downloads\bfg-1.13.0.jar --strip-blobs-bigger-than 100M

6. git reflog expire --expire=now --all

git gc --prune=now –aggressive

7. git push --force