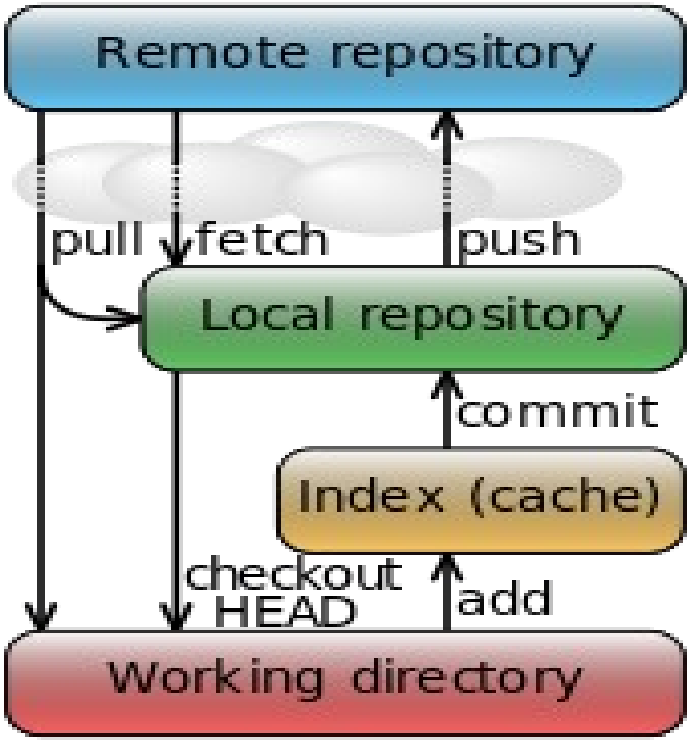
Git && GitHub

Git and GitHub allows groups of people to work on the same documents (often code) at the same time

Git is under the General Public License (GNU) version 2. Meaning its free to use. So what is important to understand is that GitHub does not own Git, they are just using the Git technology.



For Debian and their kinds like Ubuntu

- sudo apt-get install git-all

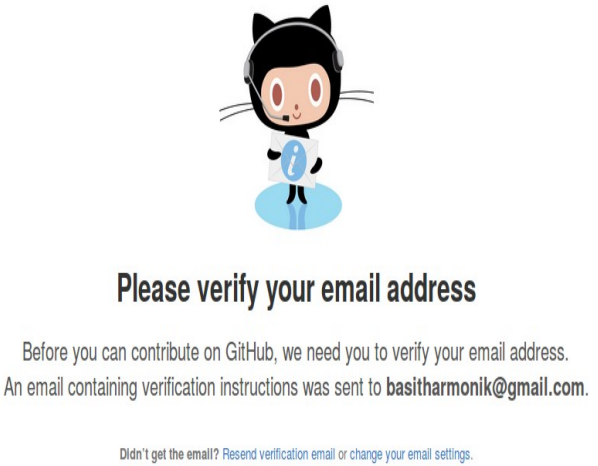
For Red Hat and their kinds like Fedora

- sudo yum install git-all

Creating git-hub

Join git-hub

- https://github.com/join?source=header-home



set your account's default identity for git

- git config --global user.email "you@example.com"
- git config --global user.name "Your Name"

Connecting to your git-hub with ssh

- ssh-keygen -t rsa -C "you@example.com"

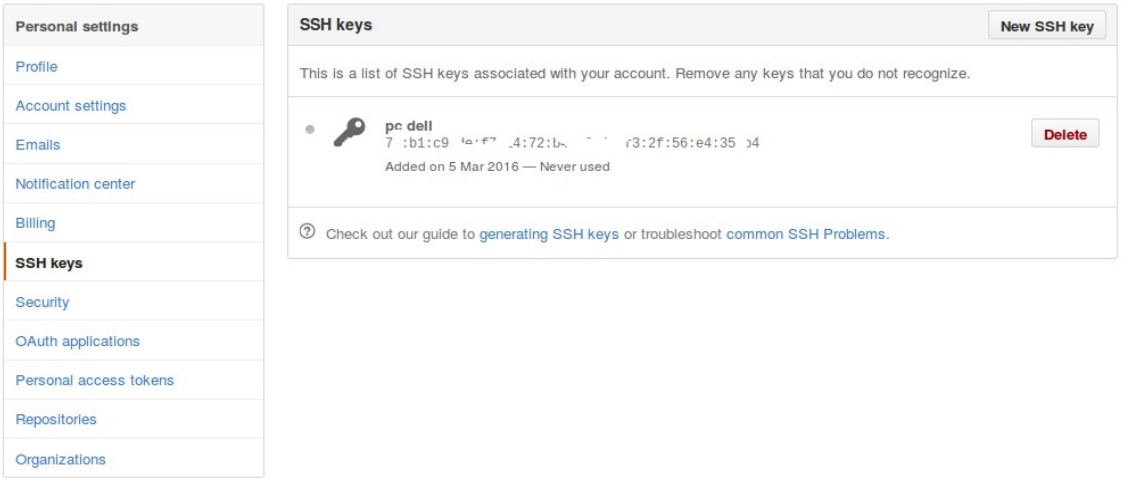
Output is gonna be like this

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/basitharmonik/.ssh/id_rsa):
Created directory '/home/basitharmonik/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/basitharmonik/.ssh/id_rsa.
Your public key has been saved in /home/basitharmonik/.ssh/id_rsa.pub.
The key fingerprint is:
                                                    tharmonik@gmail.com
The key's randomart image is:
+--[ RSA 2048]----+
|                . |
|                .|
|               :+|
|               -B|
|        ,    . *=|
|             ' .o|
|           .   o ^|
|           o     |
|            . ..o|
```

Creating git-hub

Copy your new public ssh key and paste it into git-hub's ssh keys section which is located in your personal setting

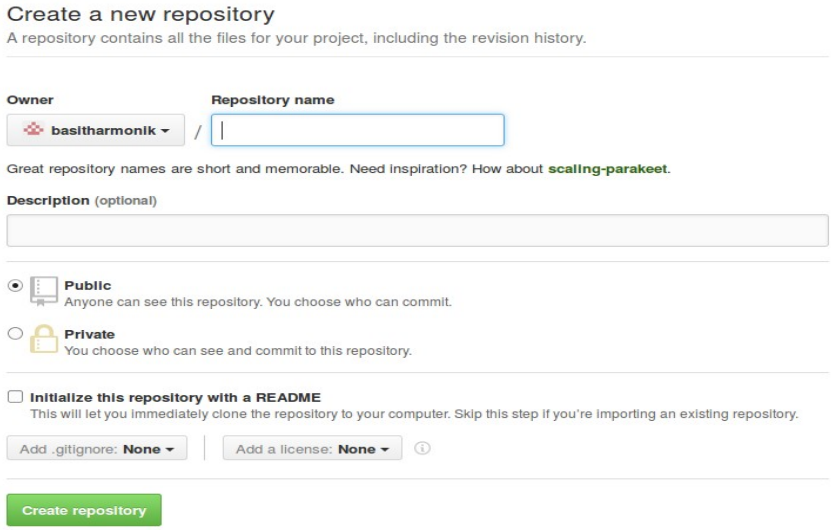
- vim /home/basitharmonik/.ssh/id_rsa.pub
- https://github.com/settings/ssh
- ssh -T git@github.com

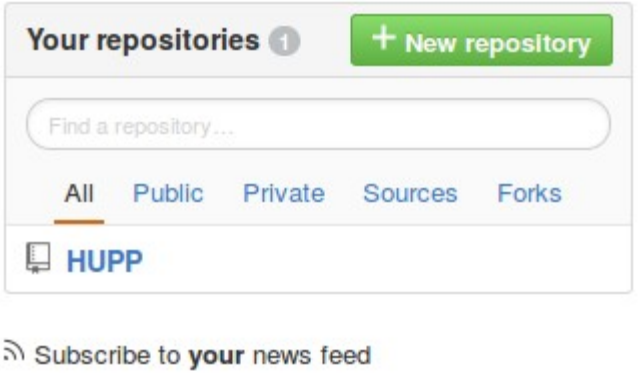



```
vim ../../.ssh/id_rsa.pub
basitharmonik@basitharmonik:~/Desktop/Hupp$ ssh -T git@github.com
The authenticity of host 'github.com (192.30.252.129)' can't be established.
RSA key fingerprint is 16:27:ac:a5:76:28:2d:36:63:1b:56:4d:eb:df:a6:48.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'github.com,192.30.252.129' (RSA) to the list of known hosts.
Hi basitharmonik! You've successfully authenticated, but GitHub does not provide shell access.
basitharmonik@basitharmonik:~/Desktop/Hupp$ ls
```

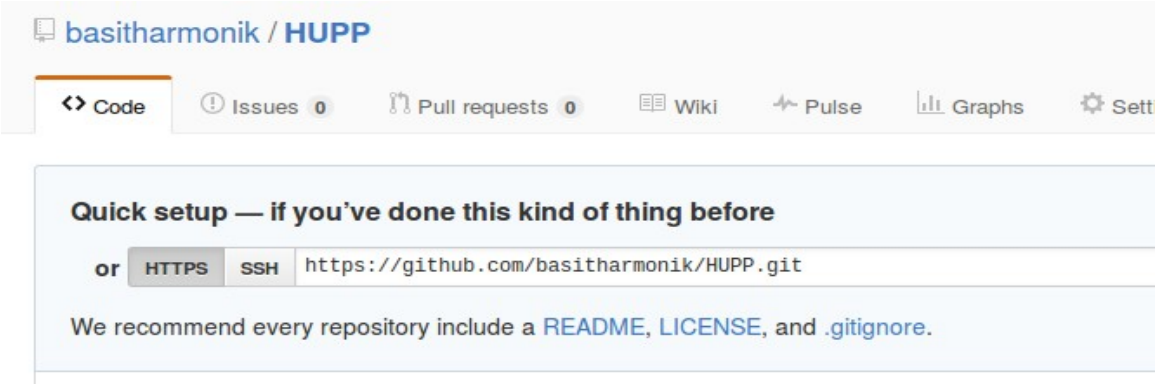
Repository Request

https://github.com/





https://github.com/new



Git Commands

- git init (now git tracking our file, we can see that with ll command)
- git add . (add command adding files into stage level, "." means that everything)
- git add text1 (we've just added one file to stage level)
- git commit -m "first commit , this part using for comment "
- git commit -m "first text added" (commit adding staged files into local repo.)

If wanna learn what is the difference between committed or stageded version and our version which is located in working directory , we can use these two

- git diff or git diff --staged (second one useful)

Deleting file from repo

- git rm text2

Deleting from your directory than updating your status

- git commit -m (text2 deleted)

Renaming is can menage with mv command just like linux

- git mv text1 text2

UNdo

- git checkout or specific one git checkout HEAD

if we modified file and than added staging status after that we can make undo from staging status with:

- git reset HEAD text1

Git Commands

- git remote add origin https://github.com/basitharmonik/HUPP.git
- git push origin master (uploading all files into github)

```
basitharmonik@basitharmonik:~/Desktop/Hupp$ git remote add origin https://github.com/basitharmonik/HUPP.git
basitharmonik@basitharmonik:~/Desktop/Hupp$ git push origin master
Username for 'https://github.com': basitharmonik
Password for 'https://basitharmonik@github.com':
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (5/5), 420 bytes | 0 bytes/s, done.
Total 5 (delta 0), reused 0 (delta 0)
To https://github.com/basitharmonik/HUPP.git
 * [new branch]      master -> master
```

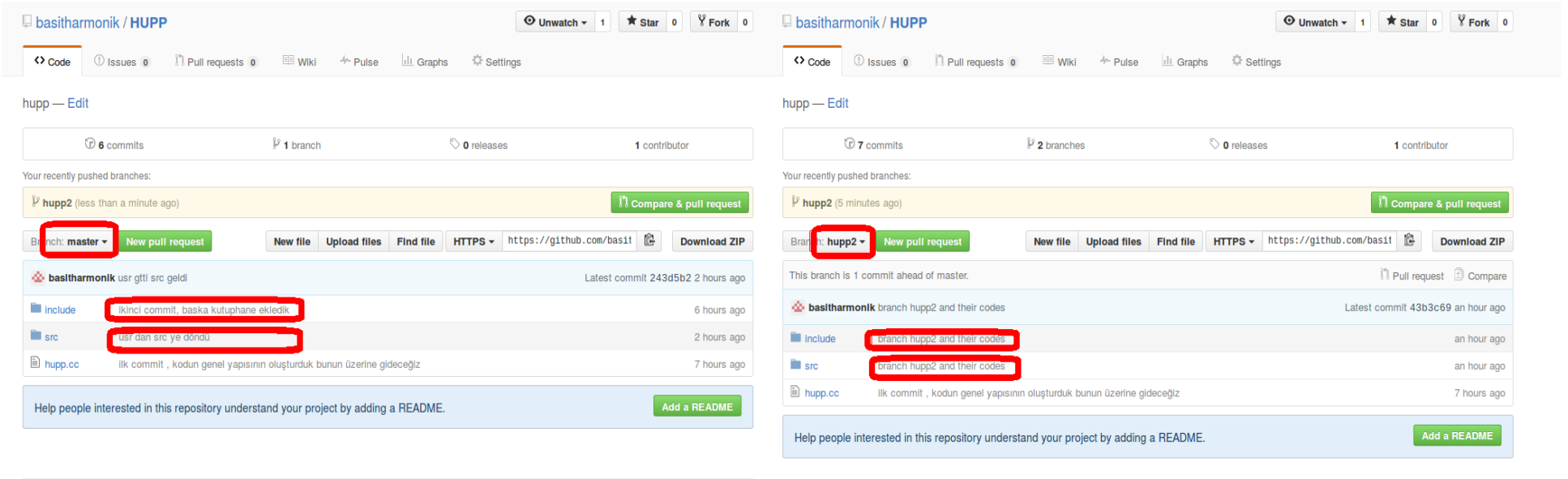
Lets assume that we have another study group and the want to add new methods

- mkdir Hupp2 && cd Hupp2/
- git init
- git clone https://github.com/basitharmonik/HUPP.git
- cd HUPP/
- git branch output : *master
- git branch hupp2
- git checkout hupp2 output :Switched to branch 'hupp2'
- git branch output : *hupp2 , master (now you are in hupp2 branch)

Git Commands

Let 's assume that we are adding new codes into src and include file

- touch src/hupp2lib.cc include/hupp2lib.hh
- git add .
- git commit -am "branch hupp2 and their codes"
- git push origin hupp2 (please notice that we have just changed master into hupp2)



As you can see now we have two branch and two different codes

Git Commands

Lets merge these two branch
Before merging you may want to sync all your codes with github
Please note that we are going to merge under the master branch so that we can get new classes from hupp2

- git branch — output : hupp2 , *master (now you are in master branch)
- git pull origin hupp2 — (sync command)
- git merge hupp2

```
basitharmonik@basitharmonik:~/Desktop/Hupp$ git branch
  hupp2
* master
basitharmonik@basitharmonik:~/Desktop/Hupp$ ls ./ include/ src/
./:
hupp.cc  include  src

include/:
hupp2lib.hh  hupplib2.hh  hupplib.hh

src/:
hupp2lib.cc  hupplib2.cc  hupplib.cc
```

As you can see now we have same codes. Now we can upload into github

- git push origin master

Git Commands