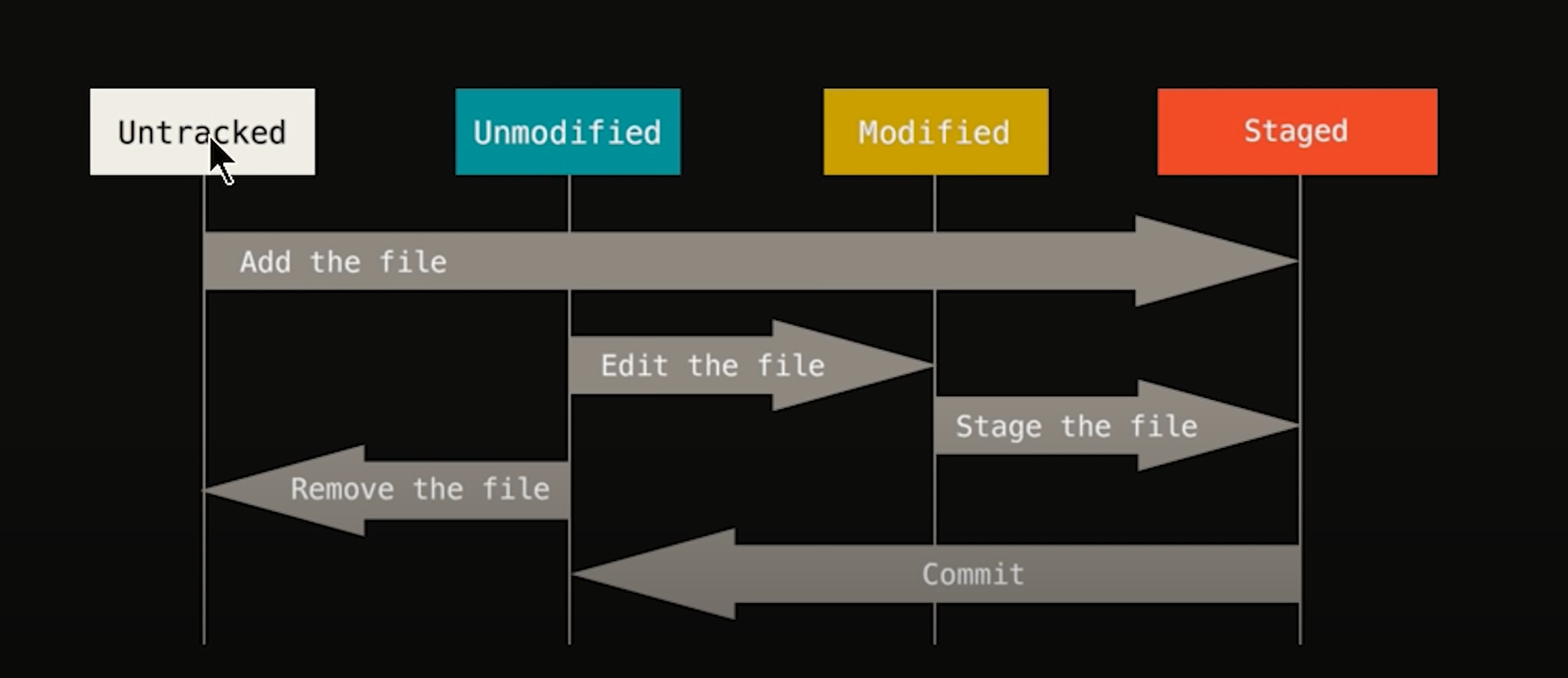
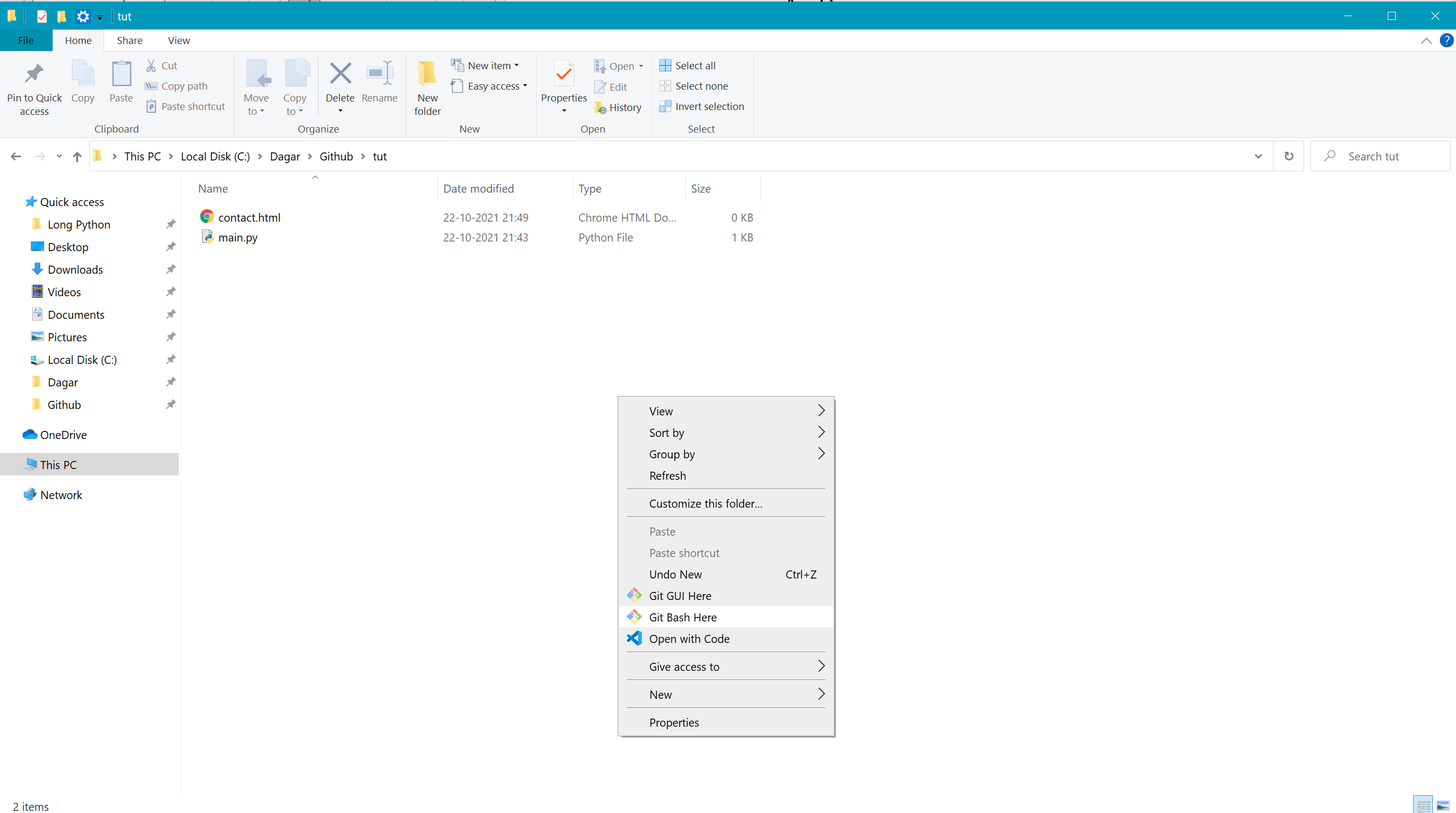
Github notes made by Dagar Singh

First, the file is always untracked means when we open the git bash in any folder at the first time and we type git init then we type git status the files always are untracked, then we add them then they are staged files then we commit them, after then they become unmodified, then we edit them then they become modified, after modifying them, then we stage them(add them), then we commit it, And some time we remove them(by rm command), then this cycle going on and going on…(As you can see in the below diagram)

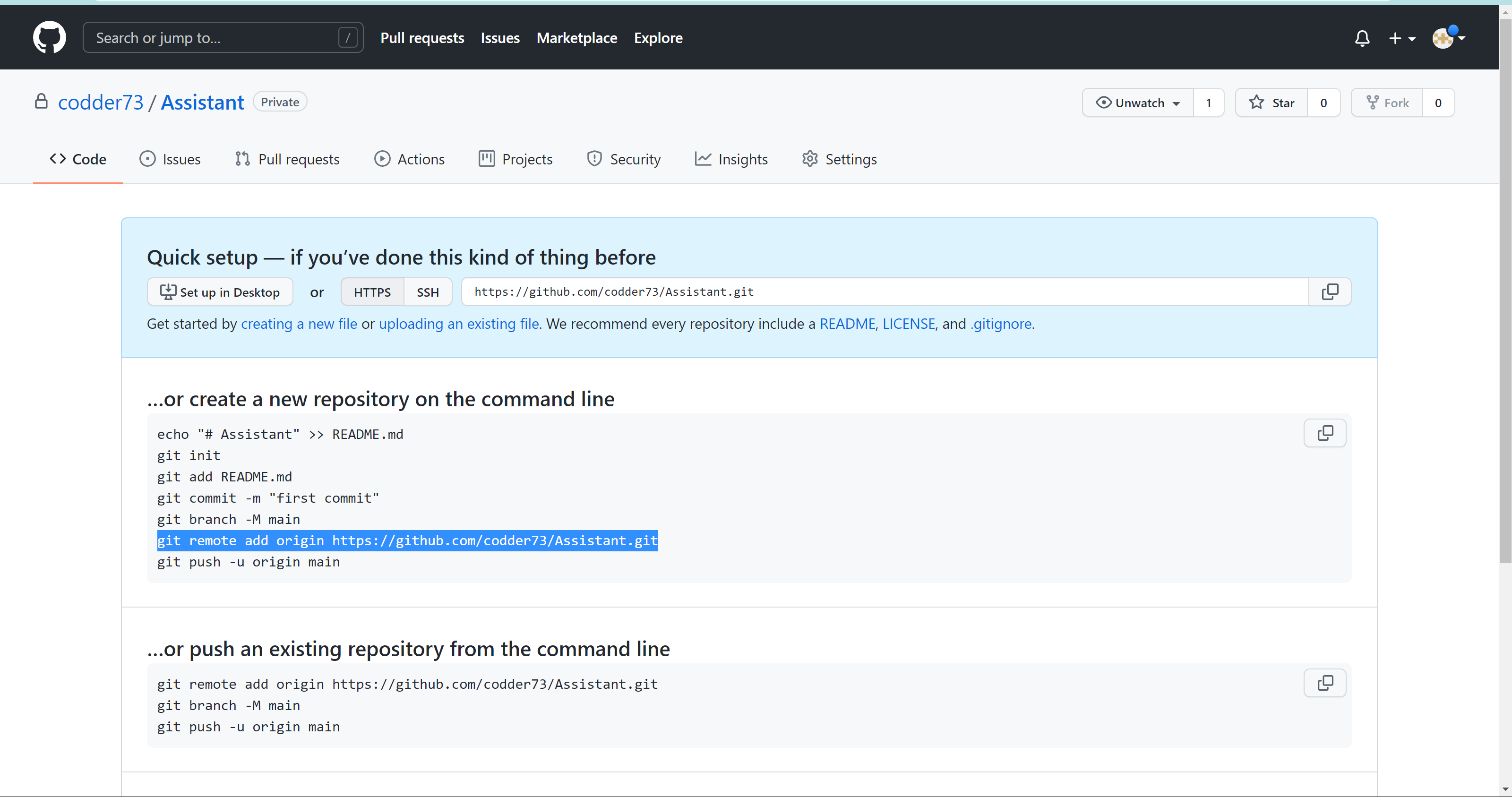


Starting 

* First go in your folder in which files you want to commit
* Then right click on mouse
* Then click on - open git bash here 
* The terminal will open
* Then – type git init in the terminal
* Then type = git status(all the files will be untracked)
* git add \_name of the file\_ **or** -A (we stage the files)
* **Git commit 1st method **
* git commit (the vim editor will open) -- then press ‘i’ Then type what you want to type then press ‘insert’ then type ‘:wq’ and then hit enter key
* **We have a shortcut of git commit that is **
* We can type - git commit -m (And type those things what you want to type in the Quotation mark that is-**“”** )
* your changes have been comitted(these are be unmodified files)

Sending files to github

First make a repository in the git hub

Then type this in the terminal 

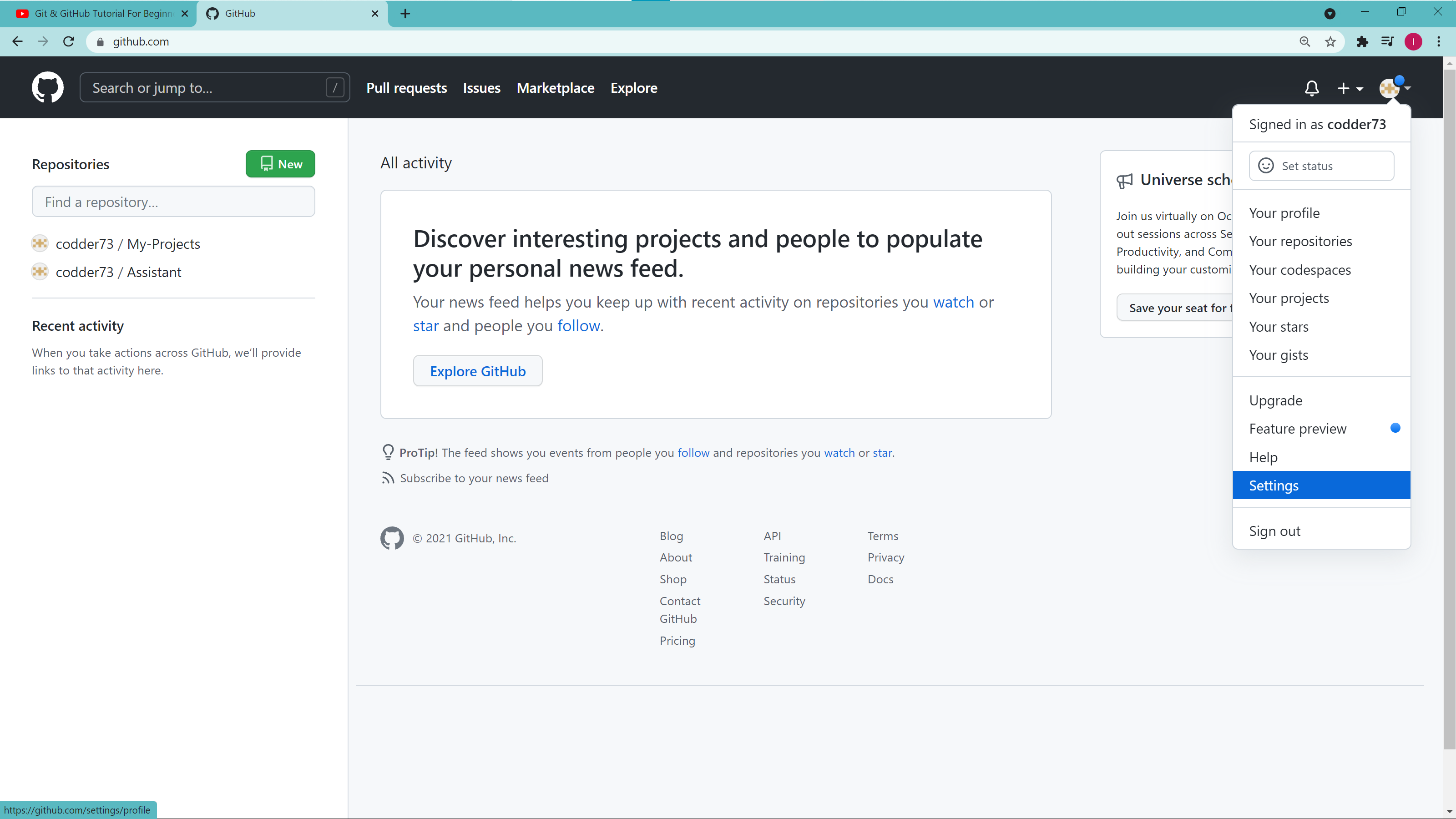
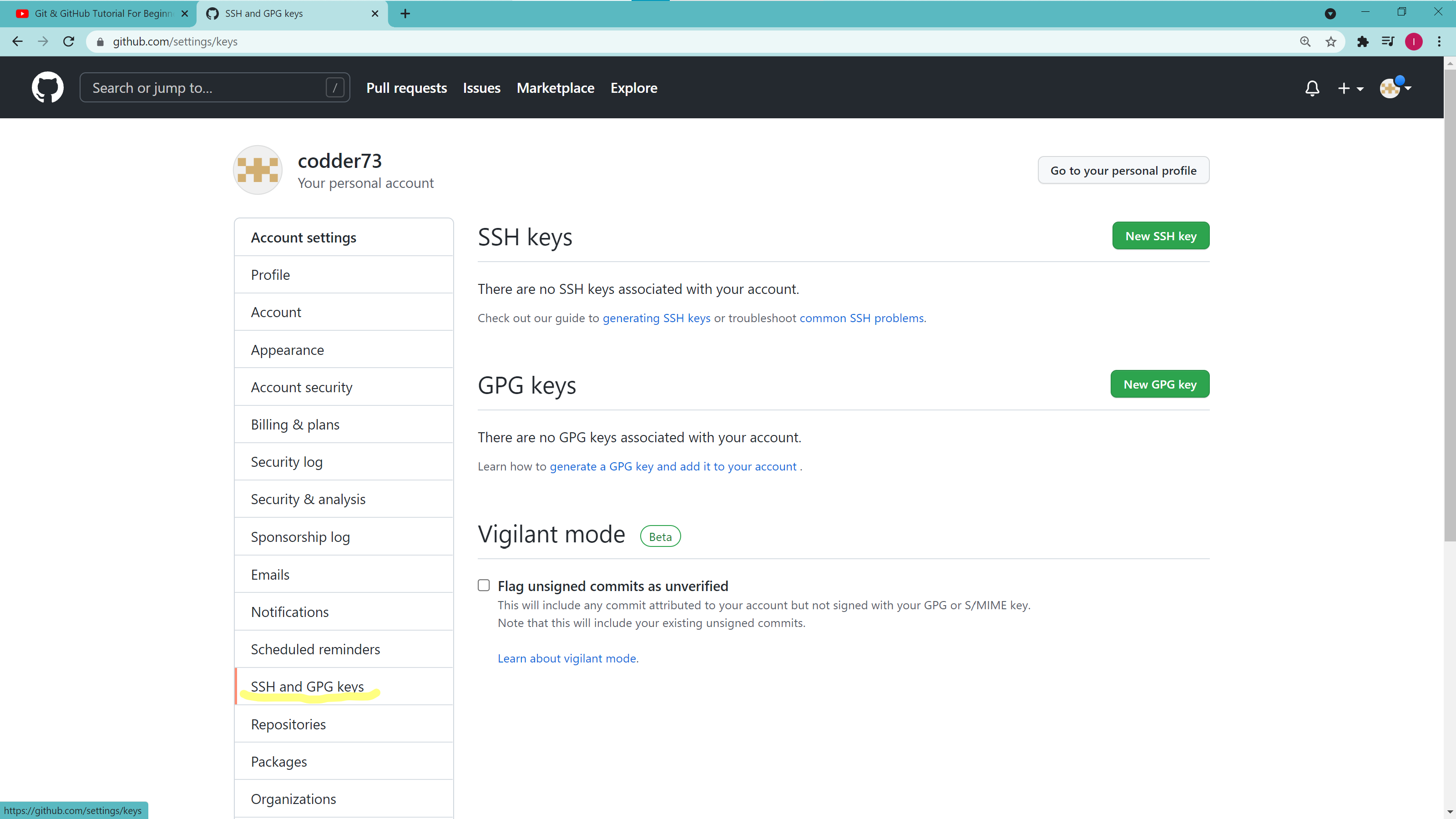
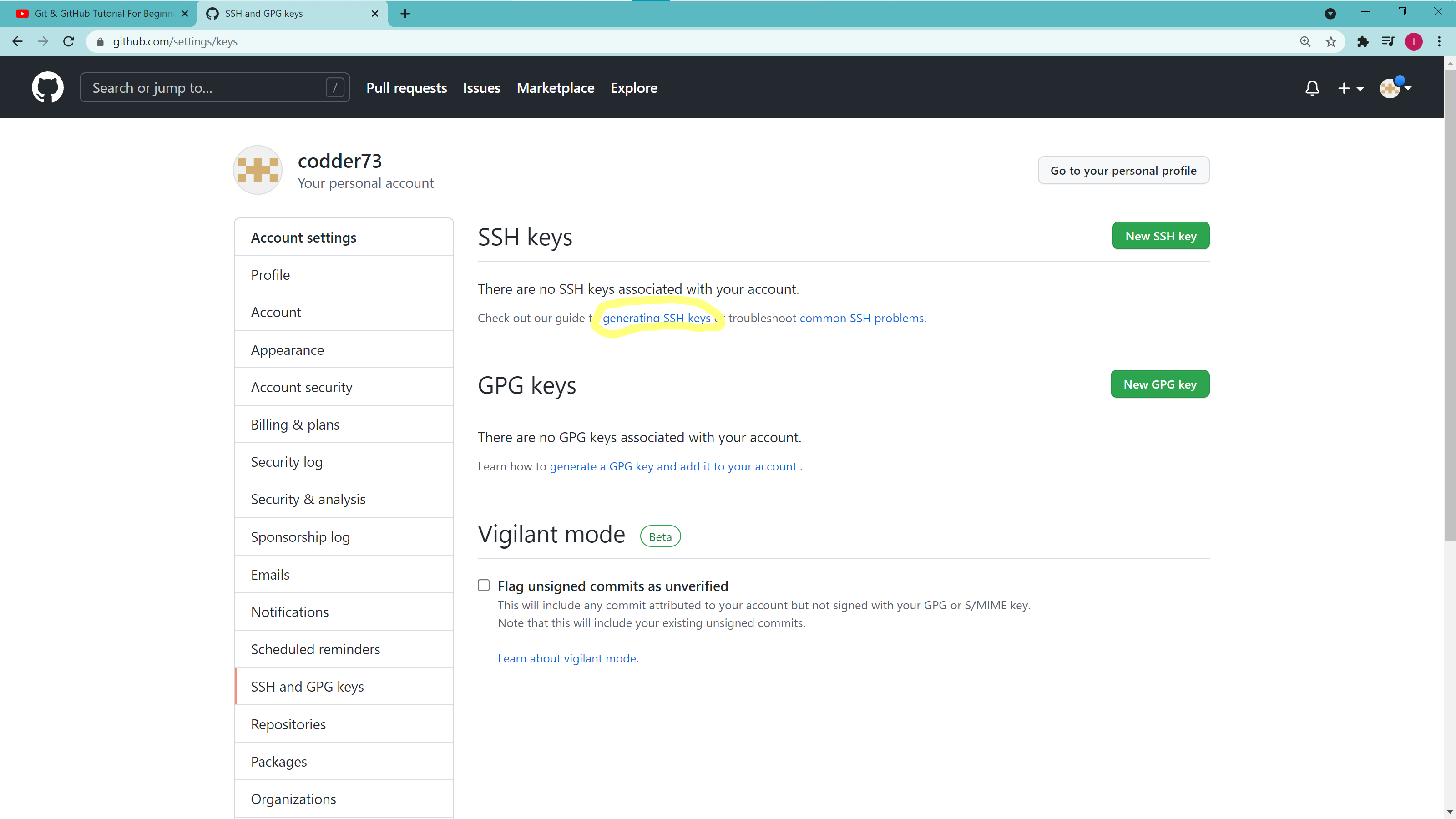
* Copy that selected line and then paste it on the terminal (Sorry its image is not available),It will be proportional to the user as you will have your repo name it will also changed, and then Hit enter –

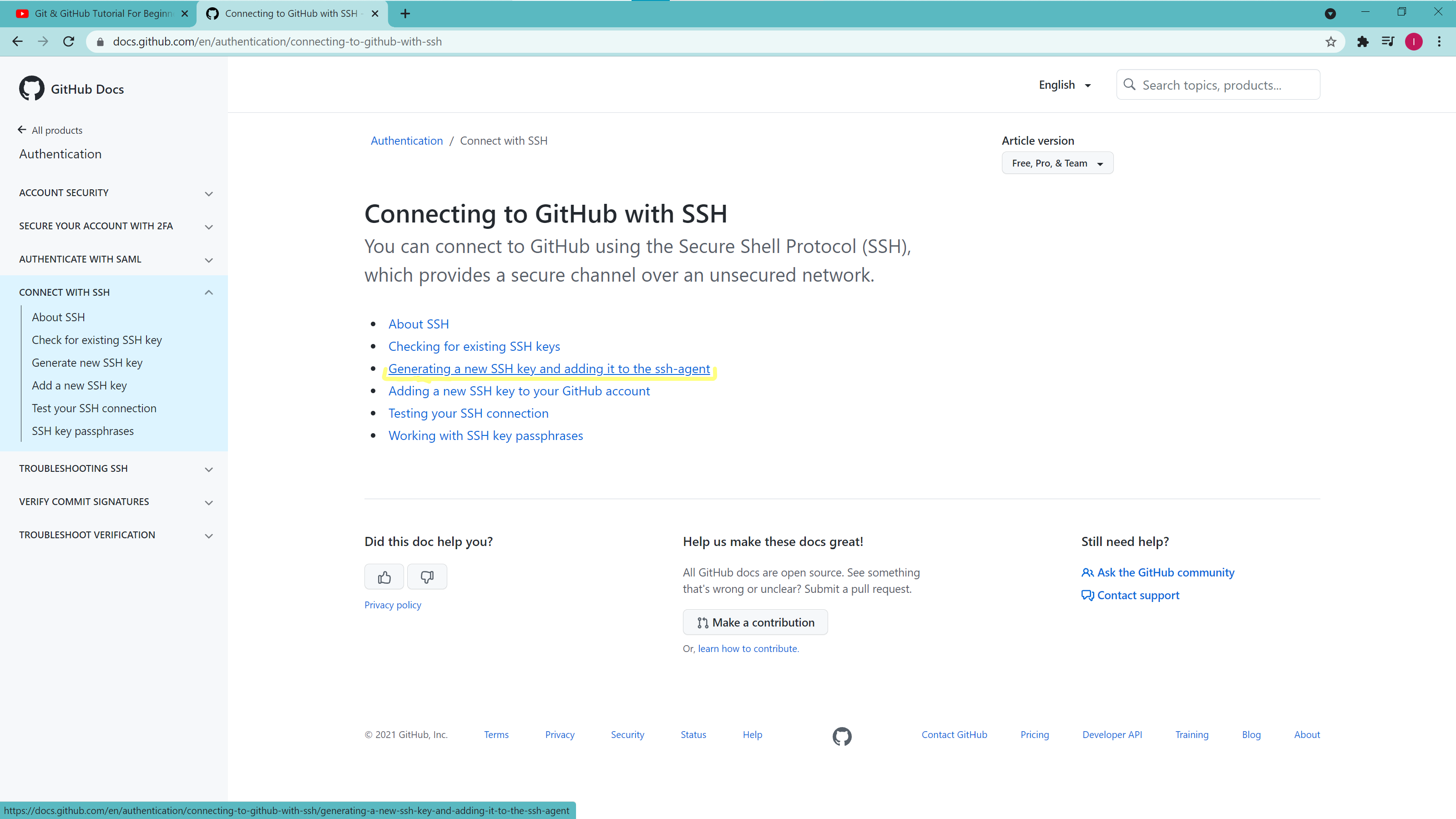
Then type git remote == your output should be origin

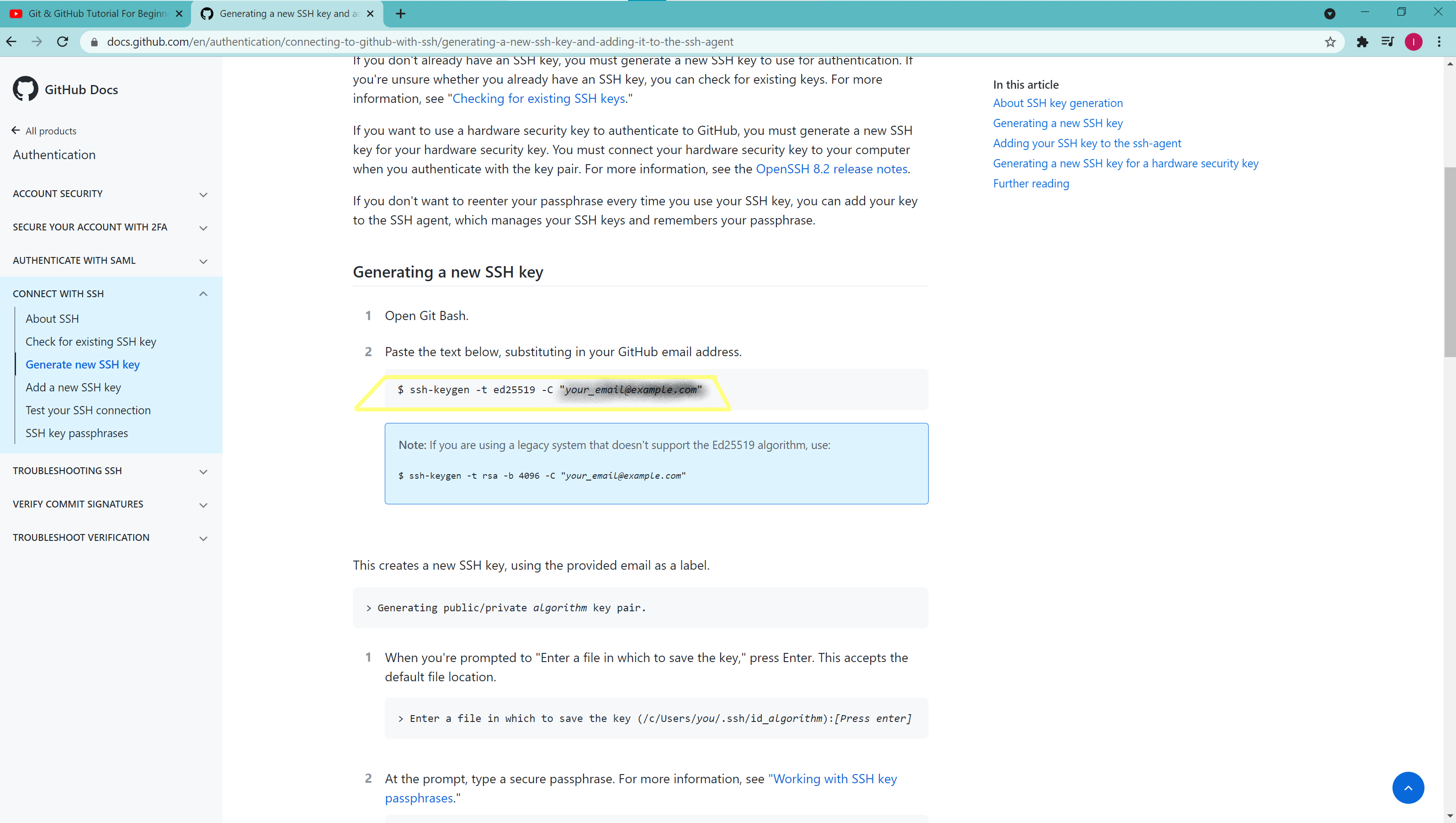
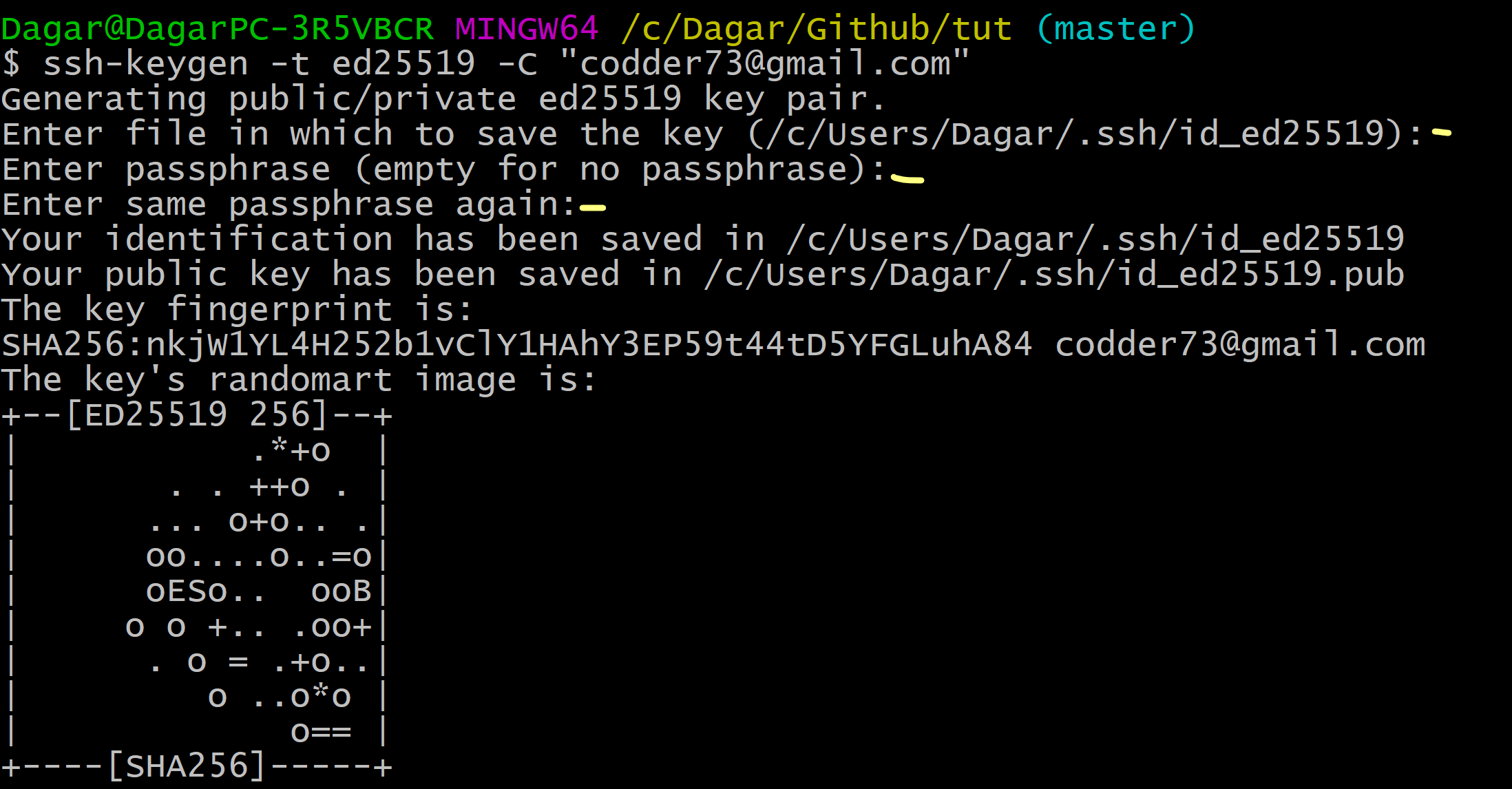
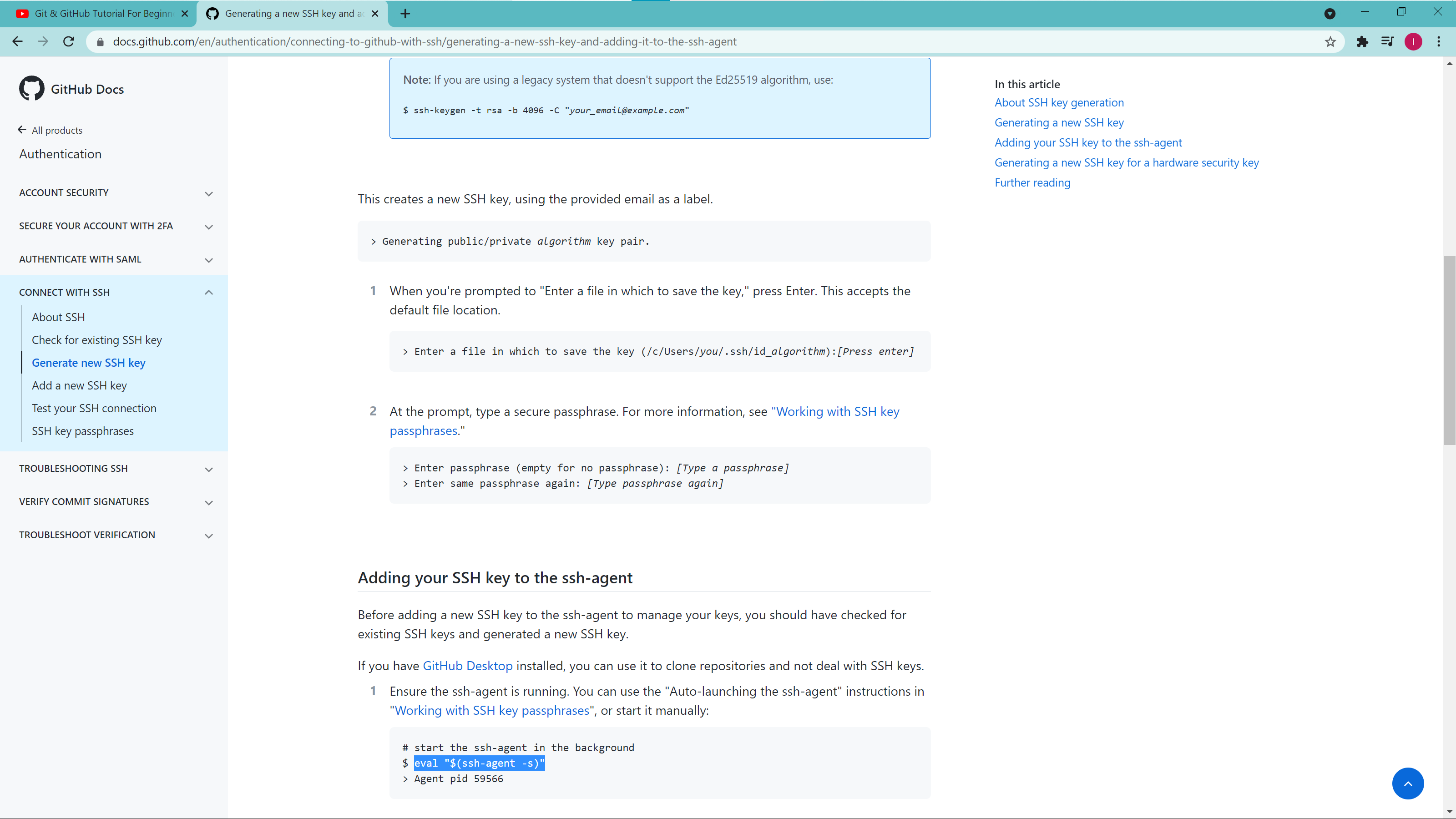
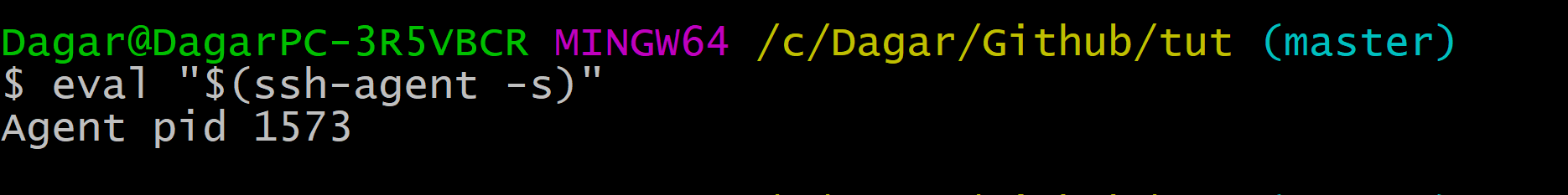
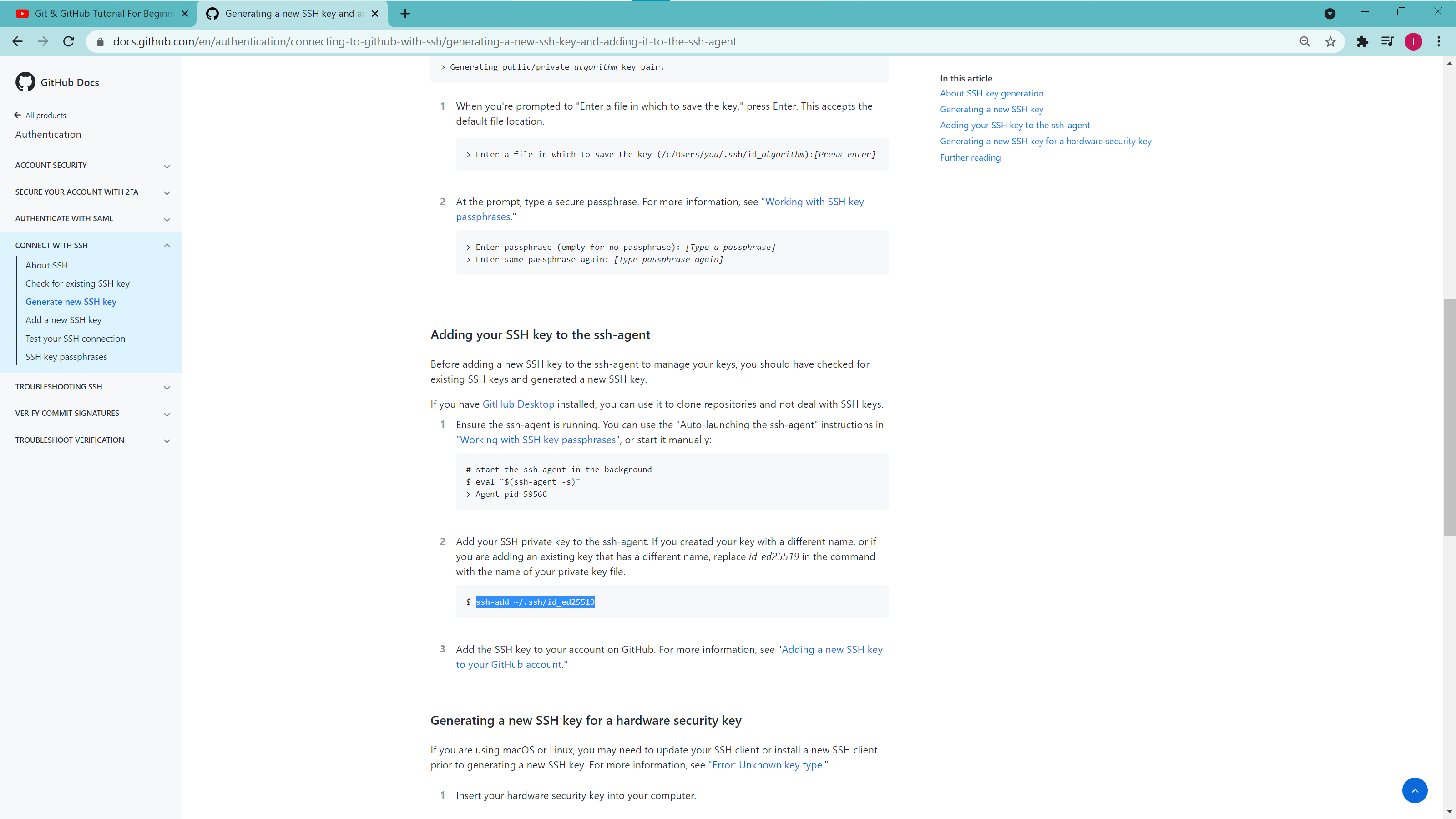
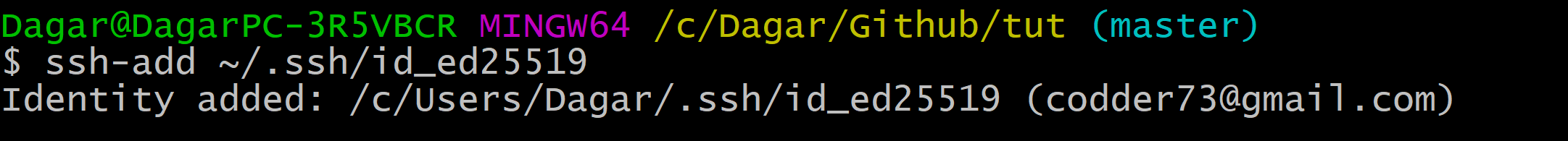
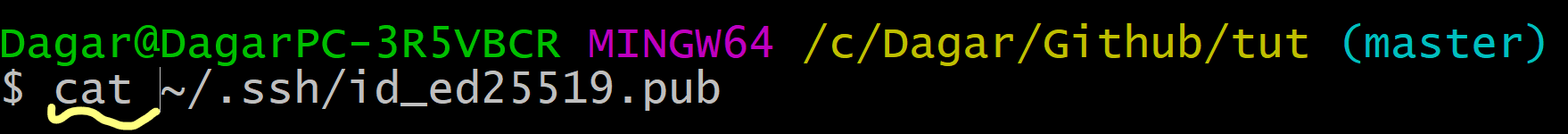
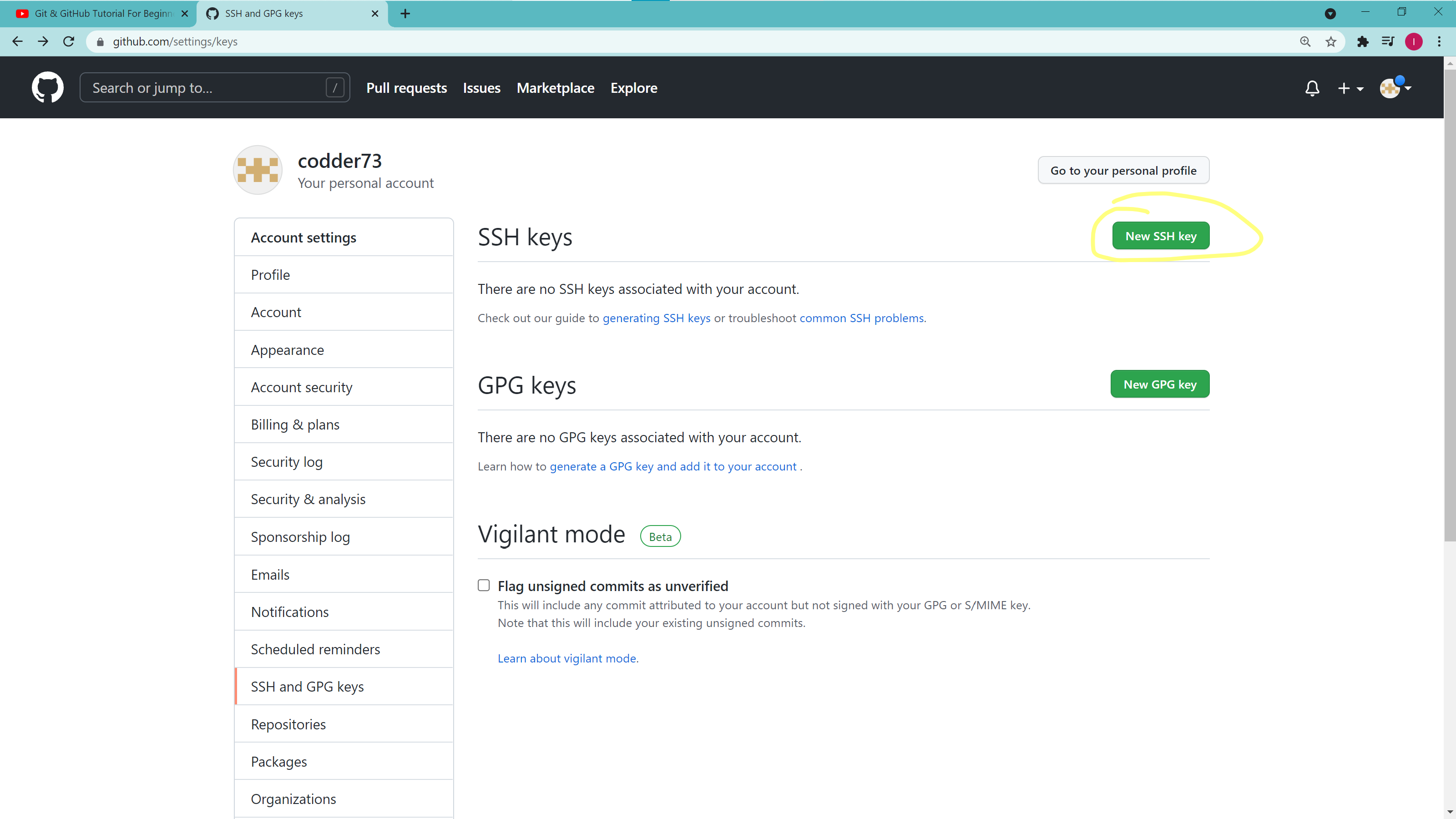
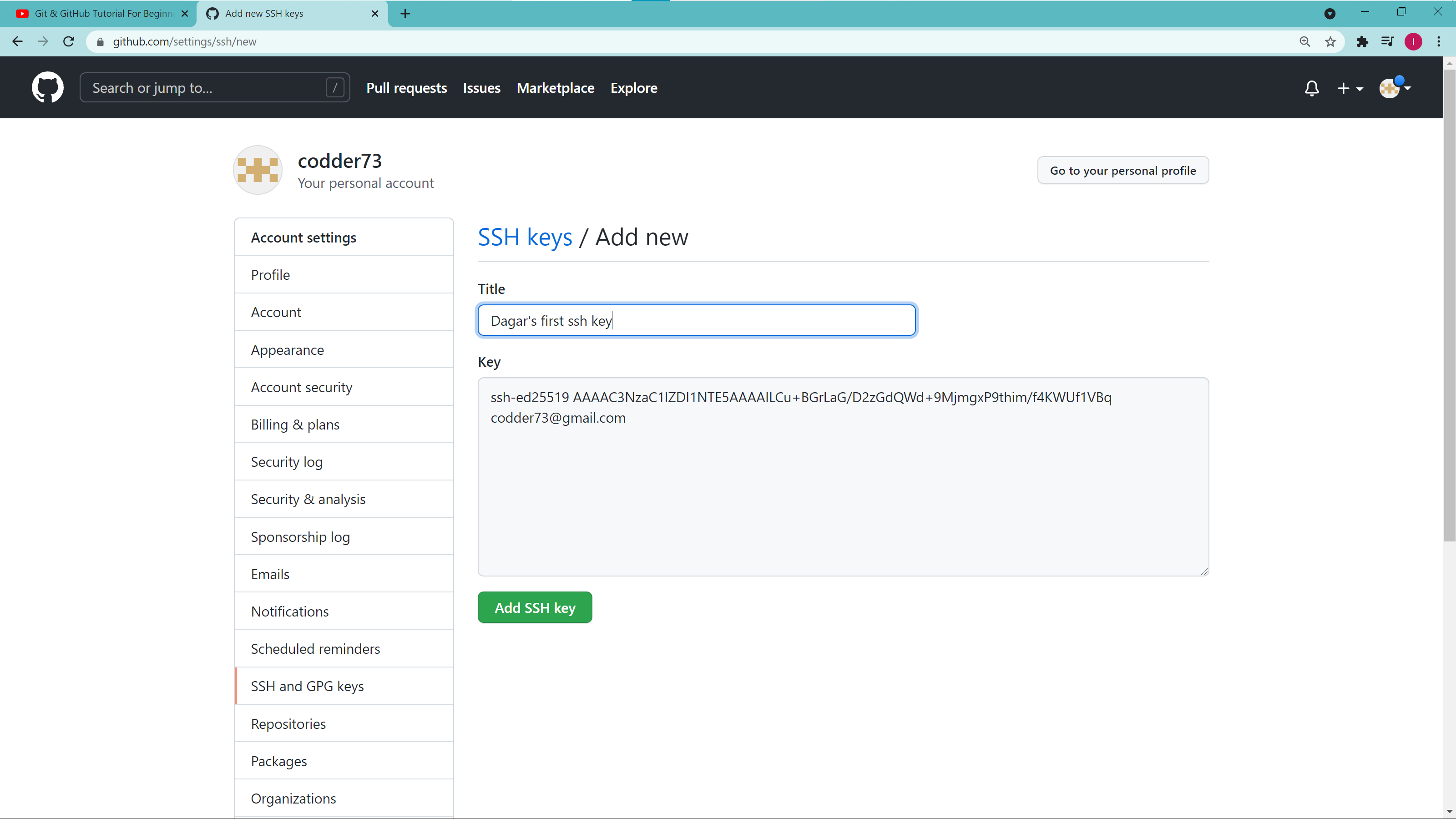
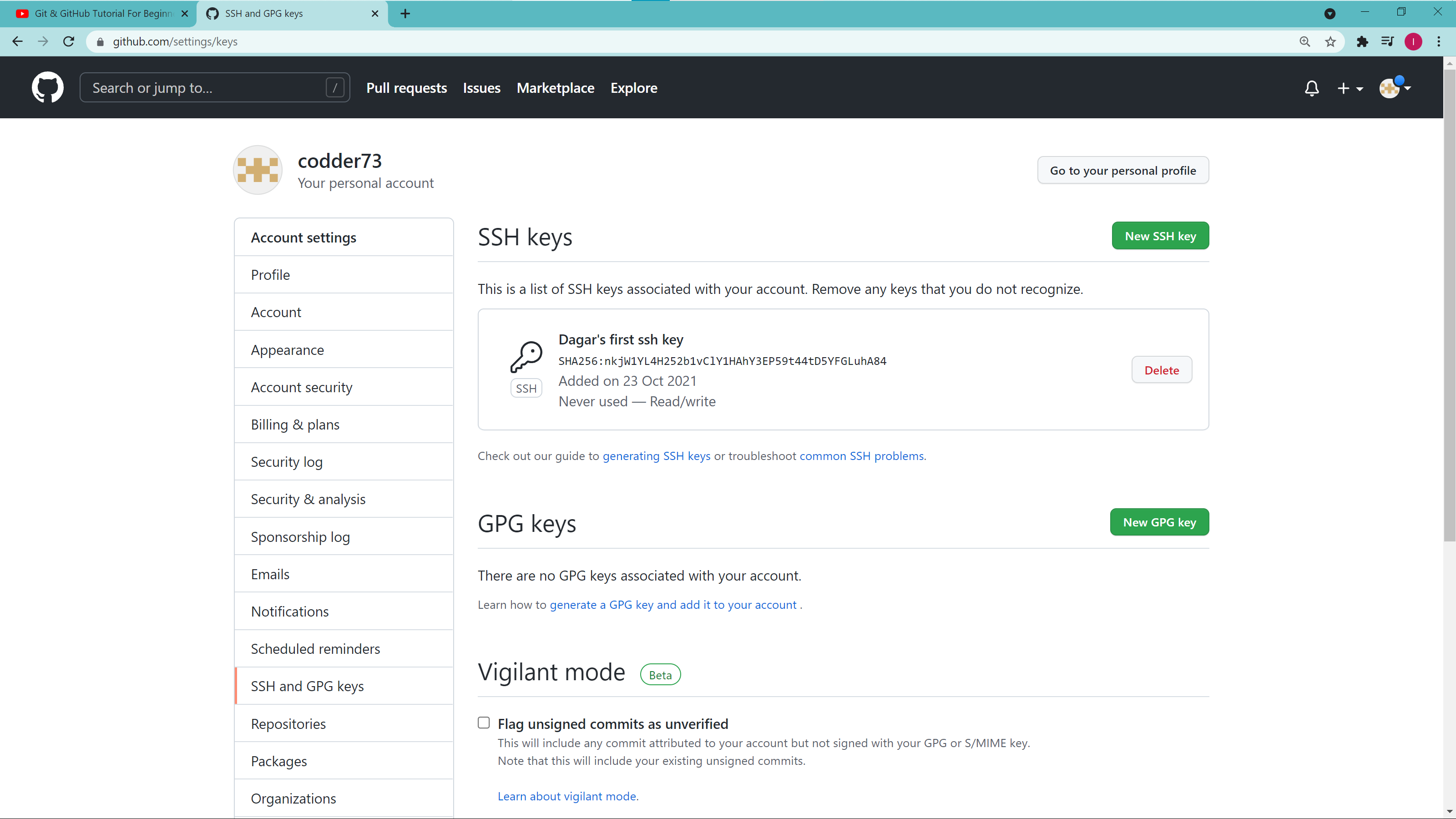
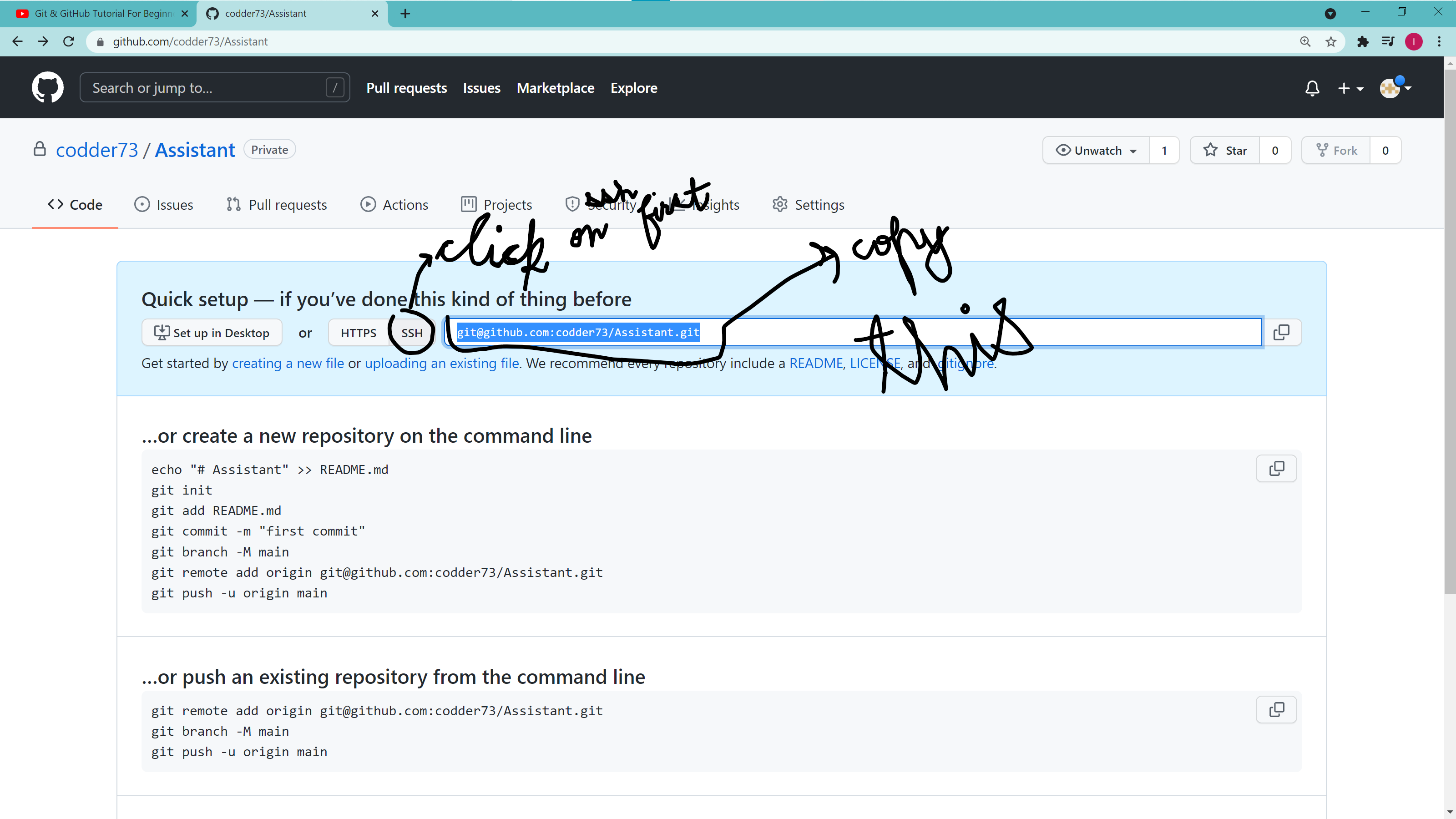
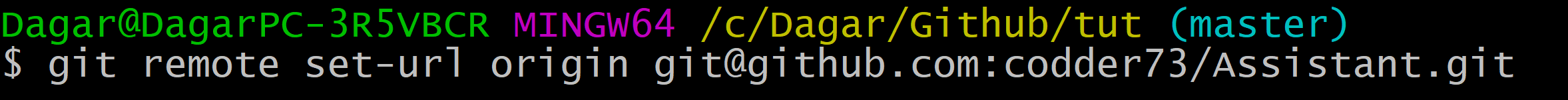
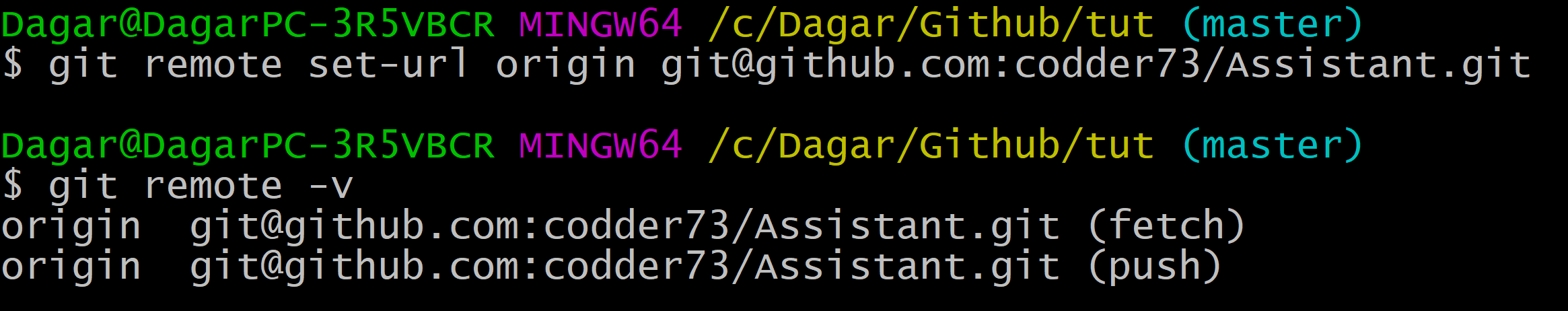


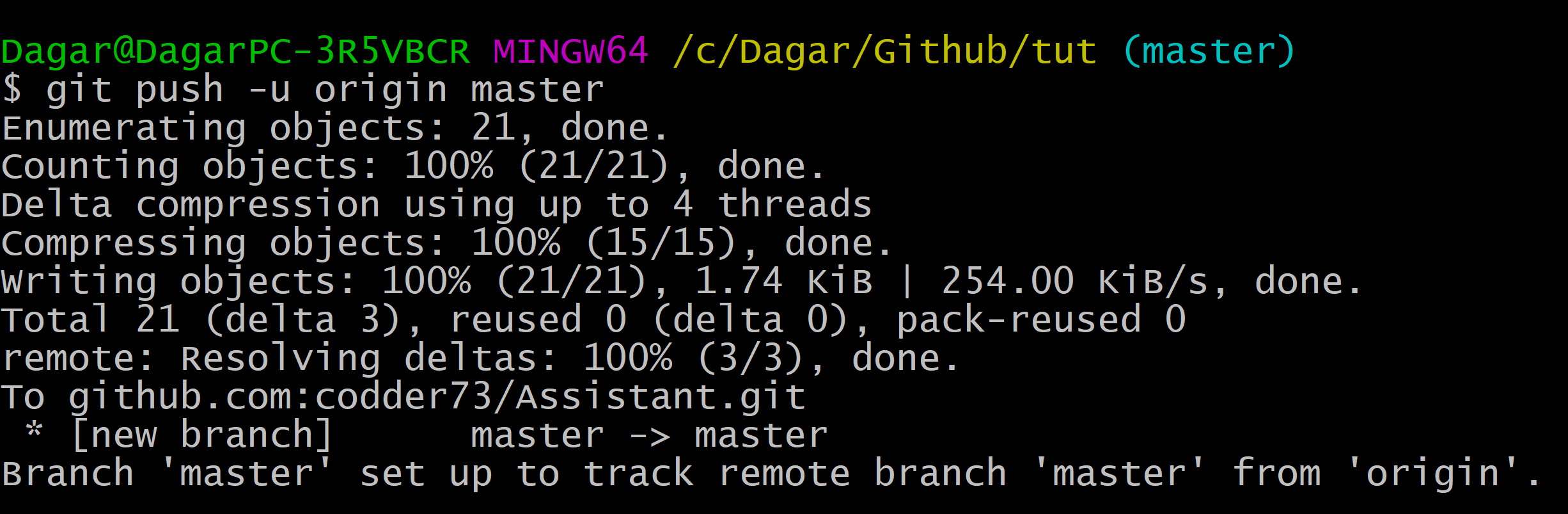
Now continue with these step if you have new account and not made any ssh key before, then make a ssh key by following these steps, And **if you have made ssh key before there is no need to make any new ssh key And continue with the 20th step-**

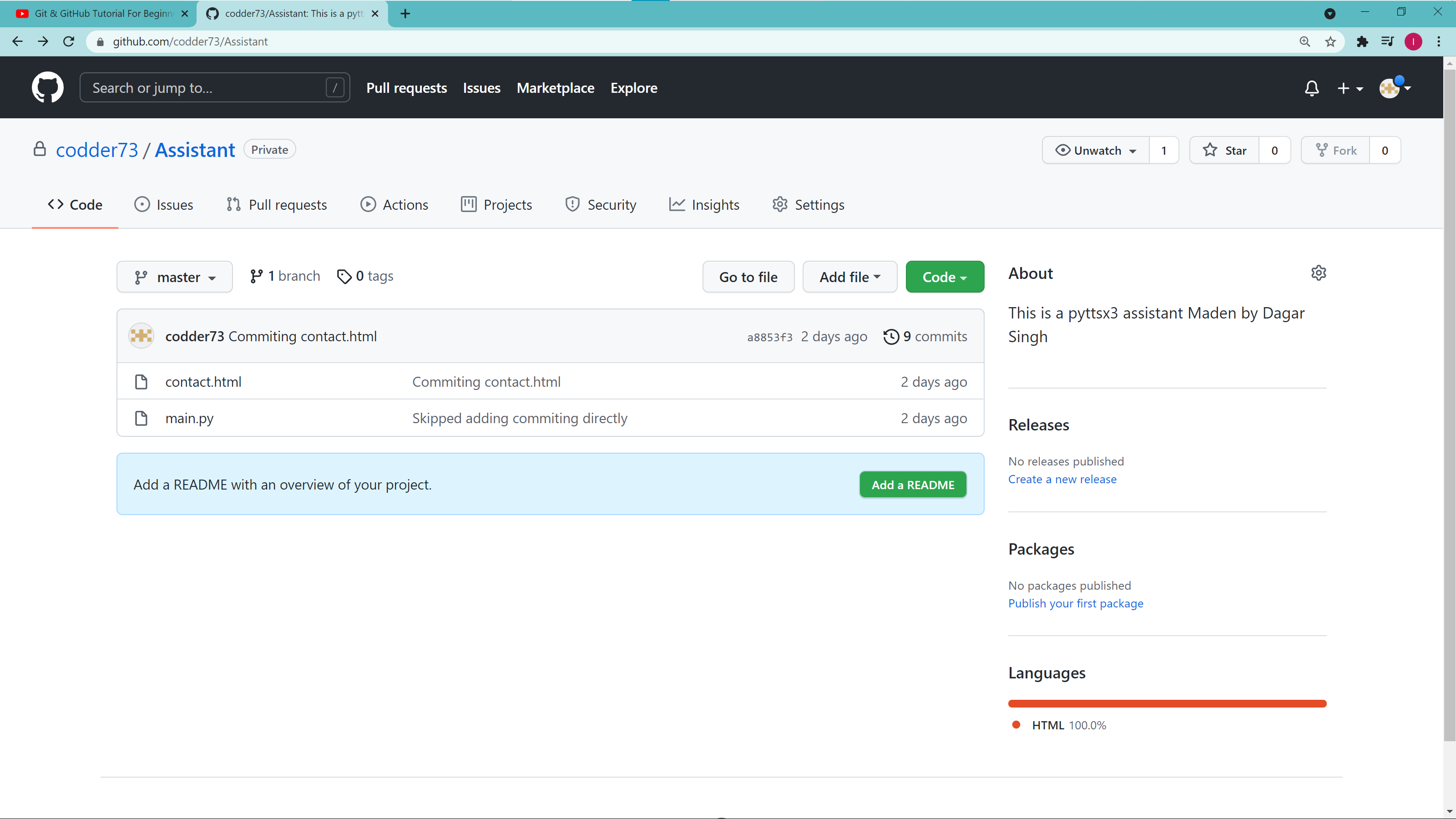
\*) GENERATING A NEW SSH KEY

1. Go to settings 
2. Go to ssh and gpg keys (underlined by yellow line ())
3. Now click on generate ssh keys 
4. Then click on generating a new ssh key and adding it to the ssh-agent



1. Then copy the second line and replace the shaded part by your id 
2. Paste this at the git bash and leave the questions which it asks from you blank: 
3. And now we have to deploy this by – go some down side it is just here 
4. And paste this eval in the git bash and you will ge your agent process id 
5. Now copy this ssh and paste it to the git bash -- 
6. Paste this ssh in the git bash like this -- 
7. Now we have to deploy the ssh key to our github account— now you go back you will options like this -- 
8. Click on adding a new ssh key to your github account and copy the highlighted part 
9. Type cat before pasting like this -- 
10. Copy the output
11. Now you can also follow the steps present in the website
12. Come to the ssh and gpg keys and the click on new ssh key 
13. The output I have said you to copy(step 14) paste that output here at the key part choose any title click on add ssh key 
14. Confirm your password
15. Now you will have a read and write access to your github account like this 
16. Now go in your repository(repository’s home screen), which we have made just now we will have to change our url, 1.Clink on ssh and then copy that link which is highlighted 
17. Go to the git bash and then type – git remote set url-origin master \_paste what you have copied in the previous step\_ then hit enter key \*Your url will have changed\* : 
18. Type – git remote -v \*see your url had changed, ‘these must be same’\*:
19. \*\*\*PUSHING THE FILES ON GITHUB\*\*\* TYPE – git push -u origin master like this one problem can take place here, I don’t want to type or describe it, So I would tell you just one thing that I think error could be that -

--- please change the origin master to origin main by – git branch -M main ; 

1. Reload your reopsitory page and then see your files are there ✌: 

SOME EXTRA GIT COMMANDS WHICH YOU SHOULD KNOW 

**git checkout===**

**If someone change your code and saved and shut down you computer and go away and then you want that that is impossible but if you have committed the file so you only have to type**

**(‘‘‘The checkout command only work on which files are modified!’’’)===**

=== git checkout \_the name of the file\_

**and there is one more command which checkout all the files present in the folder that is**

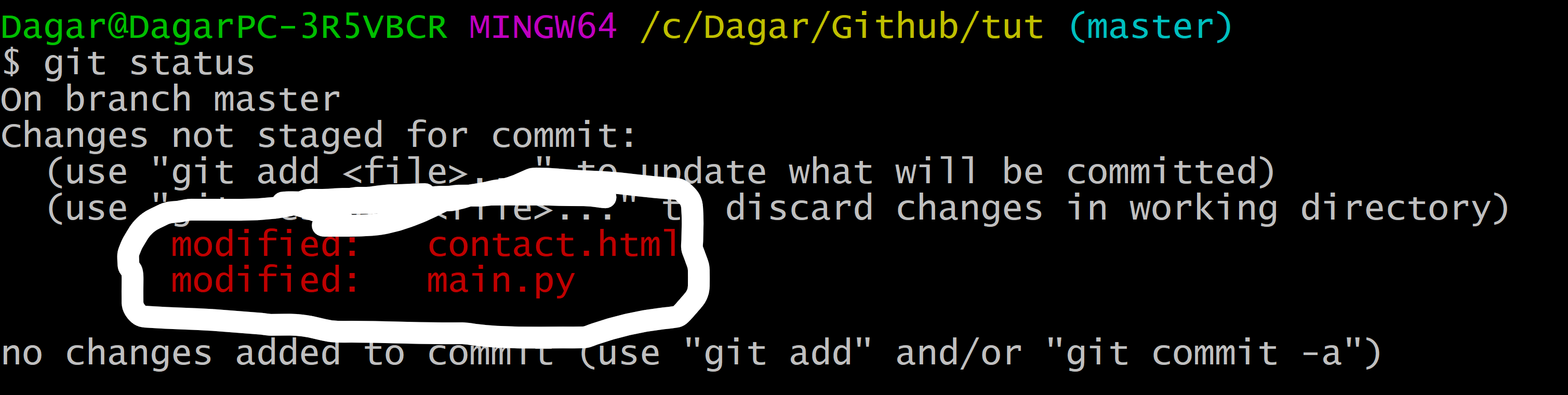
===git checkout -f #It will checkout all the files present in the folder

(this command is like a undo command)

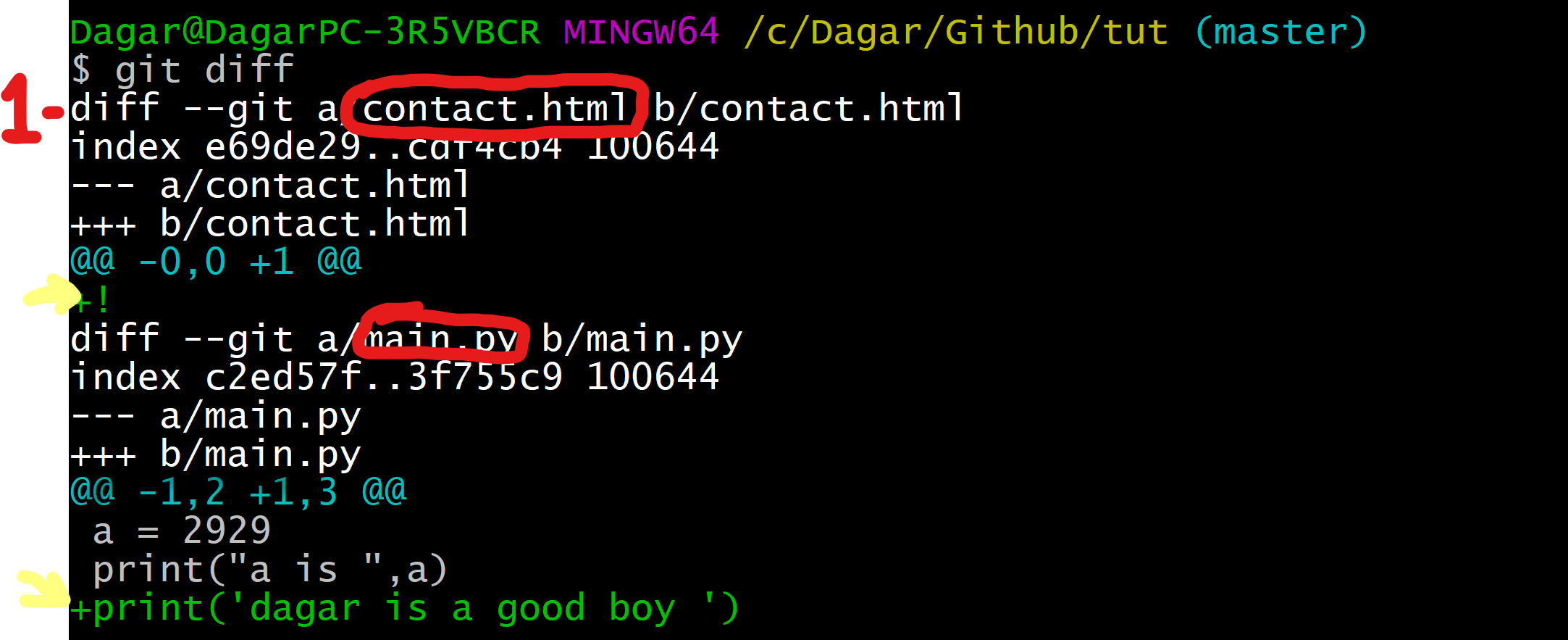
**Git diff === (I am telling you about a shortcut that diff means difference in the below para )**

It differentiate between the modified file by the previous(unmodified file(That file’s previous version)) and tells us about the diff… it only differentiate the modified file even if the file is only added(git add -A) it will not differentiate it with any file like we have 

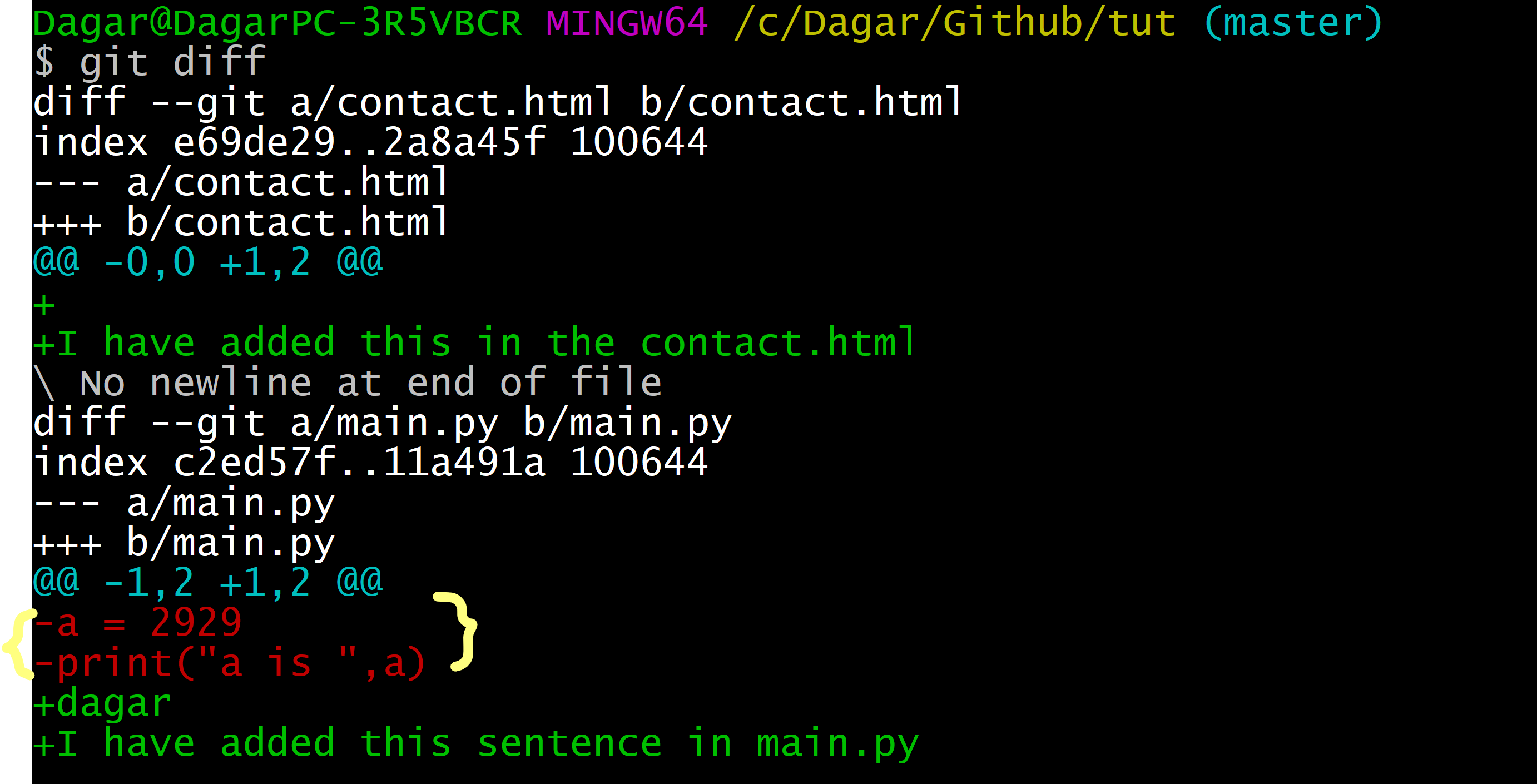
* Contact.html and main.py modified



* And if we now type git diff then it will tell us about the difference



* And if we remove any thing from the code so it will be colored as red like👇👇👇



The thing which is written in the red color is that which I have removed from the file and the code which is in the green color is which, that is new written

**Adding removing file and folder using git bash (using git bash as a terminal) **

If we want to add a file in the folder in which we have opened so type 

Write the name of the file you want to make after touch like = touch main.html

\*it is a untracked file \*

Then we can use short hand to commit it or we can do manual things to do this

**AND** if we want to delete any file we will use the rm command 

Type === git rm --cached \_name of the file\_ #if we use cached so then the file will be untracked it will present in the folder (in will only get out not killed)

And if we want to kill it from the folder also so

Type == git rm \_the name of the file\_

**COMMIT SHORT HAND**

IF YOU WANT TO DIRECT COMMIT WITHOUT STAGING IT 

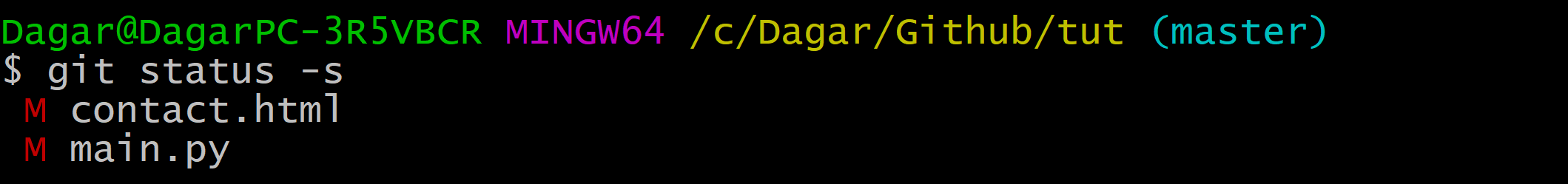
It is called short hand \*\*WE SHOULD NOT HAVE TO MAKE A HABBIT OF IT IT SHOULD BE DONE ONLY SOME TIME OR FOR THOSE FILE WHICH ARE EMPTY OR BLANK\*\*

TYPE --- git commit -a -m “\_any thing you want to type as commit\_”

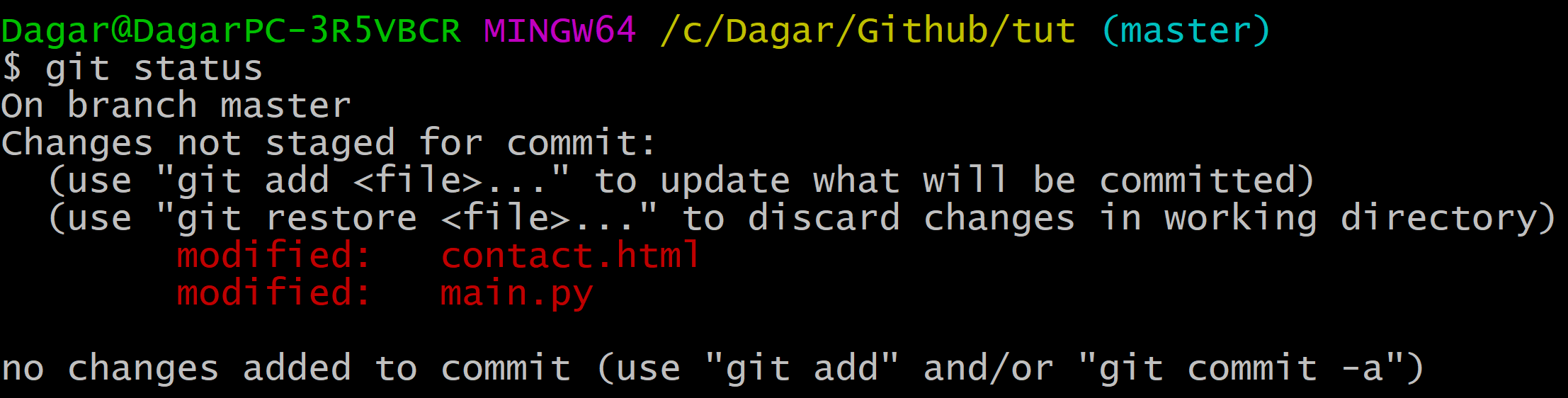
**NOTE** - if you modify a file you have to first add it (git add -A(stage the file)) and then you have to commit it (git commit -m “\_Write any thing what do you want\_”) **and** You can use comiting short hand for these works

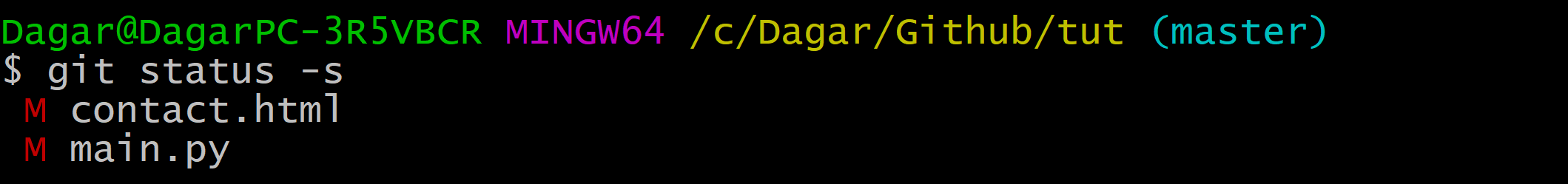
**Short hand of git status (git status -s) ===**

it is a short form of git status in this we can easily read the output(readability of the code or output)>

* If we type === git status -s #if there is nothing in the output it means that > working tree clean, And if in the output we say the M like this it means that the file is modified 

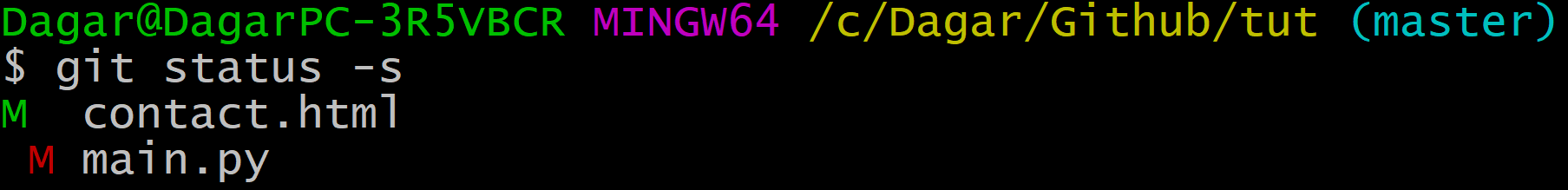
It means that the contact.html and main.py are modified…

These both are same  



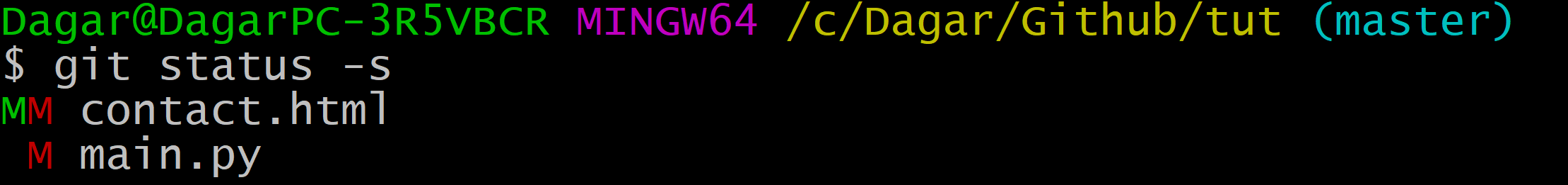
The only difference in these is that the -s status’s readability is better than the other

And if the M is like this type this means 



It means that the contact.html is staged (means it is added ‘git add contact.html’)

* If the any file is like contact.html it means that it is added and then it is modified



First it was modified and then it is added and after adding it is modified and it is same as👇👇👇