# Introduction to Git: Remotes

## Agenda:

1. To understand the concept of remote in Git and working with GitHub.
2. Hands-on :
    - creating a GitHub account
    - cloning remote branch
    - fetching and merging from remote branch
    - pulling from remote branch
    - pushing into remote branch
    - adding and removing remote branch.

## Activity:

Here you are going to learn Git and GitHub using the following two assignments:

**Assignment 01:** Clone your local git repository from your remote GitHub repository. Here you are the owner of the project.

**Assignment 02:** Fork MLApp project from others github repository to your remote github repository. Then clone your local git repository from your remote GitHub repository. Here you are not the owner of the project, but want to contribute to the project.

### Remote Git Repository

When you are working in a Git repository, a remote is simply a repository in another location from where you are currently working. To be able to collaborate on any Git project, you need to know how to manage your remote repositories. A remote repository can be:

- A repository in another directory in the same laptop.
- A repository in your group member's laptop.
- A centralized server with bare git repository.
- A web-based git repository. E.g. GitHub, GitLab, Gerrit, BitBucket etc.

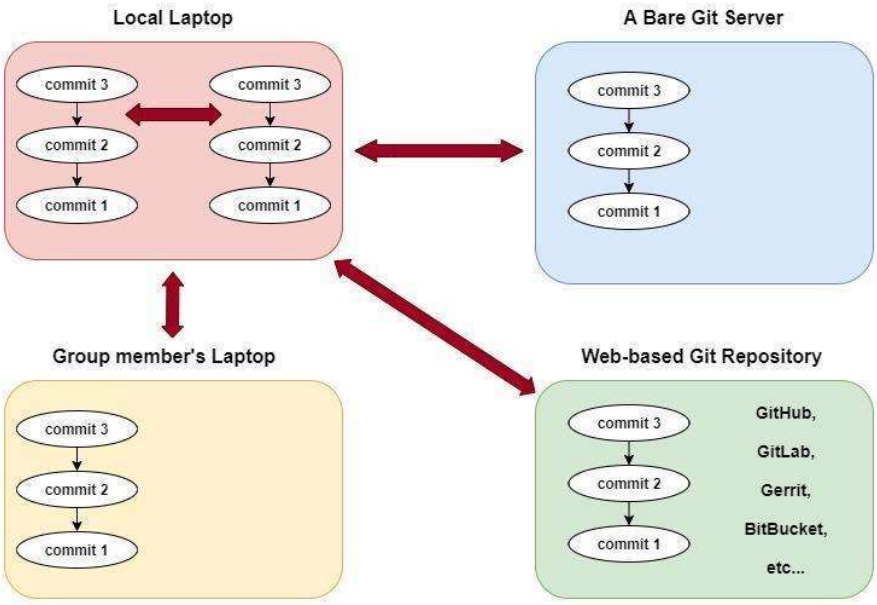


Figure 1

**GitHub**

GitHub is the single largest host for Git repositories, and is the central point of collaboration for millions of developers and projects. While it's not a direct part of the Git open source project, there's a good chance that you'll want or need to interact with GitHub at some point while using Git professionally.

## Assignment 01: Clone your local git repository from your remote GitHub repository.

## 1. GitHub - Account Setup and Configuration:

### Step 1 - Setup a free user account in GitHub

Visit https://github.com, choose a username that isn't already taken, provide an email address and a password, and click the big green "Sign up for GitHub" button.

Connecting to GitHub with SSH:

You can connect to GitHub without supplying your username or password by using SSH keys: https://help.github.com/en/github/authenticating-to-github/adding-a-new-ssh-key-to-your-github-account

**Step 2 - Verify your email address**

GitHub will send you an email to verify the address you provided.

**Step 3 - Create a Git repository in GitHub**

1. To create a new GitHub repository, click on the 'New' button on the left side of the home page and near the Repositories Tab.
2. Name the repository as 'MLApp'.
3. Select the repository type as public or private.

   Note: You have to pay to get private GitHub repository
4. Select the 'Initialize this repository with a README' option.
5. Click on the 'Create repository' button.

This will create a repository with a single commit. GitHub made this first commit to add the first file i.e. README.md file.

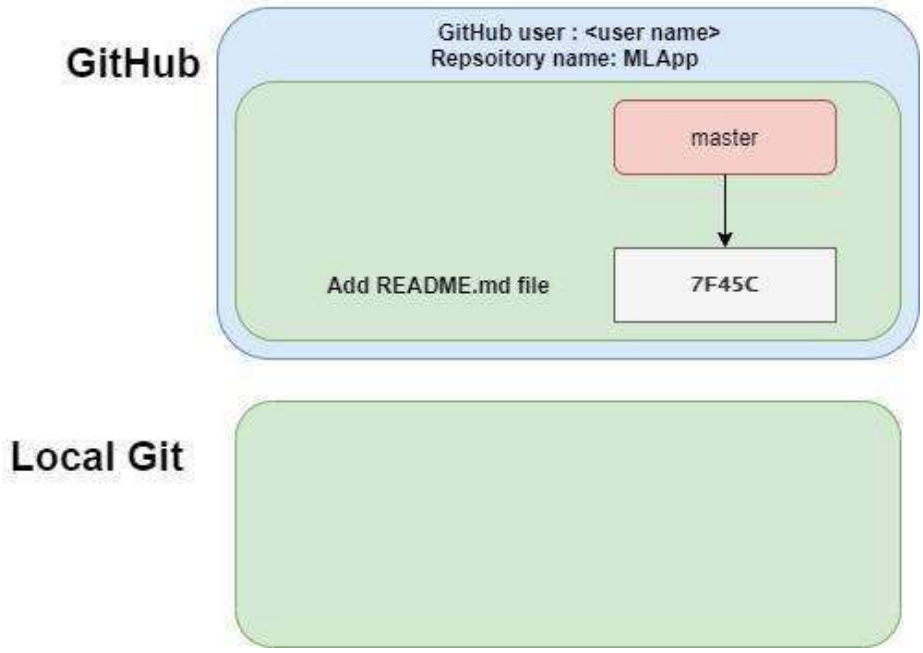


Figure 2

## 2. Retrieve the repository to local system:

**Step 1 -** Click on the 'Clone or download' button and copy the url present under the 'Clone with HTTPS' option.

**Step 2 -** Clone the repository using git clone command.

git clone <<URL>>

E.g.:

git clone https://github.com/<<User>>/MLApp.git

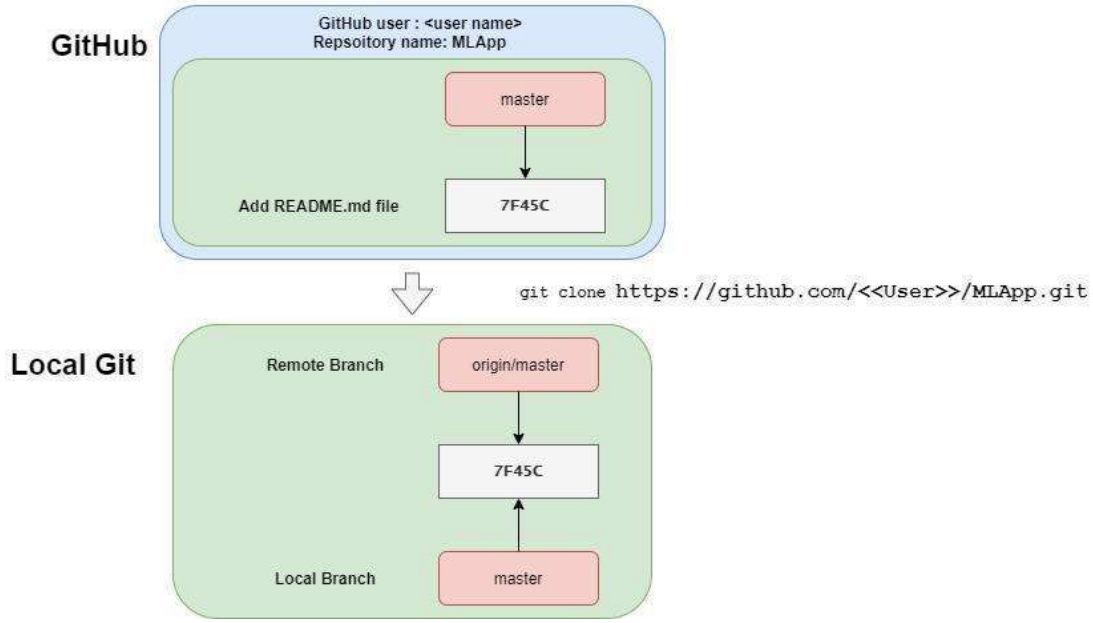


Figure 3

When you clone a repository with git clone, it automatically creates a remote connection called origin pointing back to the cloned repository.

**Step 3 - Verify the commit graph**

git log --all --oneline --graph

Locally we only have the master branch which points to our latest commit and the symbolic HEAD pointer which is pointing towards the latest commit. In addition to the local pointer, we have origin/master and origin/HEAD. Origin, i.e. the default name Git gives to the server you cloned from.

Origin/master is a specialized branch and is called a remote tracking branch. The job of the remote tracking branch is to tell us what the master branch looks like at origin. The remote tracking branch (origin/master) will tell us whether the local master branch and the master branch in GitHub are pointing to the same commit or not.

## 3 Git remotes:

git remote

This command lists the remote connections you have to other repositories.

git remote -v

-v option displays the URL of each connection.

## 4 Modify the remote branch through GitHub:

**Step 1 -** Go to the GitHub web service (https://github.com) and go to the MLApp project repository and edit the README.md file

**Step 2 -** Commit the changes through GitHub itself.

## 5 Verify the commit graph through GitHub and your local system:

The latest commits will be visible in the GitHub server, but your local branch is not aware of the changes.

## 6 Fetching from Remotes:

git fetch <<short name of the remote branch>>

E.g.:

git fetch origin

It fetches any new work that has been pushed to that remote server since you cloned (or last fetched from) it. It's important to note that the git fetch command only downloads the data to your local repository — it doesn't automatically merge it with any of your work or modify what you're currently working on. You have to merge it manually into your work when you're ready.

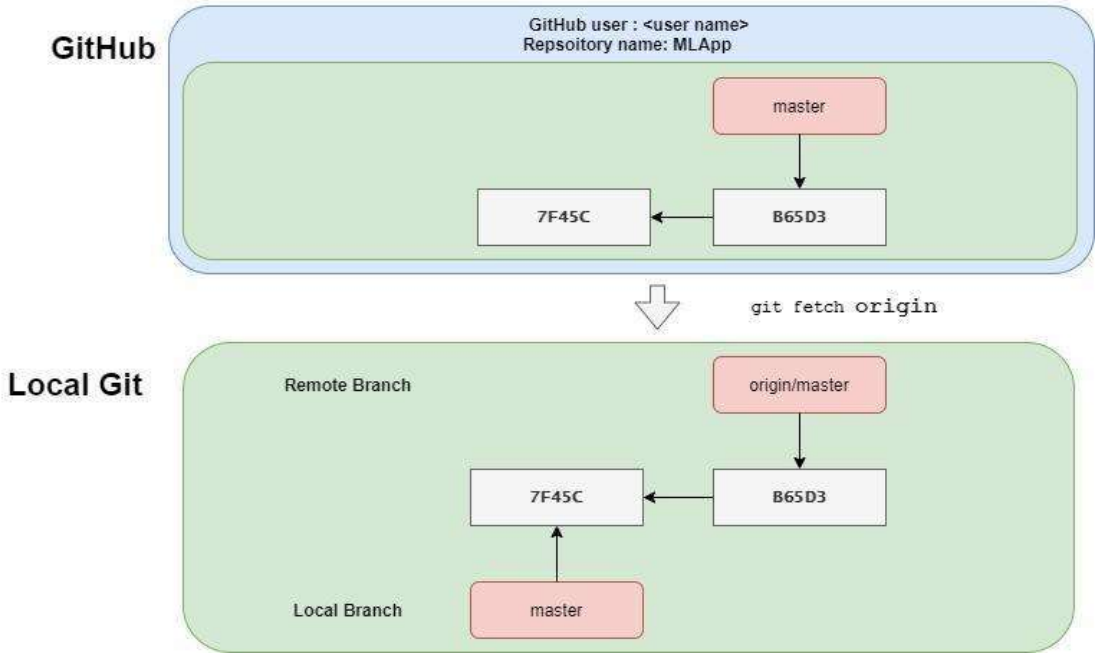


Figure 4

## 7 Merge the local master branch:

git merge origin

Merge the local master branch with the origin

# 8 Pulling from Remotes (Optional):

git pull <<shortname of the remote branch>>

E.g.:

git pull origin

Use the git pull command to automatically fetch and then merge that remote branch into your current branch. i.e. **git pull = git fetch + git merge.**

Note: To have better control it is recommended to using git fetch and git merge

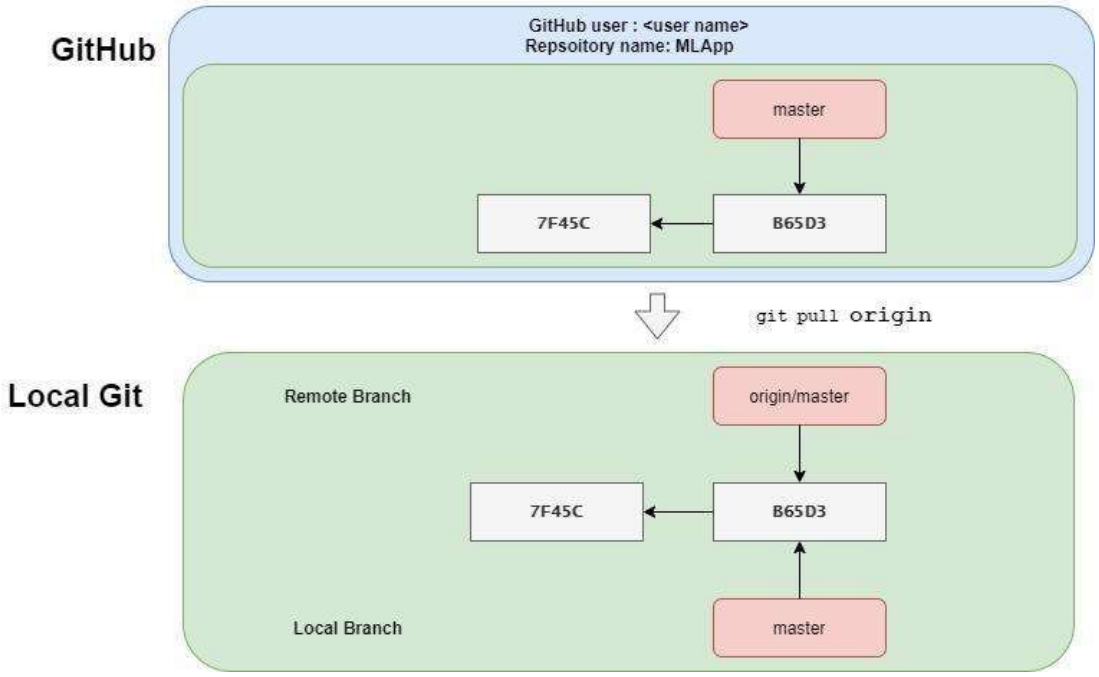


Figure 5

# 9 Make changes in the local master branch:

**Step 1** - Edit the README.md file and commit the changes to the local master branch.

**Step 2** - git commit -a -m "Modifying the README.md file."

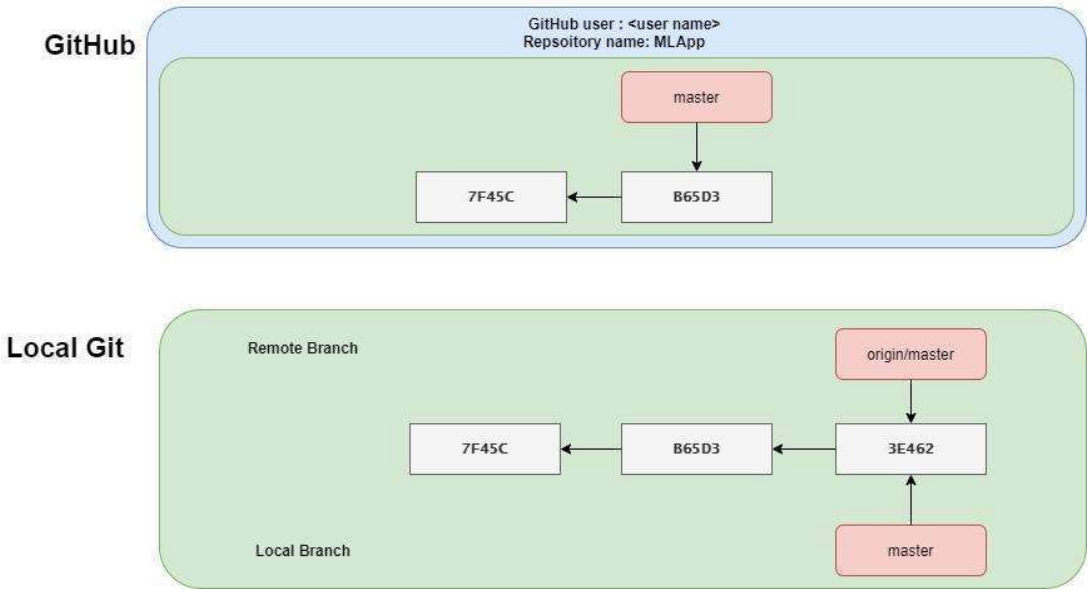


Figure 6

## 10 Pushing to Remotes:

git push <<shortname of the remote branch>> <<Branch Name> >

E.g.:

git push origin master

When you have your project at a point that you want to share, you have to push it upstream using this command. This command works only if you cloned from a server to which you have write access and if nobody has pushed in the meantime. If you and someone else clone at the same time and they push upstream and then you push upstream, your push will rightly be rejected. You'll have to fetch their work first and incorporate it into yours before you'll be allowed to push.

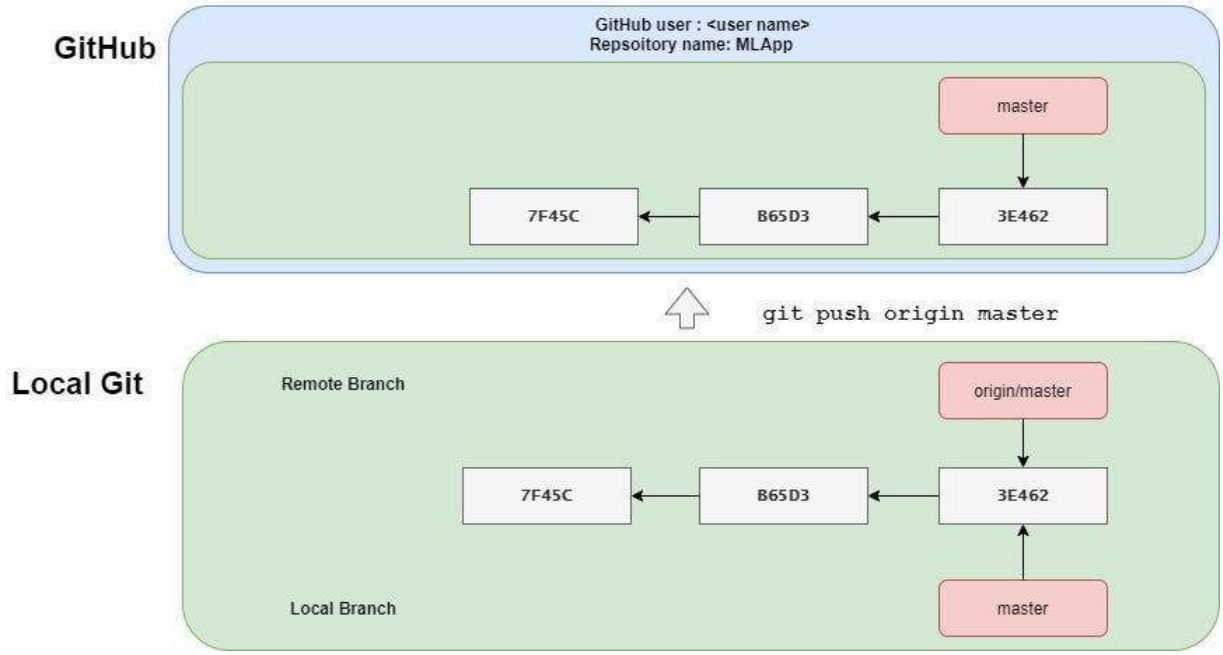


Figure 7

**Scenario:** Usually when we work as part of a team, we might need to work on remote repositories for which we won't be the owner and we might not even have write permissions. So we won't be able to directly make changes in the repository. Given below are the steps to follow in these scenarios:-

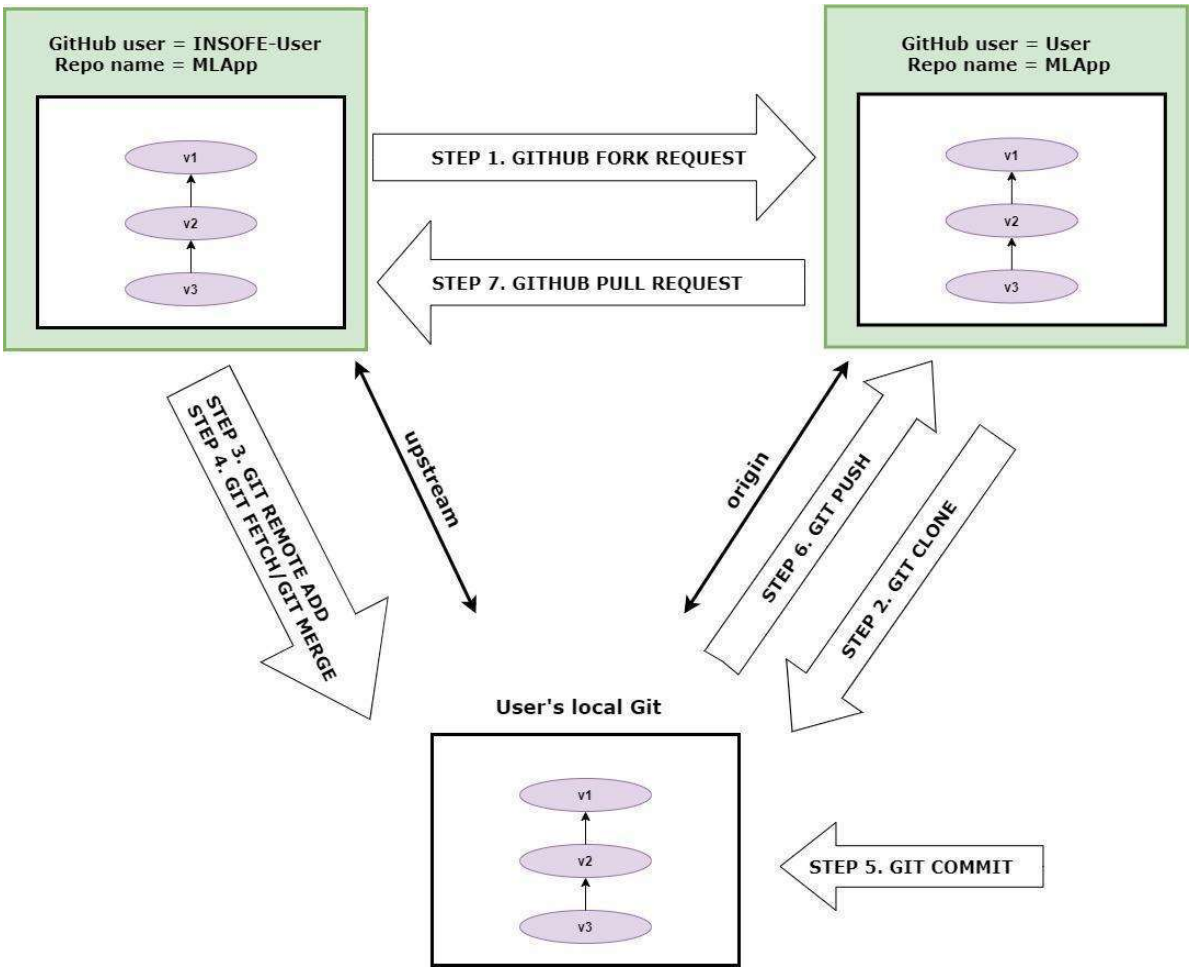


Figure 8

**Assignment 02:** Fork 'MLApp' project from INSOFE github repository to your remote github repository. Then clone your local git repository from your remote GitHub repository. Here you are not the owner of the project, but you want to contribute to the project.

**Step 1:** Fork the Git repository from GitHub:

Search the GitHub for INSOFE-User1/MLApp public repository and fork it into your GitHub account. To fork a repository you need to click on the Fork button at the top right corner.

A fork will create a copy of the repository in your GitHub account and you will be the owner of that copy.

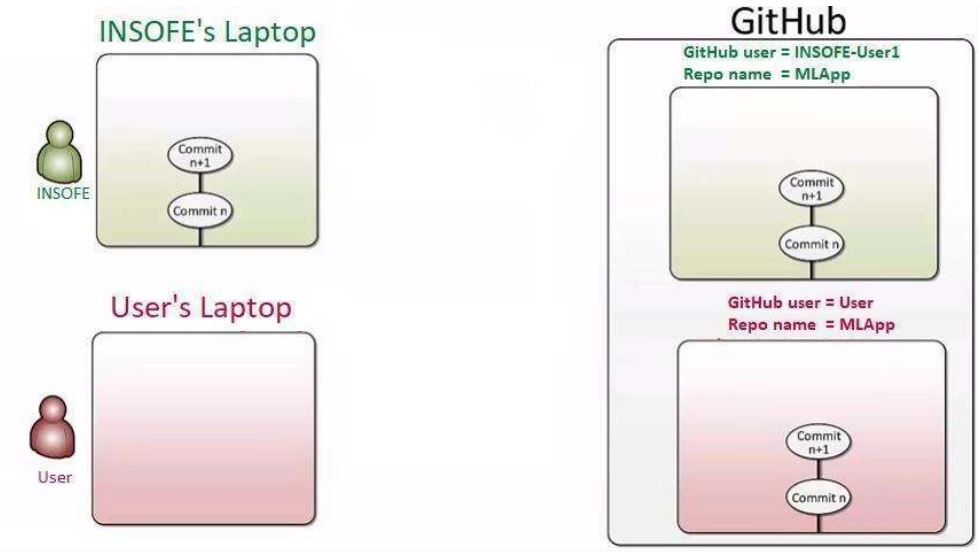


Figure 9

**Step 2:** Clone the repository to your local Git.

git clone https://github.com/<<User>>/MLApp.git

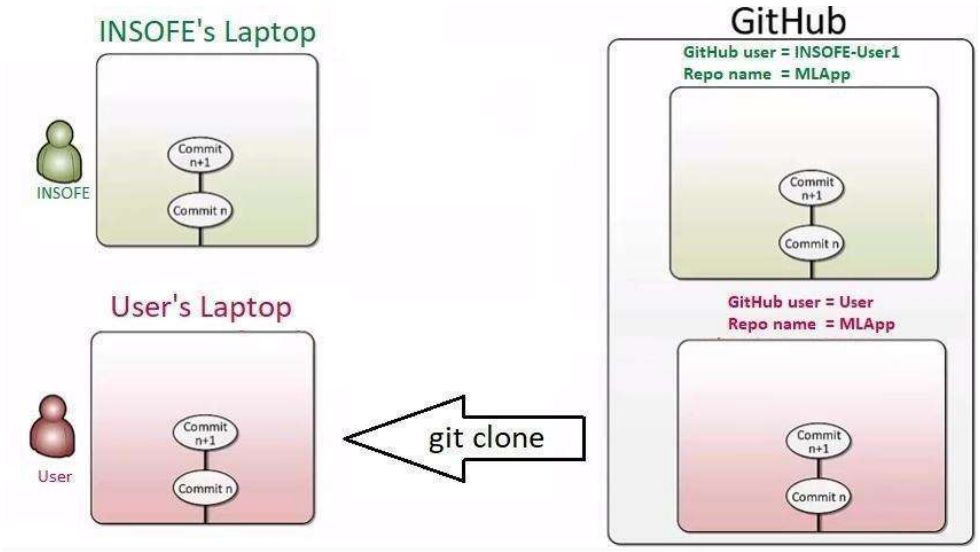


Figure 10

**Step 3:** Add the original remote repository into your branch using the git remote add command. You can call it the 'upstream' remote branch.

git remote add <<shortname of the remote branch>> <<URL>>

E.g.:

```
git remote add upstream https://github.com/INSOFE-User1/MLApp.git

git remote –v
```

git remote -v will show us two remotes. The origin is pointing to the user's fork and the upstream is pointing to INSOFE's repository.

NOTE: Similarly, you can use the Git remove command to remove a remote Git repository.

```
git remote remove <<shortname of the remote branch>>
```

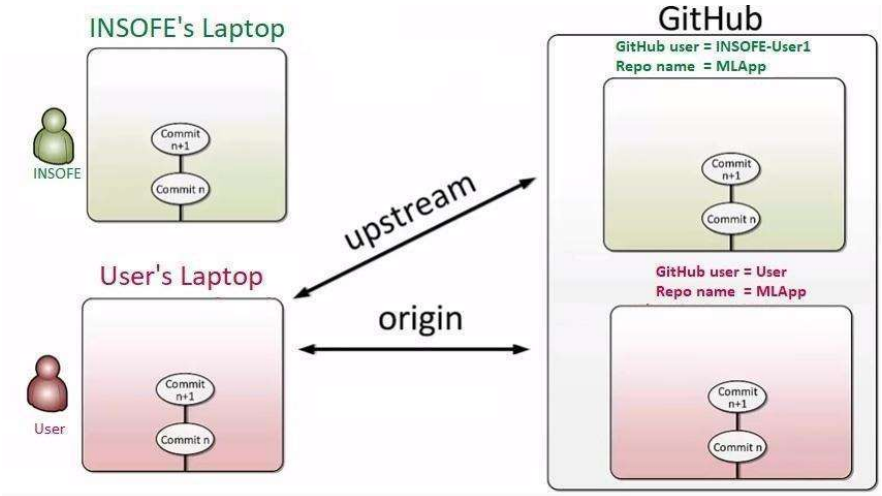


Figure 11

**Step 4:** Sync the upstream remote branch with your local branch by using git fetch command and git merge command.

Note: This makes sense only if other's GitHub repository has new commits.

```
git fetch upstream

git log --all --oneline --graph

git merge upstream/master

git log --all --oneline --graph
```

**Step 5:** Work on the local branch of your remote repository and commit changes in the local branch.

```
git add *
```

git commit –m “Committing the changes”

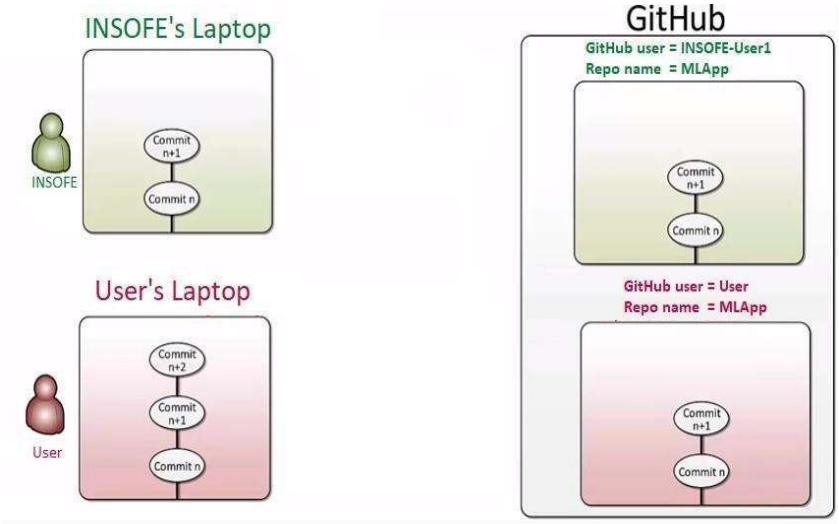


Figure 12

**Step 6:** Push all the changes into your GitHub repository using the git push command.

git push origin master

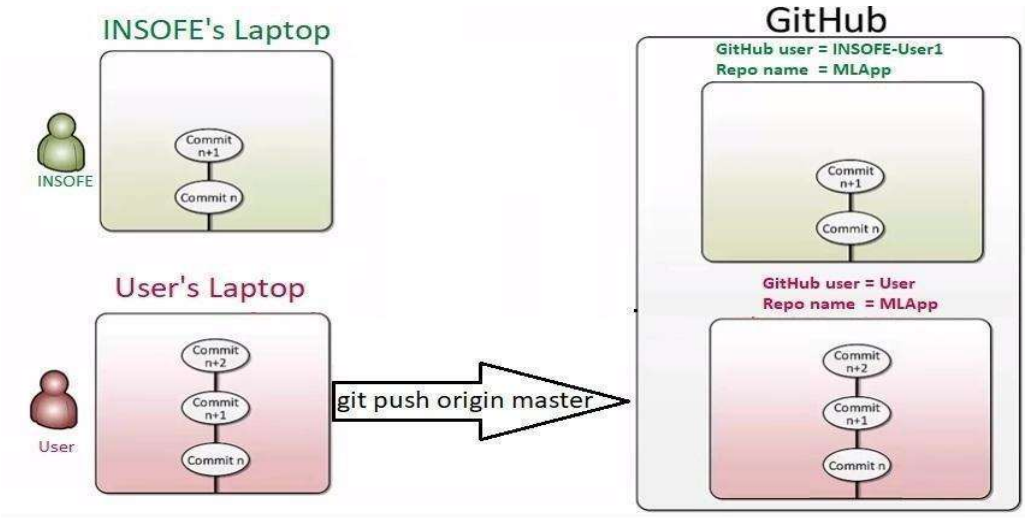


Figure 13

**Step 7:** From your GitHub account, raise a ‘Pull Request’ to accommodate your changes in the other’s (i.e. in this case INSOFE_user1) GitHub repository. Insofe_user1 will review your changes, if acceptable will complete the ‘Pull Request’ by accepting your new changes.

## References:


1. Git-Book: https://git-scm.com/book/en/v2
2. https://www.youtube.com/watch?v=Gg4bLk8cGNo