GIT global information tracker

* Git also called as version control system (VCS). A Set of software that track and manages changes happens in a files this git allow user to undo changes, commit changes, merge the changes and compare the files lot more. It is free and open source

What is git repository

* Git repo is a workspace which track and manages files in a folder. Each repo has different history they are not linked.

Along with GIT we have few more version control system

1 Subversion 2 CVS 3 Mercurial



Git helps us to

* Track files between multiple files.
* Compare the version of the projects.
* Time travel back to old versions
* Collaborate and share changes
* Combine changes

Difference between git and github

Git is version control system software run locally github is web service that host git reposito

On your machine. You don’t need sign up for an ries in the cloud. You need to sign up for ac

Account you can use git without touching github count. It is online place to share the work

Which is done in git

To interact with Git we use 2 ways 1 terminal and another GUI

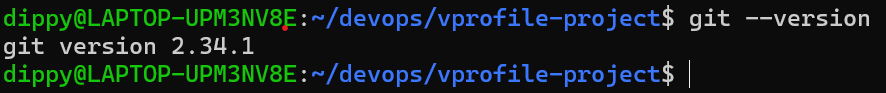
For terminal we use Git Bash

For GUI : Github Desktop Source tree Tower GitKraken

For text editor we use Visual Studio Code

Once Git Bash is installed check version by using below command

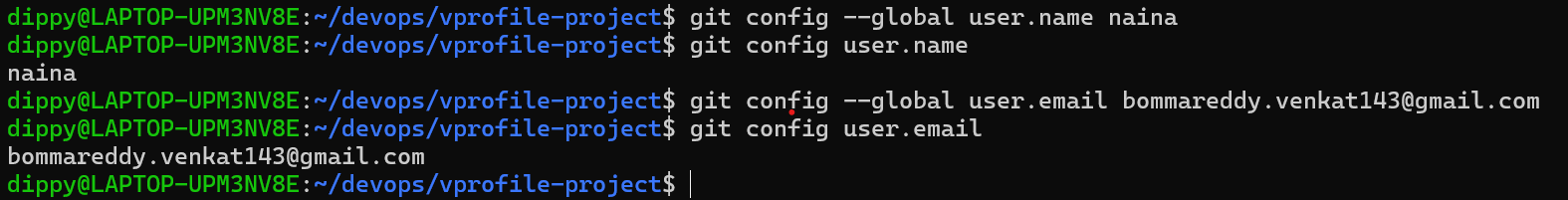
Git –version



Configuring your name and email id:

git config –global user.name naina

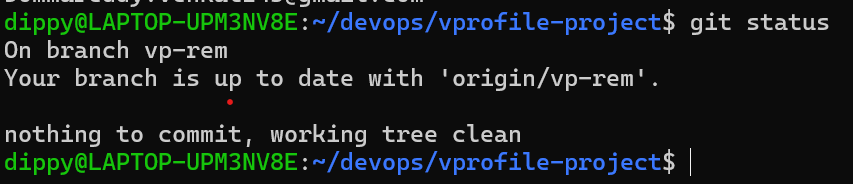
git config –global user.email bommareddy.venkat143@gmail.com



Git status

* it gives the information on the current status of a git repo and its content

git status

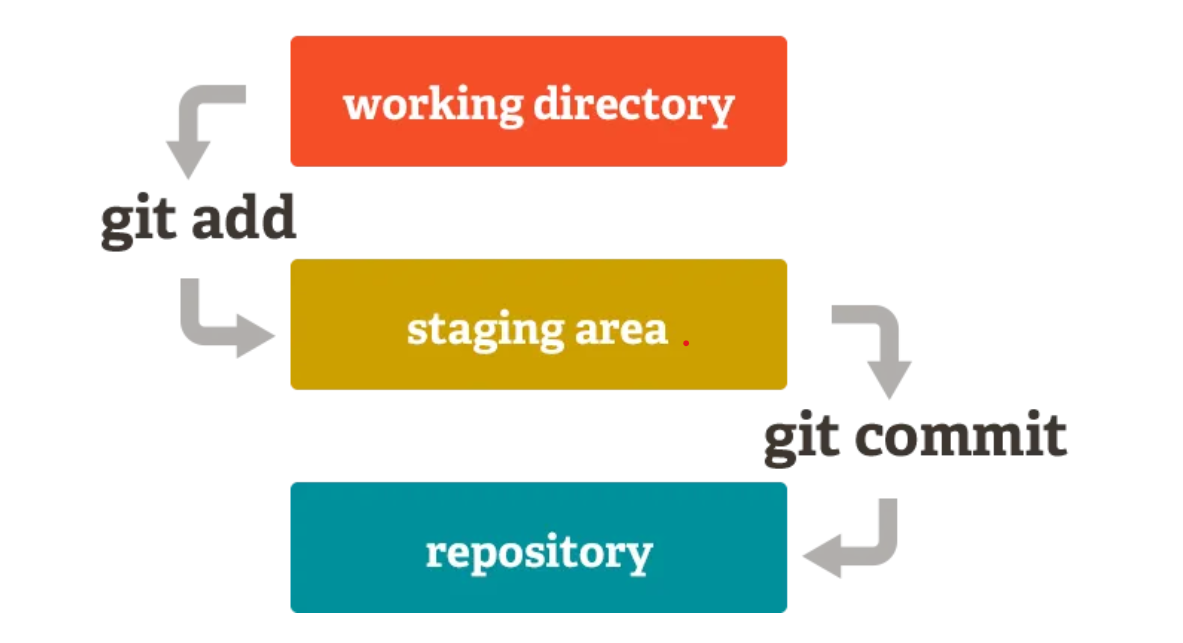


Git init

* used to create a new repository before we execute any git activity we need to initialize a repo first. If we execute same command twice it will re initialize git repo but some cases git will get confused . Before running a git init command you should be inside a directory . when you execute git init command it will generate hidden directory .git . If you delete that hidden directory current repo will be deleted. This hidden directory will be present in every repo. Git init will track subdirectory as well so no need to run git init command in subdirectory.



The workflow of git



Working directory: it is place where you currently working

Staging area: at this stage we are going to add the selected commits by using git add command and add commit will move to staging area.

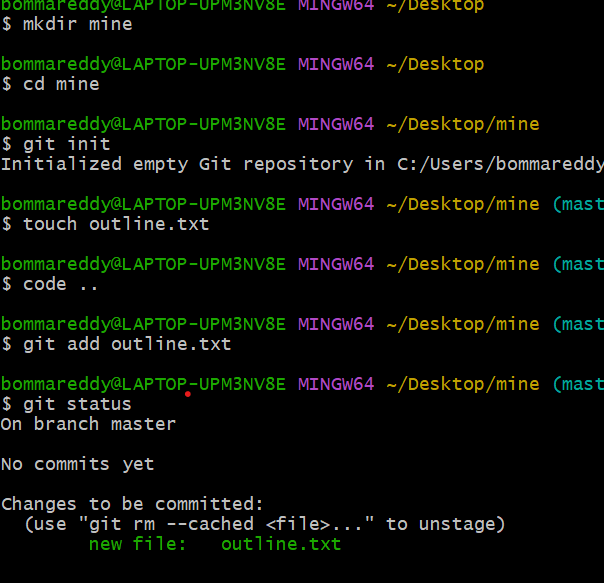
Repository: after doing the commit files will come to repository using git commit command git commit is nothing but snapshot of file after we do necessary changes at current time .

Git add command

* tracked files means git is monitoring the files. When we started adding file using

git add filename

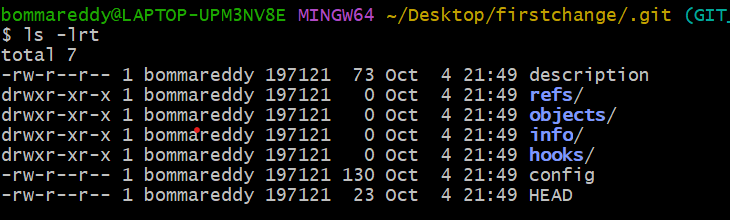
Then git start tracking the files.



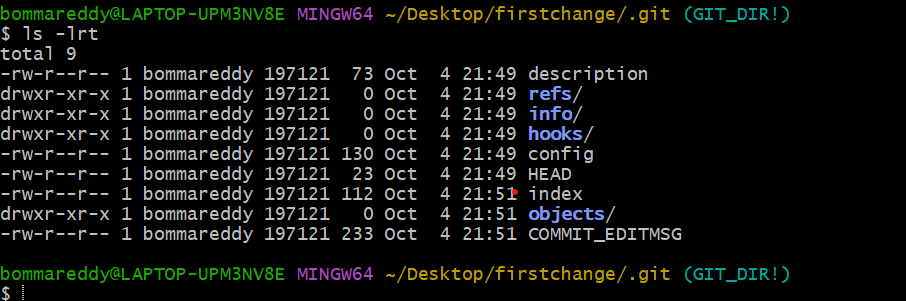
.git directory files

As we discussed earlier when we initiate git init .git directory will be created

Before executing git commit command files in .git directory



After executing git cmmit command files in .git directory.



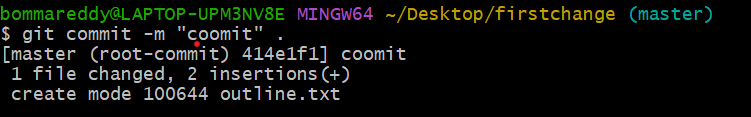
Git commit command

* Git commit -m “first commit” filename

m is noting but message always try to write a good commit message so that we can remember what we wrote in that file .We can also git commit filename but it will open your default text editor and until you close the editor commit cannot be saved.

Note : so it is good practice to use -m option will using commit command

Another way to add and commit at same time git commit -am “red” color.txt



Log command

* it is used show commits changes happed in repo

git log

commit a47fcb2538759b336498281d45b3ed5b14f94948 ----------- every commit had uniqe identity it is called as hash.

(**HEAD** -> **master**) by default we have branch called Master

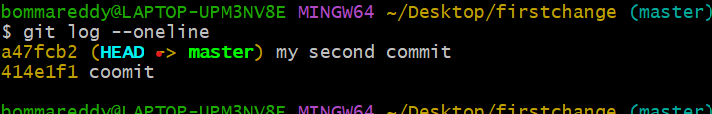
Author: bommareddy <bommareddy.venkat143@gmail.com>

Date: Fri Oct 4 22:16:25 2024 +0530 ----- the date when this commit is made

my second commit ------ the message which we gave for a commit

git log – oneline

this command will give shorter details about commits.



Amending commit

* Suppose you made a commit and forgot to add a file or you made a typo mistake for committing a message that you want to correct it. As I mention message will play a critical role to identify our commit this amending commit is so useful and you can redo you changes.

Note : this command work for 1 commit before it

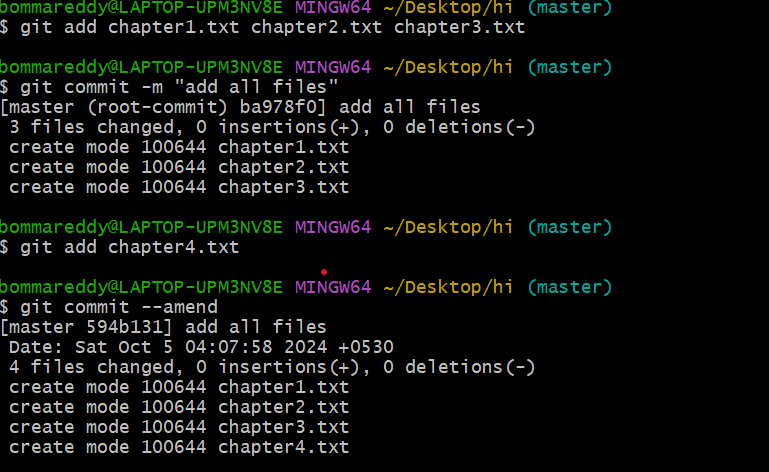
Git commit --amend

This is for file which we forgot to add with remaining files

You have add that missing file and use commit amend command

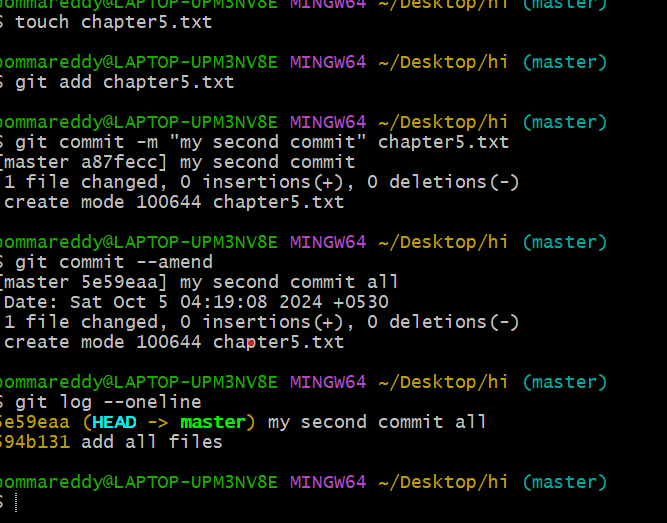
Git add filename(missed filename)

Git commit –amend then a text editor will open just save and close the editor



If you want to change the message after commit just run amend command

Git commit –amend



Ignoring files

* Sometimes we want files or directory to be ignored or not tracked by using .gitignore files.

When we are working on application code we don’t want others to track that files like API keys credentials packages etc. create a file called .gitignore in the root of repository or anywhere after creating .gitignore file open and keep the files and directory which you want to ignore it. For directory add / at the end of the dir name – foldername/ and \*.log all files with .log will be ignored

Vi .gitignore

Keep files and save it

Branches

* Branches are essential part of git. They allow us to work on separate ideas parallel. If made a change in one branch they do not impact on another branch (unless you merge then)

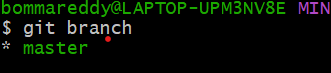
By default you are always on branch called as master branch.

Head

* Head always refer to our current location. Head always point to our last commit we made (recent one)
* commit 3c7e98ba2b769de466d496084dbbbefefddf3698 (**HEAD** -> **master**)
* Author: bommareddy <bommareddy.venkat143@gmail.com>
* Date: Sat Oct 5 05:44:20 2024 +0530

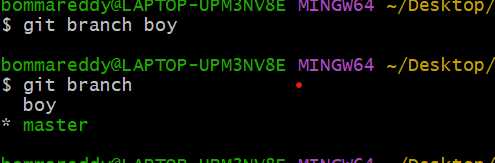
Viewing Branches

* Use git branch command to view you current branch. The default branch in git repo is master look at \* it represent the current branch



Creating branches

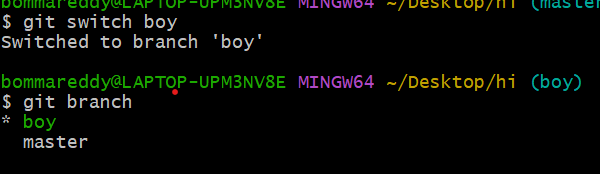
* Use git branch brranch name to make a new branch it is just creating a branch not switching to another branch.



The above screenshot \* is towards master means we are in master branch we just created a branch but we didn’t switch to that “boy” branch.

Switch Branches

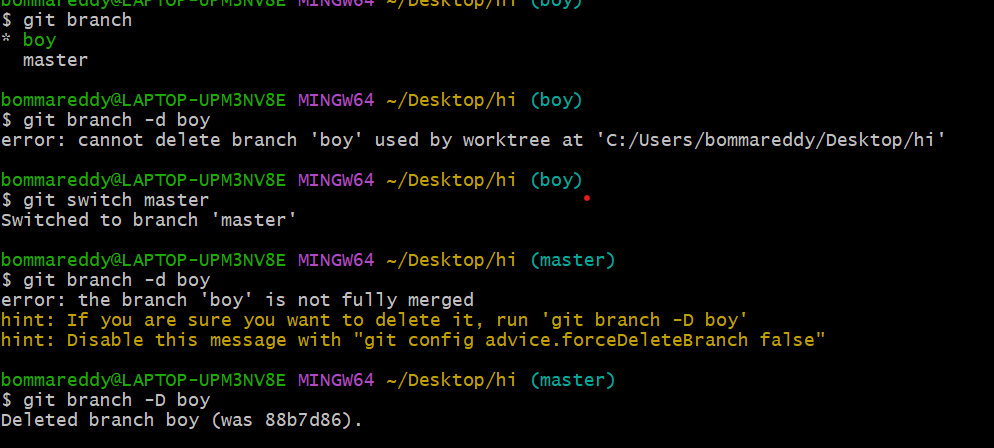
* To switch branches we use git switch branch name



Delete a Branch

* To delete a branch we use option -d or -D git branch -d branch name
* Git branch -D branch name

Note : for deleting a branch you should not be inside of it you should move away from that branch



Renaming a Branch

* To rename a branch we use -m option git branch -m newname

Note: to rename a branch you should be in the branch not somewhere else



Merging Branches

* We merge branches not commits. You have to move to branch where you want to merge

Example you have 2 branches 1 is master and another boy if you want to move boys data into master then you have to switch to master and run the merge command.

Git switch master ------------------------ receiving branch

Git merge boy

After merge is complete

Git add .

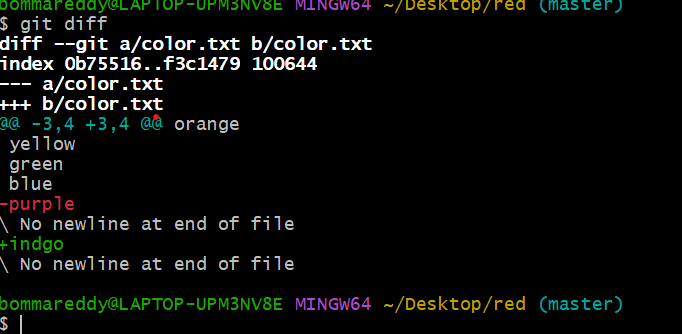
Git commit

Diff command

* Diff is all about showing changes made in git changed between branches files etc.

This command don’t impact git repo.

Git diff this command will show only unstaged changes. Below example will show you difference between last commit and change we made. If the changes are saved or staged it wont show anything.



After adding to stage if you want to view the difference

Git diff HEAD

If you want to see the staged changes

Git diff –staged

Git diff –cached

If you want to see difference between branches

Git diff branch1..branch2

Git diff branch1 branch2

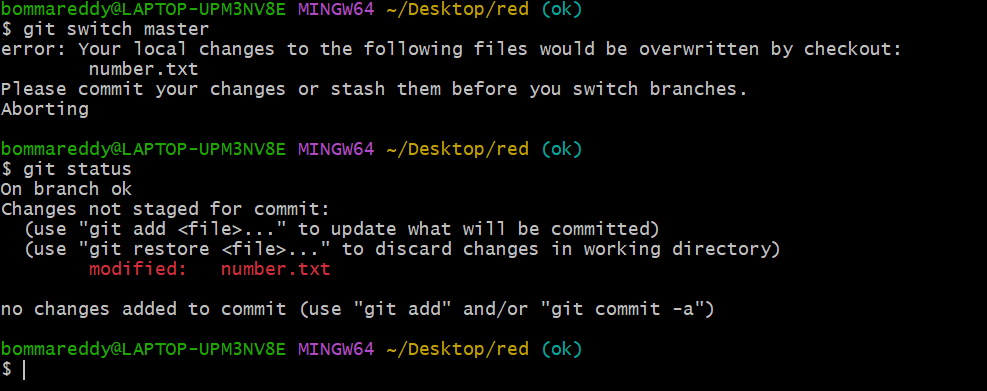
Similar way we can do it for commit as well by giving commit hash

Git log –oneline

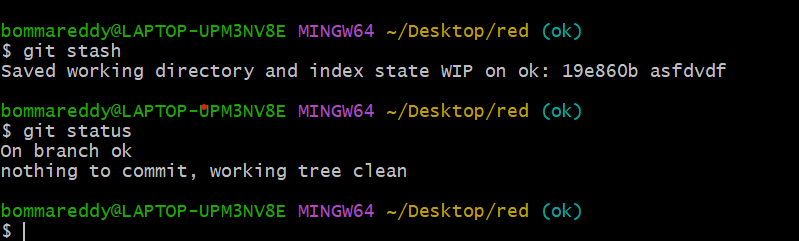
Git diff commit1 commit2

Stashing

Suppose you are working on master repo you made some changes and forgot to save them and to switched to new repo and the changes you made in master branch will come along with you or you have some other important work to do in another branch and what want to do this work later without saving. While switching branches it wont allow you switch or changes also come with you.



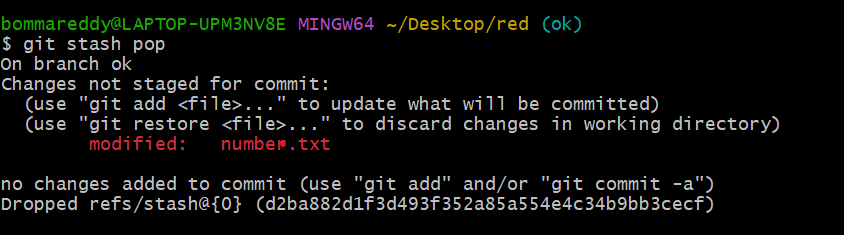
Git stash this command will hide this file but it will remember them later we can unstash them.we wont be able to see those this change anymore.



Removing stash

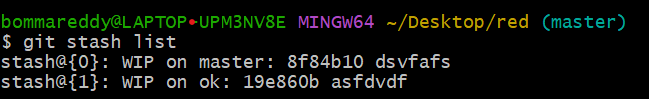
To remove the stash changes we use the command

Git stash pop



We can view the number of stash by using

Git stash list



Working on multiple stash

Every stash has a stash id like above screenshot

stash@{0} is the stash id

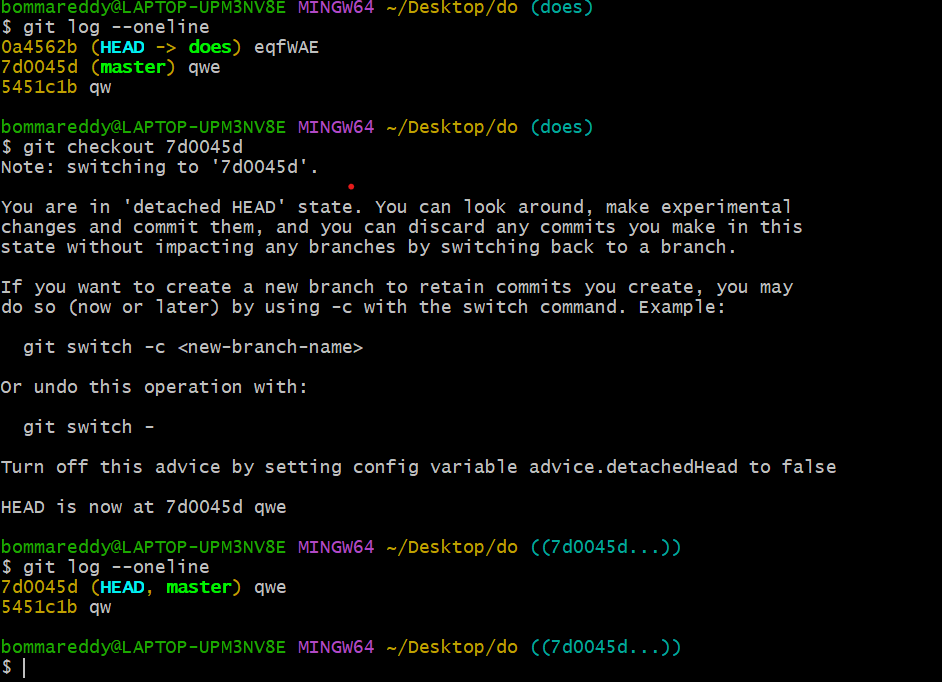
to apply 1 stash id

git stash apply stashid

checkout

* If you want to view you previous commits we use this checkout command.

Git checkout hashnumber



If you see the above screenshot you can note that 3 commit before you run checkout command.

After running you can see only 2 commit because you move back in time.

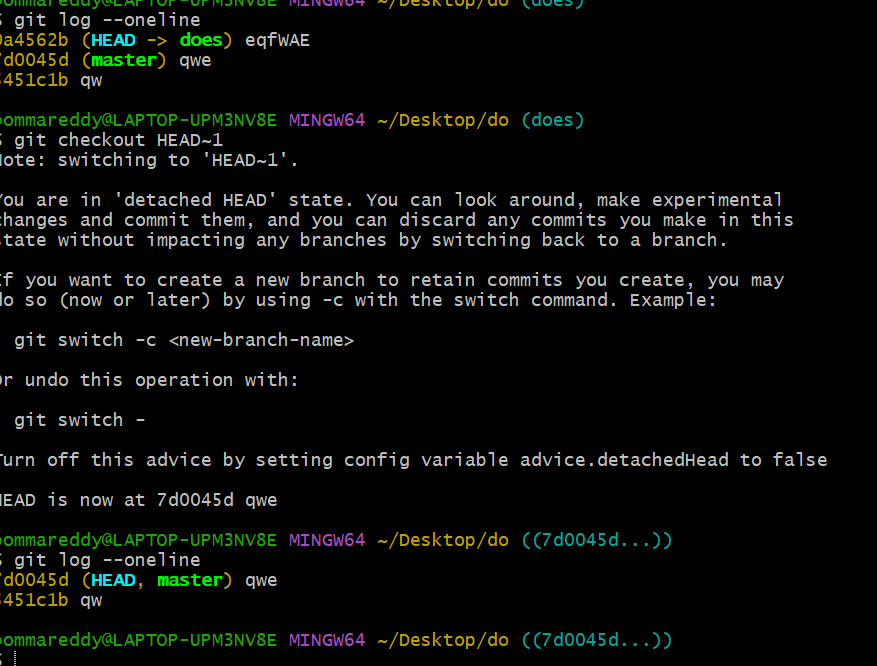
If you want to come back from that commit

Git switch branch

There is another option instead of using this hash

HEAD~1 refer to the commit before head

HEAD~2 refer to 2 commit before head



Discard changes

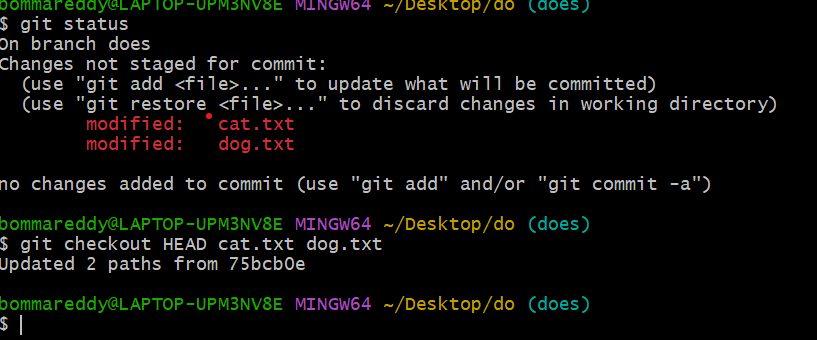
* Suppose you made some changes in the files but you don’t want to keep them this apply for uncommited changes.

Note command will apply before you add and commit the changes we can manually remove the changes in the text but what if you made changes in several place

Git checkout HEAD filename or

Git checkout –filename

Git restore filename

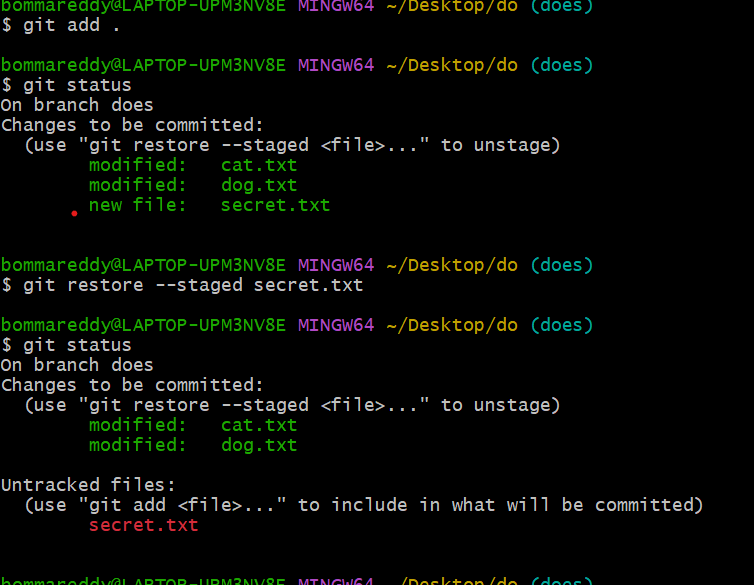


Unstaging a file after adding it using git add

* You added a files using git add but you didn’t commit and you want unstage the file

Git restore –staged filename

The apply git restore filename



Reset

Suppose you made couple of commit but that commit has to be in separate branch or you want to undo the commit we use command git reset you want to reset the repo back to a specific commit

We using command git reset commit-hash and commits are gone.

Git reset –hard commit hast this will delete the commit and it working directory also

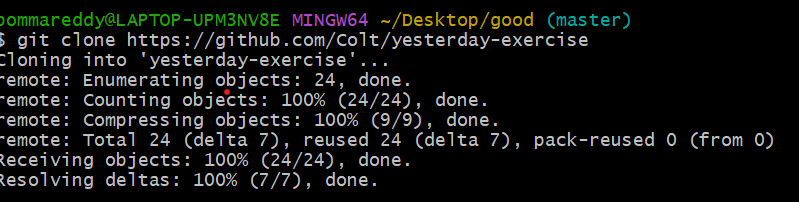
Github

Githib help us to keep our git repo in cloud so that we can share with anyone and collaborate with people.

Cloning

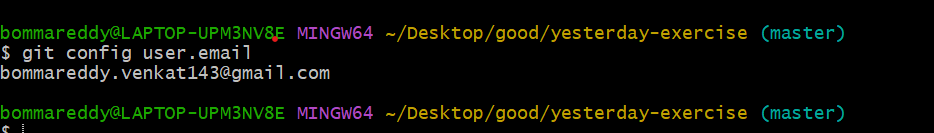
We don’t have a repo in our git and you want to get a repo from another source like github for that we use a url. Permission for clone if you are able to see then you have the access to download.you no need to be a owner and contributor

Git clone url



Register a account in github

Go to github and create a new account while creating a account please try to give a email which you config In git



Ssh key

Everytime when you push or pull something from your command line to github you have to give your email and password if you don’t want to do this generate a ssh key so everything will be password less

<https://docs.github.com/en/authentication/connecting-to-github-with-ssh>

first you have to check if you have any ssh key

1. Enter ls -al ~/.ssh to see if existing SSH keys are present.
2. $ ls -al ~/.ssh

# Lists the files in your .ssh directory, if they exist

Check the directory listing to see if you already have a public SSH key. By default, the filenames of supported public keys for GitHub are one of the following.

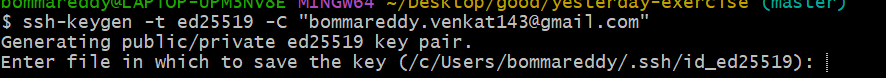
* *id\_rsa.pub*
* *id\_ecdsa.pub*
* *id\_ed25519.pub*

if any key is not present generate a ssh key by using below command

ssh-keygen -t ed25519 -C "your\_email@example.com"

**Note:** If you are using a legacy system that doesn't support the Ed25519 algorithm, use:

ssh-keygen -t rsa -b 4096 -C [your\_email@example.com](mailto:your_email@example.com)



It is asking for file where you want to save this key but I am giving the default file name

Then key will be generated .

Next step is add your ssh key to your ssh agent

Copy the ssh key by using the below command

clip < ~/.ssh/id\_ed25519.pub

go to github --- go to setting --- ssh and gpg key option------ and go to add shh key------ paste the key and save it

how do I get my code to Github

there are 2 option

1 option -- if you have a existing repo in your local and you want to push it to github

Create a repo new in github

Connect your local repo (by adding remote)

Push you code to github

Option 2 -- if you didn’t start any work in local then create a repo in github

And make same changes and push then to your local by using clone.

Then do changes in your local and push them back to github

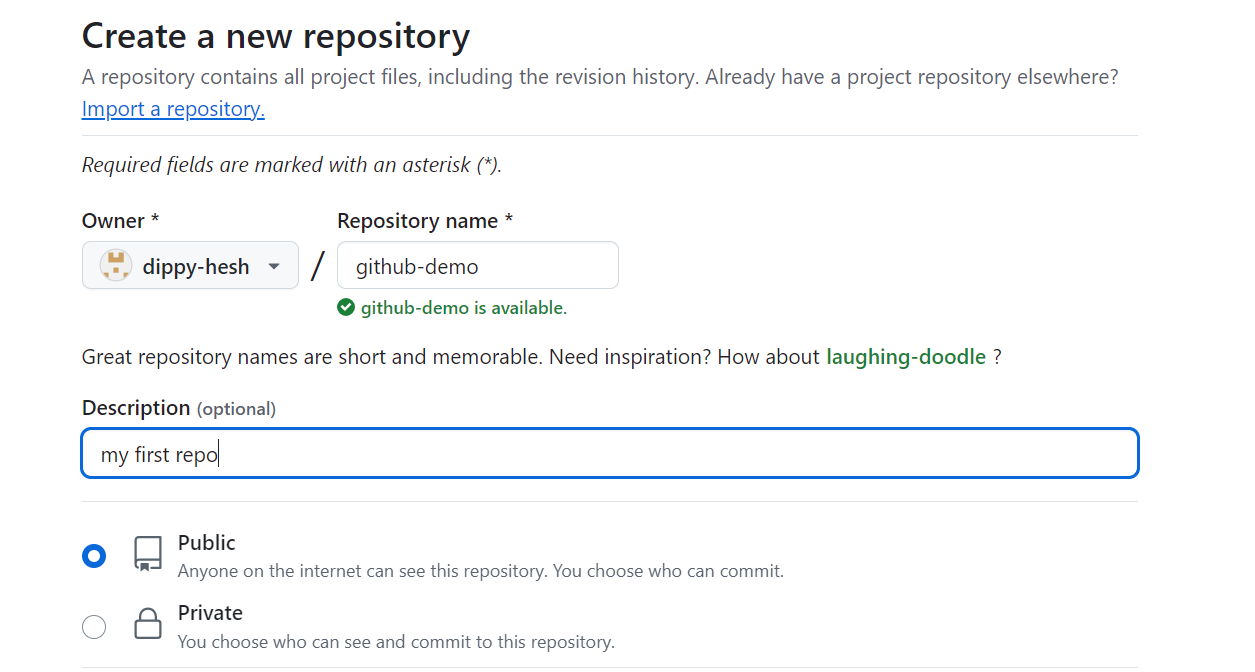
Creating your first github repo

Go to github click on + symbol on the top click on new repository

Then you will see a new page like below in repository name give a repo name as you like.

Description it is upto you if you want people identify or search to repo you can given a name

If you want people to see your repo keep it as public

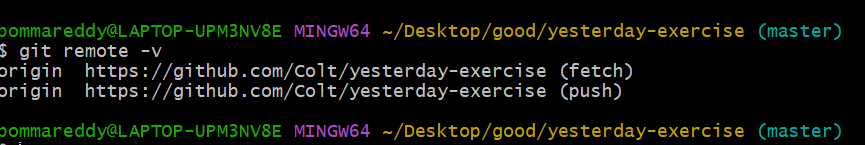


Then click on generate a repo

Remote

Remote is a fancy name destination url it helps git to push or pull the repo

Git remote -v will list the current remote url we have



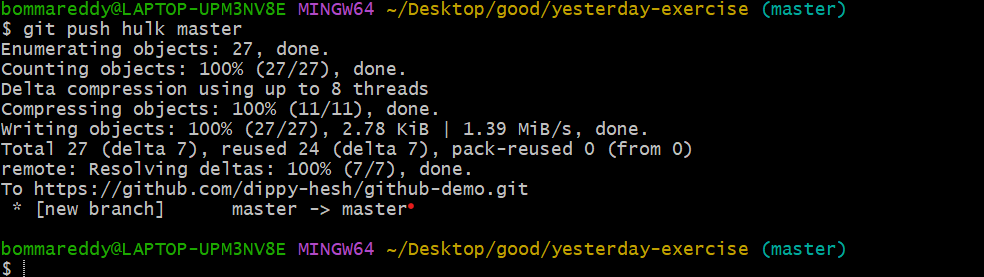
Adding new remote

Git remote add origin url we are git to remember the url with name or label as origin . the name origin can be anything else we are asking git to remember url with that name

Introducing git push

It better to check git status before pushing because it is better to check if there is any uncommitted changes

Git push hulk master --- here we are pushing master branches changes. You no need to be have to in that branch every time when you push



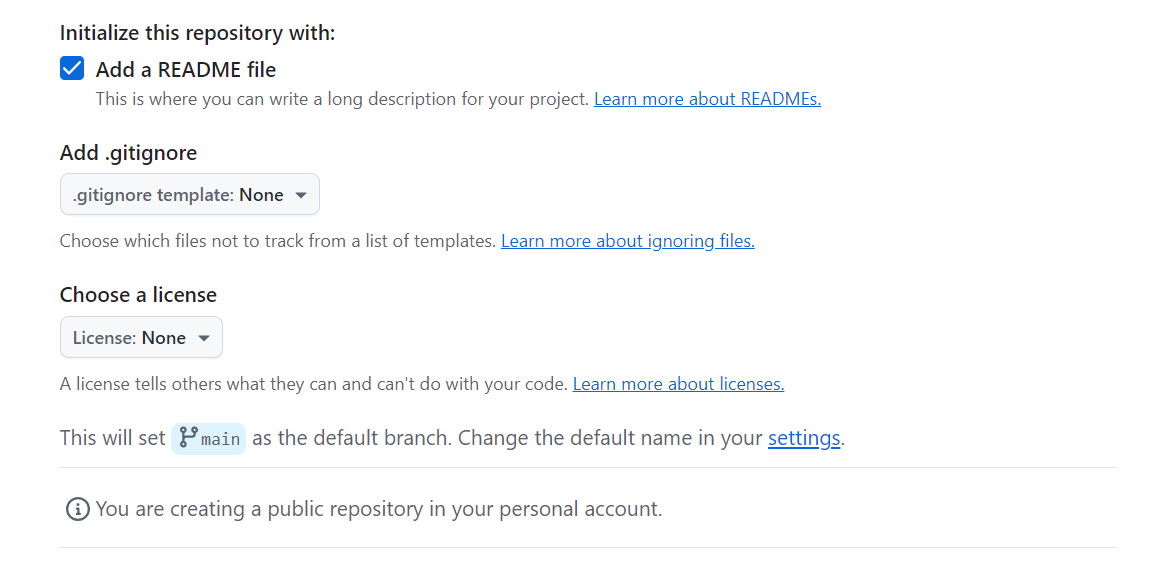
How to push to another branch which is not created

Git push hulk master:doit --- this will push changes from master branch to doit branch if the doit branch doesn’t exit it will create a new branch called doit.

How to push from github to git

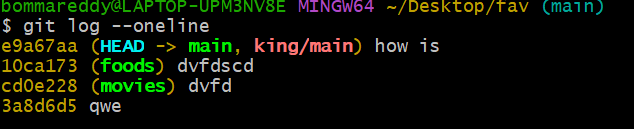
Git clone url it will push it will create a remote automatically we no need to add it

By default master branch in git but by default main is the branch name in github



Remote Tracking Branch

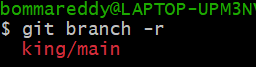
Similar to git github also track there branches



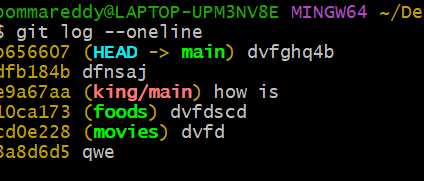
Main is our git branch name and king/main is our remote tracking

To checking how many remote tracking branches we have

Git branch -r



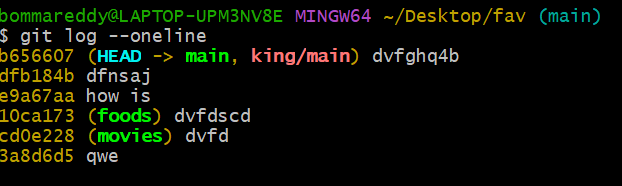
If we made a push and after we did any 2 commit github cant track it until you push the changes again



If you want to see how my project look like when we first clone it

Git branch king/main

After pushing it again you will see remote tracking is upto date



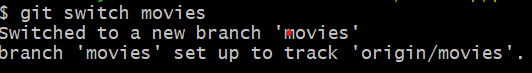
What if you are working on a repo which have multiple branch

By default when you clone it will copy the branch which is selected if you want to copy remaining branches first we have to know how many branches we have using below command

Git branch -r

Later use the switch command it will copy the content of github to ur local git

Git switch branchname



Git fetch basics

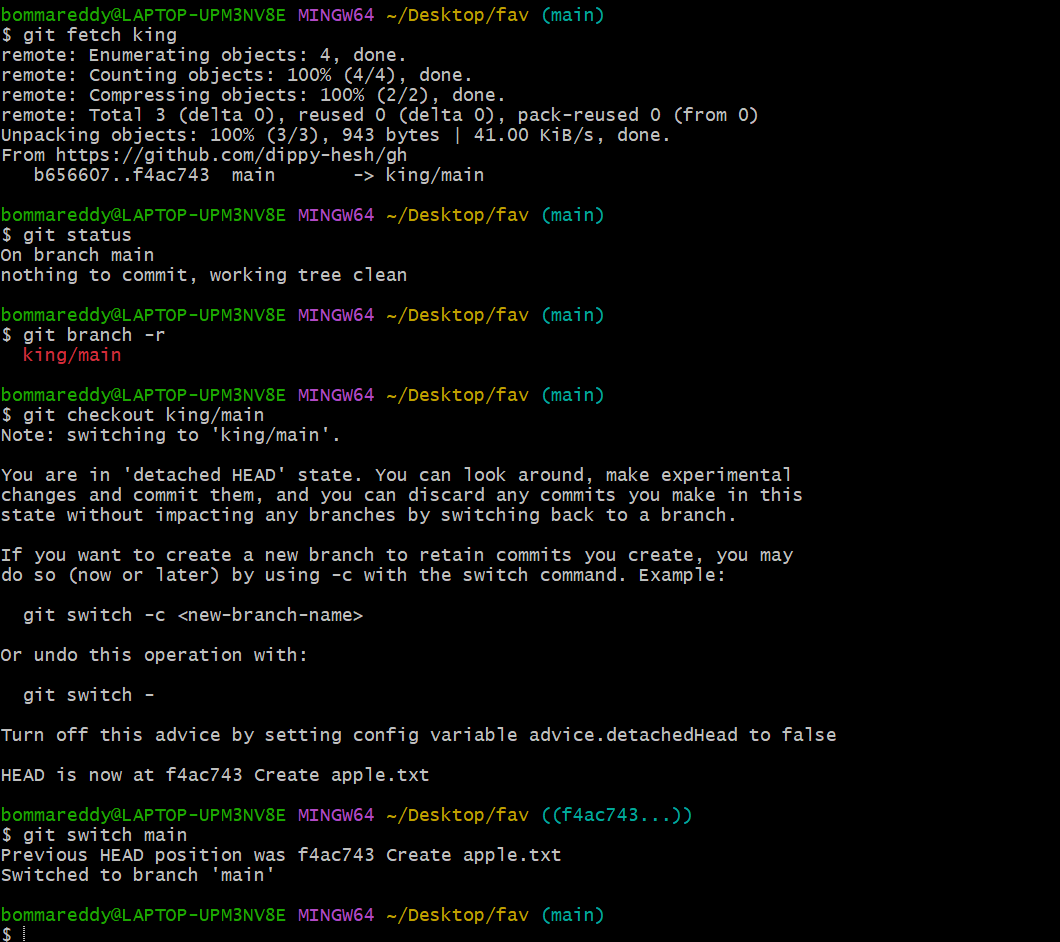
Fetching allow you download the changes from a remote repo but those changes wont automaticall integrate with our working directory other collaborates will make changes and I just want to see them

Git fetch remote for all

Git fetch remote branch to view one branch

Git fetch remote

After that git checkout remote branch name then u will be able to see the changes then git switch



Git pull

This pull is another command to retrieve the data but it will update the current working directory

Git pull = git fetch+ git merge where we execute the command also matter we have to be inside the repo where we want to place the data sometimes it will conflict the data because this will merge the data.

Git pull remote main

Conflicts suppose you have file in github with a name man.txt and you have a added same name file when you pull the data from github there will conflict then manually resolve them and save the file.

And push them back to github again

Public and private Repos

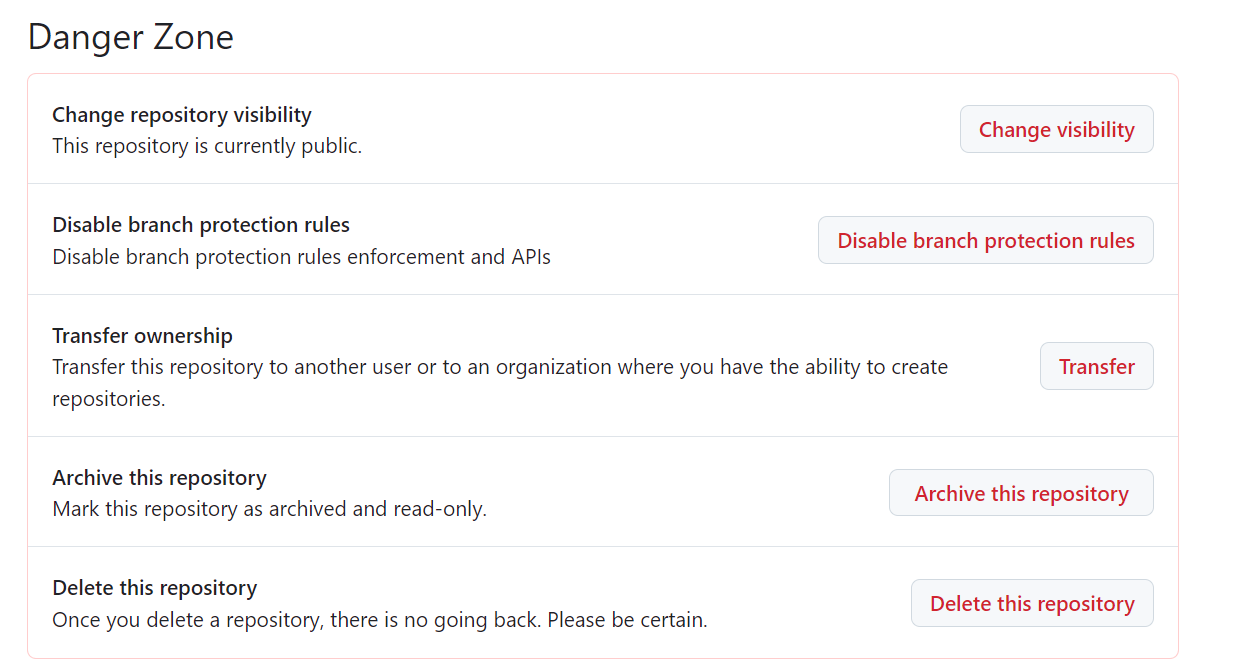
Public anyone can view or browse the repo and they clone this repo and access the repo in internet

Private only the owner has the access and the people who are granted will be having the access

If you want to make a public repo to private repo

Go to settings at the end

Danger zone select change repo visibility



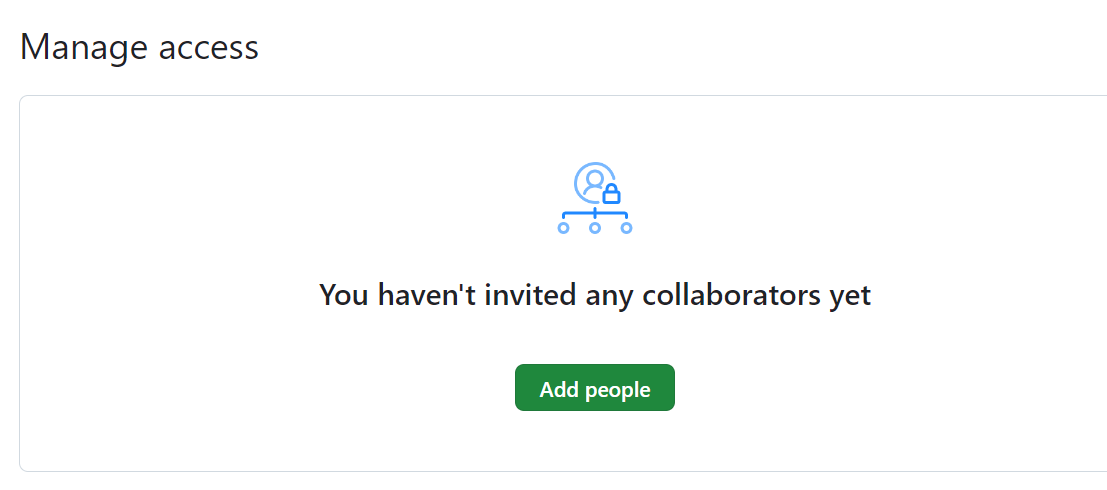
And make it as private

Adding collaborates

Adding other collaborates to you repo means the changes what they made will added to you repo

Go to settings

Click on manage access or collaborators



Click on add people give there github user name or github email id this is just sending invite to him he has to open his email and accept it.

README file

This is one of the important file which explain about project to convey the importance about the project to new people we put in this file. Usually people keep this file in the root (main) but it is upto you.

Readme.md are makedown files is text to html for web writers it is plan text formate which convert to html or xhtml

Git gists

Gists is place where people share there code with others and post your questions

Github webpages

Using github repo we can host a webpages but there is some restrictions we can only do for statics web pages just like html css and js but we cant host high capacity webpages.

There are 2 types of github pages

User site and project site

Project site you get unlimited websites. each git repo can have corresponding website the default urls follow this rule username.github.io/repo-name

User site

You git 1 user site for a account this is where you can build your portfolio website. url follow this rule

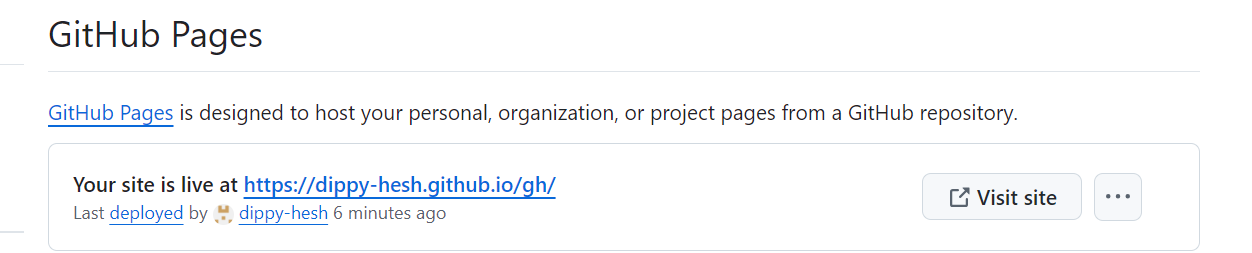
Username.github.io

Creating your first webpage

* Create a index.html file in your repo go to settings and **Code and automation**

And go to pages in build and deployment section

In branch select where you index.html is present and in folder select root or doc as u like and save it



Then you can see url where your webpage url

Centralized workflow

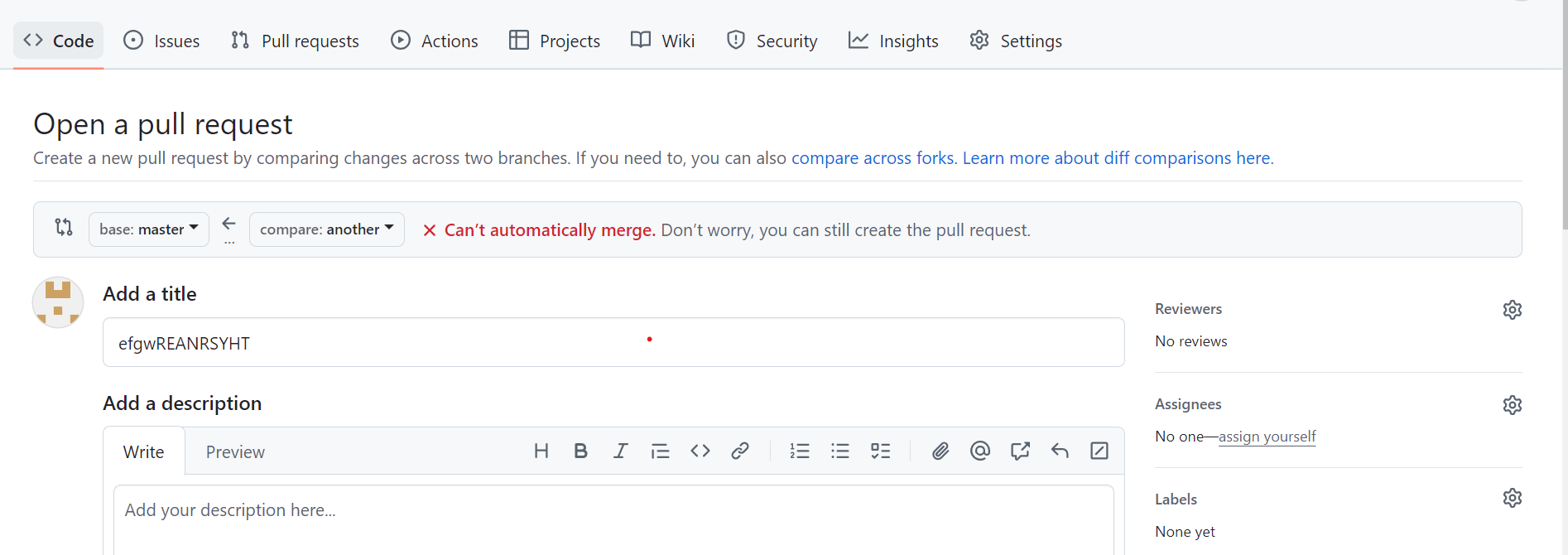
Everyone works on master or main branch or a single repo. If there is any broken code we don’t push them master to check with others because pushing will merge into out present files. So we create a another branch and collaborate with others and get help from them once we are good we will do code review with the team and merge them to the master or main branch usually in company th keep there main data about the project in the master or man branch.

Pull request

Before merging your data in to master or another branch we send a pull request nothing but taking approval from you manager or asking for any review so that they review leave a comment on it based on it we merge them to another branches

On github go to you branch click on pull request base is nothing but where we want to merge

Compare is nothing but our branch. A lot pattern should be followed like naming convention



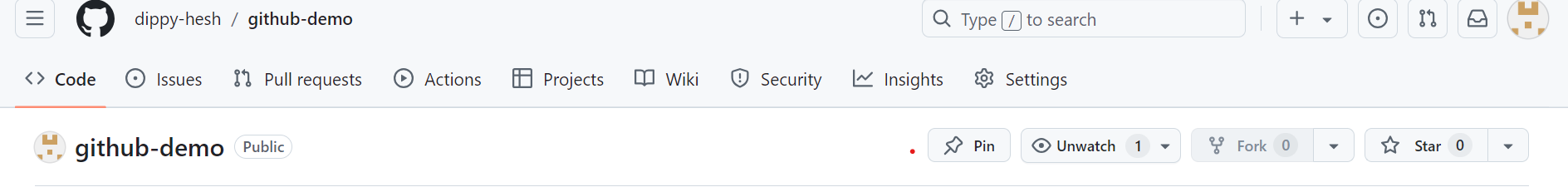
Configuration of branch rules

When we have a lot of collaborators we can set rules for our branches

setting and select branches

Forking

This allow to make a copy of other people repo. It is a gihub it is not a git feature



If you clone someone repo you can make changes in our git but you cant push that up because we are not a collaborator but fork will make a copy we can change as we like and push it back and you can add collaborator . you can make a pull request to the original copy of the file . you have the access to pull changes down to your repo if there is any changes made in original repo.

Rebasing

Rebasing is used into ways

1 as a alternative to merging

2 cleanup tool