**Git**

It’s a source control system. It’s a distributed Version control system. It is like a single place where any developer can have all the essential source needed for the Project.

Advantage

1. Fast
2. Distributed- you can work even if disconnected, every developer has a local copy, asa work is done at the central place we can update changes.
3. It is powerful and Easy.
4. Branching- entire work is based on branching, we can create delete and merge branch. Master branch contains quality code.
5. Pull Request- a developer can ask for a review, and merge back in another branch. It helps in improvement of code quality.

**What is GitHub**

It is a hosting service using GIT, runs on top of Git. It has free (it will be open for all users) and paid plans. Source Management is the main moto.

**Base Concept of GIT**

It is a distributed Source Management System. In Central Source Management System, We have a central copy at the central server.

There are many Local systems as well, which will be merged to Central one.

In a Distributed system, work is little different. Every local system can get a clone of the central repository, this way every developer can work on the entire repository and later on, can merge it with the Master repository. For Merging we can use something called as push, and once the code in Main repository is available others call pull it to their system.

3 States of GIT

1. Committed-data is stored in local db
2. Modified –file has been changed but hasn’t commited to local db
3. Staged-modied file is marked to be pulled to the next commit snapshot

All states are local

3 Areas of GIT

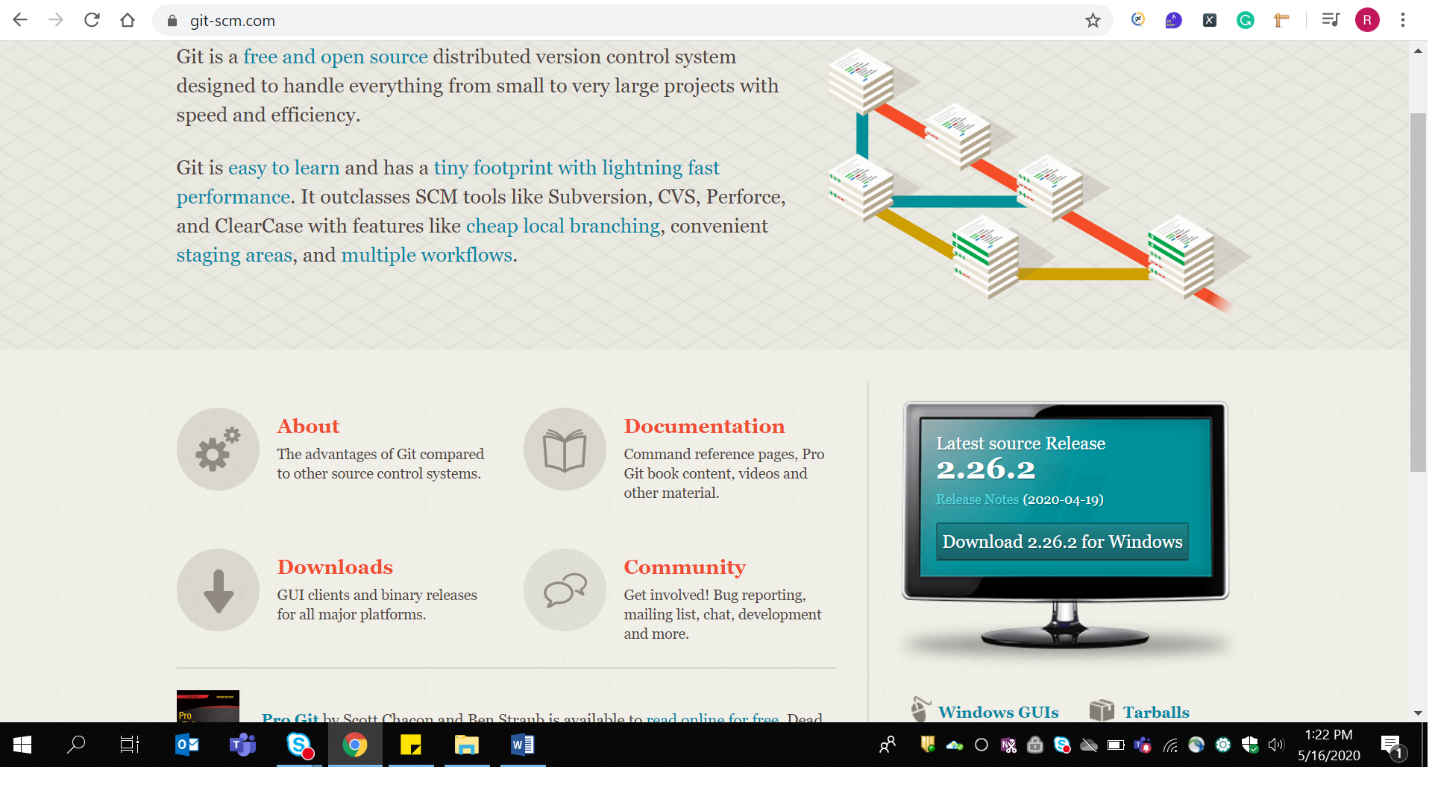
1. Working Directory- content is created, modified and deleted
2. Staging Area-before committed, it’s staged. They are waiting for be committed.
3. .git repo- this is a directory entirely managed by git.
4. Remote Repository Git HUB

Working with Git🡪 using command line

UI🡪GitHub desktop client (Source tree, GitKraken)

Gettiing Ready with Machine (For Windows)

1. Go to “git-scm.com”



Download it for windows.

1. Add Notepadd++ as editor
2. Git installs GIT BASH(which is the command prompt for GIT)
3. To configure global username-🡪 git config --global user.name "Rincy-Cheriyan"
4. To configure mail id🡪 git config --global user.email [Rincy3230@gmail.com](mailto:Rincy3230@gmail.com)
5. To open the global setting we can use 🡪 git config --edit –global

This will open our default editor with the set username and mailed

1. Enter🡪 notepad++ (This will open your default editor, but you need to set its path in environment variable)

Understanding Git commands

All commands start with “git”

$ git

$ git config🡪allows us to get and set config settings

$ git init🡪 to create an empty repository

$ git clone🡪 will clone a repository,into a local directory and will also create a branch

$ git add🡪 we can add a file to a staging area (which could be committed later)

$ git commit🡪 commit changes to local repository

**Demo for all the commands**

1. To change the directory use -🡪 cd “directory name”
2. Git init DemoGitDirectory--🡪 will create an empty directory
3. Cd DemoGitDirectory🡪 will change our path to our directory
4. If you want to check which directory has created use ls –la
5. To find the status use “git status” command
6. Lets create a file using notepad++
7. So type notepad++ README.md, this will open a file; you can type something there, save it and again run the command “git status”; now you will see a different result

On branch master

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)

README.md

nothing added to commit but untracked files present (use "git add" to track)

1. Now to add that file, we will use the same command “git add README.md”
2. Again enter “Git status”

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: README.md

1. Now we can commit the file using git commit –m “Initial commit”

[master (root-commit) 58f63fe] initial commit

1 file changed, 1 insertion(+)

create mode 100644 README.md

**GIT HUB**

Is a web based hosting service for GIT. It is used by developers to put their code there. It extends, what we can do with GIT. It provides features like

🡪Distributed Source Code Management

* Version Control
* Issues and Project Management Tool
* Got a nice system for notification
* We can also have Teams, assign tasks to Team members n so on
* Offers integration with many tools (ATOM editor)

**Getting Started**

1. Create Account
2. Your Git Hub Profile
3. Create Repository

🡪It’s a folder, in which your project resides, entire history of file is stored.

🡪 When you create a repository it is owned by only you. Later on it could be collaborated.

🡪it could be public or private

1. Create a new Repository (public) without README.md file

IT WILL SHOW YOU COMMANDS (THAT YOU HAVE ALREADY USED IN GIT LOCALLY)

1. To Merge your local file to your Remote repository

We can use “ git remote add origin <https://github.com/Rincy-Cheriyan/FirstRepo.git>” command

To check the status, if GIT is linked with any remote repository

“git remote –v”

As you probably know, git is a distributed version control system. Most operations are done locally. To communicate with the outside world, git uses what are called remotes. These are repositories other than the one on your local disk which you can push your changes into (so that other people can see them) or pull from (so that you can get others changes). The command git remote add origin git@github.com:peter/first\_app.gitcreates a new remote called origin located at git@github.com:peter/first\_app.git. Once you do this, in your push commands, you can push to origin instead of typing out the whole URL.

1. To push the file(local content ) to remote we can use other command

“**git push -u origin master**”

This is a command that says "push the commits in the local branch named master to the remote named origin". Once this is executed, all the stuff that you last synchronised with origin will be sent to the remote repository and other people will be able to see them there.

It will ask you for username and password

And it will upload data to your remote repository

1. Go to your GitHub and refresh it and you can check your file there.
2. Check the status of your GIT again using “git remote –v” command

**Connecting Over SSH**

Uptill now, we were connecting with GIT(Local machine) to GITHUB using https:, but Git is providing other options too. While using https:, while we are pushing the code to github, everytime it will ask for username and password to ensure security, to overcome that we will use SSH, wherein we can have ssh key attached to our account.

**Demo**

1. Use “git remote –v” to check the status of GIT

**You will get**

origin https://github.com/Rincy-Cheriyan/FirstRepo.git (fetch)

origin https://github.com/Rincy-Cheriyan/FirstRepo.git (push)

1. Use the command

ssh-keygen -t rsa -b 4096 -C “[rincy3230@gmail.com](mailto:rincy3230@gmail.com)”

you will get

Generating public/private rsa key pair.

Enter file in which to save the key (/c/Users/richeriy/.ssh/id\_rsa):

It will generate Key at the above specified location

PRESS ENTER

Enter passphrase (empty for no passphrase): (Enter ur password)

Enter same passphrase again: (Confirm ur password)

Your identification has been saved in /c/Users/richeriy/.ssh/id\_rsa

Your public key has been saved in /c/Users/richeriy/.ssh/id\_rsa.pub

The key fingerprint is:

SHA256:VZx/ywNCP+NuomFbunmTauyqZQgfhjnIHoIxoV6ty8g rincy3230@gmail.com

The key's randomart image is:

+---[RSA 4096]----+

|. ... |

|.. . oo |

|+ . . o .. |

|+oo + . . =. .|

|o= \* o S o =..|

|+ + \* o . + |

| E o o o.o .o .|

| o .o\*= o |

| ...+O= + |

+----[SHA256]-----+

1. Now go to the folder where your key is stored,

And open .pub file using NOTEPAD/NOTEPAD++

Copy the entire Key

1. Go to your github account
2. Go to your profile>> Settings>> (left hand side menu)SSH and GPG keys>>New SSH Key>> give name (DesktopKey) and add the copied value in the TEXT FIELD>> Click on ADD SSH Key>> it will ask you for password>> ENTER your GITHUB password>> and your key is successfully added.
3. Go to GIT bash, enter

“ssh -T [git@github.com](mailto:git@github.com)”

The authenticity of host 'github.com (13.234.176.102)' can't be established.

RSA key fingerprint is SHA256:nThbg6kXUpJWGl7E1IGOCspRomTxdCARLviKw6E5SY8.

Are you sure you want to continue connecting (yes/no/[fingerprint])? Yes

Warning: Permanently added 'github.com,13.234.176.102' (RSA) to the list of known hosts.

Hi Rincy-Cheriyan! You've successfully authenticated, but GitHub does not provide shell access.

1. Now we can connect with github using SSH
2. If you have added any password, it will ask you for password, while connecting to GITHUB

**Searching on GITHUB**

1. Global search-
2. We can define scope of search like check in repo, or Issues
3. Adv Search Page- created page, Stars,Popularity
4. Search syntax

Like: eclipse stars:> 500

Eclipse stars: 10..50

Dotnet Not “Hello World”

**Working with GITHUB Locally**

1. Create a clone of github repo, locally (We have clone command, it will create a local copy in ur git).
2. Go to github.com
3. Your repository which you want to clone
4. It will ask you an option of SSH/HTTPS
5. Copy SSH link
6. Got to git bash
7. Add command **git clone** copiedURL
8. You will see your file being copied in your directory
9. Now using cd, shift to your directory
10. To find the files in your repo, use command ls –la
11. Check the status using “git status”

On branch master

Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean

1. Add some new files to your copied(cloned directory)
2. Again check the status
3. You will see new files being added there
4. To add it to “Staged level” use command git add .(DOT)
5. Check the status again
6. git commit -m "local commit"
7. again check the status

On branch master

Your branch is ahead of 'origin/master' by 1 commit.

(use "git push" to publish your local commits)

nothing to commit, working tree clean

1. **git push origin master**
2. again check the status

On branch master

Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean

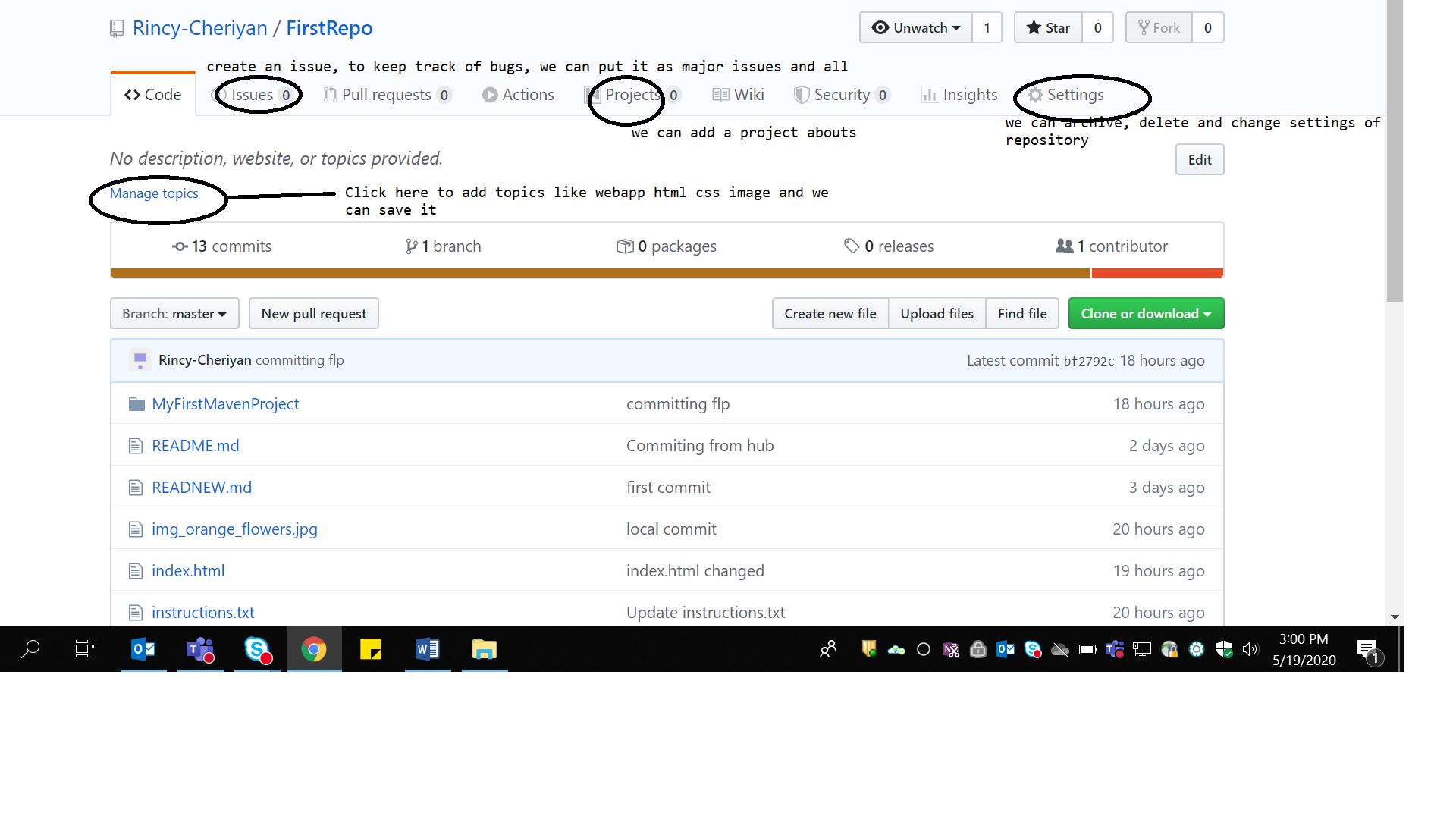
1. go to GIT HUB and refresh

**Adding and Editing data on GITHUB file directly**

1. go to github.com
2. go to your repository and edit the file
3. commit the changes by clicking on the button
4. go to the repository and you will see updated file
5. now add a new file like instructions.txt
6. add some instructions there and commit it
7. now your local repo, is lagging behind remote repo
8. so we will use fetch( to get changes in remote to local d/b)
9. then using Pull will merge the changes into local files
10. go to git bash
11. git fetch
12. git pull (it will show you changes)🡪fetch+merge (it will merge the changes to existing file + add new files as well)
13. check status
14. NOW if we are changing both at remote and local repo simultaneously
15. Change file in your github.com like instructions.txt🡪 add something there
16. Go to git bash
17. Notepad++ index.html
18. It will open the file, edit it and save it
19. git add . ( to add it to staged)
20. git commit -m "index.html changed"
21. git push origin master ---🡪 git will reject the changes and tell you to fetch, as your remote repo has got changed.
22. So use “git pull”-🡪 this will get the changes in remote repository to local one (**git pull origin master** will update all files)
23. To change the remote, as per change in local use the command **git push origin master**

**Knowing More About Repository adding features to Repo**

You can add topics to repo, which will help to search the repo later on. We can also add issues, specifying bugs and all. We can also see insights of a repository



**Archive and Delete Repository**

Go to settings, scroll down to archive or delete a rep.

**To Push Your project to GITHUB**

**Using command prompt**

**Adding project to GITHUB**

1. Put a project in the local repository
2. Git add .
3. Git commit –m “commiting to remote”
4. Check status

Getting project from gitHub to eclipse

1. Create a workspace>> windows>>show view>>Git>>Git repository
2. Copy the https:// path of github
3. Add it in eclipse path
4. And import the project in the workspace

Make changes locally, save it right click>>Teams>>Commit

Make changes in hub and using pull command get it in eclipse.

**ADDING TEAM TO GIT HUB**

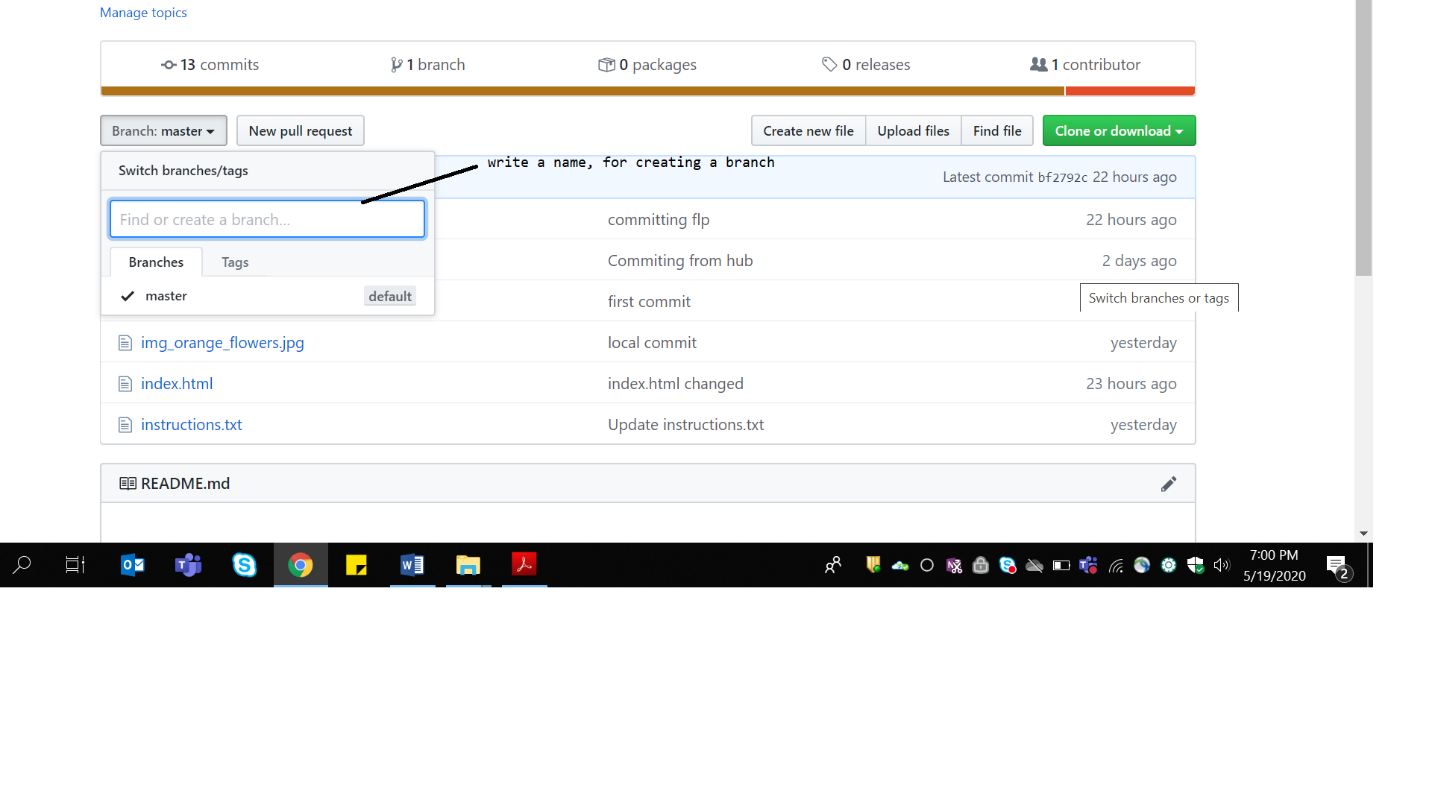
1. We can have collaborators (are fixed group of pple, development team who have more access to project, like they can change settings change code and so on) and contributors(can be everyone else, they can propose change which can be reviewed, but they cannot commit changes to project)
2. DEMO for adding Collaberators
3. When we create a file in our repository, only we are allowed to make changes to it (EDIT IT)
4. If anyone else tries to open it, they can only see it, can’t make any changes
5. GOTO Settings>>Left hand side >>Manage access(click)>>click on Invite collaborator>> search by name>> click “NAME TO REPOSITORY” on add to >> We can copy invite link>> We will also get this link on mail>> if other person open that link, they will see invite>> collaborator can ACCEPT the invite>> asa colleberator Accept the request, they can push their code into our rep.
6. Go back to original account and check if the changes are being made of not.
7. Insights give info about contributors.

**Branching Merging and Pulling request from GitHub**

Instead of adding entire code on a bramch, we are gng to add it in the side track. If the changes in side track goes on well, we can merge it with Master branch.

* Branch is a pointer to a commit

**Creating a branch in GIT HUB**

****

After giving a name to the branch, hit ENTER to create a branch.

You will receive a message

This branch is even with master.

You can also pull from master, we can also compare the two branches.

Now goto sample-branch, add changes to the file and commit it. So once you click to your repository, you will see a changed branch. But Master is still the same.

**Commands for Branching in GIT**

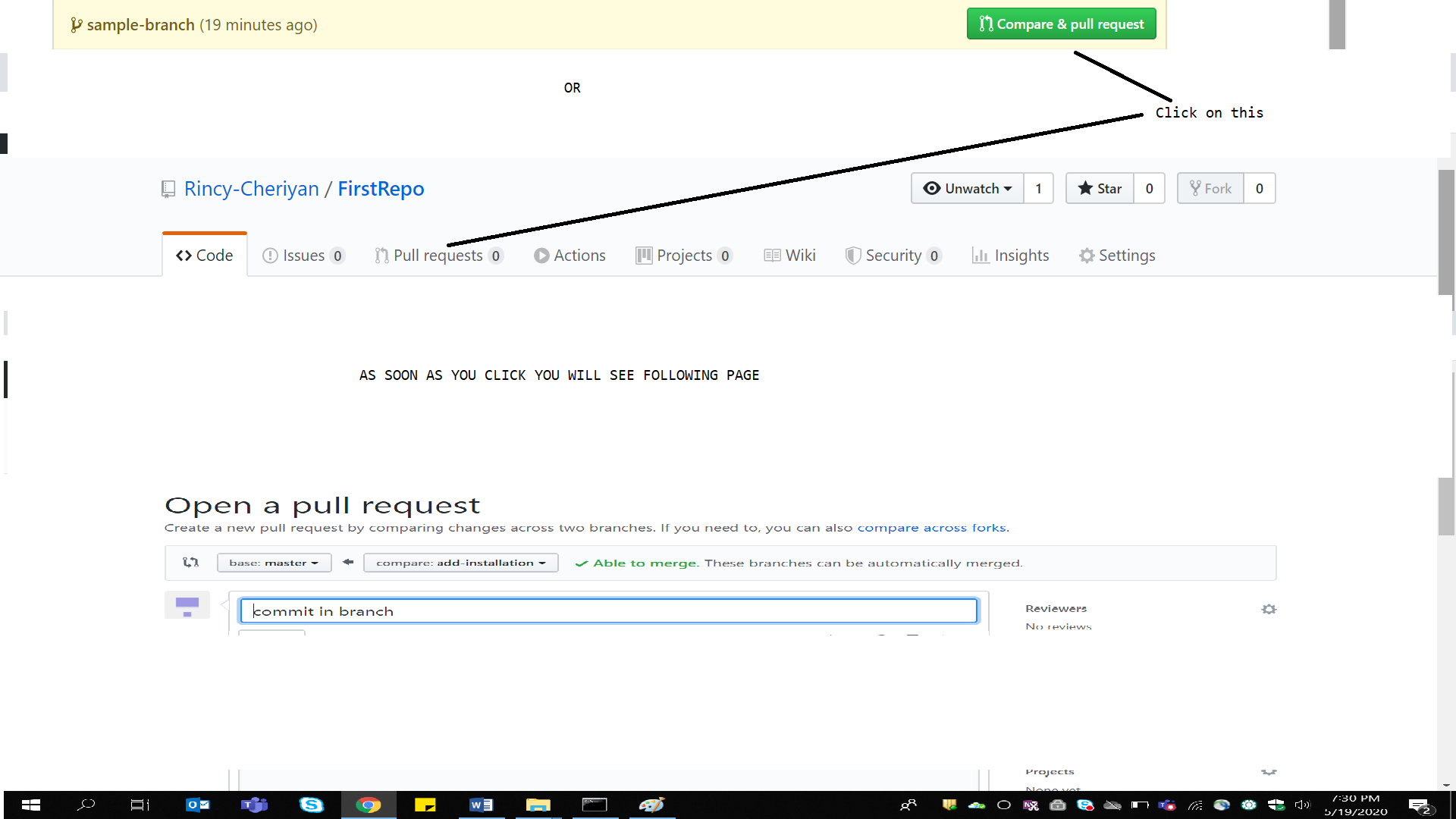
1. Git branch branchName
2. To switch to a branch git checkout branchName

**git checkout -b "add-installation" // this will create a branch and let us in that branch**

1. To add a file to that branch
2. **Notepad++ index.html**
3. Then **use git add .**
4. **Git commit –m “commit in branch”**
5. To push the branch to GIT hub we can use **git push –u origin branchName**
6. **git push -u origin add-installation**
7. **check in git hub to check your changes**

**To merge your branch with your Master**

1. asa you make any changes in the branch, it will be visible on the screen as shown below
2. while you click on **Compare and Pull,** you will see the changes(as you scroll down)



1. as soon as you make a pull request (You can also add a message stating for review by other team members)
2. you will be able to see CONVERSATIONS, 1 COMMITS, 1 FILE CHANGED
3. click on file changed to check the file changed, a reviewer can also comment on the changes made, on line by line basis
4. Go to conversation and click on **MERGE PULL REQUEST**
5. And click on **CONFIRM MERGE**
6. **After merging it with master branch, we can also delete the branch**
7. You can also go to **PULL REQUESTS** to check the status of commits

If in case there are any conflicts in case of merge, it will take you to editor. You can resolve the conflicts there and Click on **Mark as Resolved** and then **Commit Merge**

**Working with FORK**

1. Fork means creating a copy of a repo
2. Does not impact original repo
3. Changes can be merged later
4. Difference between a FORK and branch is, FORK works on different repo whereas Branch works on the same repo.
5. Example: if we are available already with a project at ASP.net site, now I want to create a similar project but with some difference. I can fork the ASP.nEt repository into mine and I can make changes in my project, but original repo is not much bothered.