We talked about git which allowed to us manage different versions of our code locally , on our machine.In this video , it’s time to bring our code to cloud. So basically we will push our code from local repository to a web based , to a remote repository, and for that we will use git hub. now lets find out how this works. Now as I said github allows us to store our local git repositories in the cloud, so we turn them into remote repositories. For this we first need a repository, so lets create one.

go to a folder. Create index.html file. Then run-

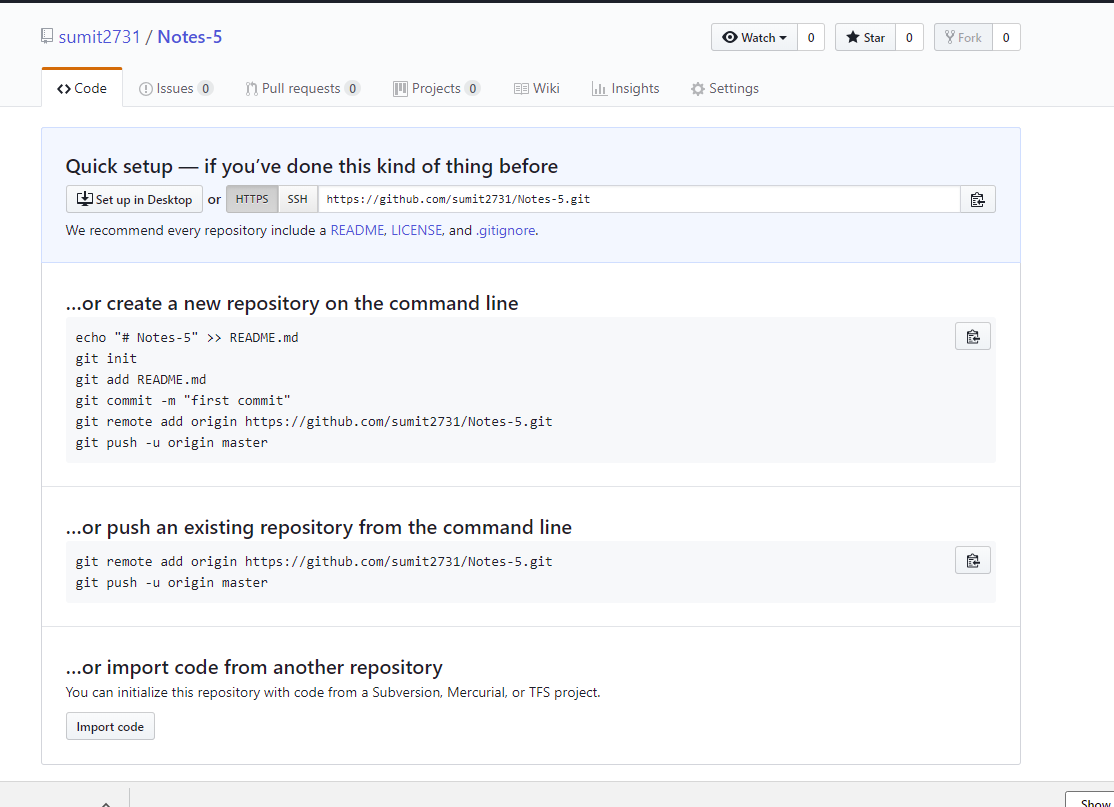
**git init**

then run **git add.**

Finally **git commit -m”first commit”**

Now make some changes in file, now again commit the code. So that we have 2 commits. Now run  **git log** and you can see your 2 commits. If we run **git branch**, we can see that we have only one branch, master. So that is all we know so far

So how can we now get this local repository to a remote repository, a repository on git hub. For this first go to github.com. create repository in github. It gives us option to add readme, add licence and gitignore file. we will skip them because they are not that important for basics. We will push the content of local repository into this remote repository. After creating repository you will see this page-

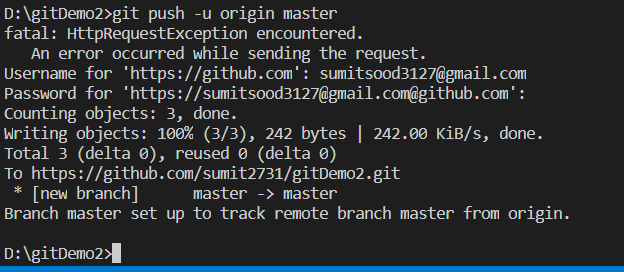


This repository does not contain any kind of code. So we have some nice hints on screen, about what we can do now. Second one talks about pushing an existing repository, that is what we want t do. So copy command listed there. These commands are-

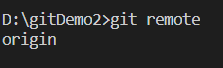
**git remote add origin https://github.com/sumit2731/Notes-5.git**

**git push -u origin master**

git command is quite clear, remote add means that we want to establish a connection between our local repository and remote repository(repository on git hub). Origin then is our name for this remote repository because we want to make sure that we establish connection between a specific remote repository therefore we have to it a name ofcourse and this typically is origin, you can ofcourse also choose a different name but for purpose of this basics video we will stick with this naming convention. Then we have a url after this. This url can also be seen on top of page also. You could also use ssh but this is also something that we will skip right now because with https we can perfectly get started with github and also push our code to github from our local repo. So run this first command.



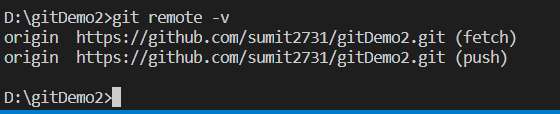
If you open repository on githib, you cannot see any files there. nothing changed basically. Actually that is partially true, run this command-



**git remote**

we get-origin

You can see that we added this remote repository already. You can be also be more specific about that, run-



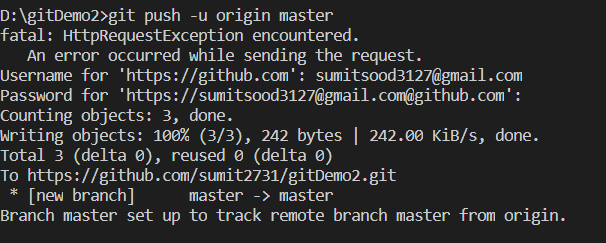
**git remote -v**

**D:\github>git remote -v**

**origin https://github.com/sumit2731/Notes-5.git (fetch)**

**origin https://github.com/sumit2731/Notes-5.git (push)**

then you can see origin right there, so the name of remote repository and additionally url behind that repository. We can see that we have fetch and push command right here. We will have look at them throughput this video. Important thing is that now we have a kind of connection between our local and remote repository but our branches and commits are not pushed to github yet. For that run second command-



**git push -u origin master**

it says git to push our branches to this remote repository , then we add  **-u** , this basically establishes an upstream , you could say between git and github. This will simply allow us to write commands in more convenient way so we can simply write git push for the next time we want to push some code and for pull, we only have to write pull, you will see all this in few seconds. So adding -u definitely makes sense right here. Then we add origin because we want to push our branches to origin remote branch(origin is name of remote repository not of branch) and then we not only have to specify a repository but also a branch and in our case ,this is the master branch that we will use. If we run above command , it will ask for username and password. Fill both of them. Now if we go to github, you can see that we have master branch there. We can also see our files in master branch. So we have pushed our code successfully. In our branch we can also see 2 commits that we have created.

Then instructor showed a info, how we can save our github password, so that we do not have to enter it when we do each push or pull. Lets get back to git hub.

As we can see that have establish connection between local branch and remote branch. If we run-

**git branch**

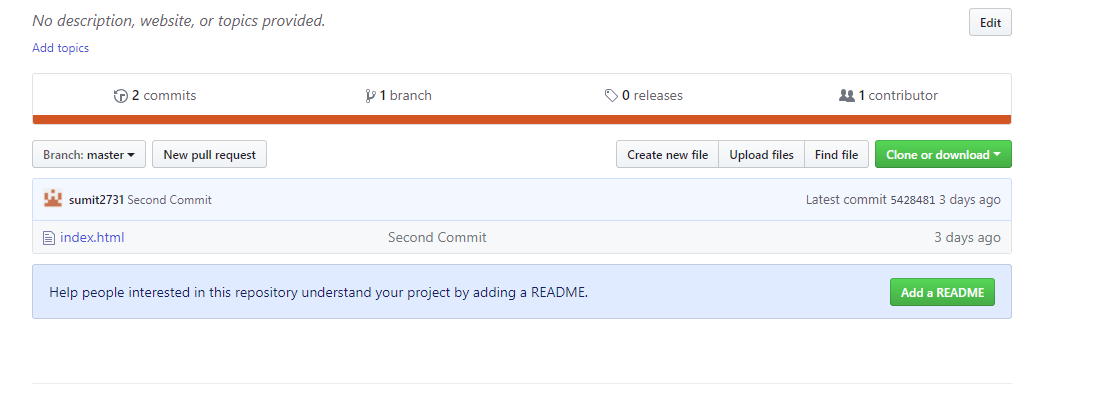
we get master. means we are in master branch in our local repository

If we run **git remote**

We get origin means we have a link, a connection with a remote repository(repository on git hub) which has a name origin that’s the name we defined previously. But we we run-

**git branch -r**

we get origin/master. What’s that? Without diving to much into details, this simply means that we have connection between local master branch and master branch in remote repository. That’s what we can see here, we basically have our remote repository which we point to with name origin and inside this repository we have master branch and this is exactly how connection is established between our local repository and remote repository. And this remote tracking branch is automatically created when we type certain git commands, or ex **git push** or **git pull**, as we was it in our case. So that’s why we have this origin master remote tracking branch right here. With that we got repository on git hub,the remote repository . What if we have a colleague now, who wants to access this code and maybe add something to it. For that click on repository on git hub.



Here we can see that we have clone or download option. If we click on that either we can download a zip or we can copy a link. We can use it to clone our repository. Let create a new folder named clone, we will clone our code here. Run this command-

**git clone** [**https://github.com/sumit2731/Notes-5.git**](https://github.com/sumit2731/Notes-5.git)

you will see that your code is copied here. Entire repository, entire content of repository is cloned or copied into this folder locally on our machine. Lets say we are working with clone folder repository. Lets we have added a new file, style.css. And we want to push that into remote branch then do this-

**git add .**

**git commit -m”new file added”**

now this a change that is saved locally. Now we want to push it to our git hub repository of course. But how do we do it. If we run **git remote,** we get origin. So remote repository is known. Run-

**D:\clone\Notes-5>git branch -r**

**origin/HEAD -> origin/master**

**origin/master**

so we have origin/master remote tracking branch also right here. This means now we can simply run –

**git push**

now entire code will be updated on git hub. You can go to git hub, you can see third commit and you can also see new file(style.css) there.

Now problem is how we get this file to our first user. That user have to download code from git hub repository. For that we have 2 options- one option is run,-

**D:\github>git fetch**

**remote: Counting objects: 3, done.**

**remote: Compressing objects: 100% (2/2), done.**

**remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0**

**Unpacking objects: 100% (3/3), done.**

**From https://github.com/sumit2731/Notes-5**

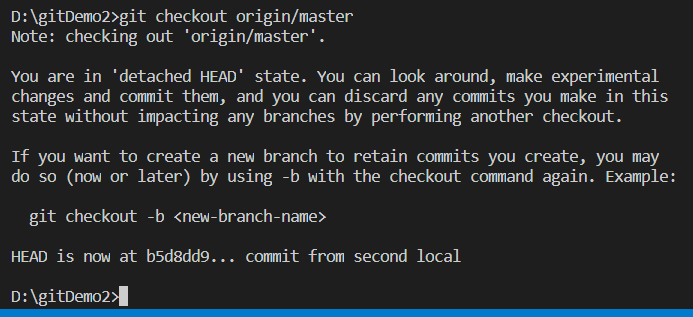
**5428481..effec2f master -> origin/master**

Here some things apparently happened right here but we still not see style.css file in our local repo. its’s strange because above it says that master is equal to origin/master. But somehow this is not reflected right here(we cannot see style.css) This is reason we had a look at remote tracking branches , a few minutes before. Because if I run-

**D:\github>git branch -r**

**origin/master**

we can see that we have our remote tracking branch right here and as we can see that our master on git hub is now basically equal to master in remote tracking branch(remote branch on local) and that means git hub and remote tracking branch kind of include same code. So if we checkout this remote tracking branch by



**D:\github>git checkout origin/master**

**Note: checking out 'origin/master'.**

**You are in 'detached HEAD' state. You can look around, make experimental**

**changes and commit them, and you can discard any commits you make in this**

**state without impacting any branches by performing another checkout.**

**If you want to create a new branch to retain commits you create, you may**

**do so (now or later) by using -b with the checkout command again. Example:**

**git checkout -b <new-branch-name>**

**HEAD is now at effec2f css added**

Now we can see style.css appears. But local code on our machine, in our local repository do not have reflect these changes. Now to reflect these changes we have to go back to our master branch so run-

**D:\github>git checkout master**

**Previous HEAD position was effec2f css added**

**Switched to branch 'master'**

**Your branch is behind 'origin/master' by 1 commit, and can be fast-forwarded.**

**(use "git pull" to update your local branch)**

Now style.css is gone.

Now run-

**git merge origin/master**

this is because we want to merge our local repo with remote tracking branch. And just now we have moved to master branch.

**D:\github>git merge origin/master**

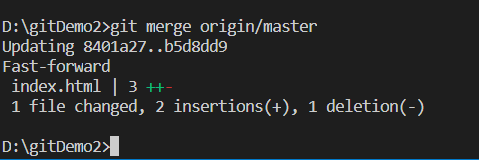
**Updating 5428481..effec2f**

**Fast-forward**

**style.css | 3 +++**

**1 file changed, 3 insertions(+)**

**create mode 100644 style.css**



now we can see that style.css is added and if we run **git branch** we can see that we are in master branch. And if we run **git branch -r**

we get origin/master.

So we can see that origin/master is also right here.

You can also combine these 2 steps, writing **get fetch** to get repository code from the git hub repository to the remote tracking branch and then writing **git merge** to merge code from remote tracking branch with our local branch, it can be done with just this command-

**git pull**

it will do both of these steps . it will fetch the code to remote tracking branch and then merge it with our local branch in end.

So that was all. Now what do you need to keep in mind.

You need to keep in mind the difference between a local repo and a github repo and inbetween these 2 repositories you also have these remote tracking branches which are kind of inbetween as the connectors you can say, between local and remote repo. And if you run **git fetch** command, then you kind of synchronize the code in git hub repo and remote tracking branch. Then we can run merge these changes to have these changes in our local repository. This is how you pull down your code from git hub branch. We can also push our code to git hub. Here you have to keep in mind that you refer to right github repository, we named it origin but as we discussed we can named it diffrenetly, and you also make sure that you refer to right branch. We always referred to master branch in our case.

Last thing is , how to delete a repo from git hub. Open repo. Got to setting, scroll down to danger zone, here we can see option of **delete this repository,** click on it. It will ask for name for repo fill it and your repo will be deleted. You can also delete the remote repository, the connection between git and remote repo in github, run –

**git remote**

you will get – origin.

So this is repository name- origin. Now run this-

**git remote rm origin**

now this repo will be deleted. You can also same thing in another folder(we have clones this github repo).

With this we have deleted the connection between git reposioty and remote repo on git hub.

Now lets say you have created a new branch, made changes in branch and commited it. Now you want to push code in that branch to git hub. But you want to have that code in separate branch o github.

Lets say we are in branch1. Run –

**D:\testGit>git push**

**fatal: The current branch geatre1 has no upstream branch.**

**To push the current branch and set the remote as upstream, use**

**git push --set-upstream origin geatre1**

so its clear from message that we need to set upstream branch for this branch. To do so run this-

**git push -u origin <branch>**

it will basically set up a remote tracking branch. note here name of branch needs to be same in both cases. If you run –

**git branch –r**

you will see new remote tracking branch is created.

If you want to create remote tracking branch for all local branches run-

**git push --all -u**