**# GitHub: Working with Remote Repositories (Learning to Push changes to Remote Repositories):**

@ Again, let's revise what are "Remote Repositories":

\* Remote repositories are versions of your project that are hosted on the Internet or Network somewhere.

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

**@ Understanding Push w.r.t GitHub / Hosting Services / Server:**

\* Pushing to Remote Repository:

---> Pushing to remote repository basically means updating the remote repository with local commits, we use "git push" command to do push changes to remote.

---> This above whole process of "Updating the remote repository with local commits" is called "Pushing to Remote Repository" aka "Push".

---> In simple terms we can say: "To move changes from the local repo to the remote, we push".

@ We've understood cloning from GitHub before when we studied "git clone" command, and that time we even cloned a remote repository to understand the commands and git clone logic much more efficiently.

---> So, now let's understand how to push to a remote repository.

---> And in next topic we will understand how to pull from a remote repository.

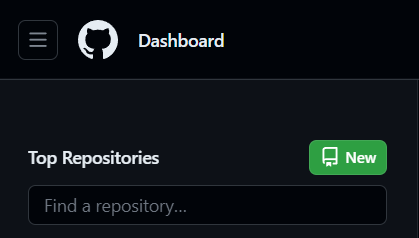
\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

--->

**@ Now let's understand how to Create a Remote Git Repository on GitHub and How to Push Changes to it:**

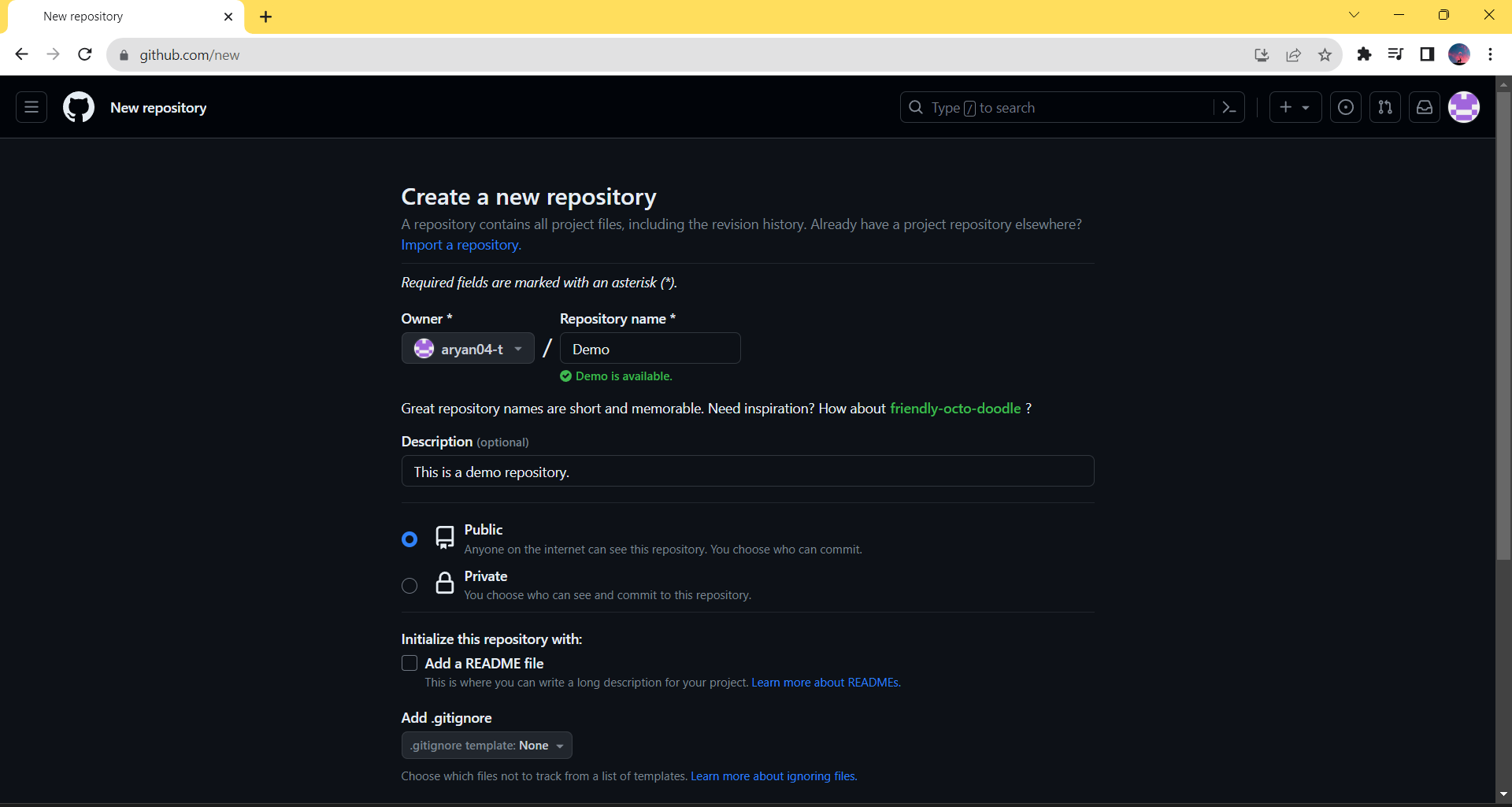
\* 1st of all, go to GitHub and make a new account if you don't have one, and if you have one then login into it.

\* Now when you're on your GitHub account's Dashboard, there you can see button named "New", it is present in the left most column.



\* Click on it and you will see a new web page, which says "Create a new repository", there just fill the mandatory details, repository name is the only mandatory thing.

\* So, give your repository a name, give it a relevant name, what codes or data you're gonna keep in this repository, name it regarding that.



\* Writing a description is optional, but if you want then you can add few lines to it, which describes your project or data which will be present in this remote repository.

\* Keep your repository "Public" for now, in future you can create private repositories but right now if you’re making your 1st repository then go ahead with public repository option only.

\* Now just ignore rest of the things on this page you can explore them later on and you can search about them on google and learn in depth about them, but for now let's just complete making this repository.

\* Scroll down and click on "Create repository" button.

A screenshot of a computer

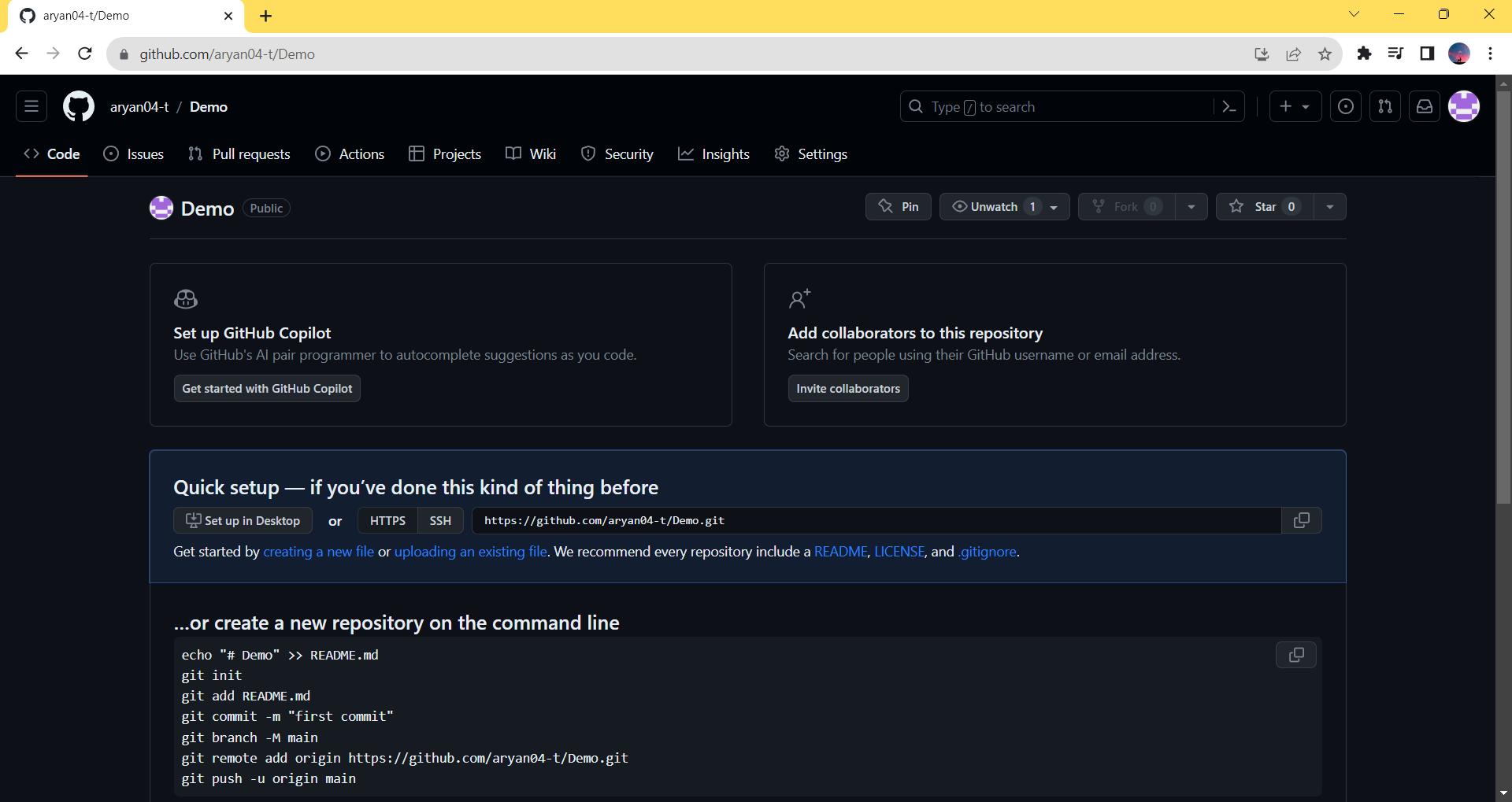
Description automatically generated

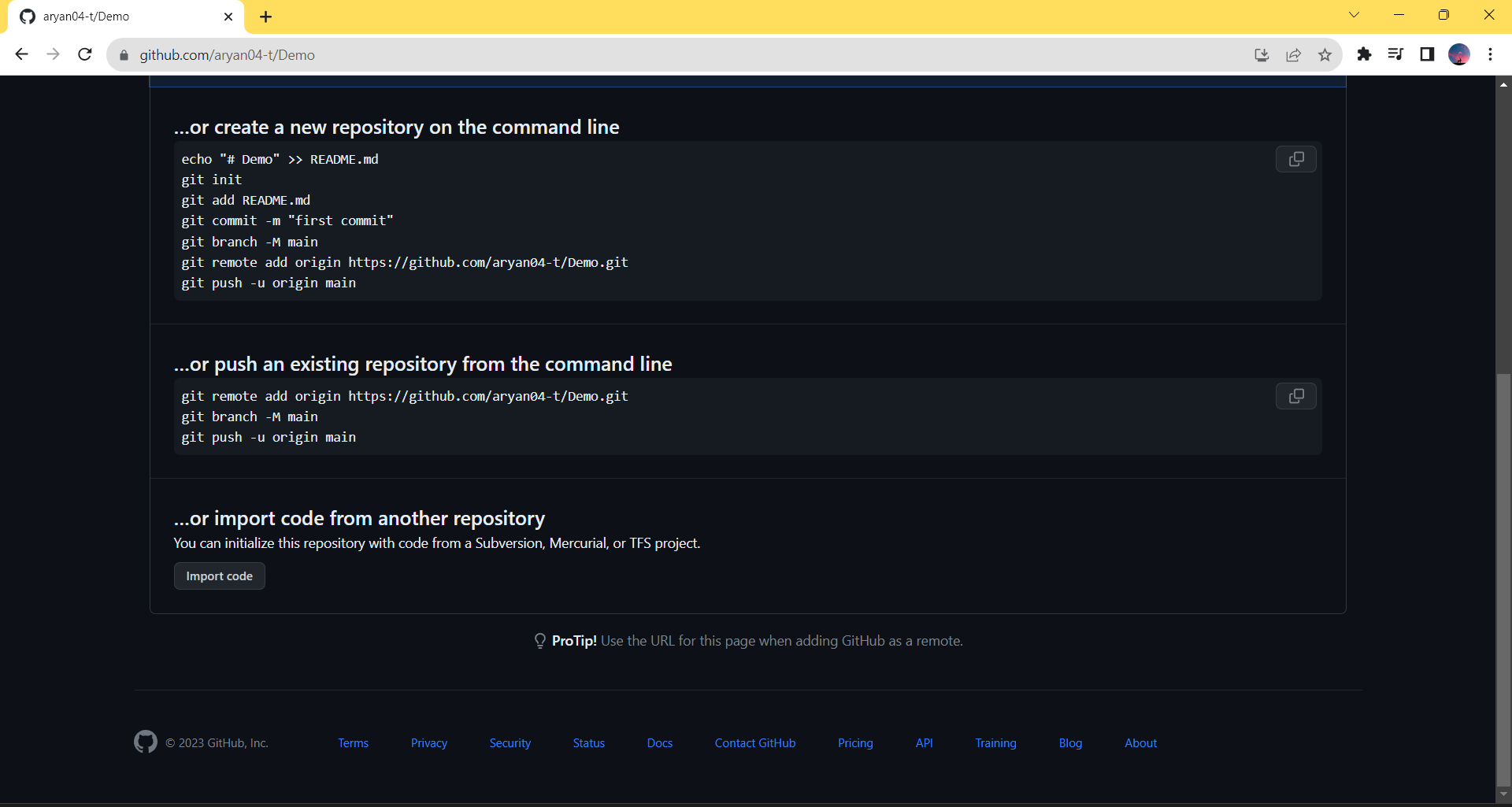
\* Your remote repository is created successfully.

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

--->

\* Now you will be directed to a new web page which looks like this:





\* Now on this new web page, scroll down and copy this below highlighted command and paste in your local git repository by opening git bash in it.

A screenshot of a computer

Description automatically generated

Command: git remote add origin <https://github.com/aryan04-t/Demo.git>

---> The link which is written in the above command is the HTTPS link of our remote git repository.

---> This above command will add a remote to your local git repository and the name of the remote will be "origin", it's not compulsory to keep the name of remote as origin, you can even change the name of remote to anything else, but as a beginner, using default name of remote - "origin" is preferred, because if you do so then chances of you getting confused with git an GitHub will be less.

\* Now basically, you have to push your default branch "master" to your remote repository, currently this branch which is named "master" is only present locally in your git repository.

\* We will study about branches ahead, we will study each and every minute detail about them, don't panic if you're not able to understand the above line very nicely, keep studying git and GitHub and by the end of these notes you will understand branches really nicely and you can come back again to this notepad to study this topic again with much more understanding, it's okay if you're not able to understand this in one go, but currently just do as instructed and try to understand as much as you can.

\* So, to push the "master" branch to the remote repository. Run this command.

Command: git push -u origin master

---> This above command means that we've pushed our default branch "master" to our remote repository, and that remote repository is known by name origin in our local git repository, because we have added our remote repository's URL in this remote which is named "origin", but if you're pushing to GitHub for the 1st time and you've not added a SSH key to your GitHub until today then you won't be able to push to your remote git repository on GitHub.

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

**# Let's understand 1st "What is SSH and What is SSH key pair?":**

\* SSH stands for “Secure Shell Protocol”.

\* When working with a GitHub repository, you'll often need to identify yourself to GitHub using your username and password. An SSH key is an alternate way to identify yourself that doesn't require you to enter your username and password every time.

\* SSH keys come in pairs, a public key that gets shared with services like GitHub, and a private key that is stored only on your computer. If the keys match, you're granted access.

\* The cryptography behind SSH keys ensures that no one can reverse engineer your private key from the public one.

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

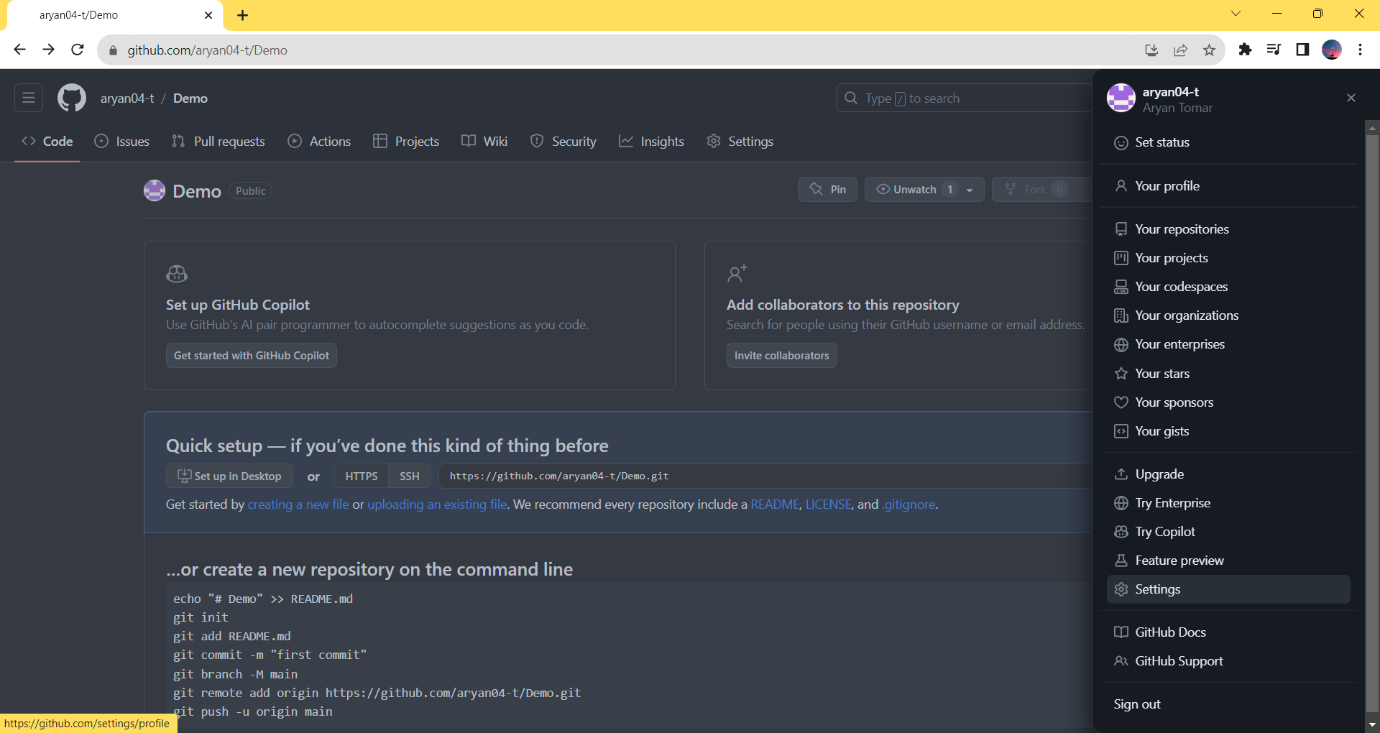
**# Adding Public SSH Key to GitHub (If you know how to generate SSH keys and how to add them to your GitHub account then skip this and next 2 sections):**

@ If you're pushing to GitHub for the 1st time, then the GitHub doesn't know your computer, and GitHub cannot allow any computer to push any changes to any repository randomly, otherwise anyone can add anything to any repository and integrity of GitHub will be lost.

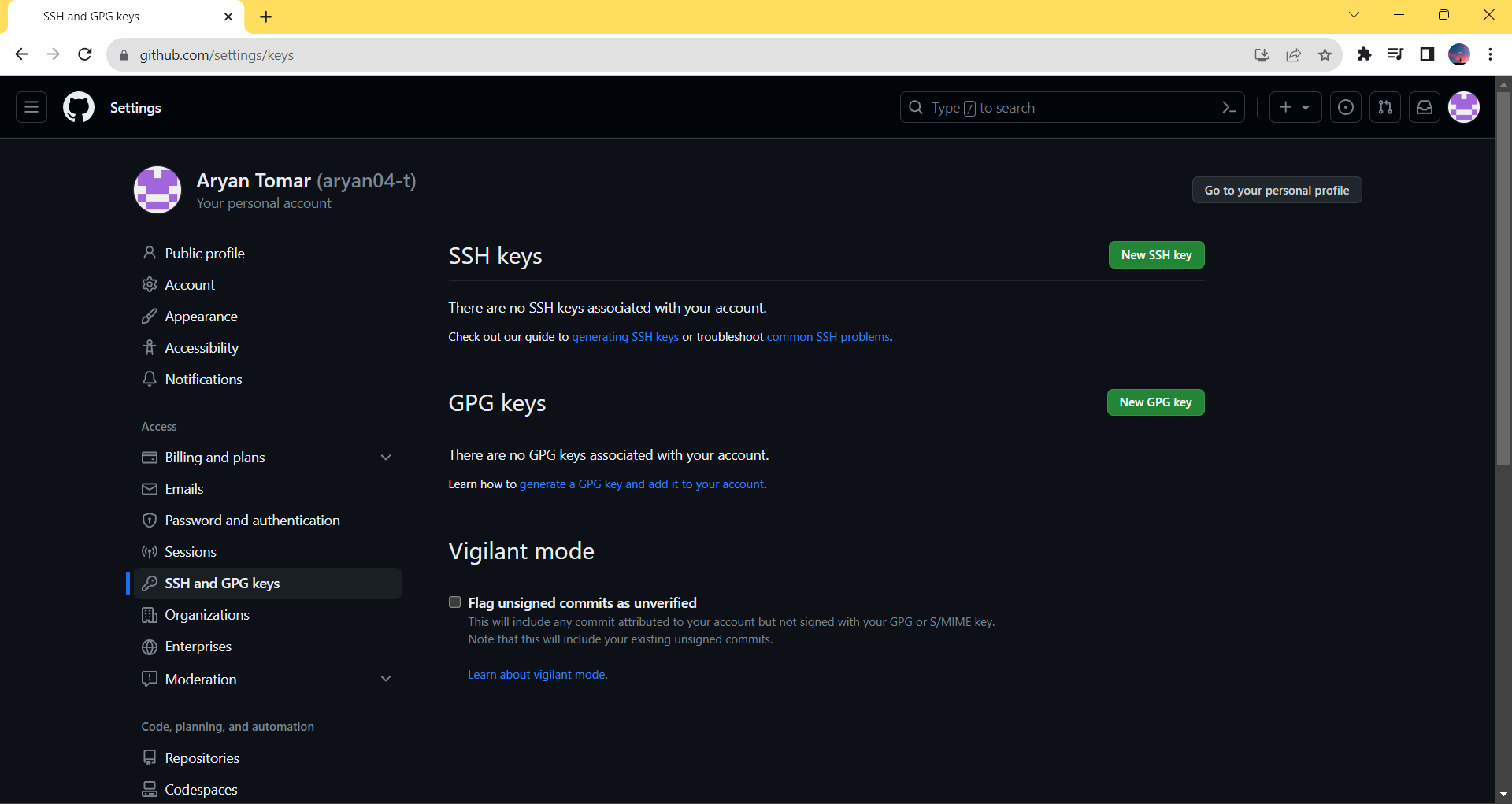
\* So, if you want to push to the remote git repositories which are there on your GitHub account, then you have to give your Personal Computer access to your whole GitHub account, so that you can push to any remote git repository of yours on GitHub using your PC.

--->

\* Now, Go to "Settings" of your GitHub account.



\* Now Go to "SSH and GPG keys".



\* Click on "New SSH key", add "Title", and to add SSH key 1st you have to generate a SSH key. So, let's 1st generate a SSH key and then we will come back to this window, and we will add it here.

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

**# Generating SSH Key Pair:**

\* Open a new tab of your browser and search "SSH keys GitHub”.

\* Click on 1st link which appears, it will be an article name (Generating a new SSH key and adding it to the ssh-agent) from official website of GitHub.

---> Link: https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

\* You can read above article to understand how to generate SSH key or let me summarize the whole article for you.

\* Open git bash and run this below command for SSH key generation.

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

---> Substitute your email in above command in the double quotes, use the email which you've used to create your GitHub account.

Ex: ssh-keygen -t ed25519 -C "myemail123@gmail.com"

---> After running the above command, you will be asked to name a folder where you wanna save this pair of SSH keys, I would recommend as a beginner just hit the enter key, a folder in your C drive will automatically be generated.

---> Then you will be asked to enter a passphrase, it is like a password which can be used to add extra security but it's already enough secure if you and only trusted people use your PC.

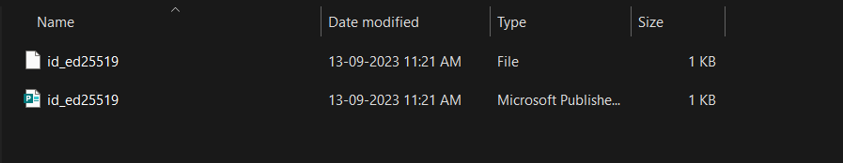
---> So, here also as a beginner I will recommend just to hit enter 2 times to skip setting-up passphrase, if you're familiar with git and if you want then you can set passphrase it's your wish, your SSH key pair will be generated after you're done with this passphrase setting-up section.

\* You can see this type of text on your git bash, which says "Your identification has been saved in /c/Users/Aryan/.ssh/id\_ed25519"

---> You cannot copy this path and paste it in your windows "directory management system". So, if your using windows then just simply open "This PC" go to your "C-drive" and follow the path graphically which is written above.

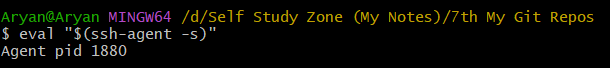
--->

\* You will reach the directory where your pair of SSH keys is saved.



\* Now use git bash and run this command.

Command: eval "$(ssh-agent -s)"



---> This above command will "start the ssh-agent in the background". You will get an "Agent pid" after running this command as an output, but we're not concerned with the output here, we want the ssh-agent to start running in the background if it's not running.

\* Now add the private "SSH key" to the ssh-agent.

\* If you created your key with a different name, or if you are adding an existing key that has a different name, replace id\_ed25519 in the command with the name of your private key file.

\* If you've not customized the name of the SSH key then use this command to add the private SSH key to the ssh-agent, we've not customized our SSH key name. So, we will run this command on git bash.

Command: ssh-add ~/.ssh/id\_ed25519

A black background with yellow and white text

Description automatically generated

‘ --->

**# Continuing Adding Public SSH Key to Our GitHub Account:**

\* You can refer to this official GitHub article to do this procedure: https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account

\* Else I will summarize the above whole article for you here.

\* Now run this below command on git bash to copy the SSH public key file to your clipboard.

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

---> This above command copies the contents of the id\_ed25519.pub file to your clipboard

---> If your SSH public key file has a different name than the example code, modify the filename to match your current setup. When copying your key, don't add any newlines or whitespace.

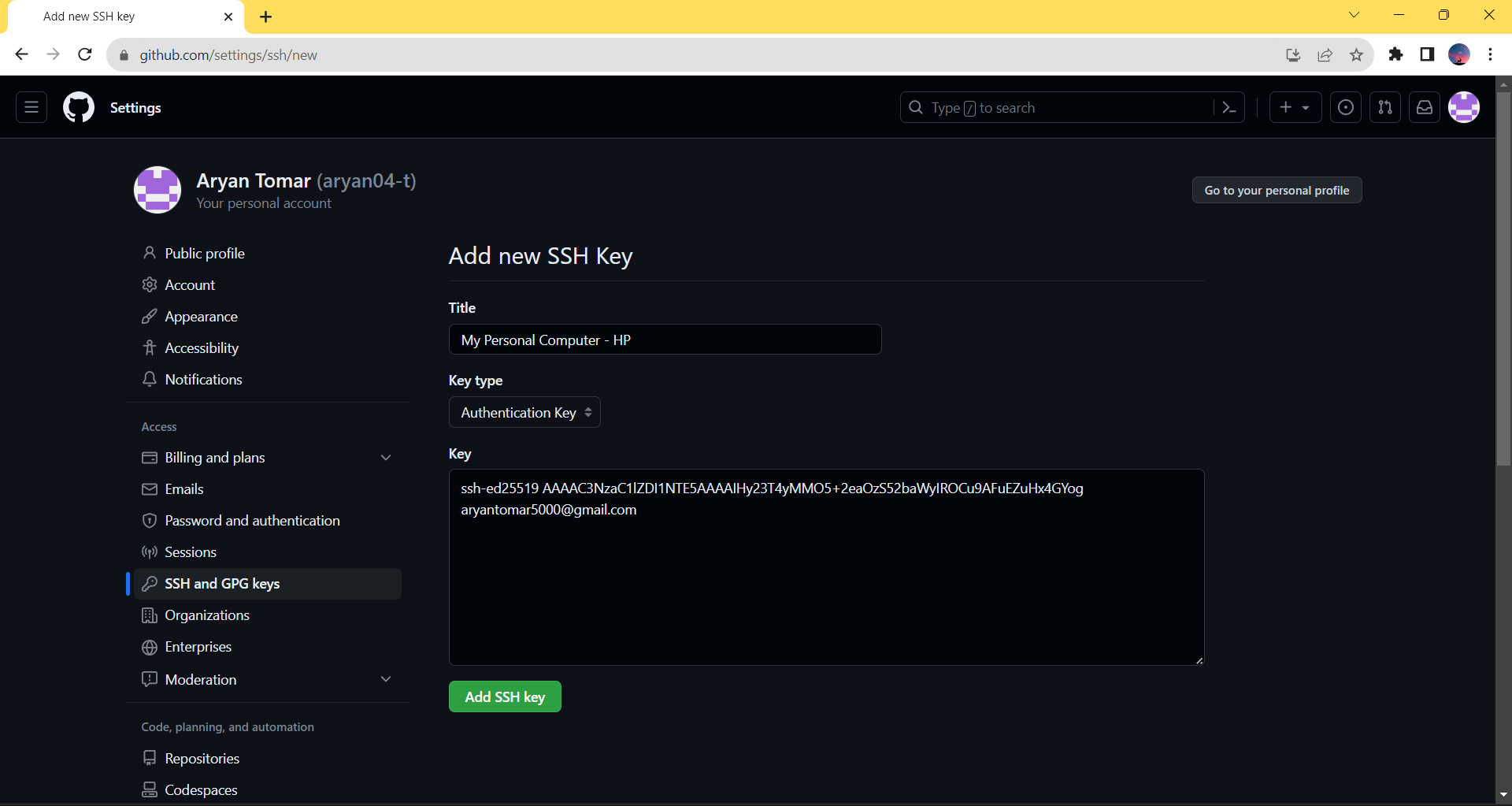
[OR]

\* You can also run this below command instead of above one, this below command prints the public SSH key on the git bash console and you can copy the from the console manually.

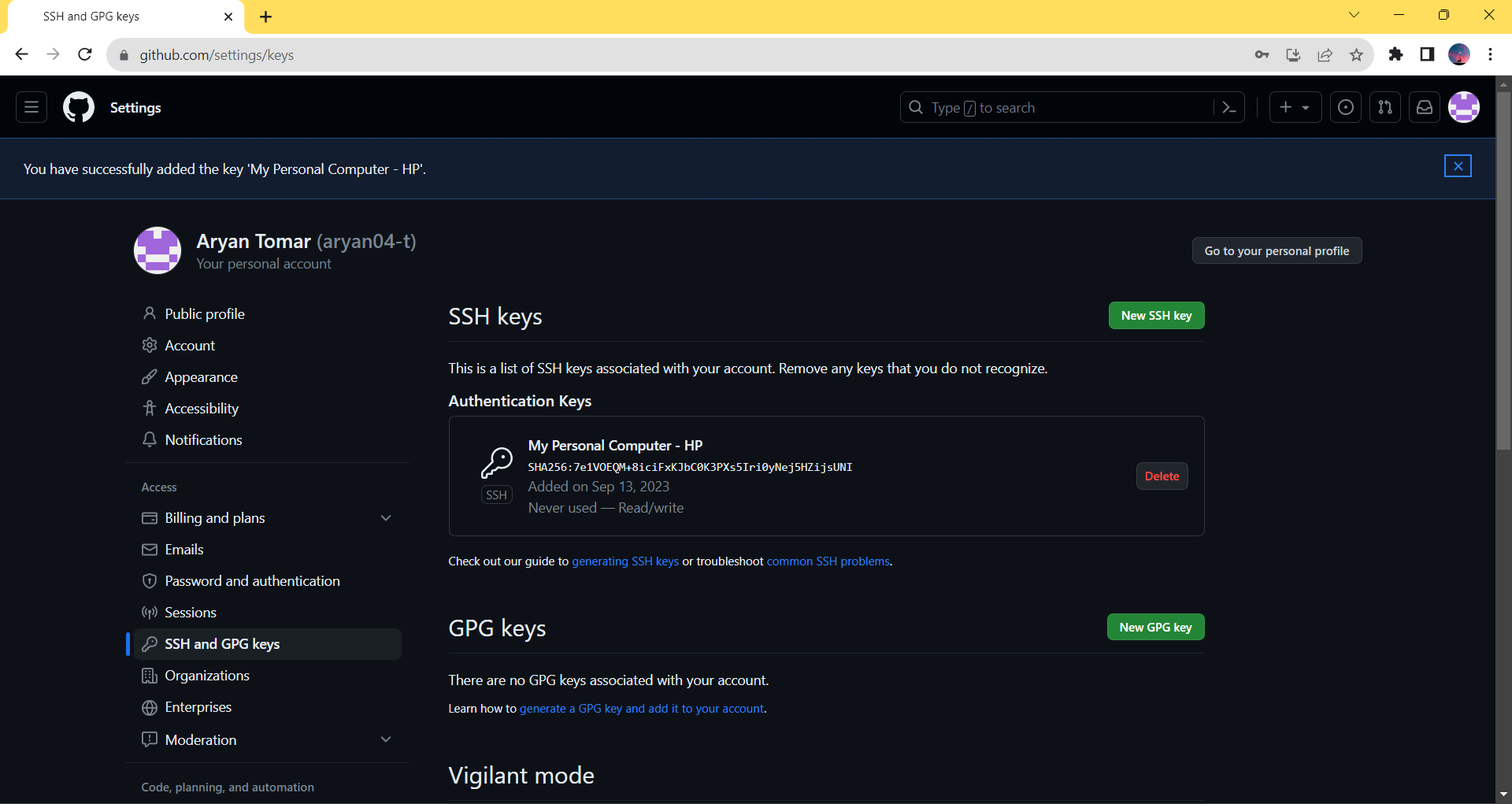
Command: $ cat ~/.ssh/id\_ ed25519.pub

\* After you run any one of the above 2 commands and perform the necessary steps, then the public SSH key is copied to your clipboard, now go back to GitHub's web page where you left “adding of SSH key” process in between.

\* Go there and paste your SSH public key in the "Key" section.



\* Click on "Add SSH key" button and now you're done.



\* Your SSH public key has been successfully added to your GitHub account.

@ Now your GitHub account recognises this current PC of yours, and you can push to your remote git repositories using this PC of yours from now onwards as much as you want.

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

**# Understanding Some more Commands of Remote:**

@ You can use this below command to check how many remotes are currently existing in your local git repository.

Command: git remote

@ Above we have seen command for "How to add a remote to your local git repository" and "How to check the current existing remotes in your local repository", but this below command will help you delete a remote if you have added any remote by mistake or that remote is no more of use then you can run this command in your git bash.

Command: git remote remove name\_of\_the\_remote

Ex: git remote remove origin

---> This above example command will remove your remote named "origin" from your local git repository.

[OR]

\* Instead of using "git remote remove name\_of\_the\_remote" command, you can also use a shorter command.

Command: git remote rm url\_of\_remote

---> This above command will also perform same action as of the earlier command "git remote remove name\_of\_the\_remote" but this command just have a little bit different syntax, it requires url of the remote to remove it, you can use of of the above 2 commands, whichever you're comfortable with.

\* Now to check whether the remote was successfully removed or not, you can run "git remote" command, this command will show you the name of all existing remotes and if the name of the remote which you removed is not present in the list then it has been successfully removed.

\* But there is one more command which you can run to verufy whether the remote was removed successfully or not:

Command: git remote -v

---> In above command -v means "verify", this command shows that how many remotes are existing there in you local git repository, and this command also shows little bit more information than "git remote" command, this above command doesn't shows just shows the name of the remotes, but it also shows the urls of the remote, that from where the remotes are pulling or pushing if we use them.

\* If you want to rename a remote then you can run this command:

Command: git remote rename new\_name\_of\_the\_remote old\_name\_of\_the\_remote

Ex: git remote rename new origin

---> This above example command will rename the name of remote "origin" to "new".

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

**# References Cited:**

\* 13th Video of CodeWithHarry (Complete Git Tutorials For Beginners In Hindi) Playlist : (GitHub: Working with Remote Repositories | Git Tutorials #13):

<https://www.youtube.com/watch?v=Thx5yerhlhs&list=PLu0W_9lII9agwhy658ZPA0MTStKUJTWPi&index=13>

\* Software Carpentry (SSH Keys for GitHub):

<https://jdblischak.github.io/2014-09-18-chicago/novice/git/05-sshkeys.html>

\* GitHub (Connecting to GitHub with SSH):

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

\* GitHub (Generating a new SSH key and adding it to the ssh-agent):

<https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent>

\* GitHub (Adding a new SSH key to your GitHub account):

<https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account>

\* Career Karma (Git Remove Remote: A Guide):

<https://careerkarma.com/blog/git-remove-remote/>

\* Beanstalk (How do I rename an existing Git remote?):

<https://support.beanstalkapp.com/article/16-how-do-i-rename-an-existing-git-remote>

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_