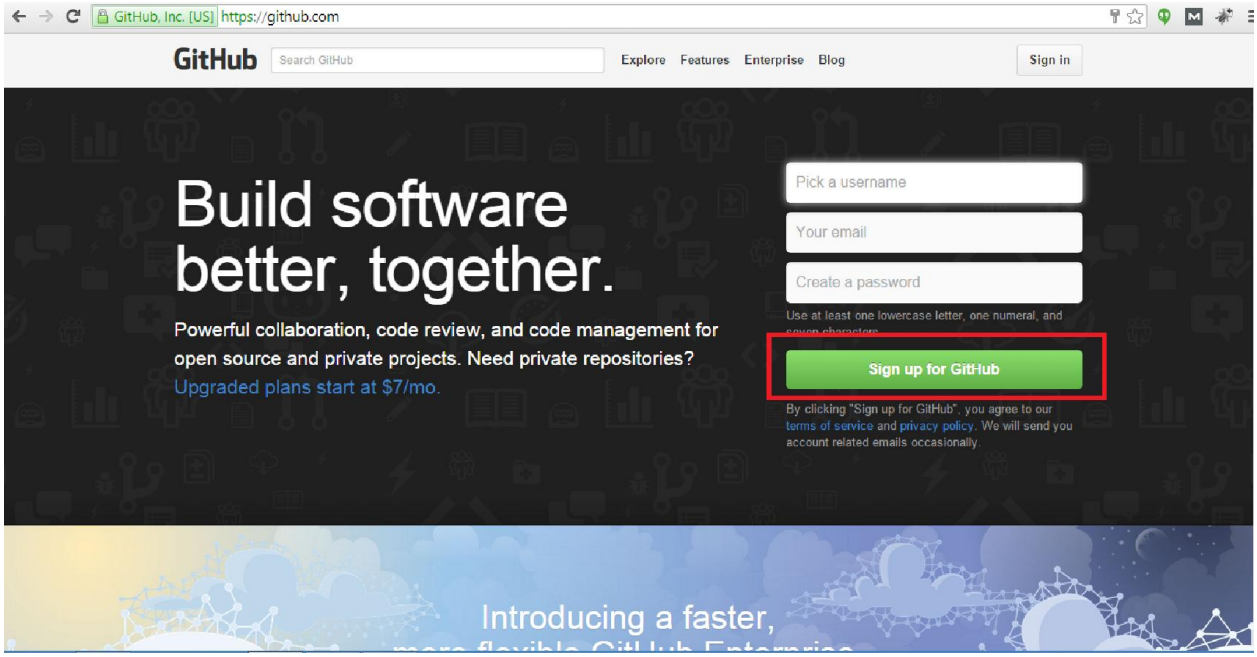
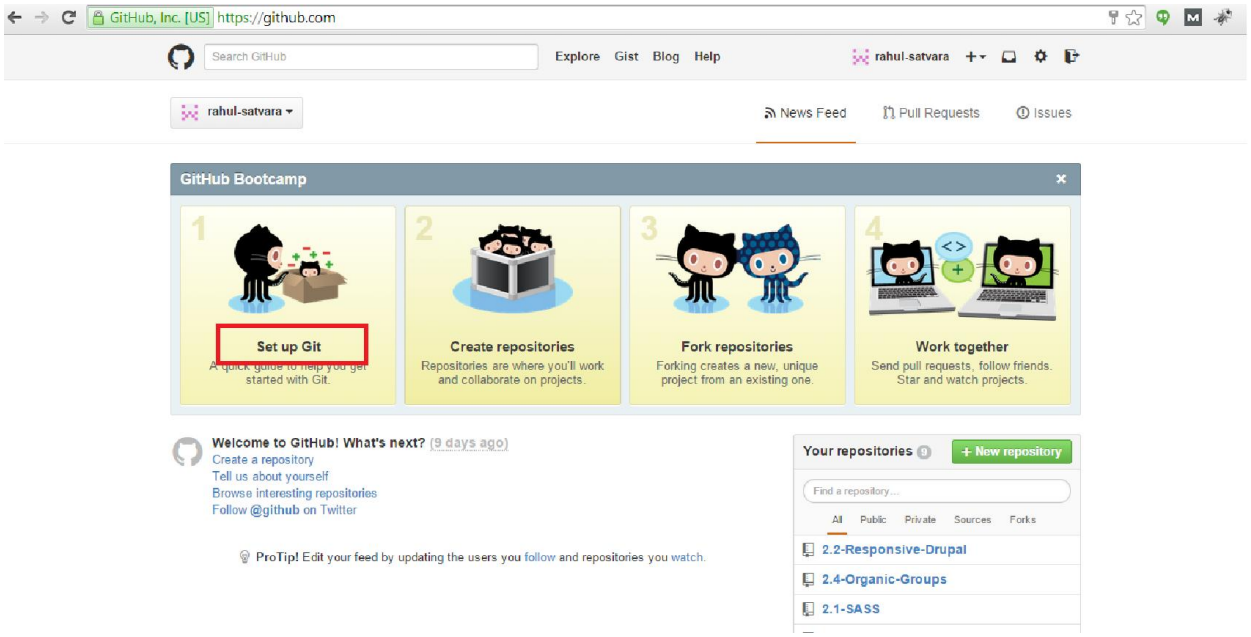
1. First create account in git go to https://github.com/ And Give username, email and password and click on signup.

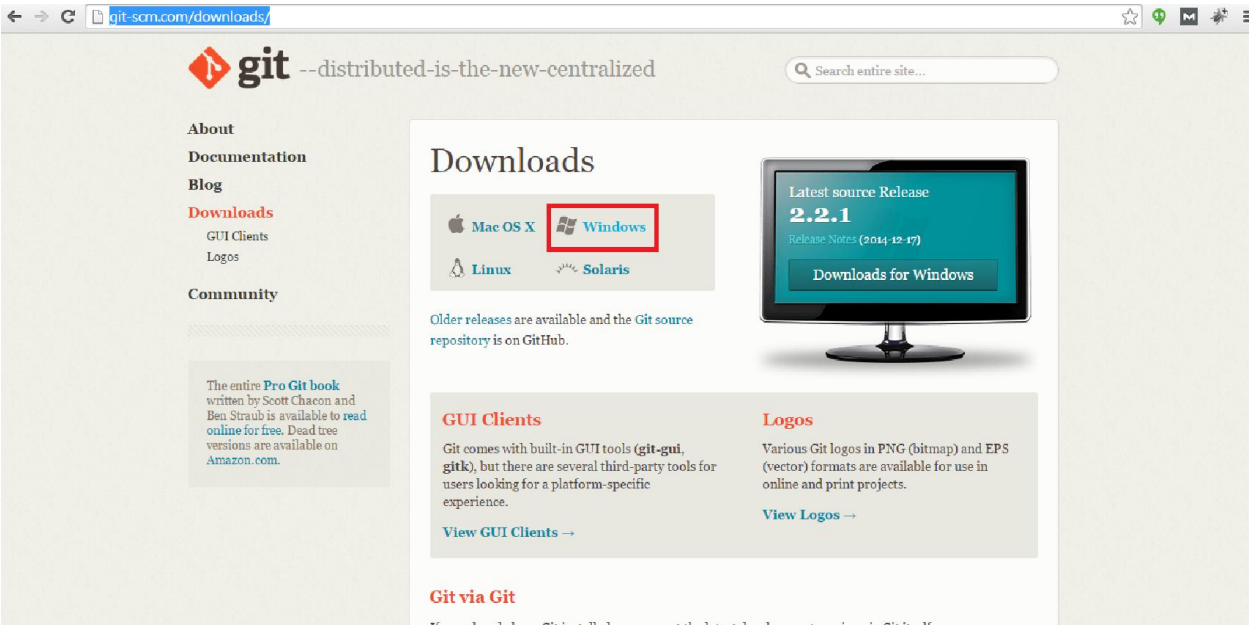


   a. Then sign in into your account.
   b. After sign-in you will get this screen.

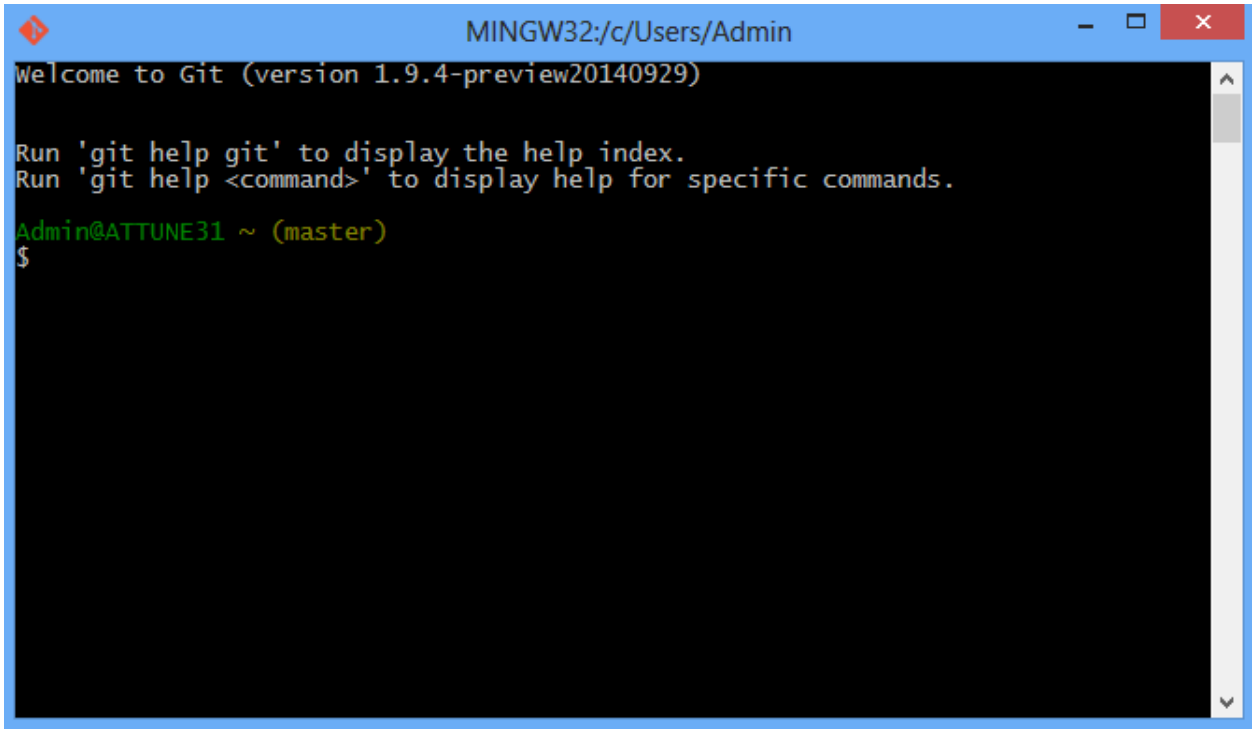


   c. Click on **Set up Git** it will show you some basic setup guide to install git.

2. Second step is to got to this site http://git-scm.com/downloads/ and download git terminal so you can run commands for git.
   You can see deferent version for deferent os. So you can download for mac, windowns or linux etc.
3. I'm working on windows so I downloaded setup for windows version.

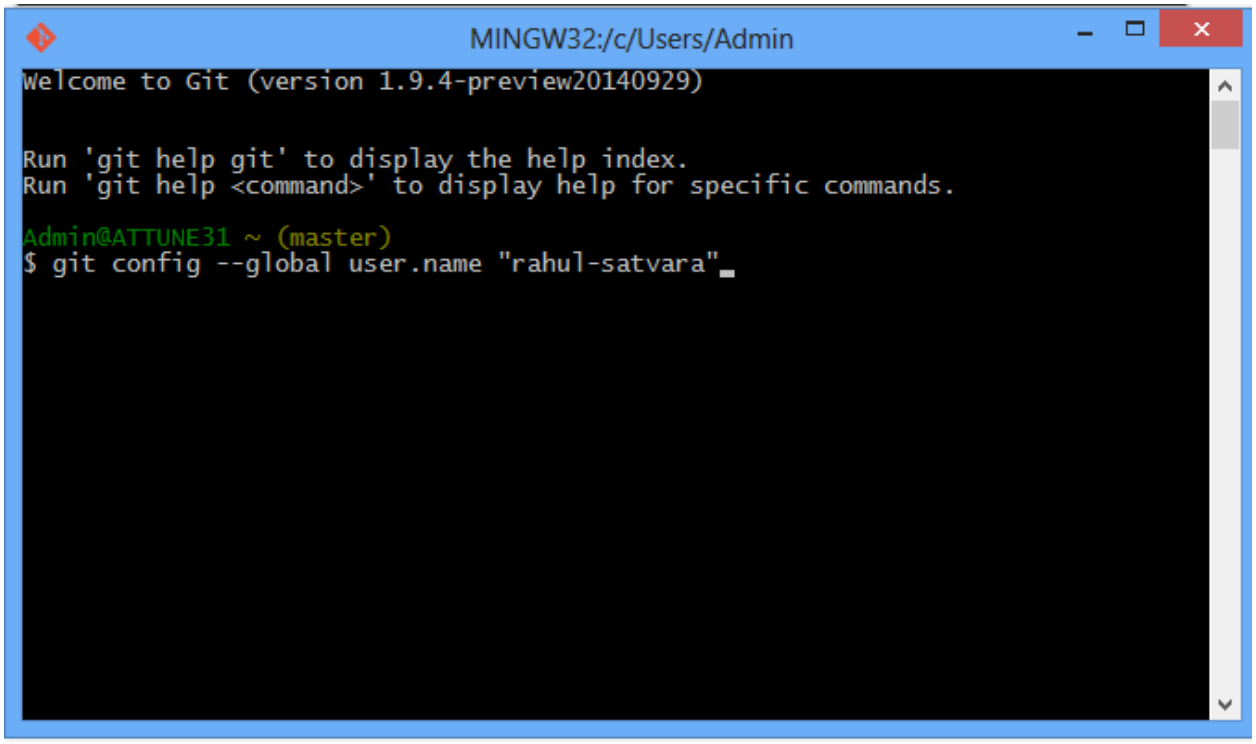


4. After downloading git software setup bobble click on It and install it.
5. After installation On your computer, open the **Git Shell** application.



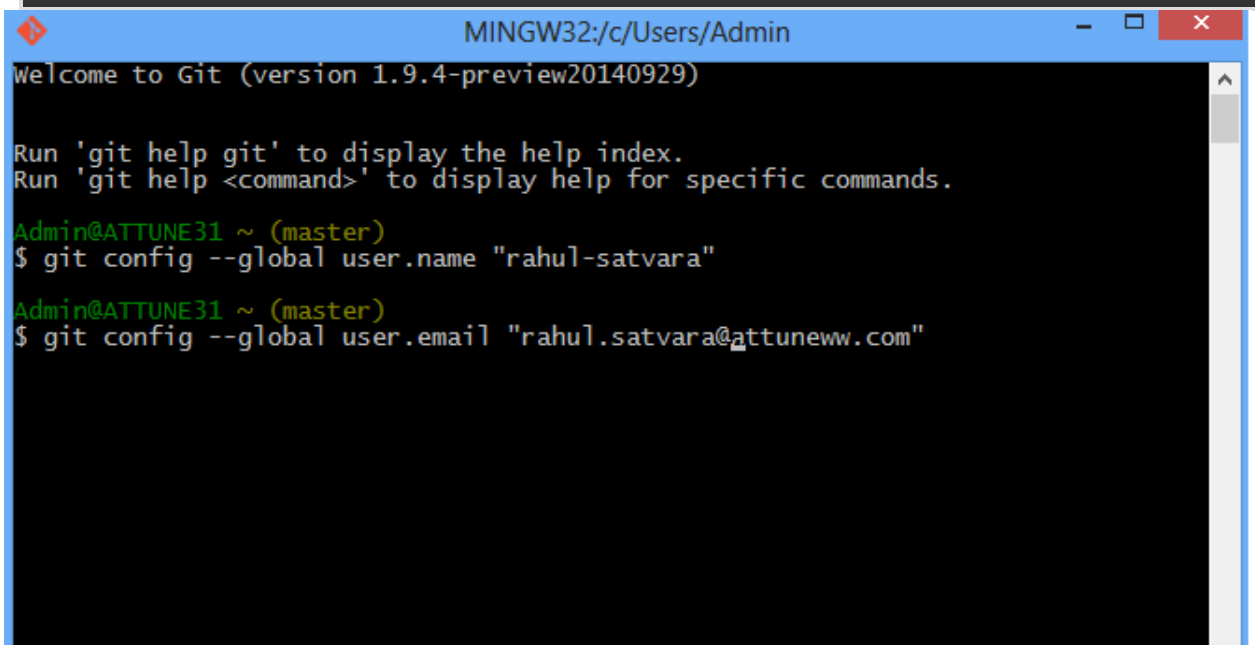
6. Next Setting up git(you can see steps on https://help.github.com/articles/set-up-git/).
    a. Tell Git your *name* so your commits will be properly labeled. Type everything after the `$` here:

```
git config --global user.name "YOUR NAME"
```



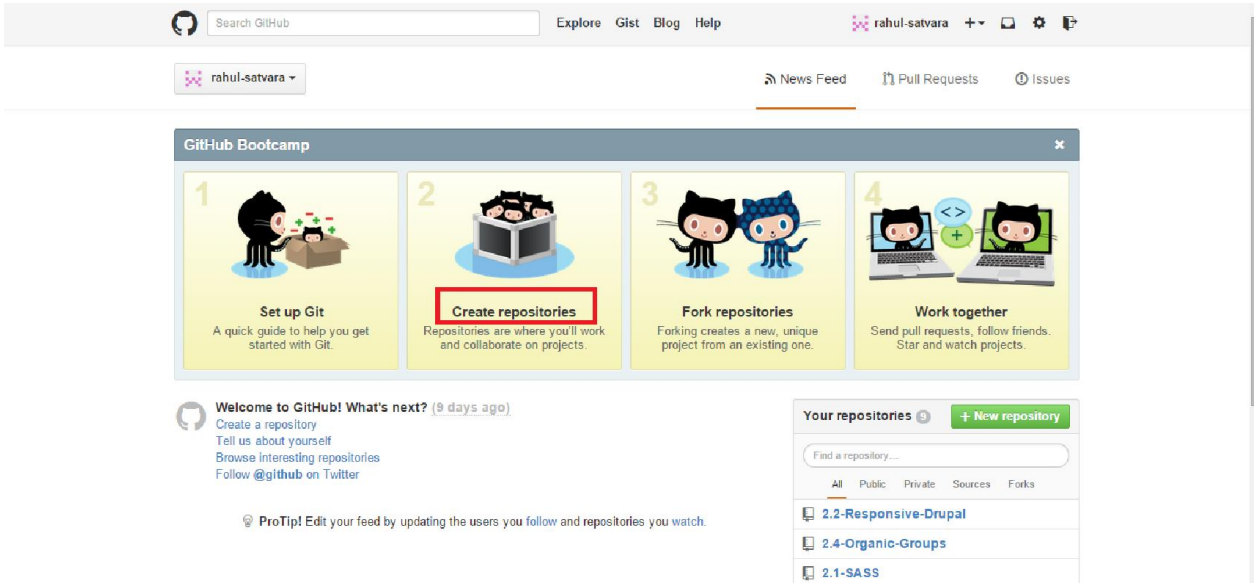
b. Tell Git the *email address* that will be associated with your Git commits. The email you specify should be the same one found in your email settings. To keep your email address hidden, see "Keeping your email address private".

```
git config --global user.email "YOUR EMAIL ADDRESS"
```

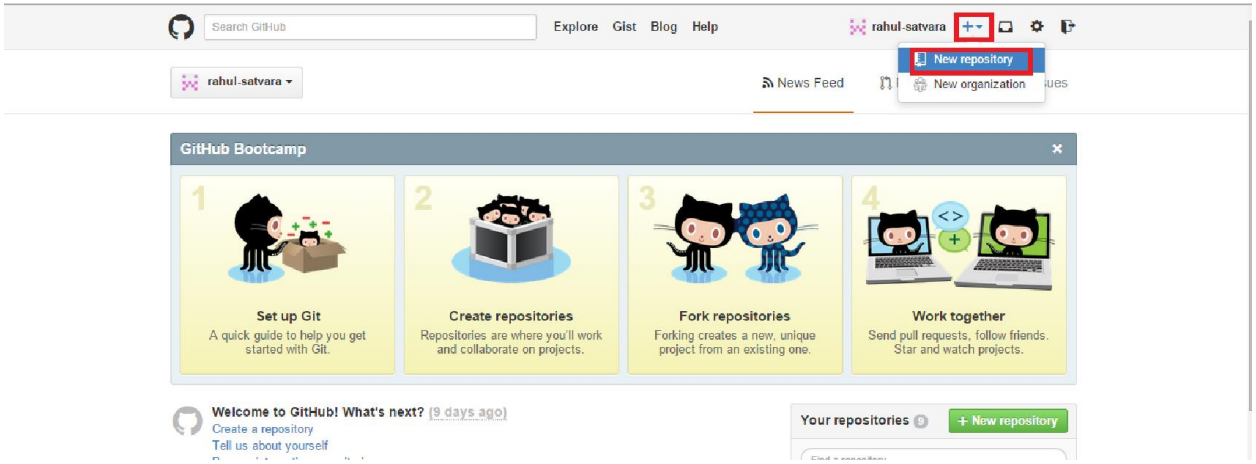


c. Now our git is set up.

7. After setup git we are going to create repository.
   Repository are where your all work collaborate on projects.
   I think I've repository as a sort of folder that is uploaded onto the github website and it contains all the code projects.

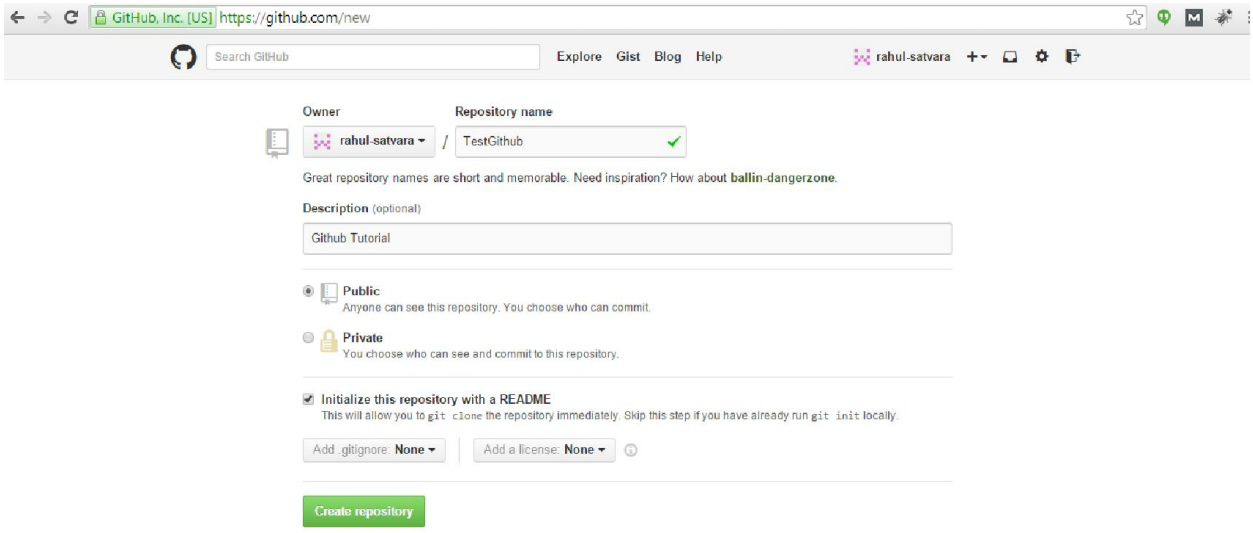


a. In the upper-right corner of any page, click , and then click **New repository**.



b. Enter any name you want to give to your repository I give it **TestGithub**. Also Give Some description to your Repository.
c. You can make repository as **Private** but for that you have to Upgrade account. Here we select **Public.**

d. **Initialize this repository with a README**., README is basically a text file about page.



e. Click on **Create Repository.**

f. After creating repository we have

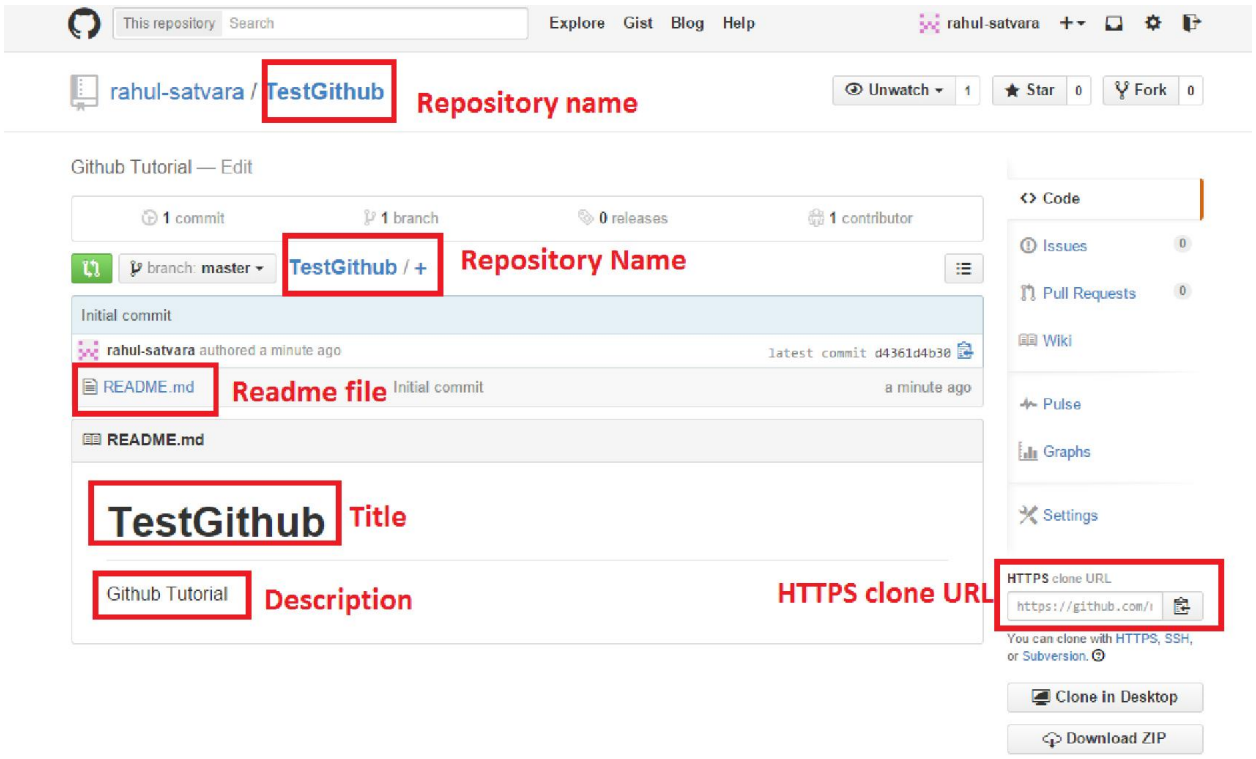
   i. TestGithub Repo.

   ii. Title

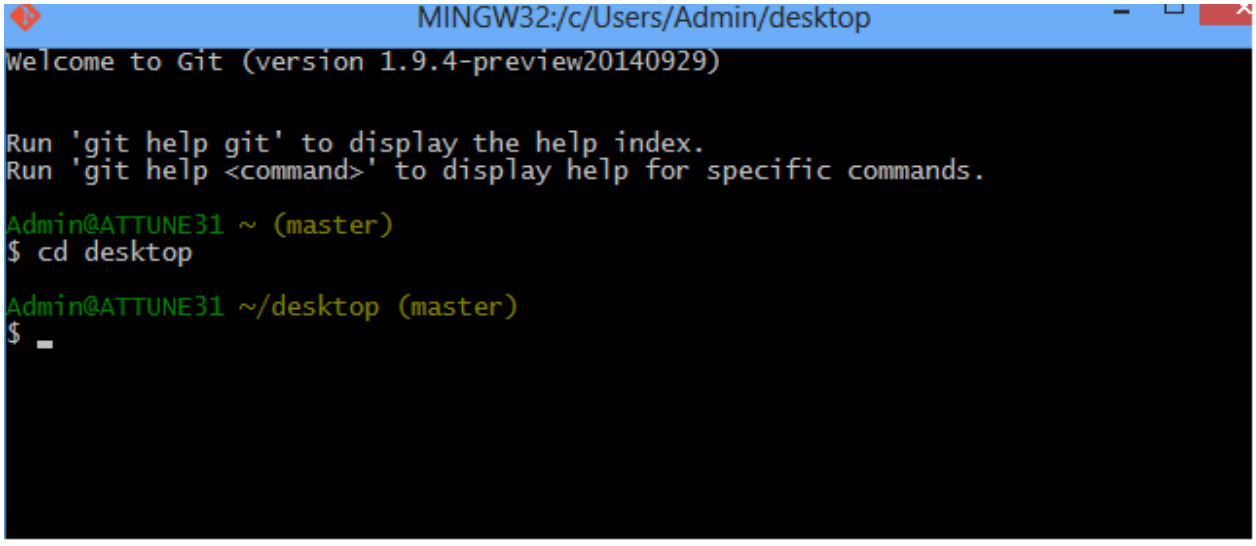
   iii. README file.

   iv. Description

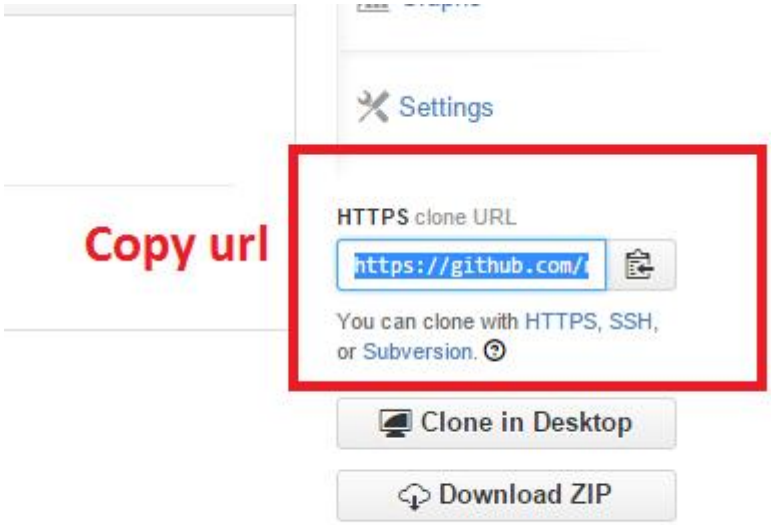
   v. HTTPS clone URL(right hand side) To clone repository.



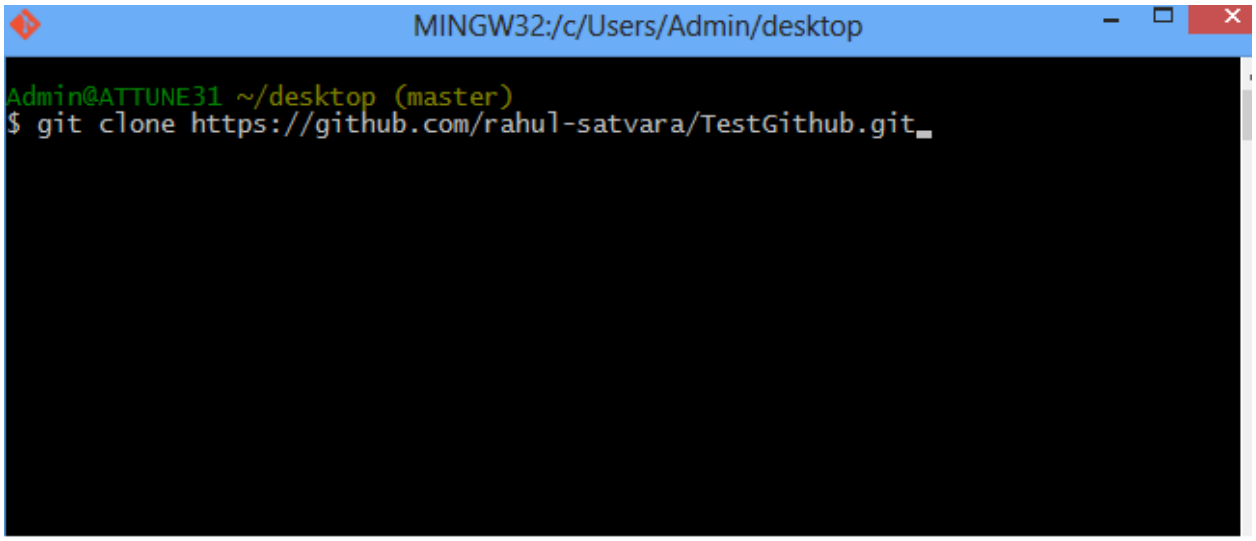
8. After done above steps let's go to git Bash.
9. Go to desktop directory by typing cd desktop.



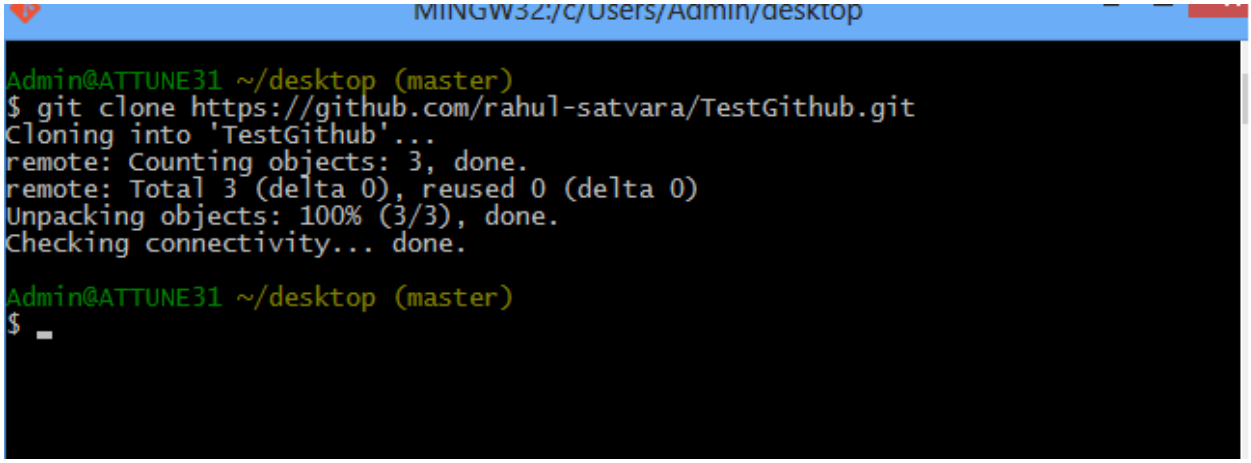
10. Now I want to clone repo that I create in github site.



Paste in gitbash.

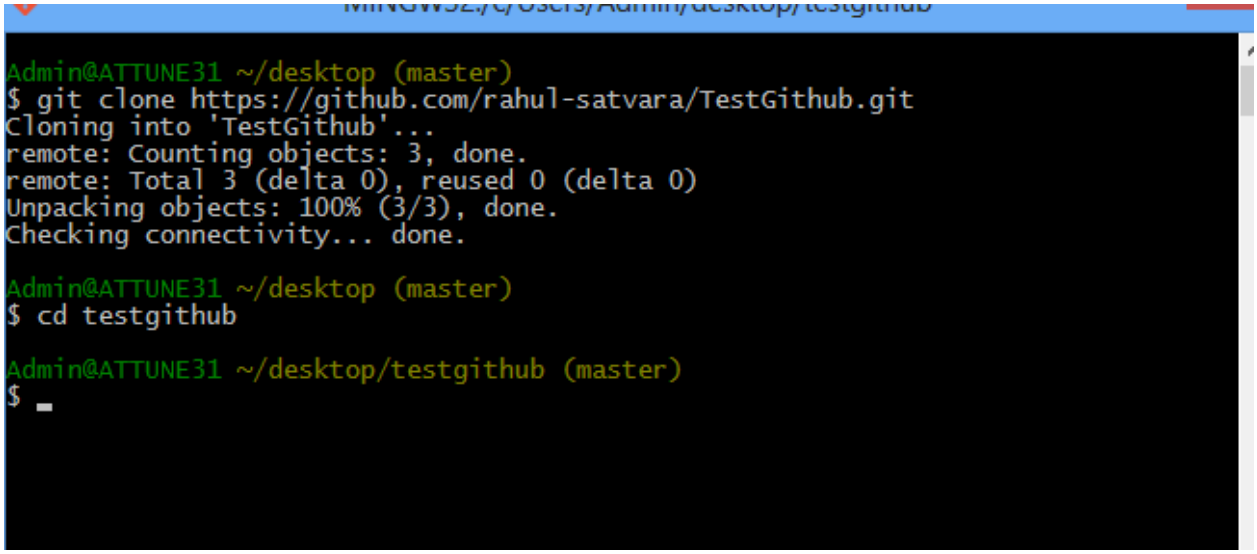


11. After running above command repository(**TestGithub**) is clone in our desktop with all resources. But We have only one file readme.

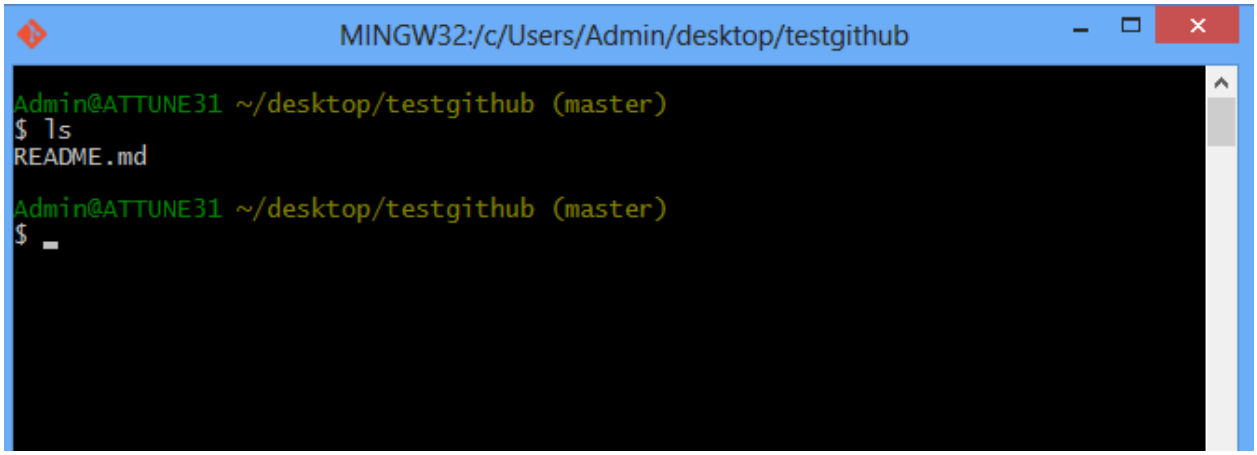


12. Now go into TestGithub Repo Folder.

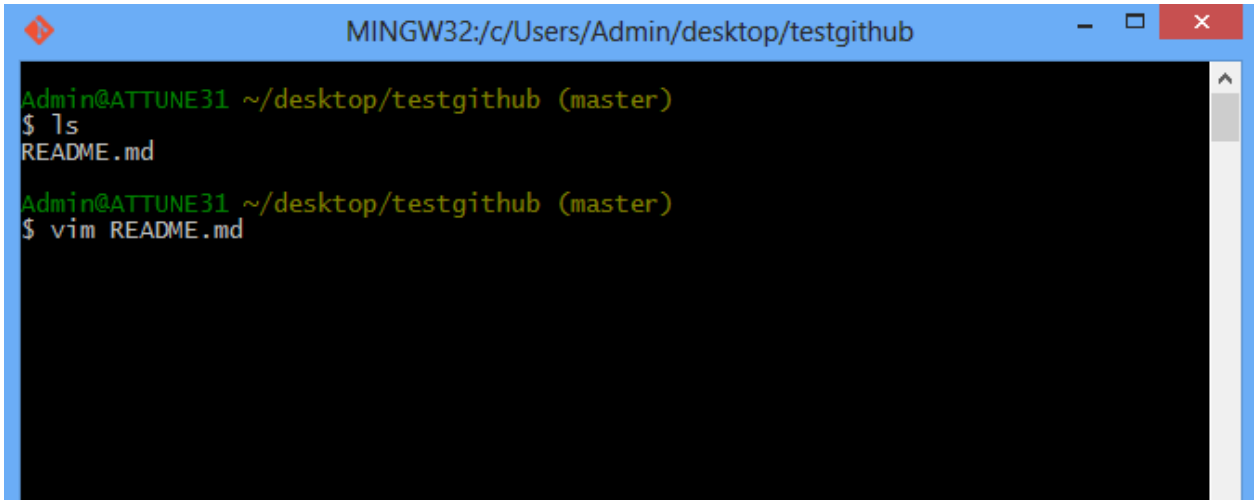
Cd TestGithub



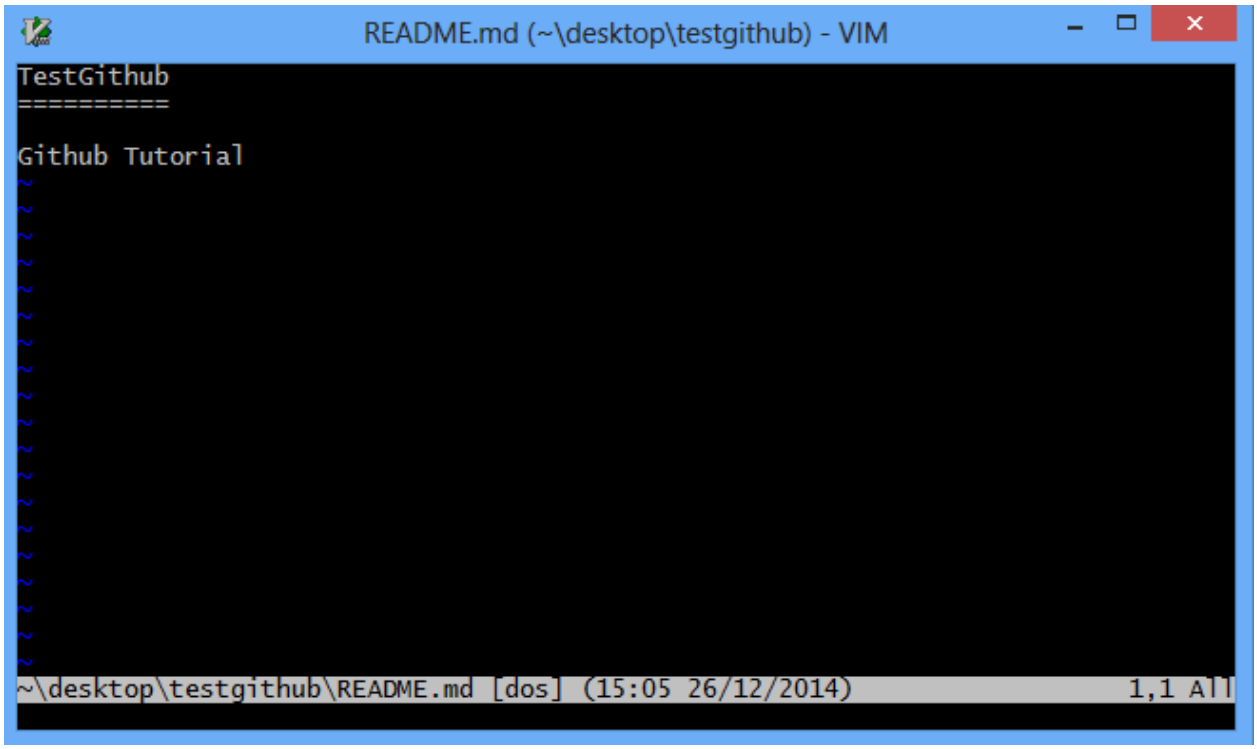
13. So now we are in the testgithub folder. It also a master branch to our current working directory.
    All implementation done here we can create other branches and that emerge into master branch for distributed environment. Where more than one person working on same project. All branchs are merge into master that is final master project.
14. Type ls command. So you can see there is one Readme.md file.



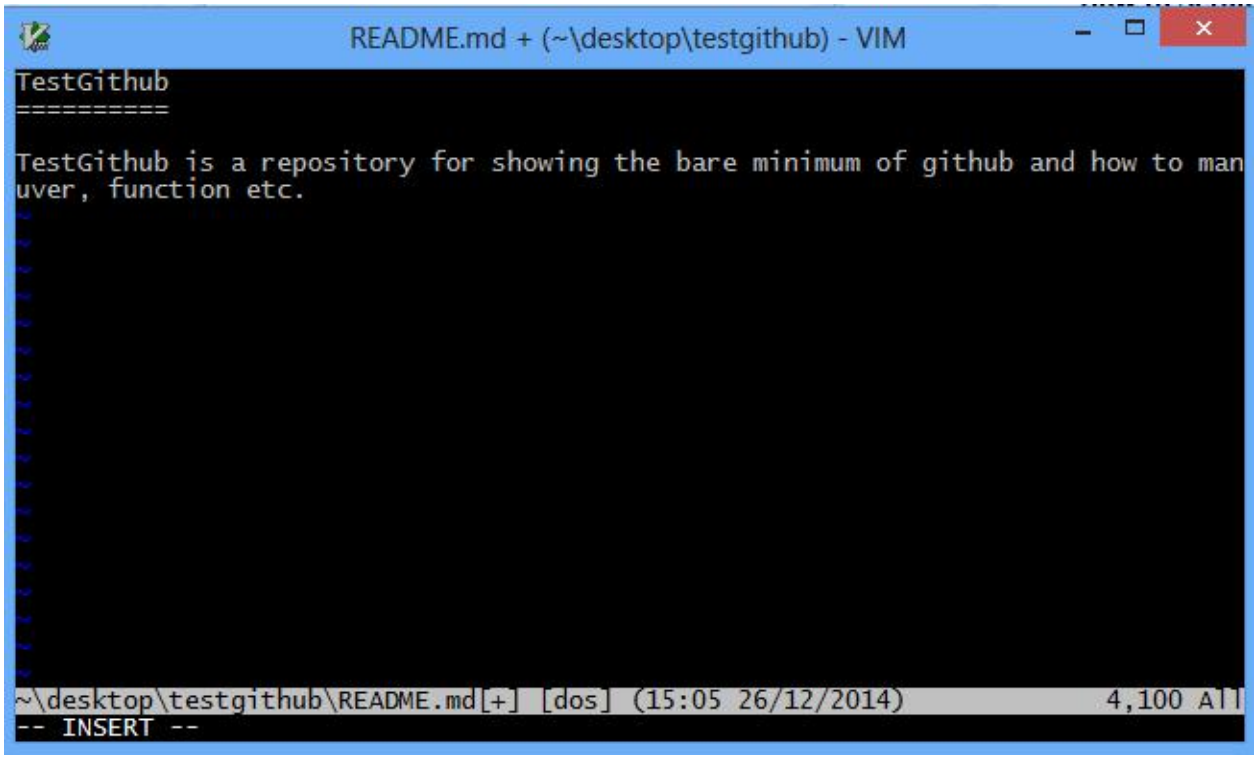
15.Now select editor to edit file.



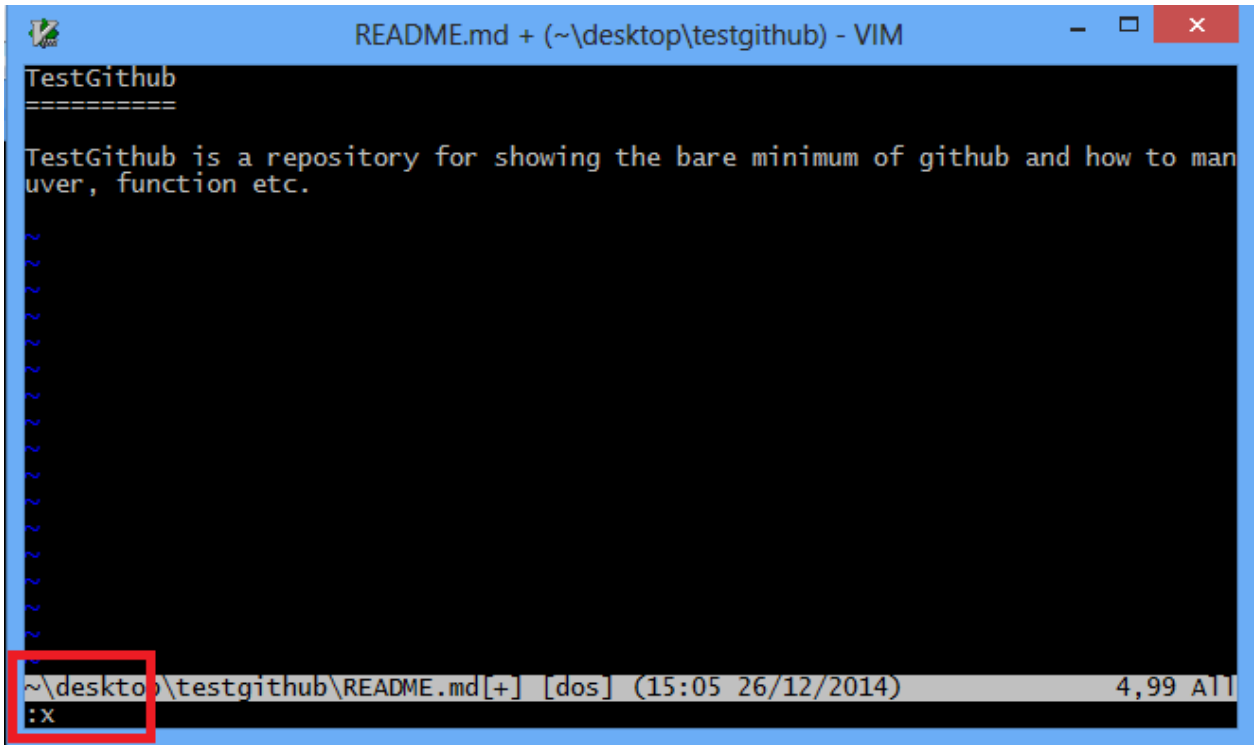
16.Now we are in vim editor so we can edit out file here. You can see there is a title and description showing in editor.



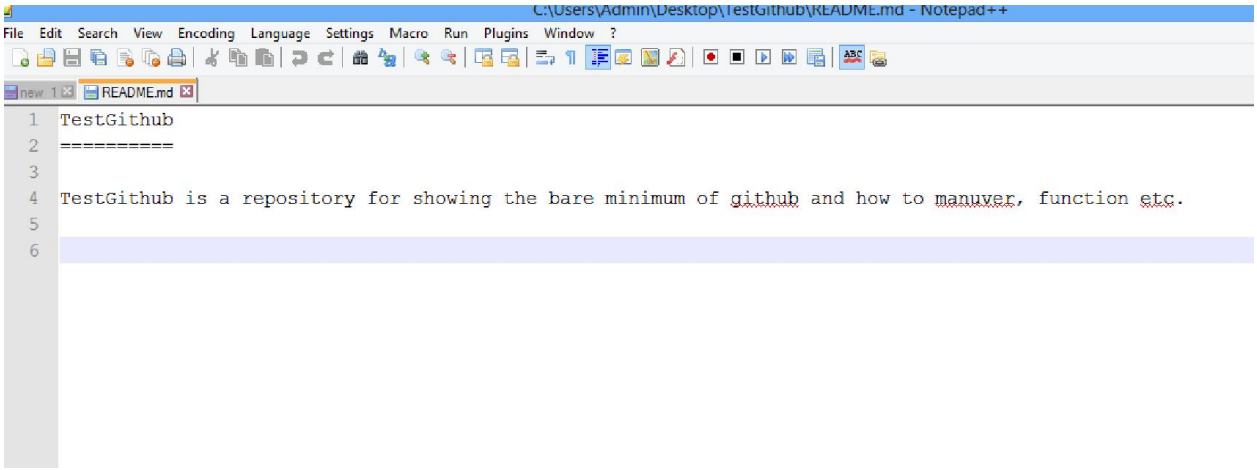
17.So now I'm modifying description.



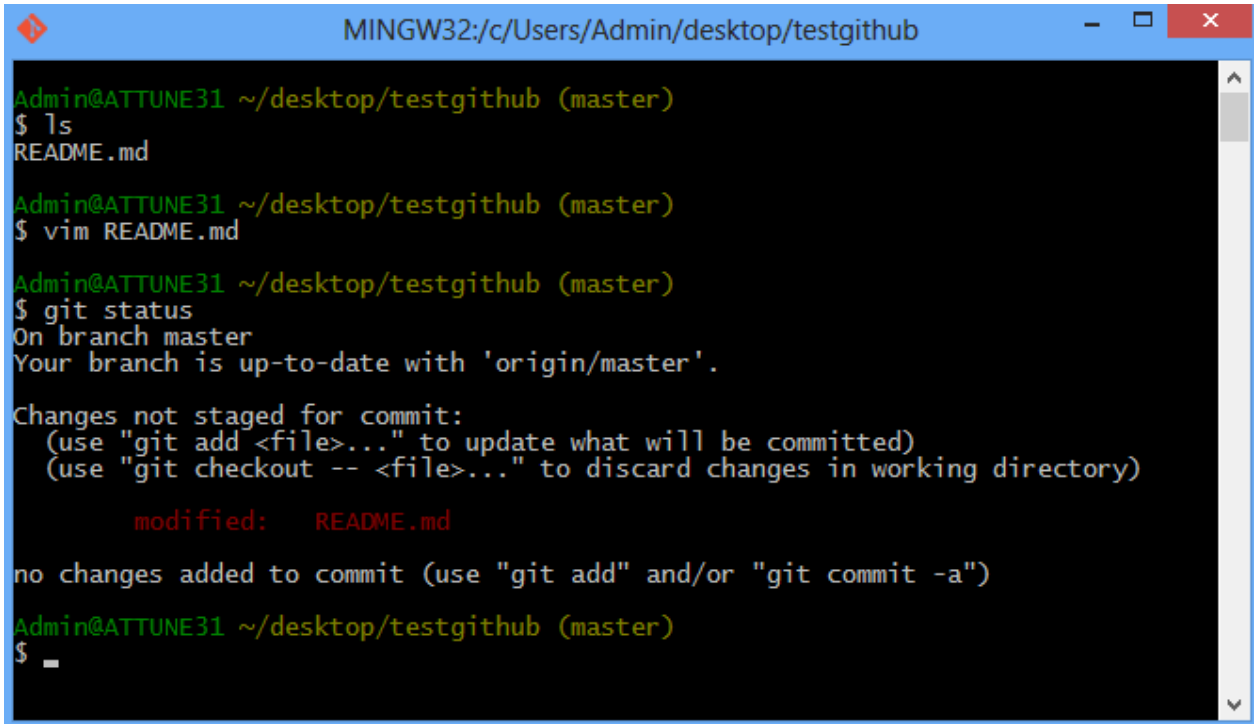
18. Now press Esc button to get out from insert mode. If you want edit then press Insert button.
19. Now type :X to save and exit from editor.



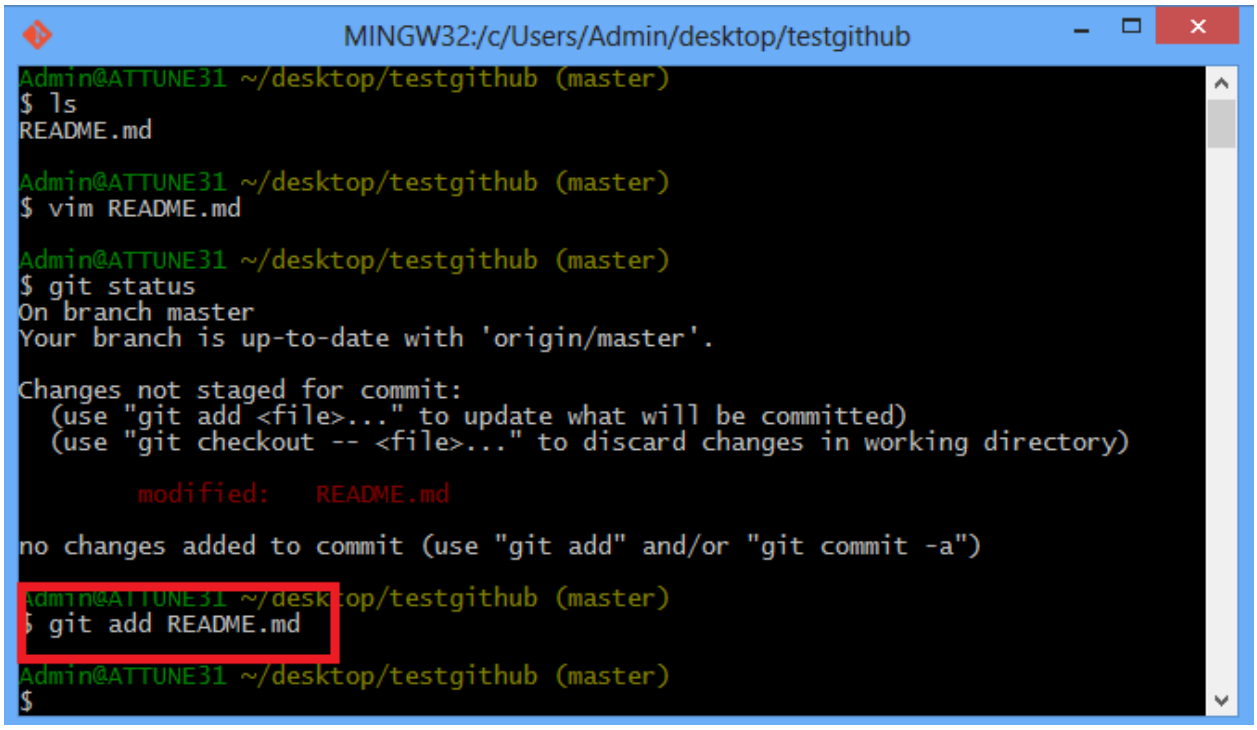
20. Now our README file is modified with description.



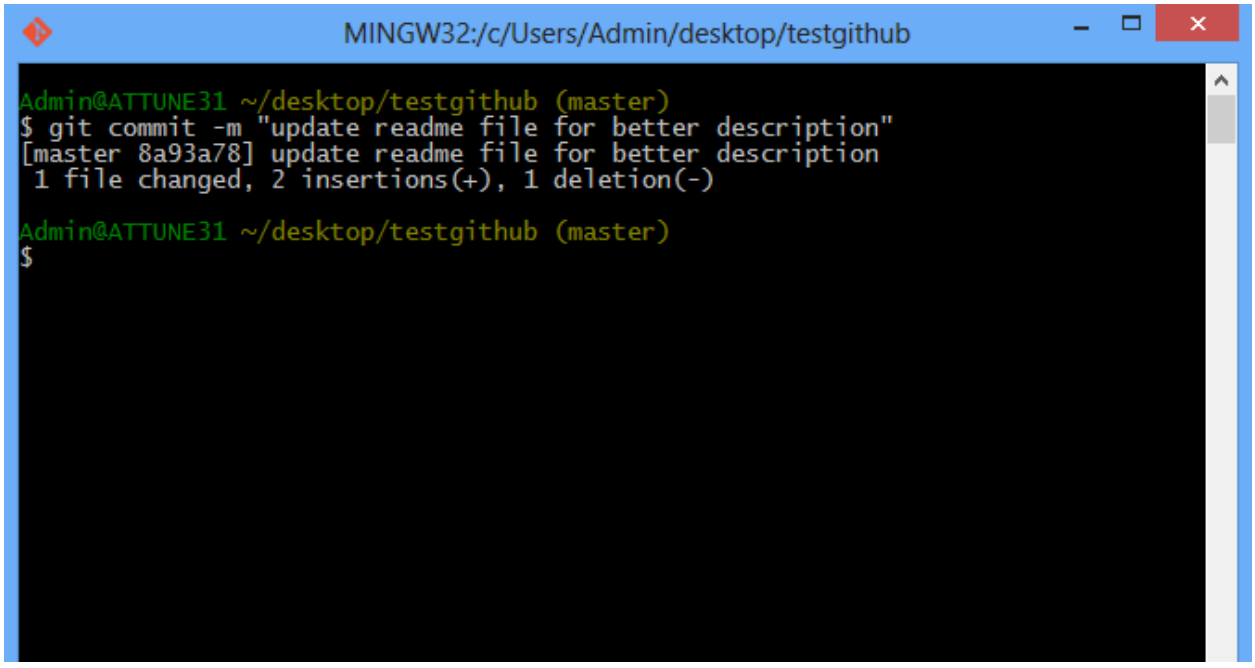
21. Now run command **git status** to know that what thing is modified in our repository.



22. After modifying file I want to update my repository online with this new readme because I did change and I want to update description I want to say that this file has a new description.
23. Now I need to add changes right. So you can use git head to add a new files into staging area think about it this way right so you are on your local directory.
    Staging area is just a place that where it's not exactly on the report on the web site yet by in between your local an online
24. So for that run command git add . to push all changes on to website.

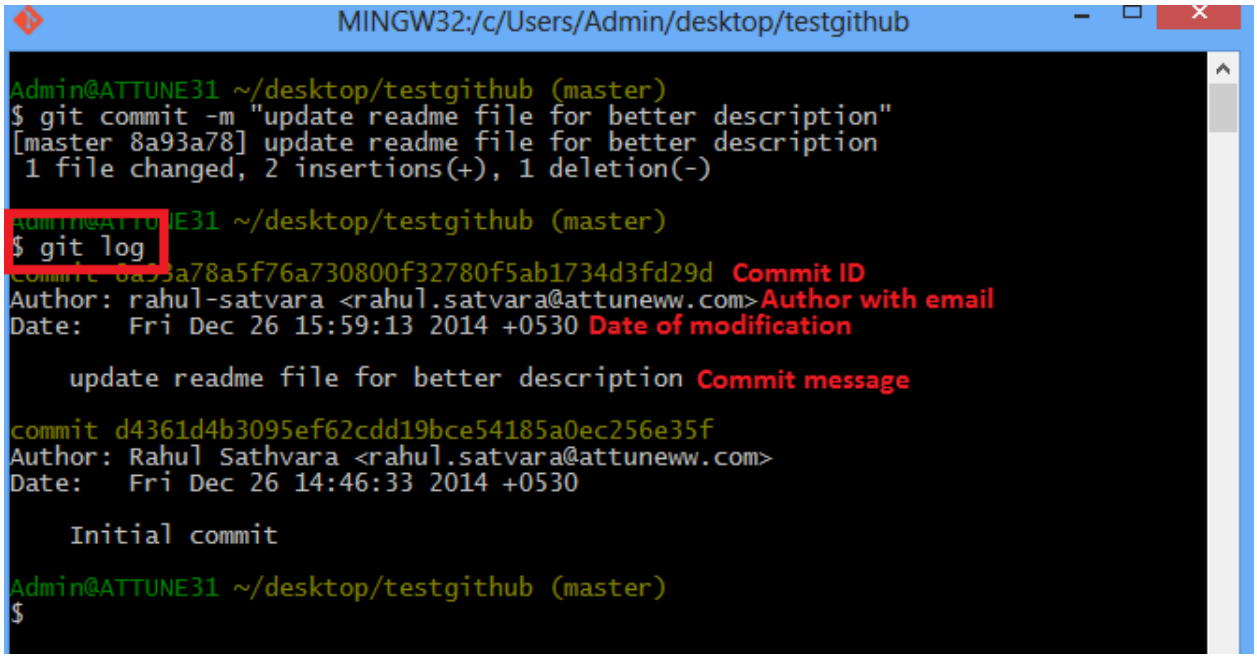


25. Now commit changes we done. Commit is a documenting the change and your finalizing staging area to be ready to push onto the rebook now. Just type **commit –m "Your Message"** message is for documenting changes to know which person done did change and what type change done.

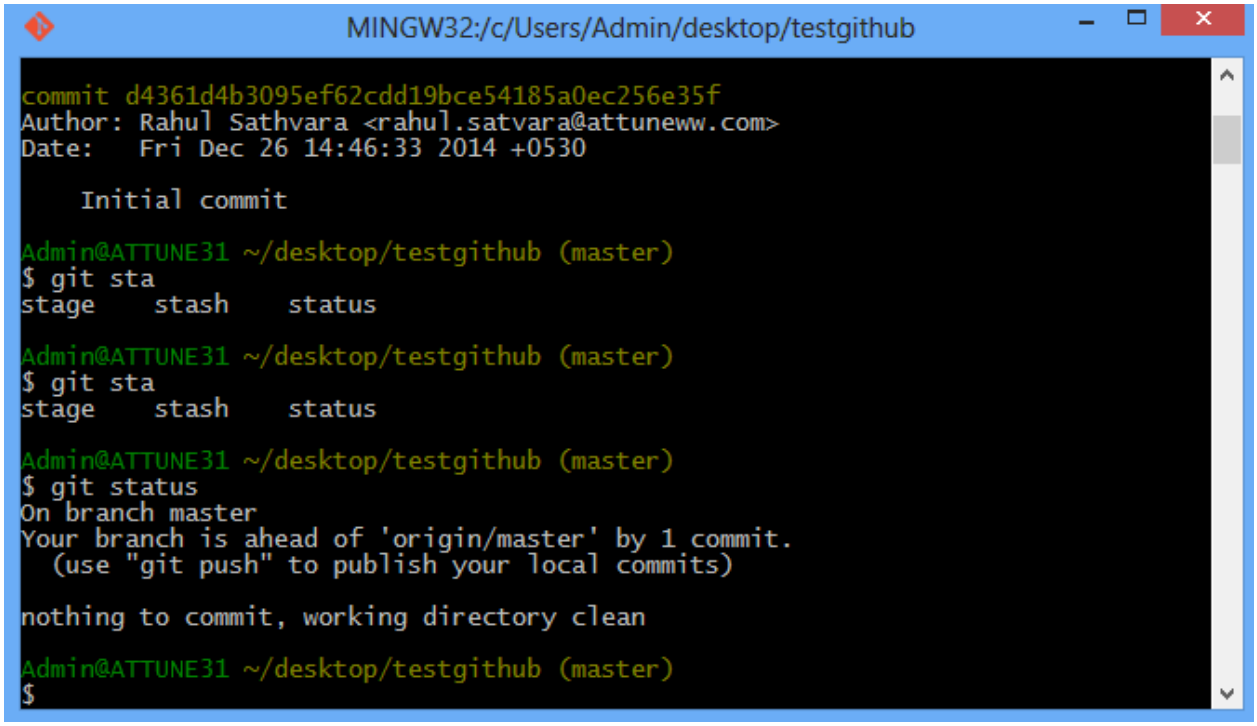


26. Now check log to know what happen.

Type: **git log**



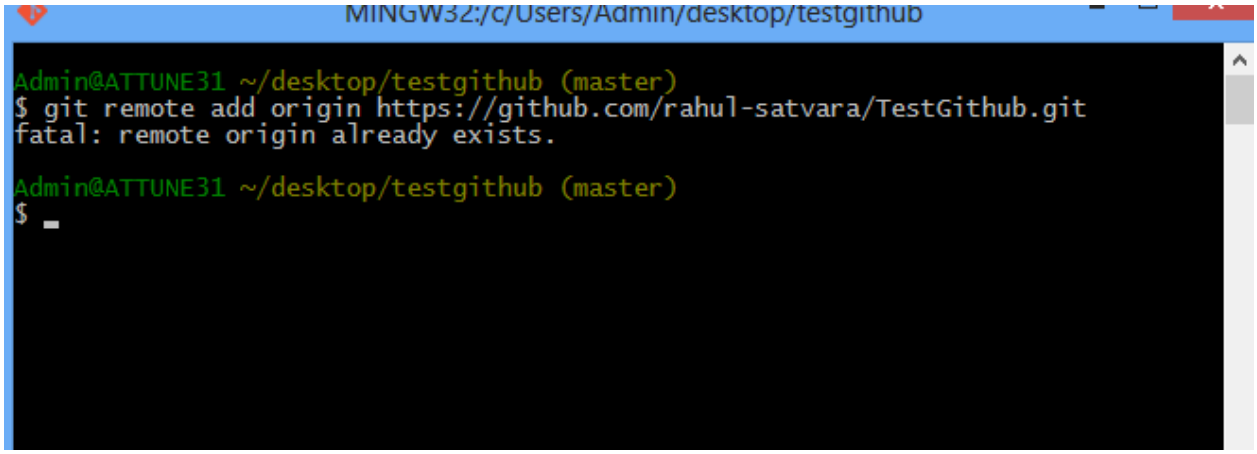
27.Now check status it is good to check status.



28.Your branch is ahead of ‘origin/master’ origin is out current working directory and master is remote directory.

29.So we are in local origin to let's move to remote origin.
30.Just type :-

git remote add origin https://github.com/rahul-satvara/TestGithub.git



We get error remote origin already exist because whenever you clone a repository it automatically create origin because of course you are cloning your are downloading it. It is going to create origin to wish you are currently at.

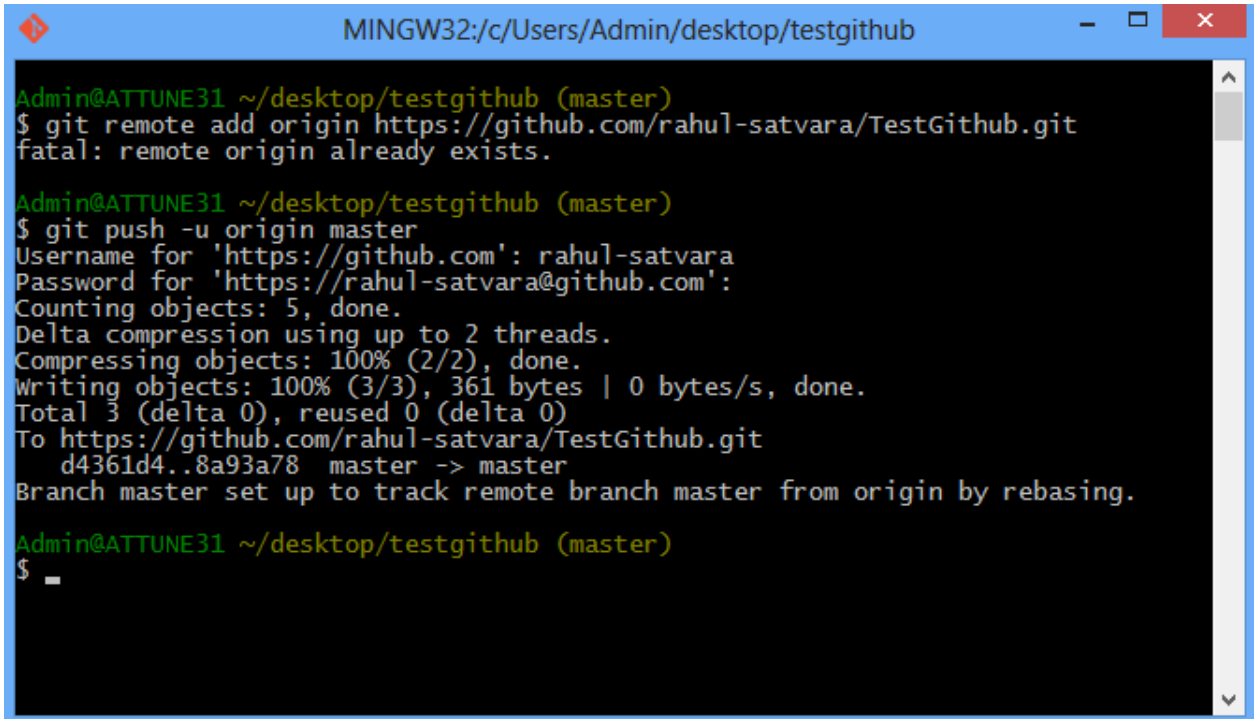
But if you are creating let's say new completely new repository and let's say I did not clone then I would have to get repo and clone.

This is necessary step if you are not cloning.

31.So after add origin we push change into master branch that is on remote location.

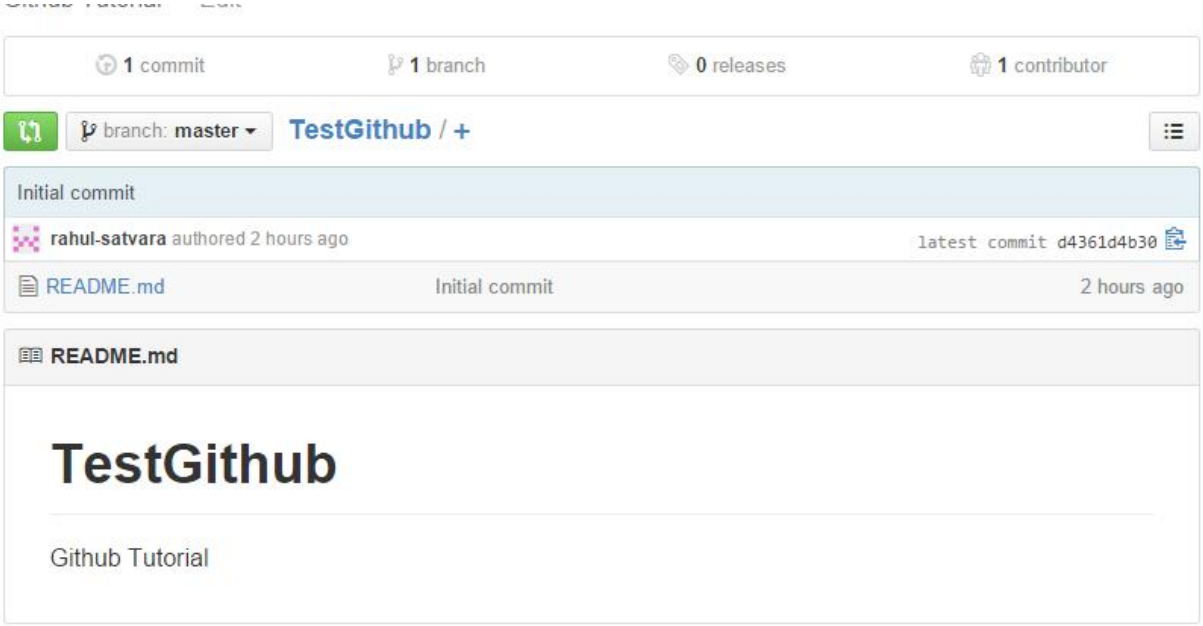
Git push –u origin master

32.It will ask you username and Password to access remote repository.

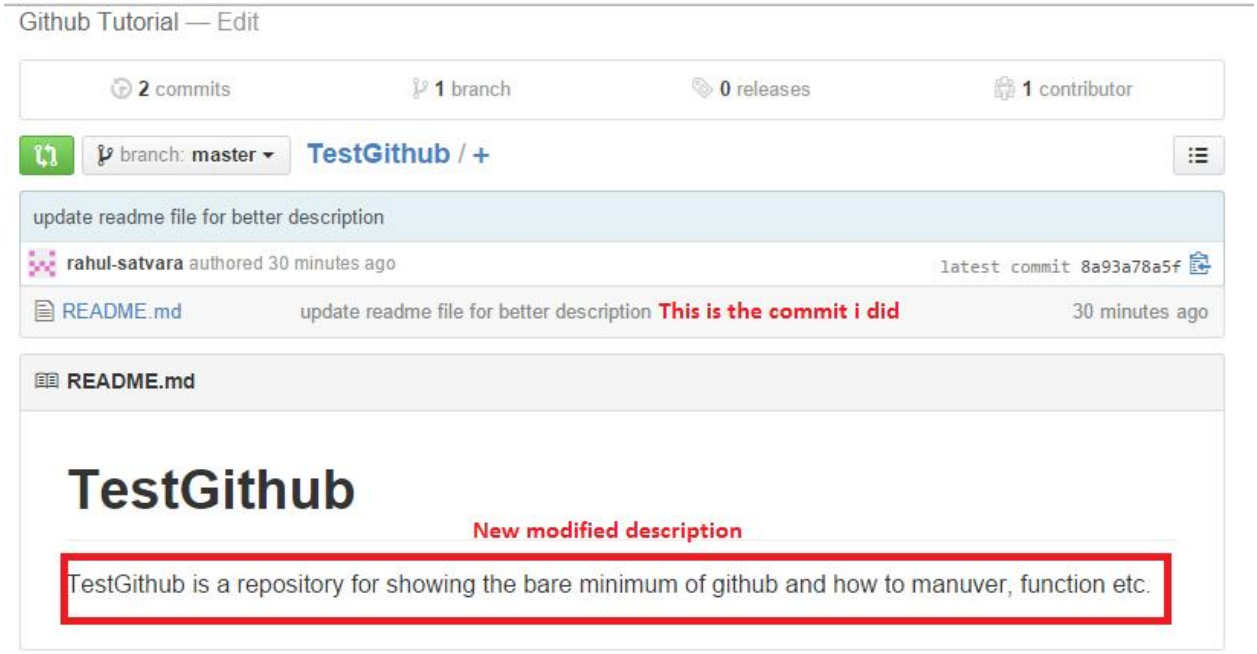


33.Now go to your browser refresh page.

Before Refresh:-

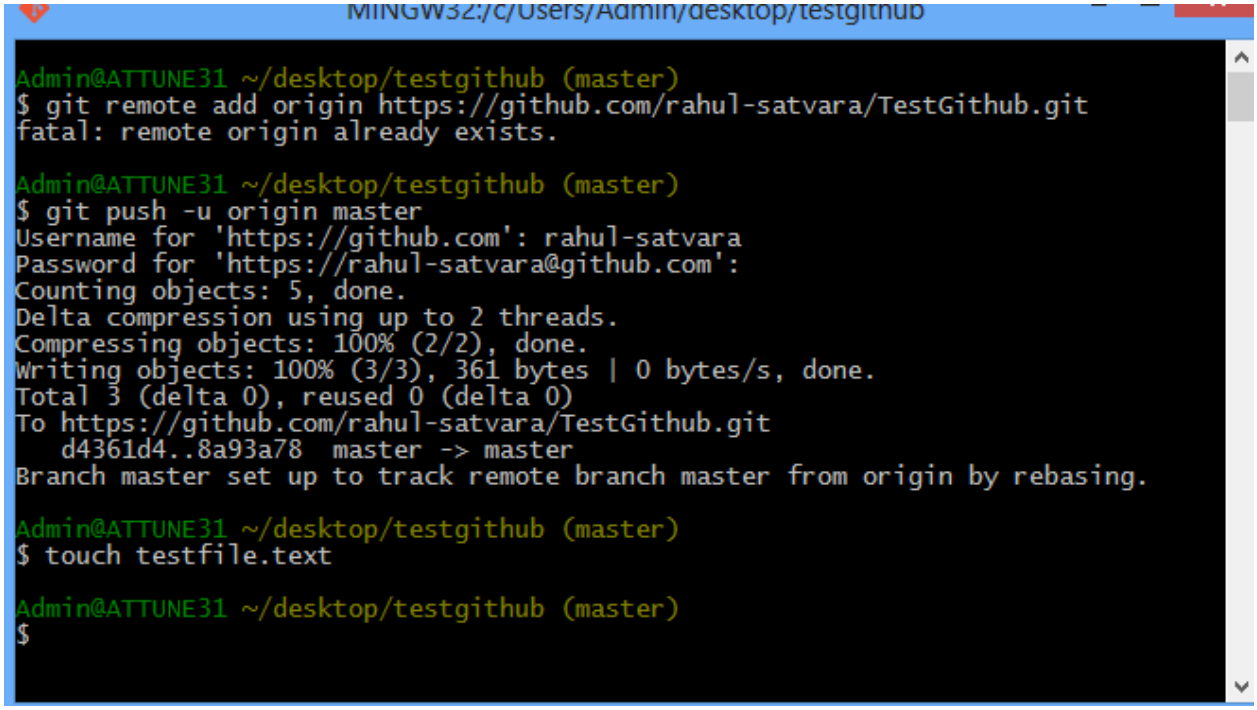


After Refresh:-

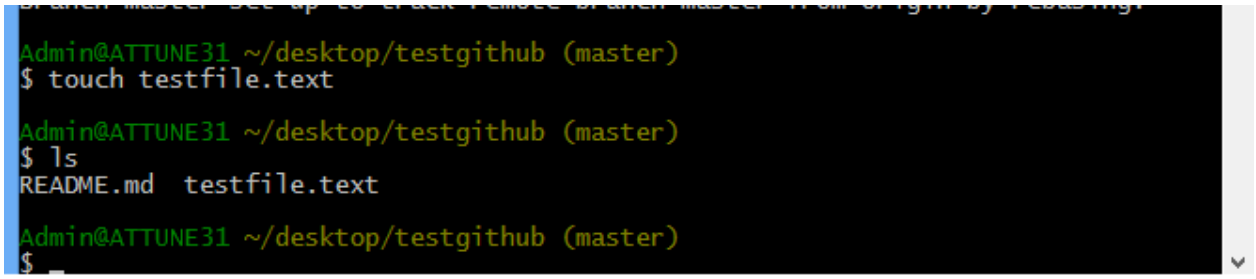


34.You can see new changes are pushed in our remote repository.

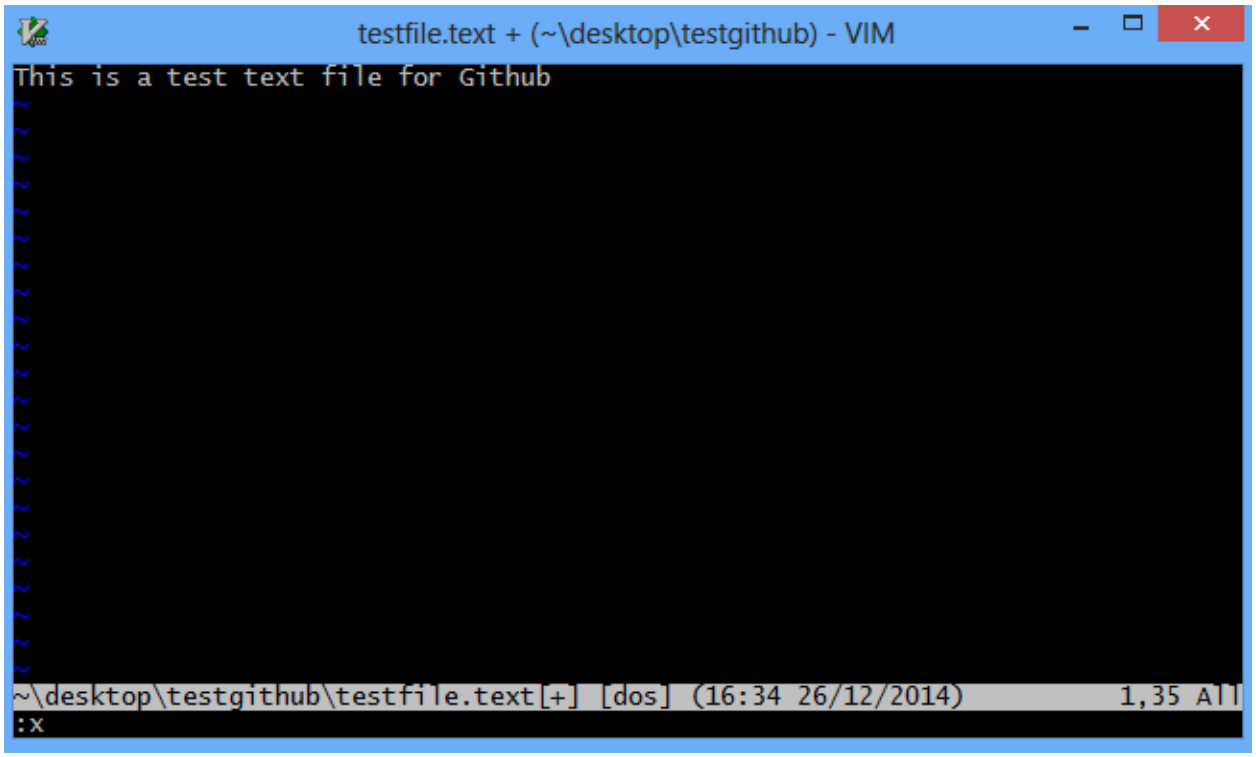
35.Now we will make a new file. You can use this commend **touch testfile.text** to create new file on local.



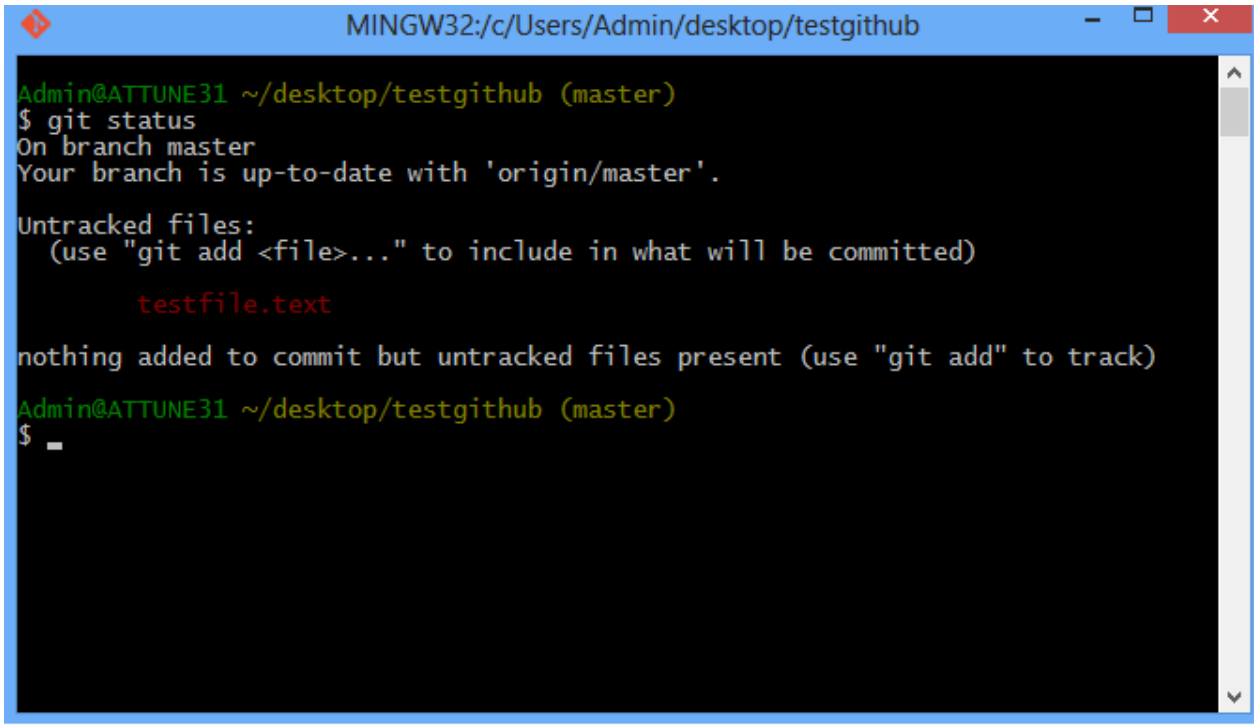
36.Now use **ls** to see new created file.



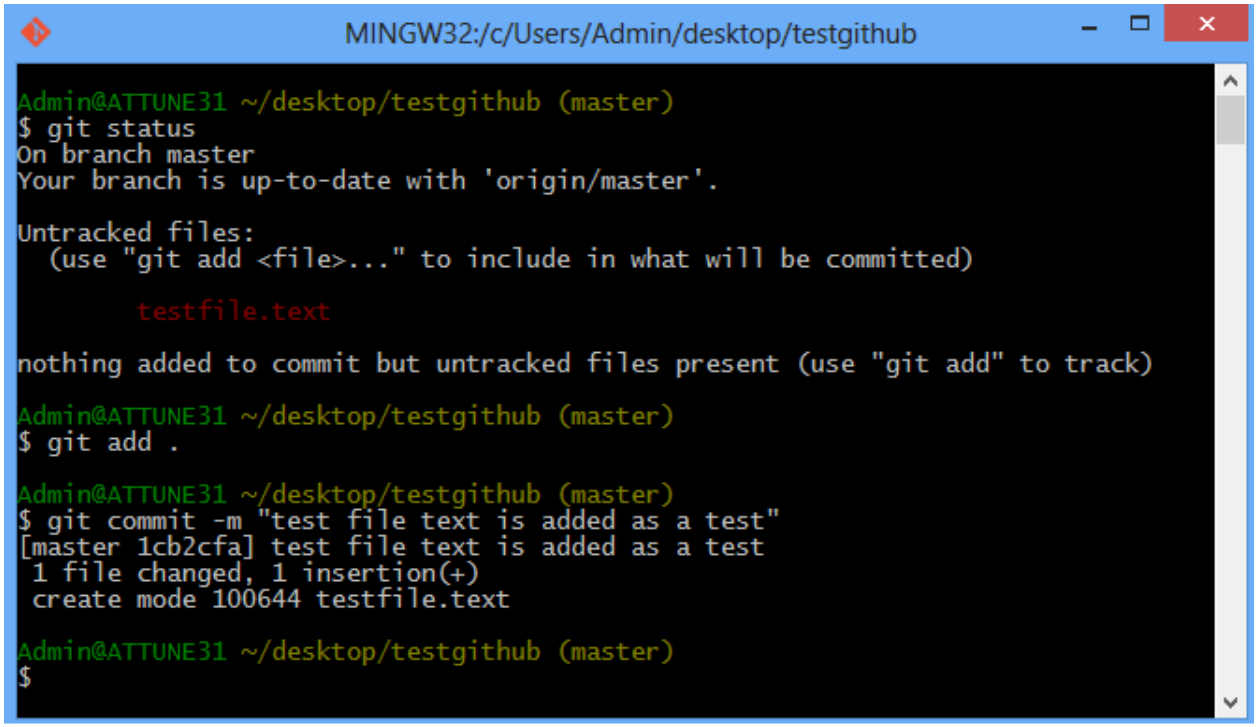
37. This file not created in remote repo. For that we have to add, commit and push to upload on remote repo.
38. First we add some text in testfile.
    a. Type vim testfile.text
    b. Type some text.
    c. Press Esc and
    d. Type :x to save and exit.



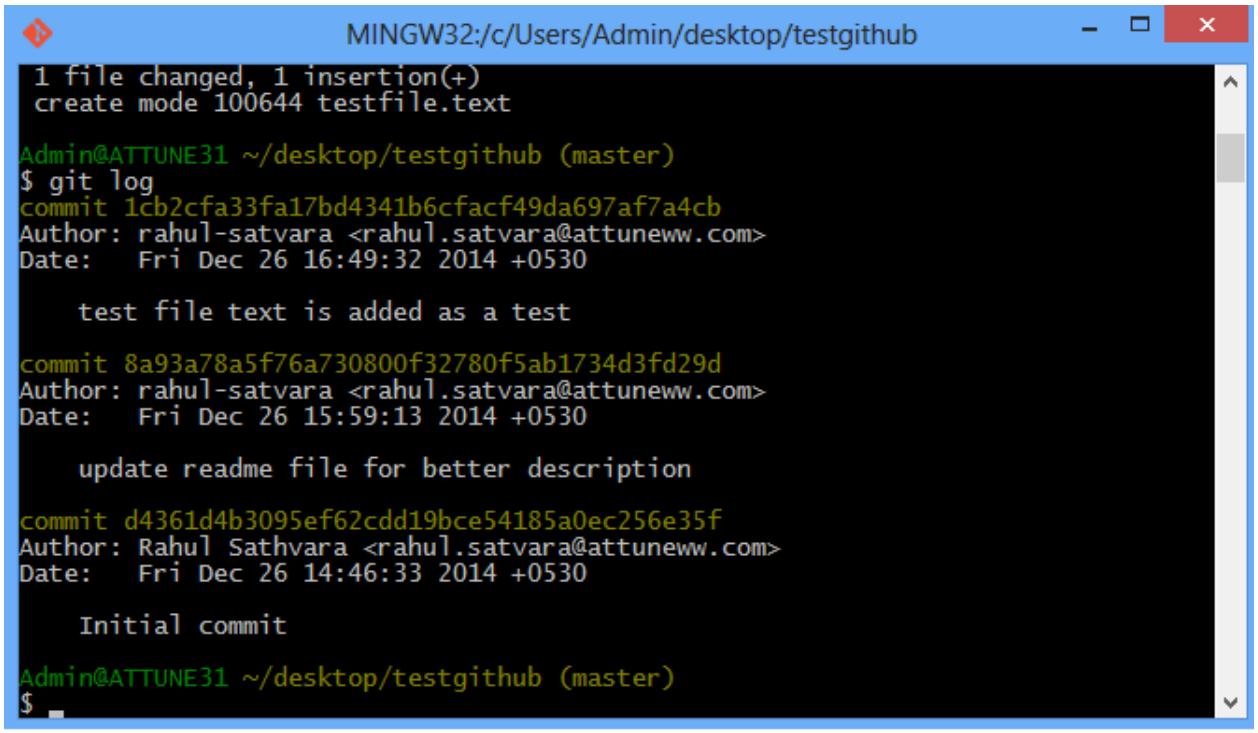
39. Let's check a status.



40. You can see that text is added in file and you can see that this file is not tracked yet.
41. So we need to add and commit.

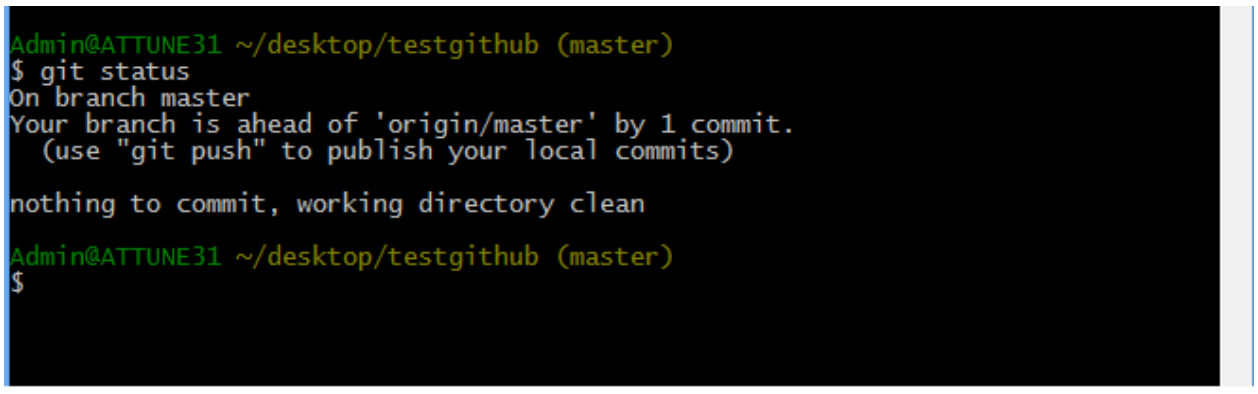


42.Let’s check log.



Here you can check all commit and commit id and Commit messages also.

43.Now check status.

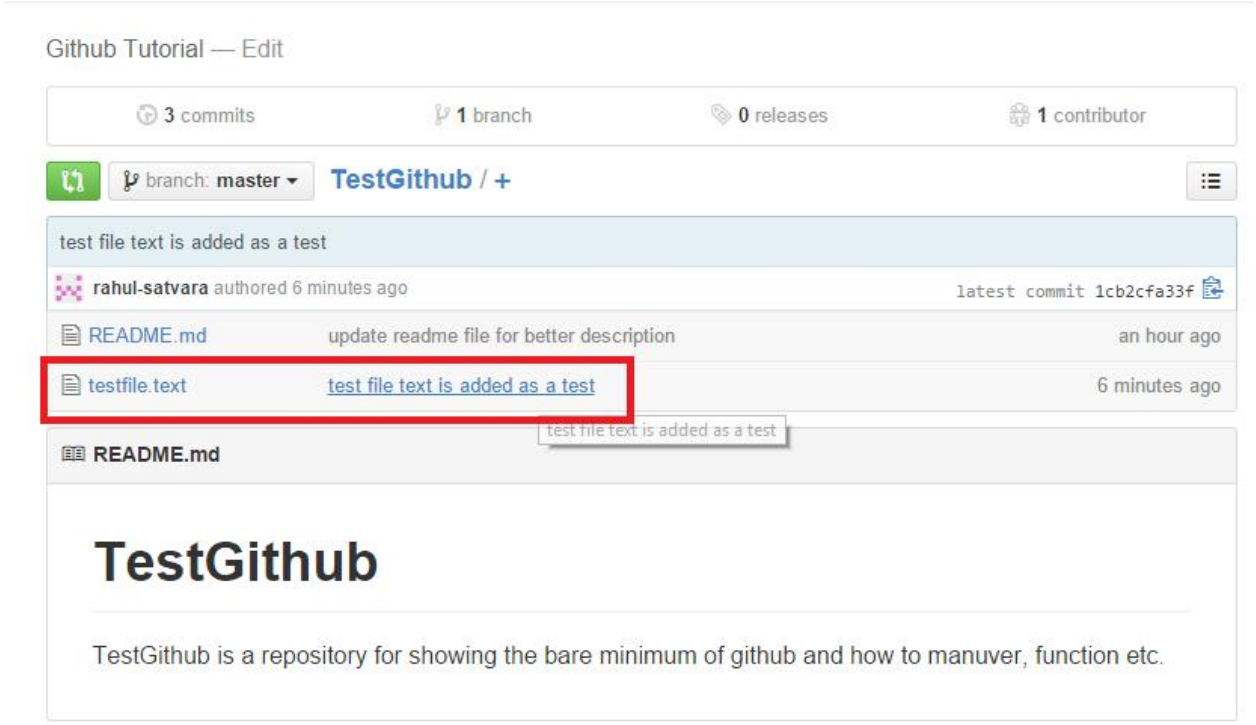


44.Nothing will be added in remote repo until run push.

45.So run push command.

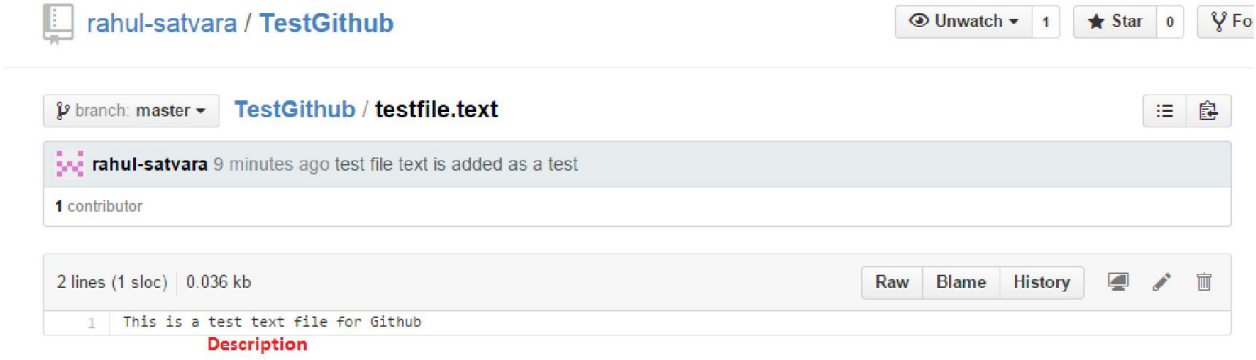
46.It will ask you username and Password just enter it.

47.Go to your browser and refresh page.



48.So you can see new pushed file testfile.text with commit message.

49.Open testfile.text



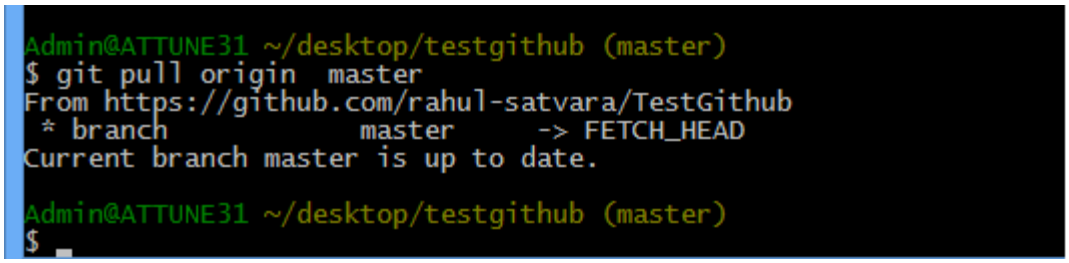
Here you can see description of testfile.

50.Now we will pull master repository to origin.

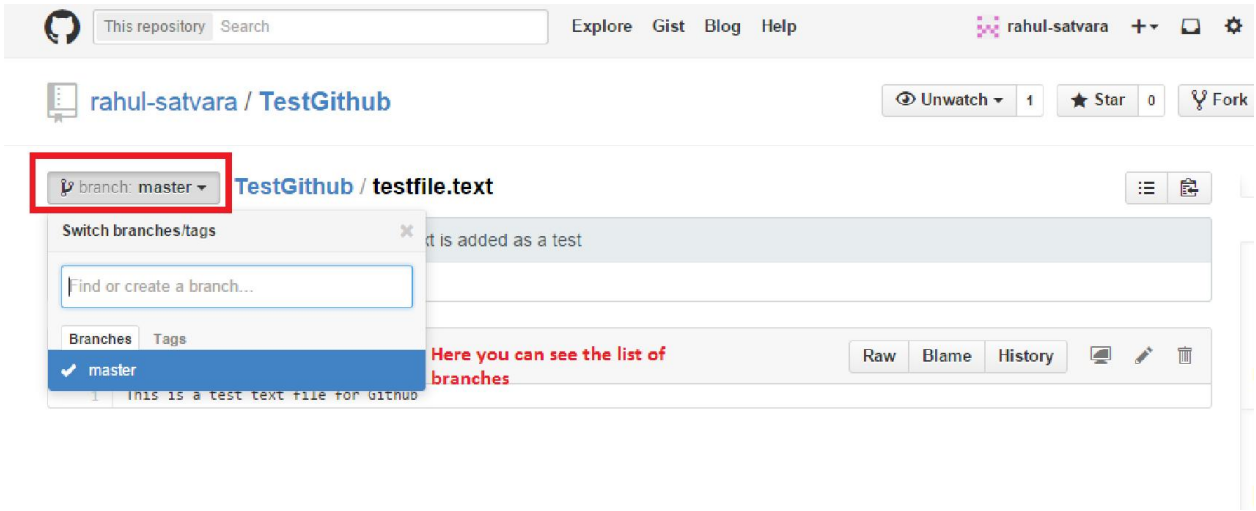
51.Just run **git pull origin master** this will pull all the content to current Repository and it will serve as Let's say that another friend was working on

a different file now let's say he was working on like **a.text** now you can do get pull origin master to basically pull that in update on your local drive.

52. I just run git pull origin master.



53. So now you can see that Current branch master is up to date because we have everything clone on the repository.
54. So that is how the pushing pulling and commit work on git hub recall using the shall.
55. Last thing the braches.



Here you can create a new branches and also see the list of branches.

And also check in which branch you are working on.

56. Lets check list of branches using command.

```
git branch
```

currently we have only one branch.

57. If you want to see all branches (including remote-tracking branches), use the `-a` for the `git branch` command.

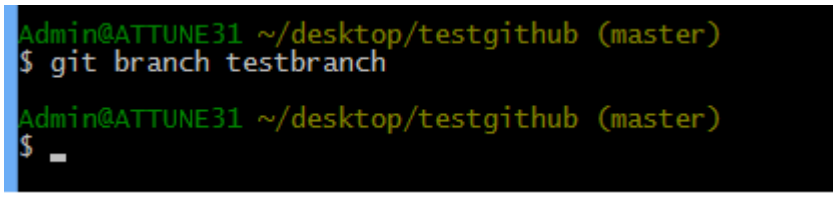
The `-v` option lists more information about the branches.

In order to list branches in a remote repository use the `git branch -r` command as demonstrated in the following example.

```
# lists branches in the remote repositories
git branch -r
```

58. You can create a new branch via the `git branch [newname]` command. This command allows to specify the starting

point (commit id, tag, remote or local branch). If not specified the commit to which the HEAD reference points is used to create the branch.



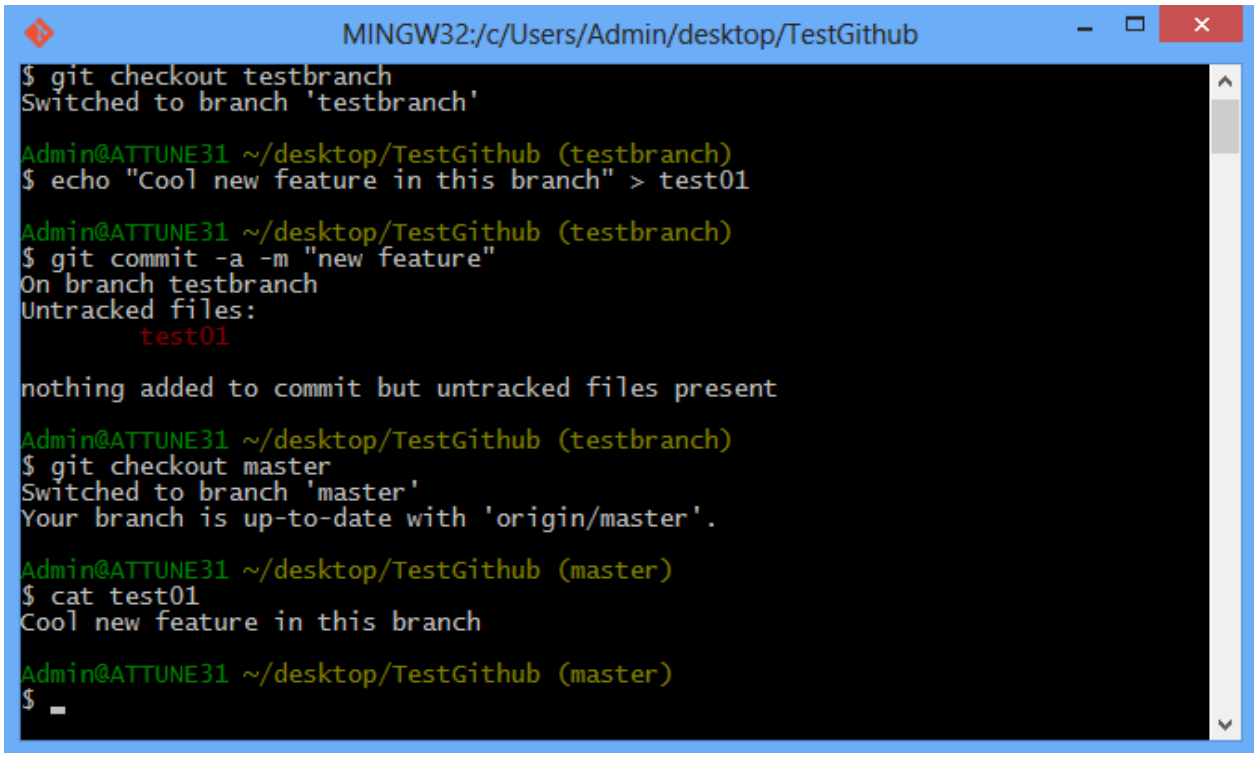
Checkout branch

59.Checkout branch:-

To start working in a branch you have to *checkout* the branch. If you *checkout* a branch, the HEAD pointer moves to the last commit in this branch and the files in the working tree are set to the state of this commit.

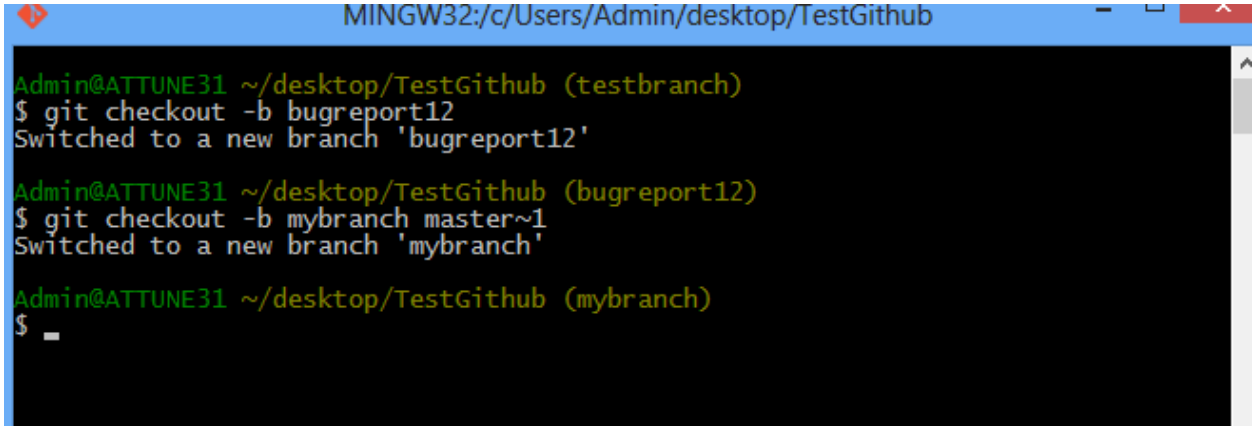
The following commands demonstrate how you switch to the branch called *testing*, perform some changes in this branch and switch back to the branch called *master*.

```
# switch to your new branch
git checkout testbranch

# do some changes
echo "Cool new feature in this branch" > test01
git commit -a -m "new feature"

# switch to the master branch
git checkout master

# check that the content of
# the test01 file is the old one
cat test01
```



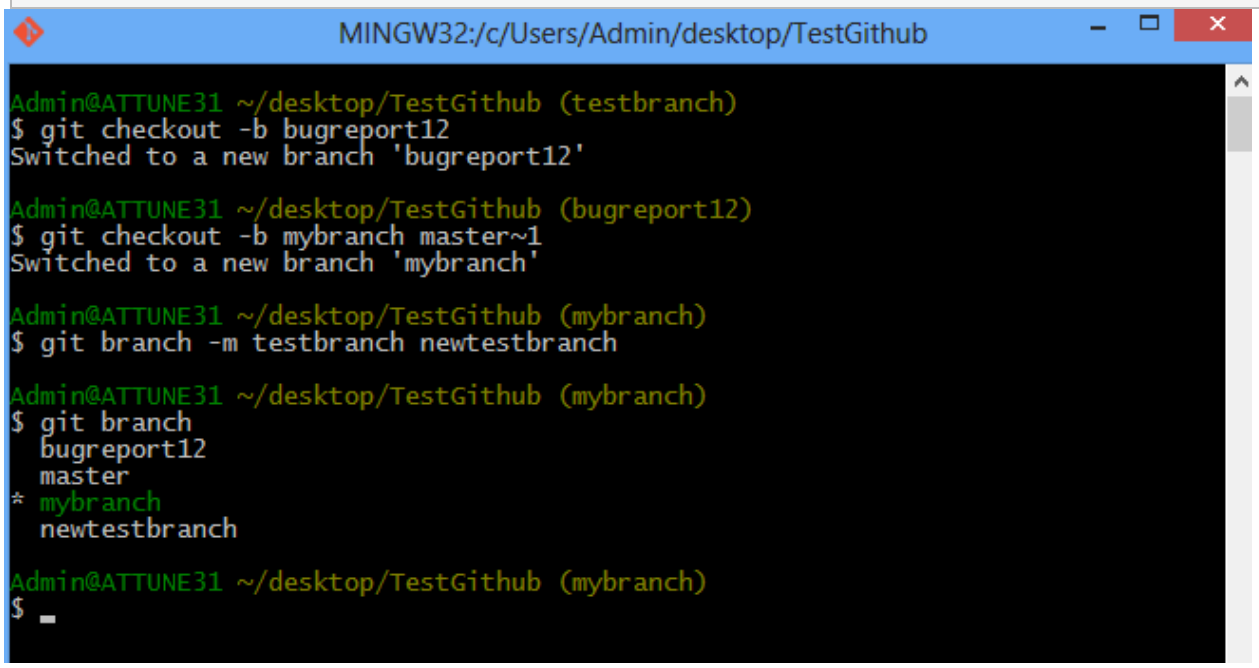
60. To create a branch and to switch to it at the same time you can use the `git checkout` command with the *`-b`* parameter.

```
# create branch and switch to it
git checkout -b bugreport12

# creates a new branch based on the master branch
# without the last commit
git checkout -b mybranch master~1
```



61. Renaming a branch can be done with the following command.

```
62. # rename branch
63. git branch -m [old_name] [new_name]
```



64. To delete a branch which is not needed anymore, you can use the following command. You may get an error message that there are uncommited changes if you did the previous examples step by step. Use force delete (uppercase `-D`) to delete it anyway.

```
65. # delete branch testing
```

```
git branch -d testing
# force delete testing
git branch -D testing
# check if branch has been deleted
git branch
```