* If we want to view all of our settings and from where it comes (i.e. path) then following command is use:

git config --list --show-origin

* If we want to set out usename and email which will displayed in every comment then following command is use:

git config --global user.name "Saloni Sindhi"

git config --global user.email [sindhisaloni31@gmail.com](mailto:sindhisaloni31@gmail.com)

if we want to change out username or email for specific project then we can write the same command without –global in that project folder.

* If we want to use different editor than git default editor then we can do it using following command:

git config –global core.editor “c://program files/notepad.exe”

* If we want to change default branch name then following command is used:

git config --global init.defaultBranch main

* If we want to see list of all settings git can have then following command is used:

git config --list

* We can set configuration variable in three ways:
* --system 🡺 It will affect the changes to all the user available in the system and their repository
* --global 🡺 It will affect to only a user who makes changes , but affected to all their repositories
* --local 🡺 It will affect only to the specific repository in which that variables are set.
* If we want to see the manual of particular command then it can be possible in three ways:

git help add

git add –help

git add –help

git help -m add

git add –h //It will just show the available option for a git command

But here man is not found in bash , so how can we add man command in git bash?

* If we want to clone a repository with other name as a folder in local system then we can do it with following:

git clone [git@github.com:saloni31/Donation-Hub.git](mailto:git@github.com:saloni31/Donation-Hub.git) Donation Hub

* **git add command can be used for** : tracking new files, to staged files or marking merged conflicted files as resolve.
* If we want to compare what is in our working directory with what is in staged area following command is used:

git diff

git diff –staged //compare staged changes with our last commits

git diff –cached // show what we staged up to now

* Commit the files:

git commit 🡺 It will opened one editor in which git status output will be displayed as commit message

git commit –v 🡺 It will also open one editor with git status output and also include output of git diff command as a commit message.

git commit -m “message” 🡺 It will directly specify commit message

git commit –a –m “message” 🡺 It will skip staging area (i.e. we didn’t need to execute git add command it will be automatically added)

* Remove the files:

git rm index.php 🡺 remove from working area as well as staged area

git rm –cached index.php 🡺 remove from staged area but not from working area(i.e. your disk)

If we already modified the file or staged the file then we forcefully remove it with –f option

* Move file or rename file:

git mv “index.php” “demo.php”

After executing this command we have to again commit and push the project to our repo

* Show logs of repository:

git log // It will show the full log of repo

git log –p // -p indicates patch it will show the committed history with lines modifications

git log –stat // it will show each commit entry a list of modified files, how many files were changed, and how many lines in those files were added and removed.

git log –pretty=oneline //display the history of all commits in one line

git log –pretty=format //display the logs with specific format

* How to create branch and perform other operations on it:

git branch [branch-name] //used to create a branch

git branch // used to list out branches

git branch –d [branch-name] // used to remove branch

git branch mv [old-branch-name] [new-branch-name]

//used to rename the branch

git checkout [branch-name] //used to switch to the branch

* If we want to unstaging the staged file then following command is used:

git reset HEAD demo.php

The same thing we can perform using git restore command also in latest versions:

git restore –staged demo.php

* If we want to see all the information about the remote server then use following:

git remote show origin

* Rename remote:

git remote rename origin saloni

* Merge commit the temp branch with main branch:

git merge --squash origin/temp

--squash is used to combines all the integrated changes into one commit

* Revert last commit

git revert -1 ae29c1728c6dc490dfb341f2e68bfb60f4030de3