Department of Information Technology

Academic Year: 2019-20 **Semester: VIII**
Class / Branch: BE IT
Subject: DevOps Lab (DL)
Subject Lab Incharge: Prof. Vishal S. Badgujar

EXPERIMENT NO. 04

Aim: To Perform Version Control on Websites/Softwares using distributed version-control system GIT

Theory:

Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency. Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.

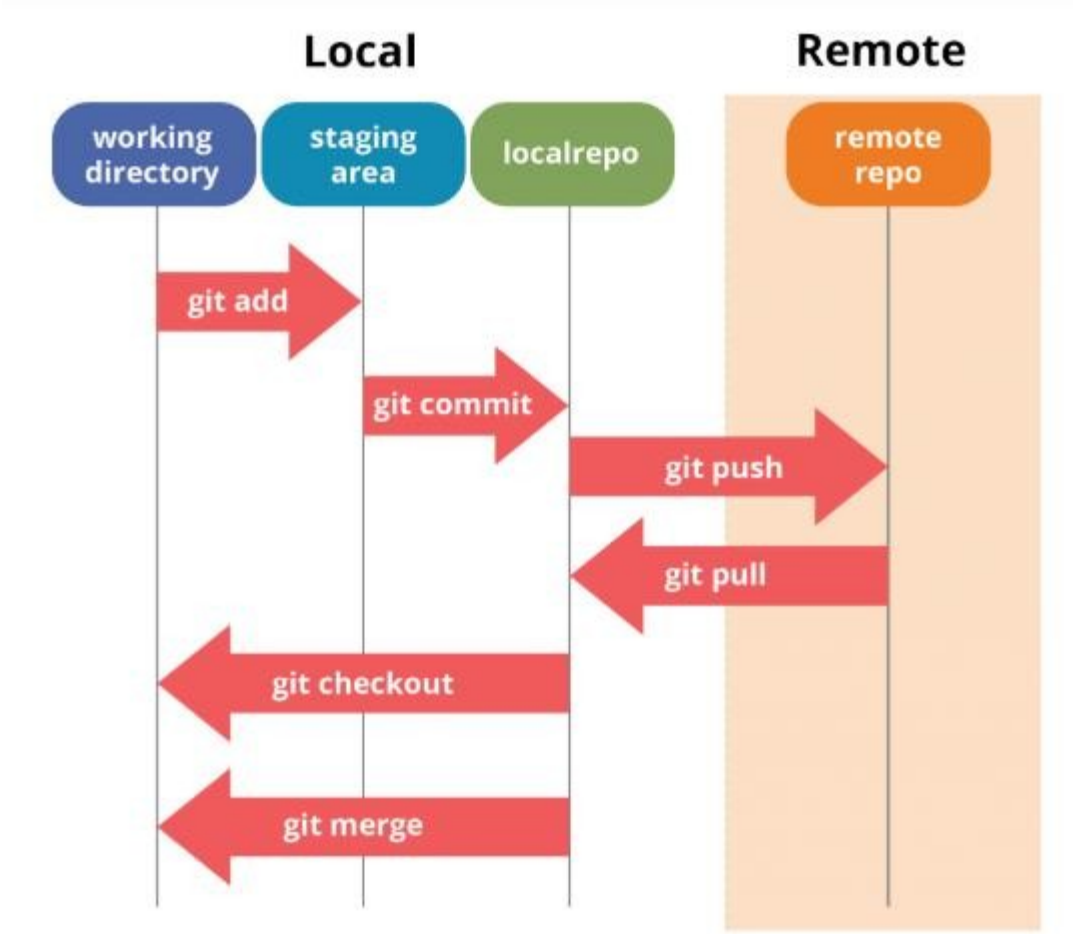
Some of the basic operations in Git are:

1. Initialize
2. Add
3. Commit
4. Pull
5. Push

Some advanced Git operations are:

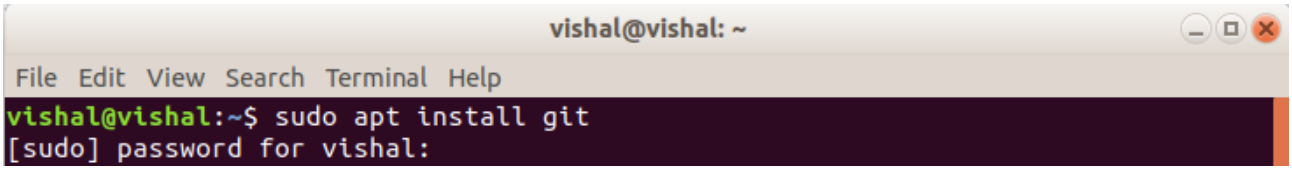
1. Branching
2. Merging
3. Rebasing

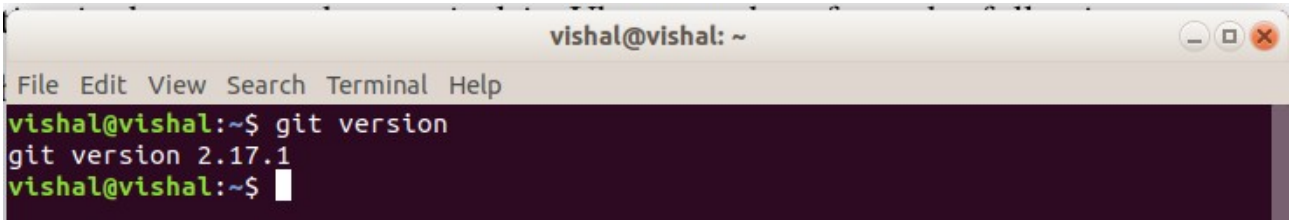
The following diagram depict the all supported operations in GIT



Installation of GIT

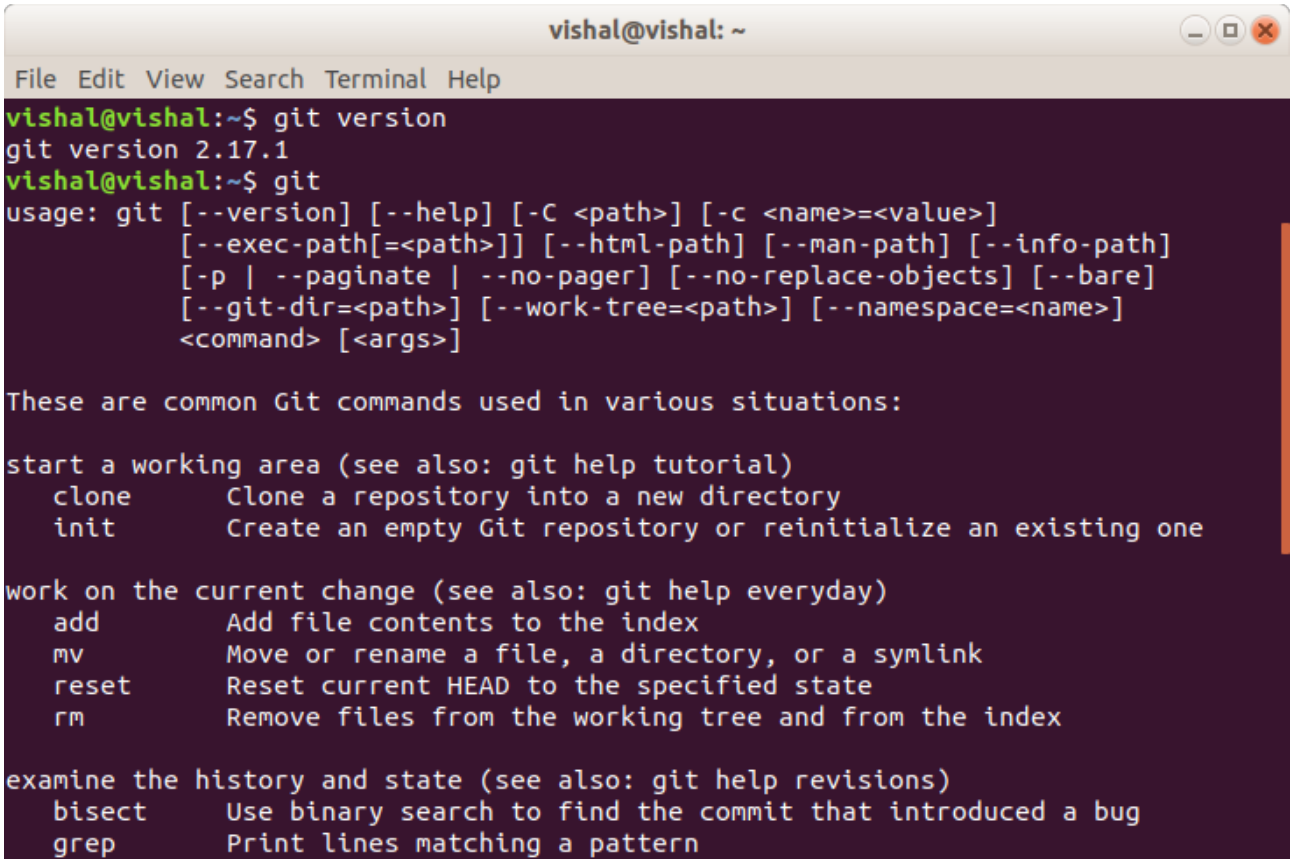
1) In Ubuntu, install GIT using $sudo apt install git, and then Confirm the version after installation using command $git version





Once installation is done, open the terminal in Ubuntu and perform the following steps

The output of GIT shell in Ubuntu is shown below



To perform version control, let us create a directory dvcs (Distributed version control system) and change directory to dvcs.

vishal@vishal:~$ mkdir git-dvcs

vishal@vishal:~$ cd git-dvcs/

Now check the user information using

vishal@vishal:~/git-dvcs$ git config --global

As there are no users defined, let us define it using following two commands

vishal@vishal:~/git-dvcs$ git config --global user.name "vishal"

vishal@vishal:~/git-dvcs$ git config --global user.email "vsbadgujar@apsit.edu.in"

```
vishal@vishal:~/git-dvcs$ git config --global user.name "vishal"
vishal@vishal:~/git-dvcs$ git config --global user.email "vsbadgujar@apsit.edu.in"
```

Now, check the list of users

vishal@vishal:~/git-dvcs$ git config --global –list

```
vishal@vishal:~/git-dvcs$ git config --global --list
user.name=vishal
user.email=vsbadgujar@apsit.edu.in
vishal@vishal:~/git-dvcs$
```

Let us create a repository for version control named “git-demo-project”

vishal@vishal:~/git-dvcs$ mkdir git-demo-project

vishal@vishal:~/git-dvcs$ cd git-demo-project/

Now, initialize the repository using following command

vishal@vishal:~/git-dvcs$ git init

```
vishal@vishal:~/git-dvcs$ mkdir git-demo-project
vishal@vishal:~/git-dvcs$ cd git-demo-project/
vishal@vishal:~/git-dvcs/git-demo-project$ git init
Initialized empty Git repository in /home/vishal/git-dvcs/git-demo-project/.git/
vishal@vishal:~/git-dvcs/git-demo-project$
```

The output of above command shown below which adds .git hidden directory in current repository.

Add some files inside our repository “ git-demo-project”

To add files in the repository by create or copy some doc,html,image files inside current directory to see index and staging area.

The add command is used along with dot (. Dot means current directory) for adding files in current repository i.e. making them in staging mode. They are untracked until we commit them.

vishal@vishal:~/git-dvcs/git-demo-project$ git add .

Index and staging area

To check the status of repository, use

vishal@vishal:~/git-dvcs/git-demo-project$ git status

Which will show you some untrack files, so untracks files can be tracked using commit command.

Now, let us commit the changes

vishal@vishal:~/git-dvcs/git-demo-project$ git commit -m "First Commit" (#here -m for message)

```
vishal@vishal:~/git-dvcs/git-demo-project$ git commit -m "First Commit"
On branch master

Initial commit

Untracked files:
        DevOps Tools.pdf

nothing added to commit but untracked files present
vishal@vishal:~/git-dvcs/git-demo-project$ git status
On branch master

No commits yet

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

        DevOps Tools.pdf

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

```
vishal@vishal:~/git-dvcs/git-demo-project$ git add .
vishal@vishal:~/git-dvcs/git-demo-project$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   DevOps Tools.pdf

vishal@vishal:~/git-dvcs/git-demo-project$ git commit -m "First Commit"
[master (root-commit) e1f8faa] First Commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 DevOps Tools.pdf
```

Add index.html in our directory by using command

vishal@vishal:~/git-dvcs/git-demo-project$touch index.html

```
vishal@vishal:~/git-dvcs/git-demo-project$ touch index.html
vishal@vishal:~/git-dvcs/git-demo-project$ git commit -m "First Commit"
On branch master
Untracked files:
        index.html

nothing added to commit but untracked files present
```

vishal@vishal:~/git-dvcs/git-demo-project$ git add .

vishal@vishal:~/git-dvcs/git-demo-project$git commit -am "express Commit" (#Here -a used for express commit)

vishal@vishal:~/git-dvcs/git-demo-project$ nano index.html

put any text in index html and save file by ctrl+o for save and ctrl+x for exit

```
vishal@vishal:~/git-dvcs/git-demo-project$ nano index.html
vishal@vishal:~/git-dvcs/git-demo-project$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")
```

```
vishal@vishal:~/git-dvcs/git-demo-project$ touch apsit
vishal@vishal:~/git-dvcs/git-demo-project$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   index.html

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

        apsit

no changes added to commit (use "git add" and/or "git commit -a")
```

Changes are Discarded by checkout

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

(use "git restore <file>..." to discard changes in working directory)

vishal@vishal:~/git-dvcs/git-demo-project$ git add index.html

vishal@vishal:~/git-dvcs/git-demo-project$ git add apsit

```
vishal@vishal:~/git-dvcs/git-demo-project$ git add index.html
vishal@vishal:~/git-dvcs/git-demo-project$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   index.html
```

```
vishal@vishal:~/git-dvcs/git-demo-project$ git add apsit
vishal@vishal:~/git-dvcs/git-demo-project$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   apsit
```

vishal@vishal:~/git-dvcs/git-demo-project$ git commit -am "Express commit"

```
vishal@vishal:~/git-dvcs/git-demo-project$ git commit -am "express Commit"
[master 380b1cb] express Commit
 1 file changed, 1 insertion(+)
```

```
vishal@vishal:~/git-dvcs/git-demo-project$ git status
On branch master
nothing to commit, working tree clean
```

Now let us see history of commits. The log command is used for seeing the commit history.

vishal@vishal:~/git-dvcs/git-demo-project$ git log

```
vishal@vishal:~/git-dvcs/git-demo-project$ git log
commit 380b1cbccdb315e33641acac0012ada86fb96ec2 (HEAD -> master)
Author: vishal <vsbadgujar@apsit.edu.in>
Date:   Sun Jan 12 22:58:51 2020 +0530

    express Commit

commit b52ffc80d553695a88bfdc5690f36647584f9f38
Author: vishal <vsbadgujar@apsit.edu.in>
Date:   Sun Jan 12 22:57:58 2020 +0530

    express Commit

commit be24cf8ae7a65f9f807cec6b42b41b9d6fe81ff0
Author: vishal <vsbadgujar@apsit.edu.in>
Date:   Sun Jan 12 22:53:31 2020 +0530

    express Commit

commit e1f8faa9cd434035d1863296e011dbca877510a9
Author: vishal <vsbadgujar@apsit.edu.in>
Date:   Sun Jan 12 22:47:32 2020 +0530
```

To see all the operation in oneline use the –-oneline option in log command

```
vishal@vishal:~/git-dvcs/git-demo-project$ git log --oneline
380b1cb (HEAD -> master) express Commit
b52ffc8 express Commit
be24cf8 express Commit
e1f8faa First Commit
```

--oneline option for particular file in log command

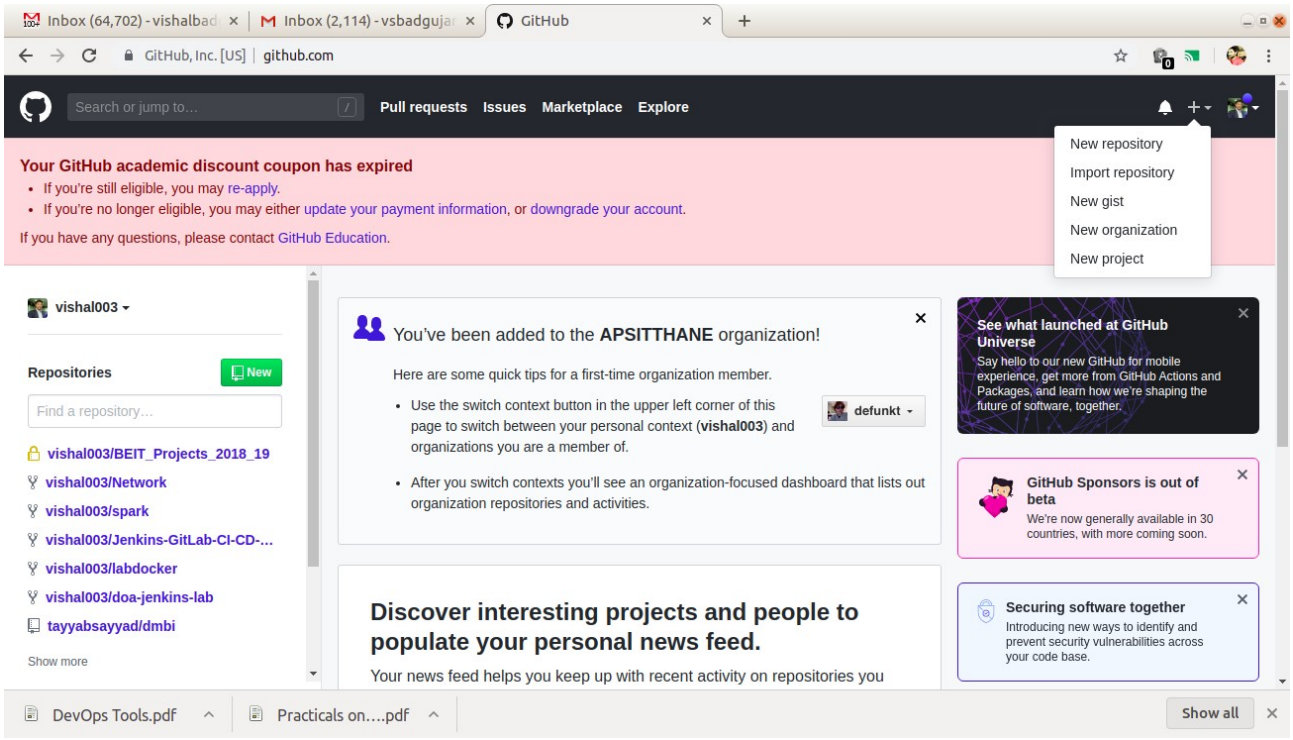
```
vishal@vishal:~/git-dvcs/git-demo-project$ git log --oneline apsit
b52ffc8 express Commit
```

```
vishal@vishal:~/git-dvcs/git-demo-project$ git log --oneline -n 2
380b1cb (HEAD -> master) express Commit
b52ffc8 express Commit
```

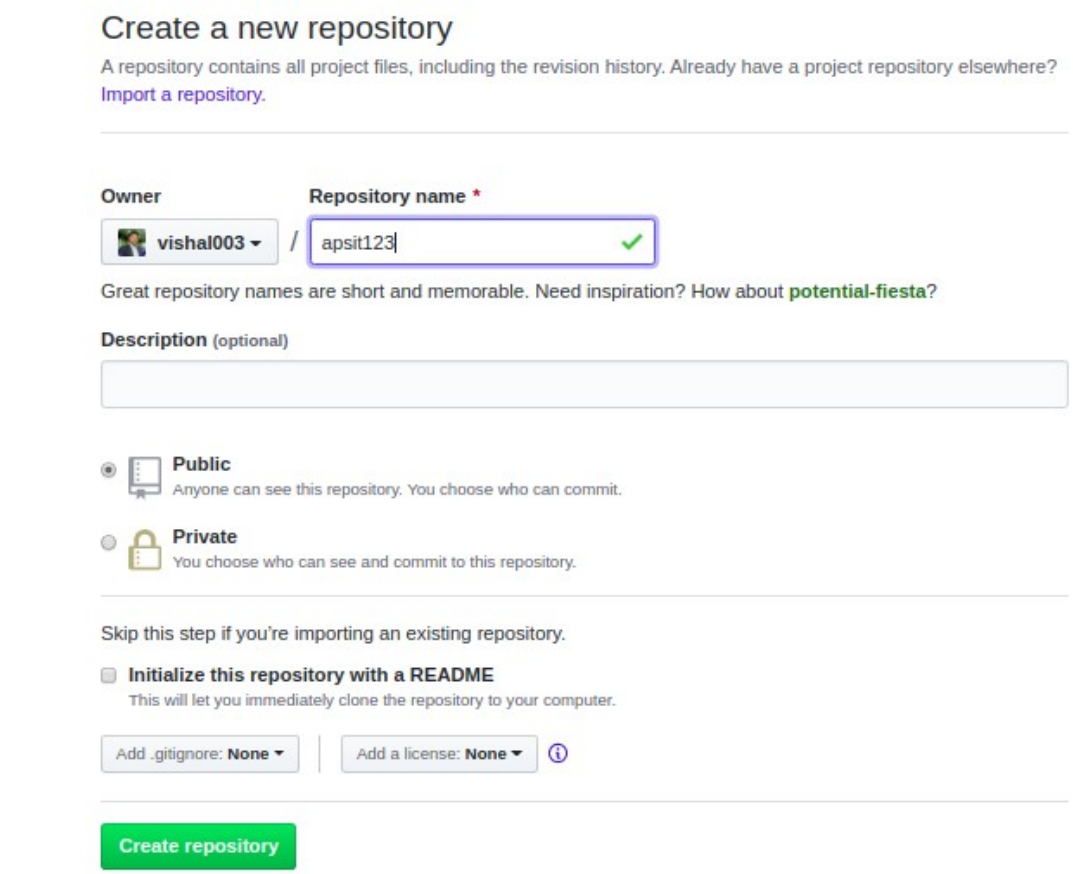
Example 2: Performing Version control in GITHUB with Pull and Push commands.

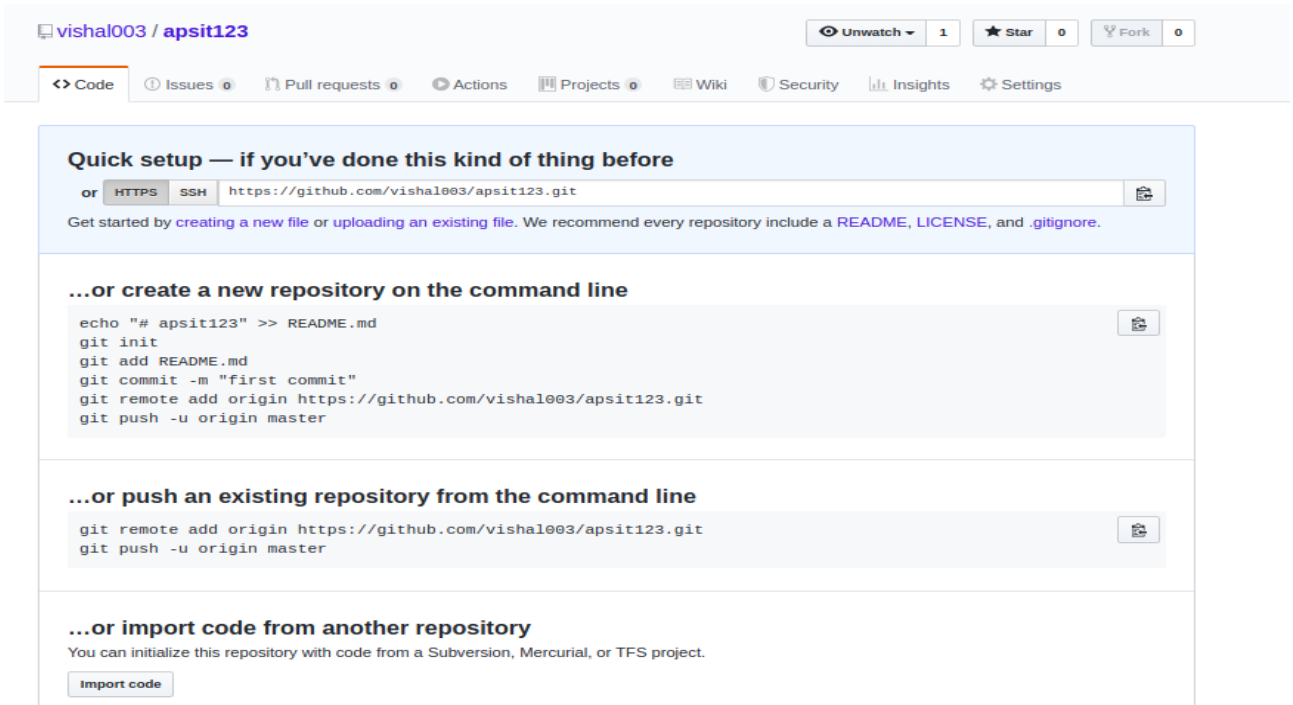
First open Github.com and create a new account.After verifying account through E-mail, create a Repository on github.com.

Open github.com → create an account → After login Select New repository from the menu.



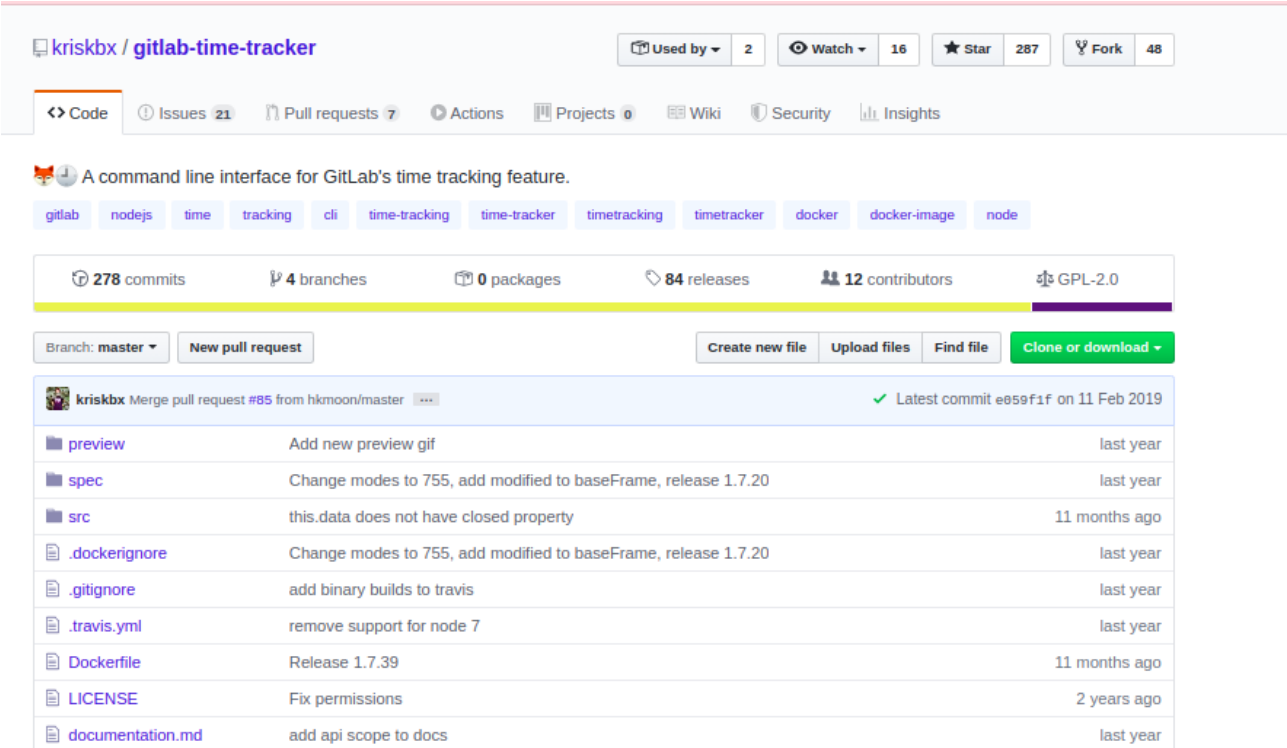
Specify a Name to repository and select public option followed by create repository


Create a new repository
A repository contains all project files, including the revision history. Already have a project repository elsewhere?
Import a repository.
Owner
Repository name *
vishal003
/
apsit123
Great repository names are short and memorable. Need inspiration? How about potential-fiesta?
Description (optional)
Public
Anyone can see this repository. You choose who can commit.
Private
You choose who can see and commit to this repository.
Skip this step if you're importing an existing repository.
Initialize this repository with a README
This will let you immediately clone the repository to your computer.
Add .gitignore: None
Add a license: None
Create repository

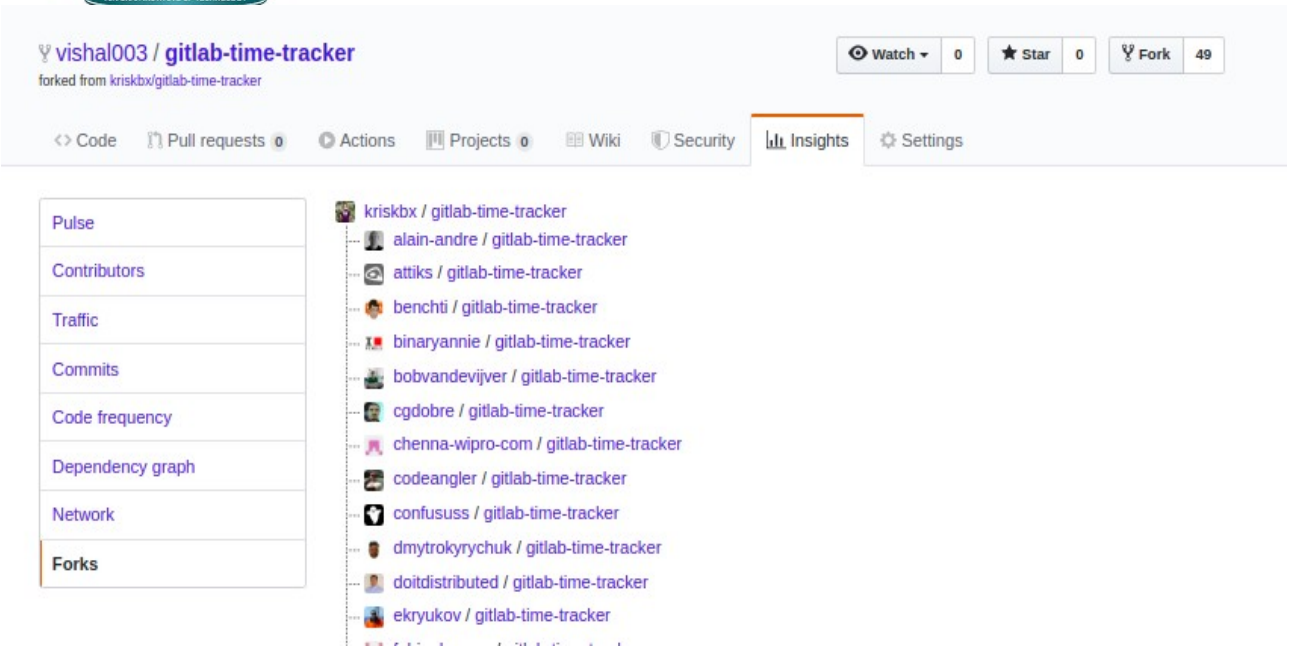

vishal003 / apsit123
Unwatch 1
Star 0
Fork 0
Code
Issues 0
Pull requests 0
Actions
Projects 0
Wiki
Security
Insights
Settings
Quick setup — if you've done this kind of thing before
or HTTPS SSH https://github.com/vishal003/apsit123.git
Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.
...or create a new repository on the command line
echo "# apsit123" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/vishal003/apsit123.git
git push -u origin master
...or push an existing repository from the command line
git remote add origin https://github.com/vishal003/apsit123.git
git push -u origin master
...or import code from another repository
You can initialize this repository with code from a Subversion, Mercurial, or TFS project.
Import code

By default, we can create public repository in Github. So we can copy the entire public repository of any other users in to own account using "FORK" Operation. Now fork the repository (Sharing with other users who wants to contribute).

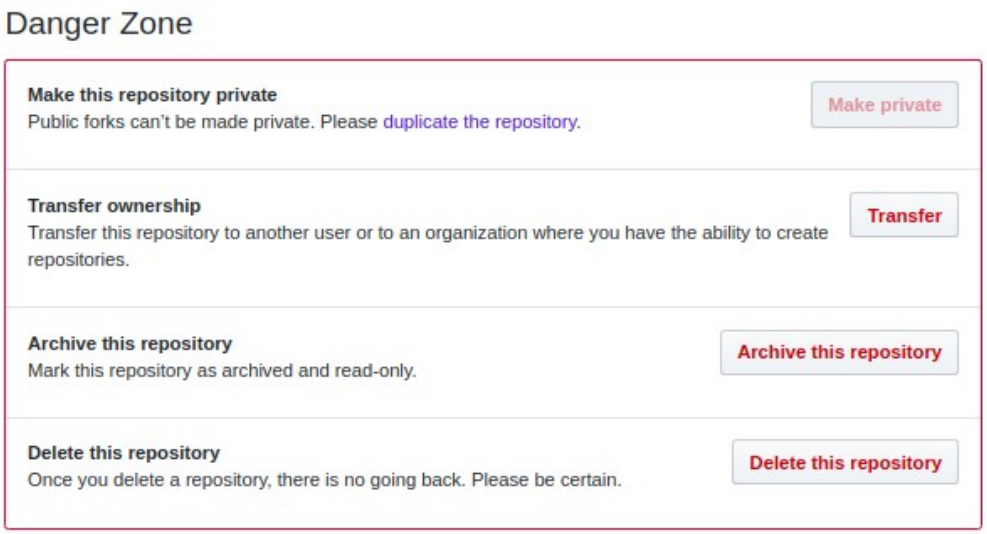
Login with another account → Copy and Paste URL of repository → then just click on fork to clone to others account. Suppose we want to fork public repository "timetracker". So search for "timetracker" github repository on google and once its opened clicked on "Fork button" from the top of the github web page as shown below.

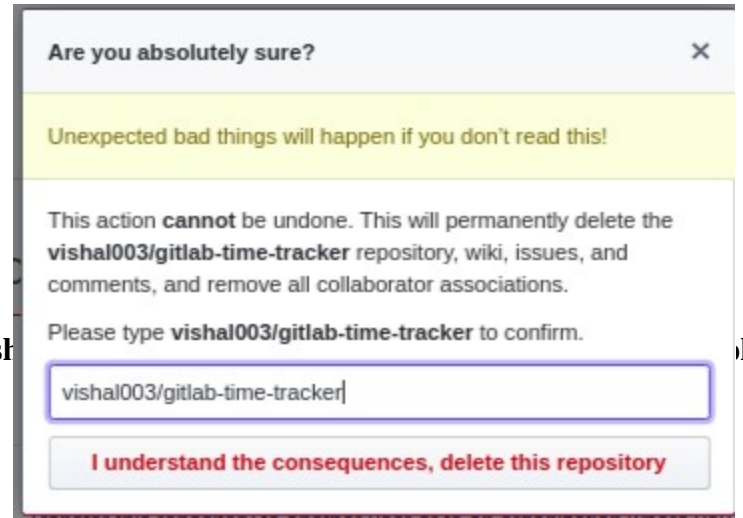


After fork it will be added in your local repository.

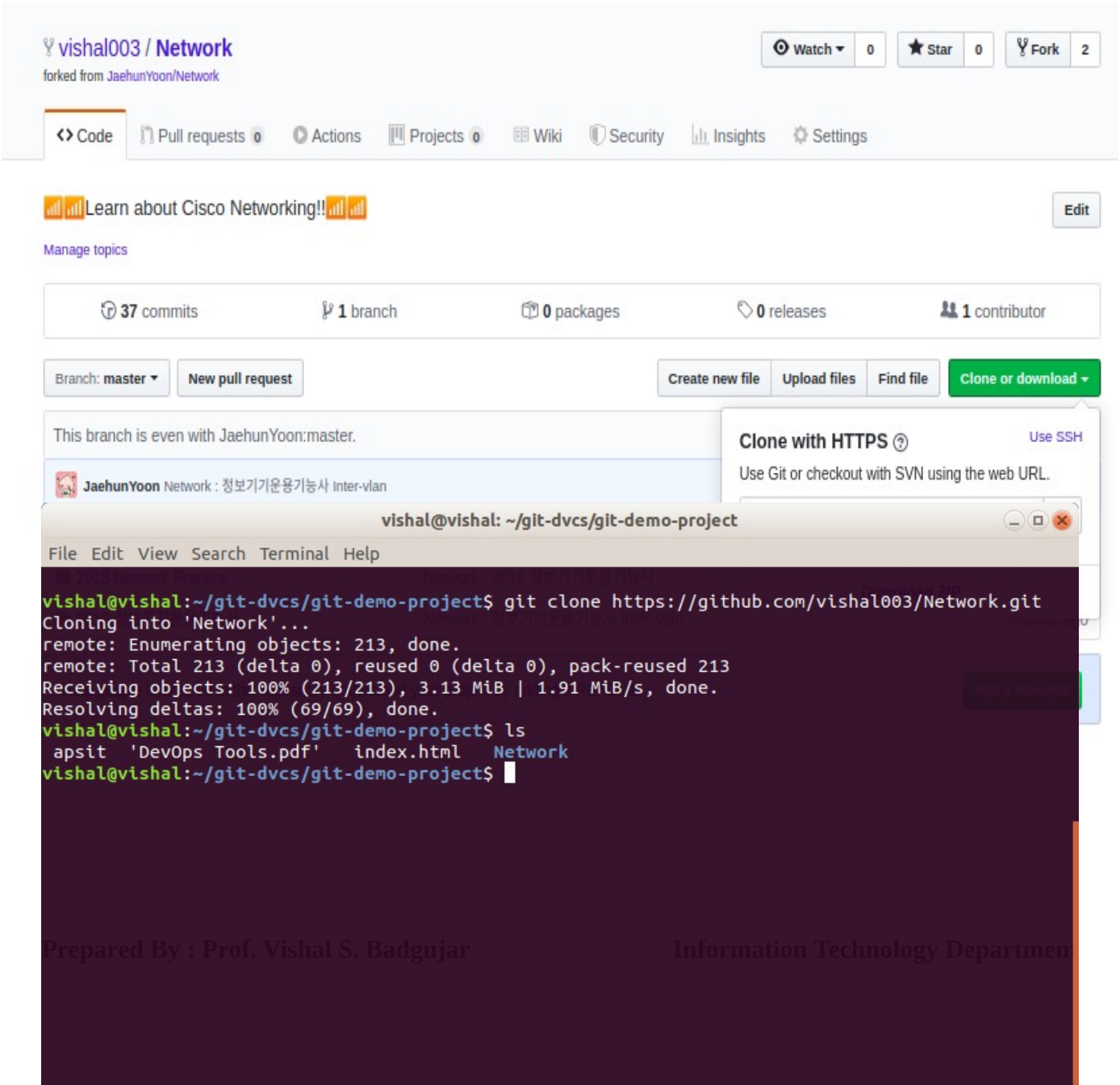


To delete the repository, open the desired repository you want to delete and go to the settings option. There you will see delete repository button to delete it.





if you want to download a repository in local machine, then git clone command is used followed by path to repository. In GitHub the path of repository can be known through clone or download button and it can be downloaded using git clone command as shown below.



To clone repository into your git local repository :

Pull and Push Processes

The pull command used to fetch the repository from github to local while push is used to commit files from local repository to Github.

Push → Push changes to Web repository

Pull → Pull changes to Local repository

The following commands are used for pull and push repositories

A) Push command

vishal@vishal:~/git-dvcs/git-demo-project$ git remote add origin https://github.com/vishal003/Network.git

vishal@vishal:~/git-dvcs/git-demo-project$ git remote show origin

```
vishal@vishal:~/git-dvcs/git-demo-project$ git remote add origin https://github.com/vishal003/Network.git
vishal@vishal:~/git-dvcs/git-demo-project$ git remote show origin
* remote origin
  Fetch URL: https://github.com/vishal003/Network.git
  Push  URL: https://github.com/vishal003/Network.git
  HEAD branch: master
  Remote branch:
    master new (next fetch will store in remotes/origin)
  Local ref configured for 'git push':
    master pushes to master (local out of date)
vishal@vishal:~/git-dvcs/git-demo-project$
```

If you add remote again then will show you fatal error.

vishal@vishal:~/git-dvcs/git-demo-project$ git remote add origin https://github.com/vishal003/Myrepository.git

fatal: remote origin already exists.

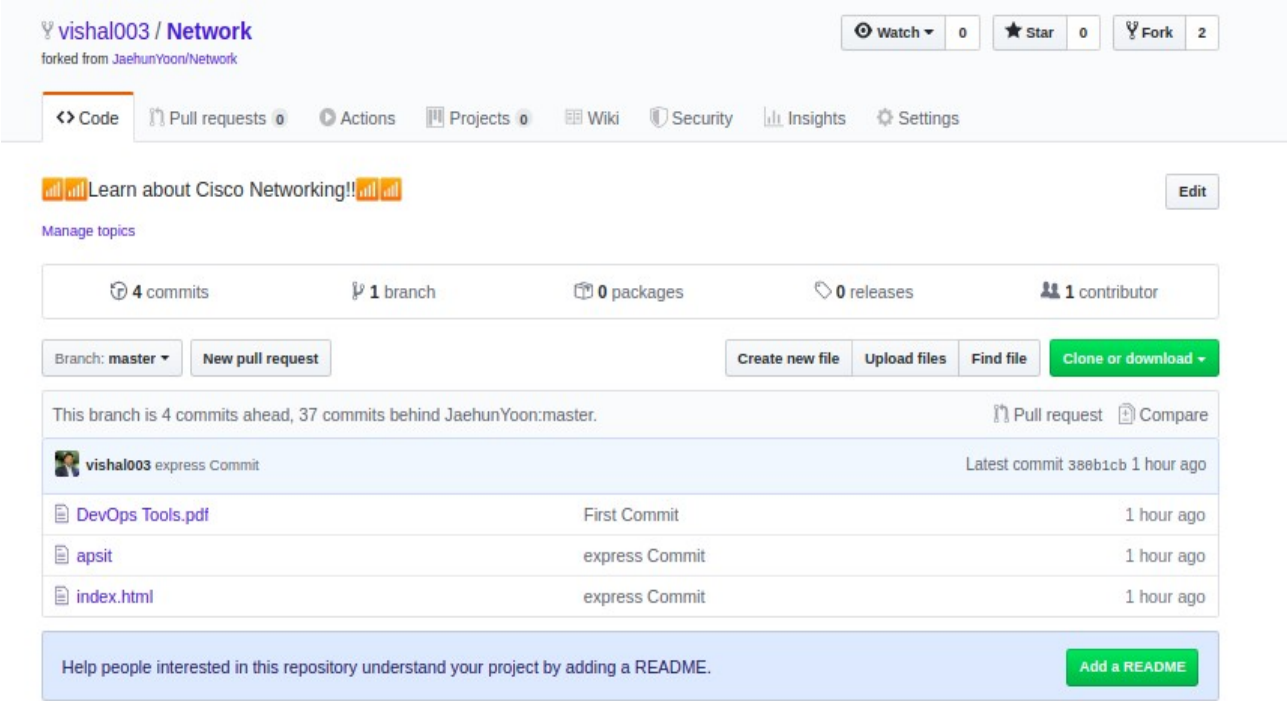
So, to delete origin rm origin command is used

vishal@vishal:~/git-dvcs/git-demo-project$ git remote rm origin

To push the local repository to remote github following command is used

```
vishal@vishal:~/git-dvcs/git-demo-project$ git push --force origin master
Username for 'https://github.com': vishal003
Password for 'https://vishal003@github.com':
Counting objects: 11, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (8/8), done.
Writing objects: 100% (11/11), 4.35 MiB | 1.69 MiB/s, done.
Total 11 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/vishal003/Network.git
 + db17c9f...380b1cb master -> master (forced update)
```

Now you can check the github for updated contents.



B) Pull Changes

Pull command is used to download the remote updated repository into local one. The command for download is:

vishal@vishal:~/git-dvcs/git-demo-project$ git pull

```
vishal@vishal:~/git-dvcs/git-demo-project$ git pull origin master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/vishal003/Network
 * branch            master     -> FETCH_HEAD
   380b1cb..5fd3e3d  master     -> origin/master
Updating 380b1cb..5fd3e3d
Fast-forward
 index.html | 1 +
 1 file changed, 1 insertion(+)
```

```
vishal@vishal:~/git-dvcs/git-demo-project$ git log --oneline origin/master
5fd3e3d (HEAD -> master, origin/master) Update index.html
380b1cb express Commit
b52ffc8 express Commit
be24cf8 express Commit
e1f8faa First Commit
```

Now you can see the changes in local repository using git log.

C) Fetch

Suppose you have a file in github and you have changes that.

Now we use fetch command to fetch the changes, which will show you both the files like original and changed in local repository.

Here fetch will not show you like updated changes file as like push. So use merge command to merge the changes so use following command for merge.

vishal@vishal:~/git-dvcs/git-demo-project$ git merge origin/master