CS352: Software Engineering II

Lab 1

Using GIT

Cairo University, Faculty of Computers and Information

Contents

Version Control System (VCS)

Revision control, also known as version control and source control (and an aspect of software configuration management), is the management of changes to documents, computer programs, large web sites, and other collections of information. Changes are usually identified by a number or letter code, termed the "revision number", "revision level", or simply "revision". For example, an initial set of files is "revision 1". When the first change is made, the resulting set is "revision 2", and so on. Each revision is associated with a timestamp and the person making the change. Revisions can be compared, restored, and with some types of files, merged.

Version control Concepts and terminologies

The following terms has a special meaning with respect to Version control systems:

- Repository (repo): The database storing the files.
- Server: The computer storing the repo.
- Client: The computer connecting to the repo.
- Working Set/Working Copy: Your local directory of files, where you make changes.
- Staging: to stage your files means getting your files ready (i.e. to specify which files you want to (save / keep track of) the changes that you have made to it). You don't always want to stage all the files that you have made changes to.
- Commit: saving the changes that was made to the files that you staged to Git history

The following Actions could be executed against version control systems:

- Add: Put a file into the repo for the first time, i.e. begin tracking it with Version Control.
- Revision: What version a file is on (v1, v2, v3, etc.).
- Head: The latest revision in the repo.
- Check out: Download a file from the repo.
- Check in: Upload a file to the repository (if it has changed). The file gets a new revision number, and people can “check out” the latest one.
- Check-in Message: A short message describing what was changed.
- Change-log/History: A list of changes made to a file since it was created.
- Update/Sync: Synchronize your files with the latest from the repository. This lets you grab the latest revisions of all files.
- Revert: Throw away your local changes and reload the latest version from the repository.

Git as a Reversion control System

Git is an open-source code management tool; it’s one of the fastest revision controls and it’s easy to use. The following steps will be used to setup a Git repository, add some files to it.

CS352: Software Engineering II

Lab 1

Using GIT

Cairo University, Faculty of Computers and Information

1- Setup Git you could go to the following link to download the latest Git software that is suitable for your system. http://git-scm.com/downloads

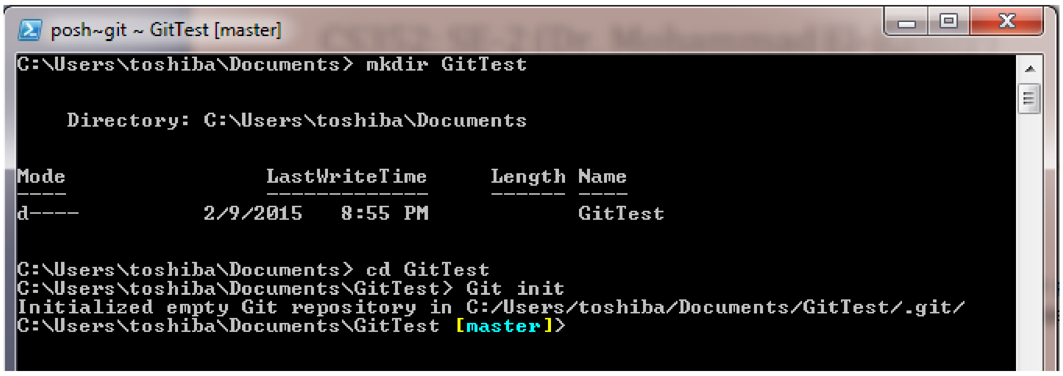
2- you can use this link https://windows.github.com/ to setup Git for windows(console + GUI).

Practice 1

Please Follow the Following Steps: During the following lab practice we will go through java application project and manage that repository under Git revision control as well as pushing this project to Github repository.

1. Using your Java IDE; create a new project name SocialNW this project will contain a number of classes which represent your Project. (we can test now with simple java project that take number from the user and check if the number is prime or not)
2. Create an empty directory on any path in your computer and open git command to initialize a repository inside it with

```
Git init.
```



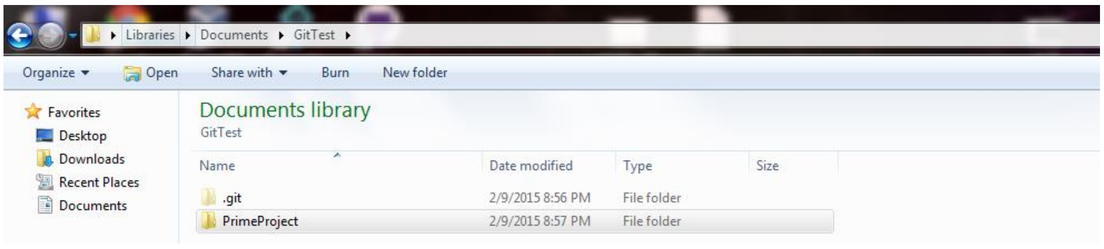
```
C:\Users\toshiba\Documents> mkdir GitTest


    Directory: C:\Users\toshiba\Documents


Mode                LastWriteTime     Length Name
----                -------------     ------ ----
d----          2/9/2015   8:55 PM            GitTest


C:\Users\toshiba\Documents> cd GitTest
C:\Users\toshiba\Documents\GitTest> Git init
Initialized empty Git repository in C:/Users/toshiba/Documents/GitTest/.git/
C:\Users\toshiba\Documents\GitTest [master]>
```

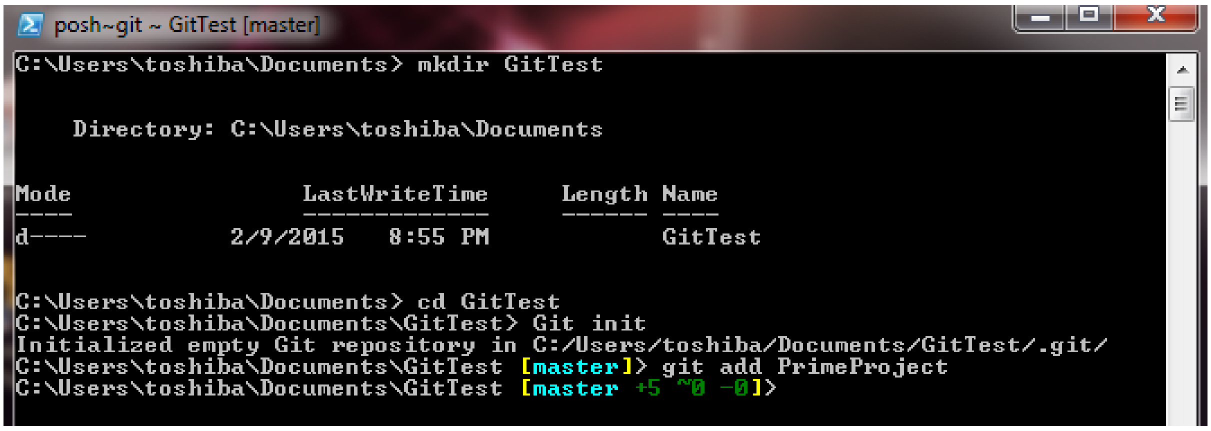
3. Copy your project files into the repository/directory created in the previous step.



Name	Date modified	Type	Size
.git	2/9/2015 8:56 PM	File folder	
PrimeProject	2/9/2015 8:57 PM	File folder	

4. To add these files to Git repository so that Git will track the changes happens to these files/projects you should add these files/projects to the repository through using Git Add <file or directory>

```
Git Add PrimeProject
```



posh~git ~ GitTest [master]

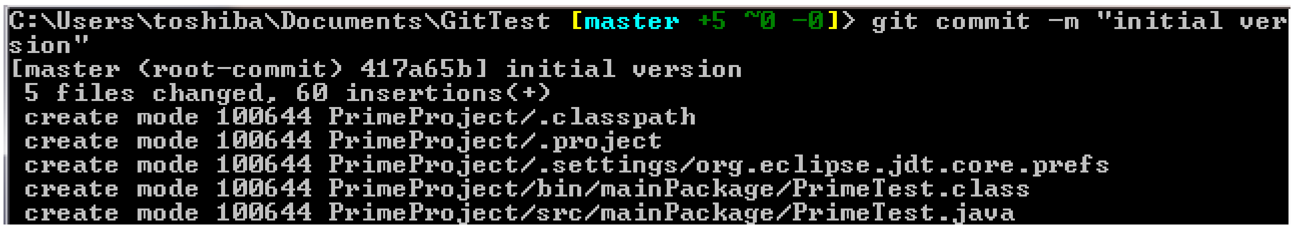
```
C:\Users\toshiba\Documents> mkdir GitTest


    Directory: C:\Users\toshiba\Documents


Mode                LastWriteTime     Length Name
----                -------------     ------ ----
d----          2/9/2015   8:55 PM            GitTest


C:\Users\toshiba\Documents> cd GitTest
C:\Users\toshiba\Documents\GitTest> Git init
Initialized empty Git repository in C:/Users/toshiba/Documents/GitTest/.git/
C:\Users\toshiba\Documents\GitTest [master]> git add PrimeProject
C:\Users\toshiba\Documents\GitTest [master +5 ~0 -0]>
```

5. To save a snapshot of your project at the current stage you could do that through Git Commit command. This command takes a description message to tag or identify the changes happens to that snapshot.



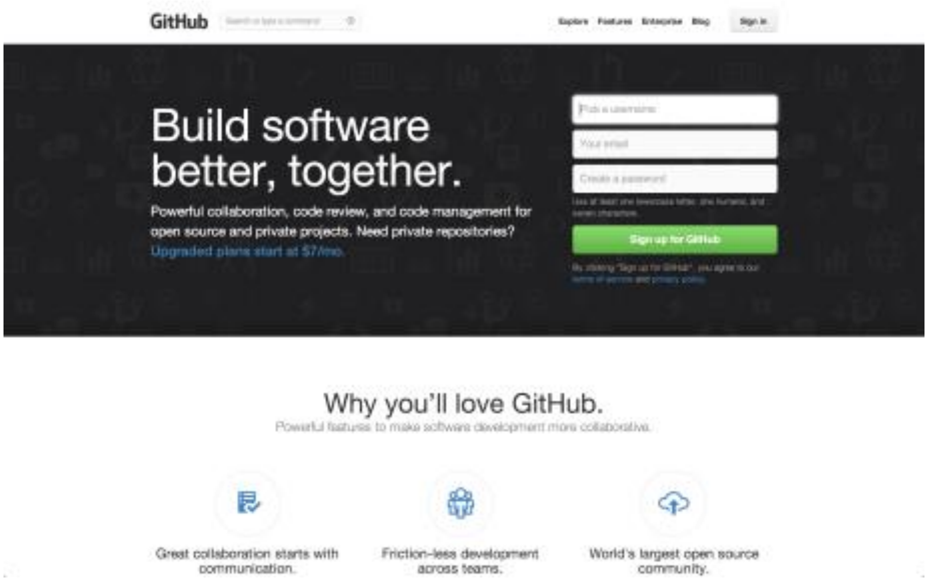
```
C:\Users\toshiba\Documents\GitTest [master +5 ~0 -0]> git commit -m "initial ver
sion"
[master (root-commit) 417a65b] initial version
 5 files changed, 60 insertions(+)
 create mode 100644 PrimeProject/.classpath
 create mode 100644 PrimeProject/.project
 create mode 100644 PrimeProject/.settings/org.eclipse.jdt.core.prefs
 create mode 100644 PrimeProject/bin/mainPackage/PrimeTest.class
 create mode 100644 PrimeProject/src/mainPackage/PrimeTest.java
```

6. You could use Git Status command to check the status of the repository at any certain point of time.

```
Git status
```

7. remote command is used to link your local version of the repository to a remote repository hosted in a remote server for example GitHub repository.
8. Before you link your local and remote repository you have to create a remote/online repository first the following steps will go through creating an online repository on GitHub:

a. Go to https://github.com/
b. Fill-in your information User name, Email and Password then Click SignUp For GitHub.



GitHub

Explore Features Enterprise Blog Sign in

Build software better, together.

Powerful collaboration, code review, and code management for open source and private projects. Need private repositories? Upgraded plans start at $7/mo.

Sign up for GitHub

Why you'll love GitHub.

Powerful features to make software development more collaborative.

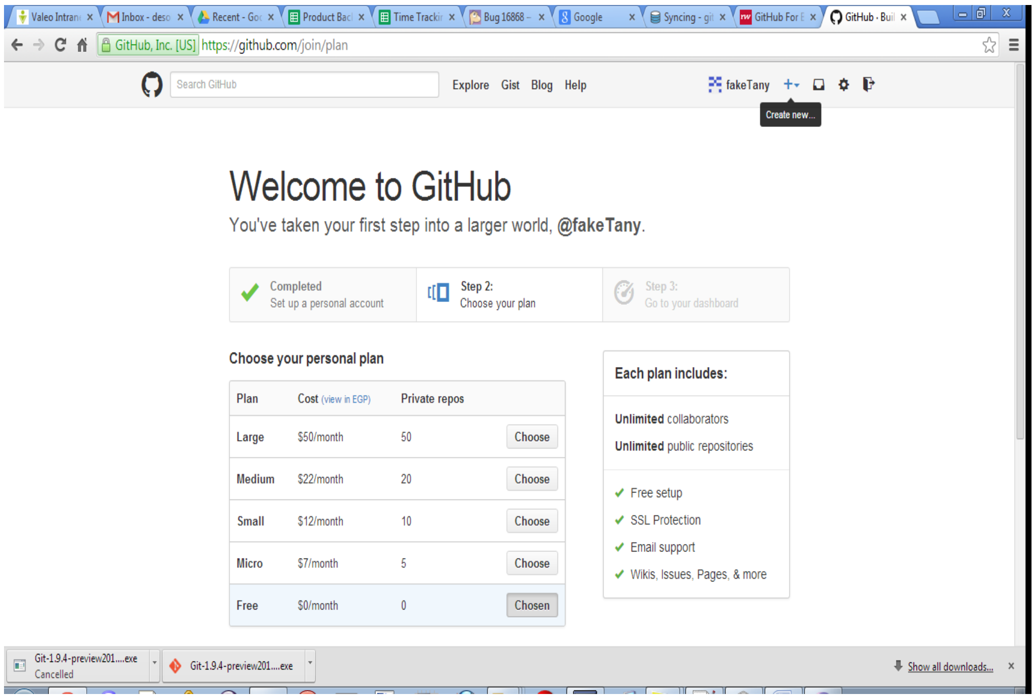
Great collaboration starts with communication.

Friction-less development across teams.

World's largest open source community.

c. Next Step to customize your repository Github provide a free repository for public projects, you have to pay to get a private repo. A side note, BitBucket is another remote repo host provide up to 4 users with a private repo.

d. To create a new online repository u could do that from the plus icon from top right create a repository.



https://github.com/join/plan

Explore Gist Blog Help fakeTany

Create new...

Welcome to GitHub

You've taken your first step into a larger world, @fakeTany.

Completed: Set up a personal account

Step 2: Choose your plan

Step 3: Go to your dashboard

Choose your personal plan

Plan	Cost (view in EGP)	Private repos	
Large	$50/month	50	Choose
Medium	$22/month	20	Choose
Small	$12/month	10	Choose
Micro	$7/month	5	Choose
Free	$0/month	0	Chosen

Each plan includes:

Unlimited collaborators

Unlimited public repositories

✓ Free setup

✓ SSL Protection

✓ Email support

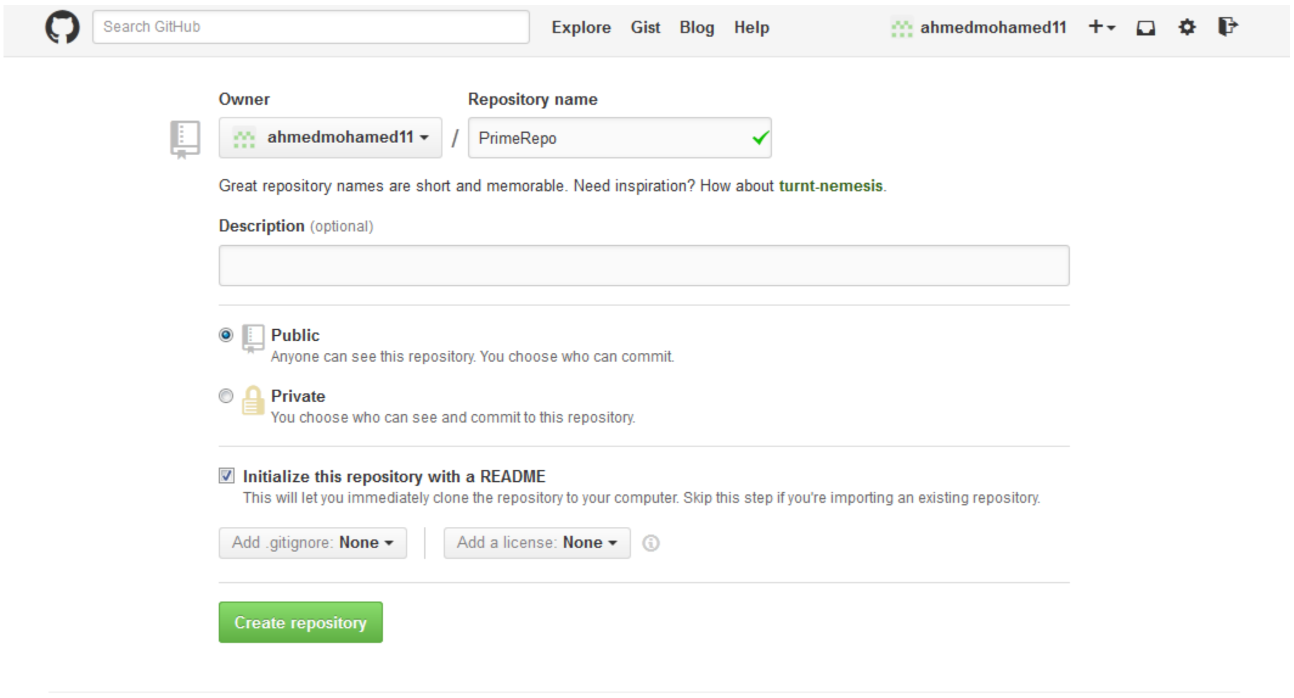
✓ Wikis, Issues, Pages, & more

Git-1.9.4-preview201....exe Cancelled

Git-1.9.4-preview201....exe

Show all downloads...

e. Give a name to a repo as well as a description for it and choose public and check add a readme file to that repository, this read me file is act as the home page for this repository it should contains project specific information and how to use it.



Search GitHub | Explore | Gist | Blog | Help | ahmedmohamed11

Owner: ahmedmohamed11 / Repository name: PrimeRepo

Great repository names are short and memorable. Need inspiration? How about turnt-nemesis.

Description (optional)

[✓ Public] Anyone can see this repository. You choose who can commit.

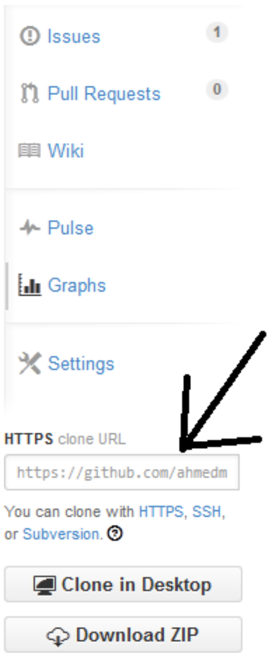
Private: You choose who can see and commit to this repository.

[✓ Initialize this repository with a README]
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None | Add a license: None

Create repository

f. Now you have created a remote repository for your project please copy the link for that repo from the left hand side part titled with HTTPS Clone URL. Something like `https://github.com/ahmedmohamed11/PrimeRepo.git`



g. Now to link the local and remote repositories we use the following Git command

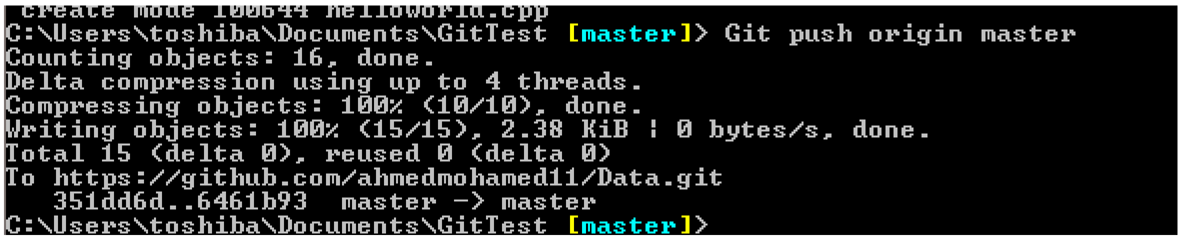
```
git remote add origin https://github.com/ahmedmohamed11/PrimeRepo.git
```

h. To push/save your local changes to the remote repository we could use Git push command

```
Git push origin master
```

i. In push/save step you may be to get all the online repository(by pull to local repository) to avoid conflict

```
Git pull origin master
```



```
 create mode 100644 helloworld.cpp
C:\Users\toshiba\Documents\GitTest [master]> Git push origin master
Counting objects: 16, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (10/10), done.
Writing objects: 100% (15/15), 2.38 KiB | 0 bytes/s, done.
Total 15 (delta 0), reused 0 (delta 0)
To https://github.com/ahmedmohamed11/Data.git
   351dd6d..6461b93  master -> master
C:\Users\toshiba\Documents\GitTest [master]>
```

j. Also We could clone an already existed repository from remote host to our local machine through using Git clone <remote repo git url> inside an new empty folder.

```
Git clone https://github.com/ahmedmohamed11/PrimeRepo.giT
```

Change Link between local repository and online Repository

1- To list your repository

```
git remote -v
```

2- To delete your origin repository

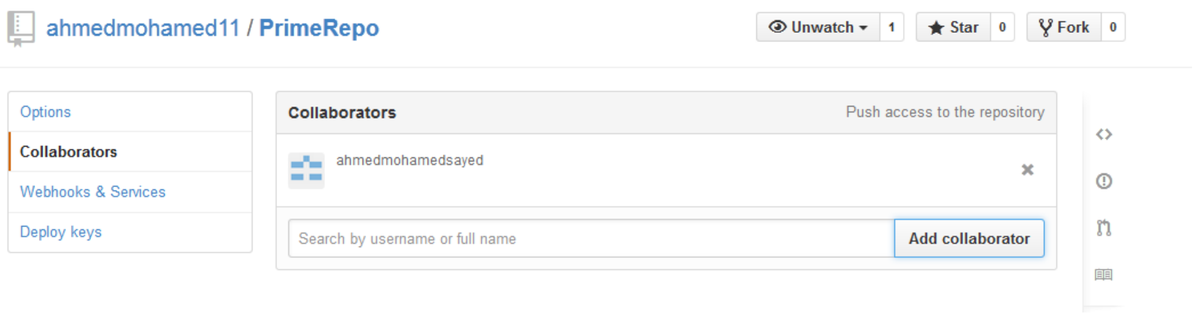
```
git remote rm origin
```

3- To Add Now to link the local and remote repositories we use the following Git command

```
git remote add origin https://github.com/ahmedmohamed11/PrimeRepo.git
```

Collaborators

You can add you team to your repository so can everyone in your team able to push to the repository


ahmedmohamed11 / PrimeRepo

Branches

The main branch of a repository is usually named master, and represents a relatively stable version of the project you're working on. So far, all the changes you've made have been on the master branch.

If you're making an app or website, for example, you might have a bunch of different features or ideas in progress at any given time – some of which are ready to go, and others which are not. For this reason, master exists as a central point to fold other *branches* of work into.

Tip: If you really want to commit straight to master, the app won't stop you. However, this doesn't lend itself well to working collaboratively, and your changes will be harder to track and maintain as the project gets larger. A branch is just a separate version of the code.

CS352: Software Engineering II

Lab 1

Using GIT

Cairo University, Faculty of Computers and Information

1- To list your branches

```
git branch
```

2- Create new branch

```
git branch myBranch
```

3- Change to your branch(to work on your branch, not the master branch)

```
git checkout myBranch
```

4- At this step you can modify your code & commit your code to your branch by the following command

```
git add filename
git commit -m "message"
```

5- Now we need to merge master with my branch to ensure that is no conflict (the conflict may be happen if the master is different from my branch which happen if the master had one or more commit after I create my branch. The conflict must be handle manually by editing your code)

```
Git merge master
```

6- To push changes you made in your local branch to your remote branch

```
git push origin myBranch
```

OR - if you finished working on this branch and it is ready to be merged into master

6- Change to the master branch to merge the new branch with the master

```
Git checkout master
```

7-Merge master with the myBranch

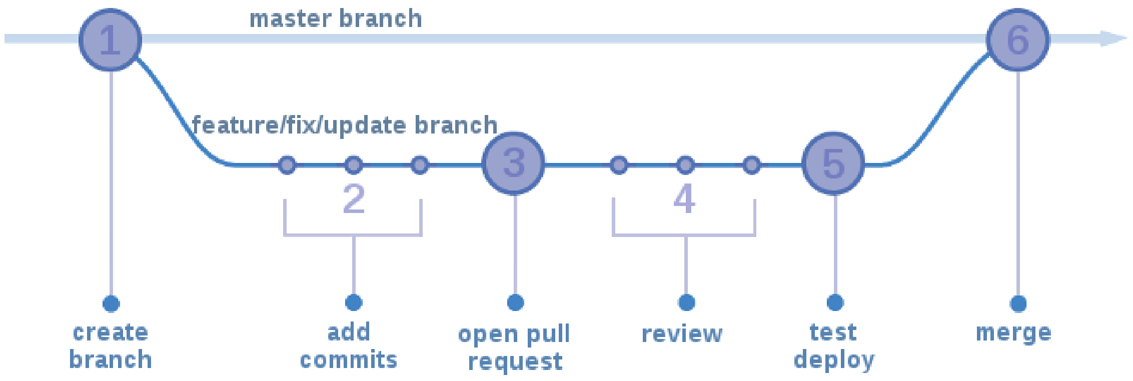
```
Git merge myBranch
```

8- Push your updated code to the master

```
Git push origin master
```

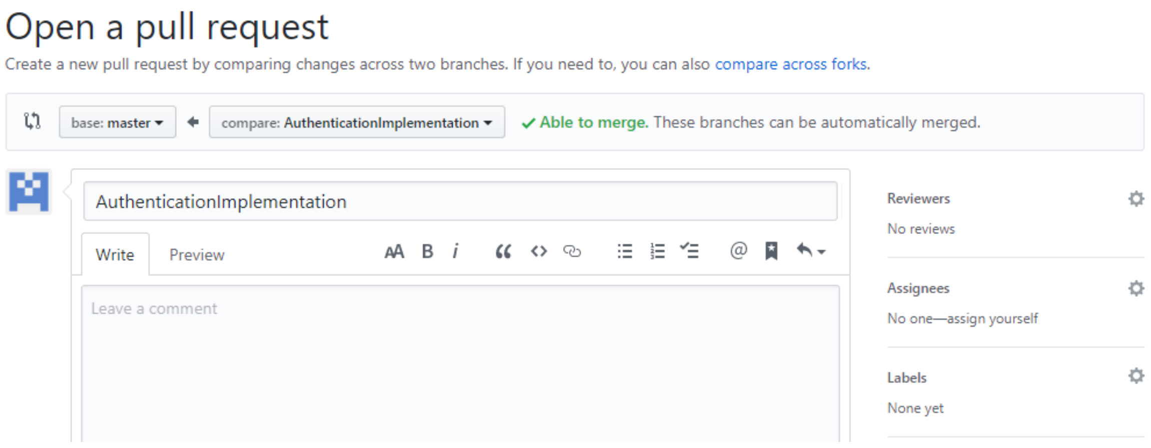
Pull Request Workflow

- The team should be able to pull a clean version of the software. This version should be tested and verified by the whole team
- Usually the software on the master branch is that software version which should be the stable and safe to pull version. Also, it should be the version that contains the correct code
- Therefore, pushing the code to the master branch directly is not encouraged.
- To increase the software reliability, any commit in the history of the master branch should be reviewed by the team and if any developer works on a specific feature, a new branch should be created for this feature specifically and the developer should push that code on this branch and after the feature is finished the developer should open “Pull Request”.
- Pull request is a request issued from the developer who developed a specific feature to the team to review a specific piece of code that represents a specific feature.
- The pull request state that a developer would like to add a new code to the software on the master branch.
- The team should review each pull request and if it seems to the team that the code is correct the team should merge this pull request.
- Merging the pull request will add the code from the branch opened by the pull request author (the developer who developed the feature) to the master branch (or the pull request author could state that the code should be merged on another branch, however, the default destination branch is usually the master branch
- The below figure states the specific steps in that workflow, these steps are



  - Pull the latest working code (on the master branch)
  - Create a branch out of the master branch. This branch is called the feature branch
  - Commit the code on that branch. You may commit any number of commits on the feature branch

- After the developer finish developing the feature and commit the changes on the feature branch, a new pull request should be opened, check the below figure


Open a pull request
Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.
base: master ← compare: AuthenticationImplementation ✓ Able to merge. These branches can be automatically merged.
AuthenticationImplementation

- In this figure, you will find that github interface offers you to request from the team to merge your branch into the master branch, you can assign a reviewer if you would like to do so
- During the review process, the reviewer can add any comments and the pull request author should handle these comments
- After all comments (if any) are handled the reviewer can merge this pull request so that new code will be pushed to the master branch normally

-

References

- [1] http://software-carpentry.org/v4/vc/intro.html
- [2] http://betterexplained.com/articles/a-visual-guide-to-version-control/
- [3] http://www.mountaingoatsoftware.com/agile/scrum/overview
- [4] http://neverstopbuilding.com/integrating-github-with-trello-with-heroku
- [5] http://www.deepfriedbrainproject.com/2010/03/configuration-management-system-quick.html
- [6] https://www.atlassian.com/git/tutorials/using-branches/
- Useful videos
  - https://www.youtube.com/watch?v=1Jd1cBn8tW4
  - https://www.youtube.com/watch?v=tHDJ_CS3rmI