# Lab 02-02 : Git Tutorial

By Nick Wang, Peter Hung, Tim, Tony, Allen last modified on 02/20, 2020.

The objective of the tutorial is to make you understand the basic functionality of Github. Github is very widely use in the managing some works or project, especially huge project which composed of many functions separated by package.

[**Lab 02-02 : Git Tutorial**](#_74gram21yqax) **1**

[Overview](#_spig986p4o7l) 1

[Topic/Activity (TA will go through this step by step)](#_5v1x8vy0upch) 1

[Topic 1: Github Accounts](#_uvo8oqg5eflu) 1

[Topic 2: Basic Commands in Git](#_h1t9fitpgpue) 2

[Assignment Tasks (DIY, either during the class or at home)](#_xhgwvyc67ksw) 3

[Task 1: Clone a Repo](#_wsbu8eqk4sb0) 3

[Task 2: git add, commit, and push](#_gf54ndob8kpv) 3

[Task 2: git branch, git checkout](#_nxii5pcl62pb) 5

[Task 3: Join Slack](#_1pccshllbh7n) 5

[1.4 Check points](#_rdrwuotvgmzi) 5

[Guidelines:](#_dj46zxfyjm7o) 6

[2.1 You Should](#_10aj89q1cugb) 6

[2.2 You Should Not](#_4rjezjw0zio0) 7

## Overview

Estimated Time to Finish: 0.5 hours

After completing this tutorial you should

* understand how to use git

## Topic/Activity (TA will go through this step by step)

### Topic 1: Get a Github Account

1. Please create your own github account, the principle of creating an account  
   創帳號注意事項
   1. You need to verify it in your email; one email can only register one account
   2. Do not use upper case -> It is usually used as a group
   3. You should use and maintain your Github account frequently; people will see how well you can code by taking a look at your Github account.
   4. Let TA to invite you to join github Sensing-Intelligent-System organization.  
      請加入Sensing-Intelligent-System github組織

### Topic 2: Basic Commands in Git

**git clone: clone github repository (從伺服器repo下載到電腦)**

**git status: check the status of your local github (確認檔案狀態)**

**git add: add some files from your working directory (增加已修改檔案)**

**git commit: commit the files has been added to your staging area (註解已修改   
 的檔案，說明此次新增內容)**

**git push: push the commited files from your local repository to the original github repo (上傳增加的檔案到github伺服器)**

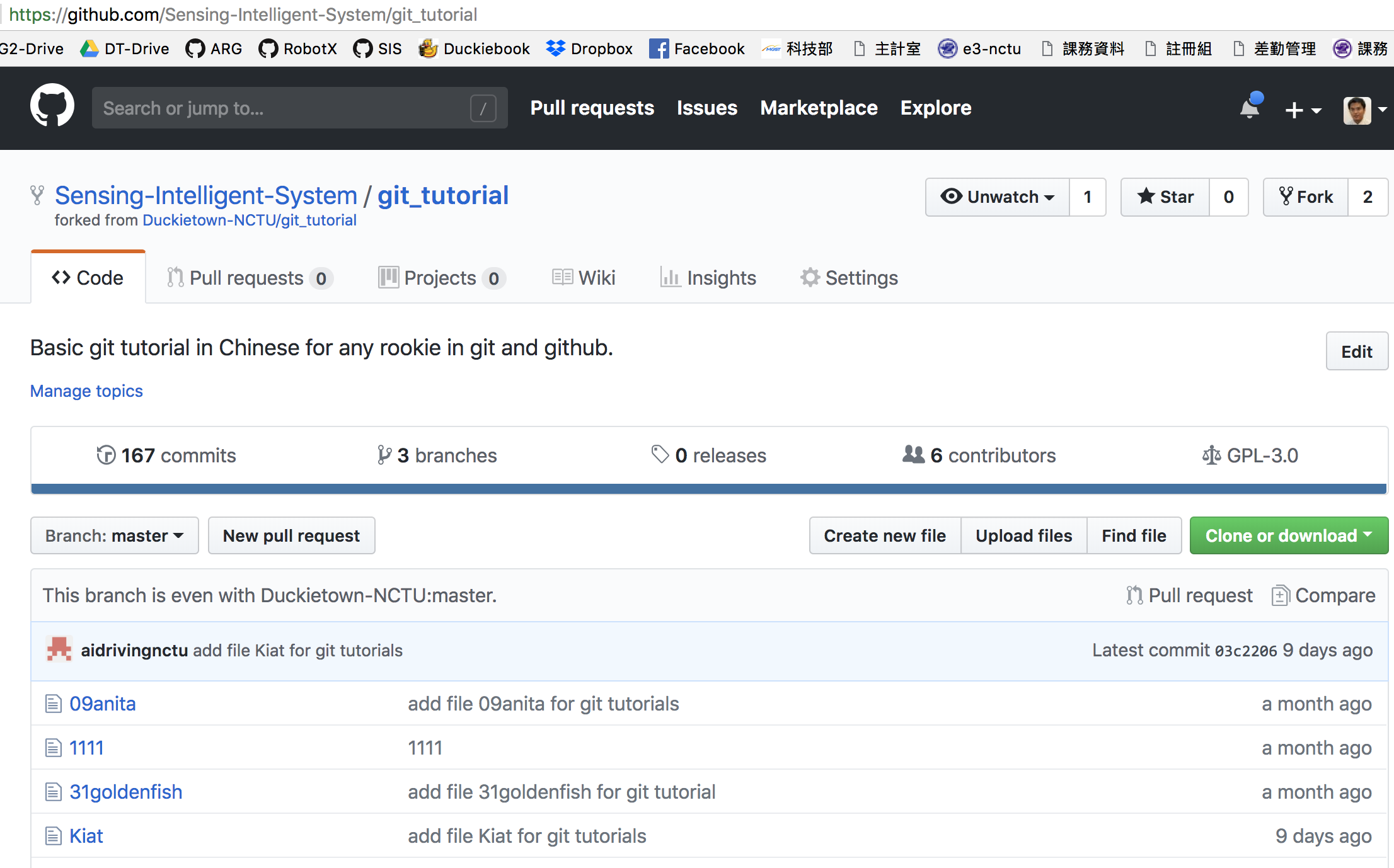
**git pull: update your local repository from the original repository (更新伺 服器repo到電腦端)**

**git branch: list all branches (列出所有分支)**

**git checkout: switch to specified branch (切換到指定分支)**

## Assignment Tasks (DIY, either during the class or at home)

### Task 1: Clone a Repo

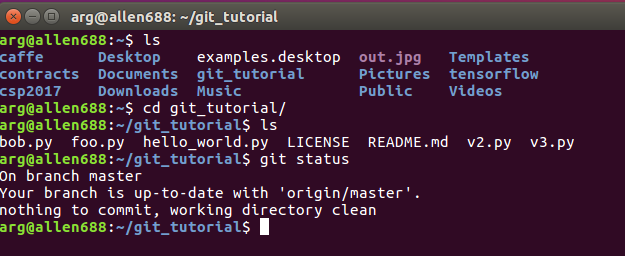
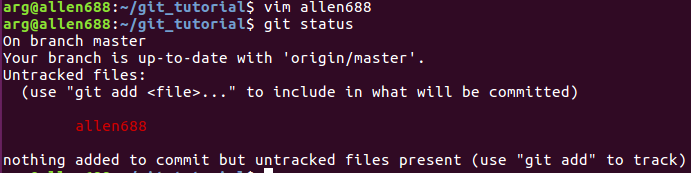
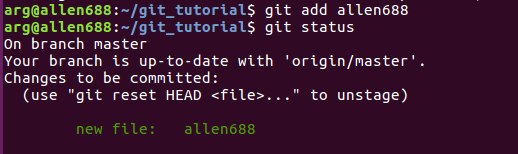
1. Clone the repo to laptop  
   從github伺服器下載repo到電腦，repo可想像成整個資料夾  
   

**laptop $ cd ~/  
 laptop $ git clone** [**https://github.com/Sensing-Intelligent-System/git\_tutorial**](https://github.com/Sensing-Intelligent-System/git_tutorial) **laptop $ cd ~/git\_tutorial  
 laptop $ ls**  
****

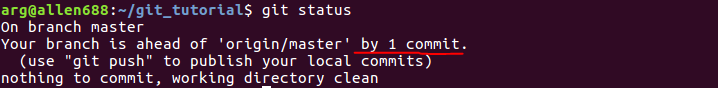
1. Keep local repo updated with original repo  
   更新github的repo到電腦端的repo

**laptop $ git pull**  
update the local repo to original repo

### Task 2: git add, commit, and push

1. Setting config file for tell git who you are  
   設定config檔案，為了在修改檔案後，告訴git伺服器誰做了這次的修改  
   **laptop $ git config --global user.email "Your email"  
   laptop $ git config --global user.name "Your name"  
   **
2. Modify file **laptop $ cd ~/git\_tutorial  
   laptop $ git status**
3. you have not modified anything, so it will show “working directory clean” **  
   laptop $ vim [laptop\_hostname]**type “hello world” and save the file **laptop $ git status**you can see untrack files below  
   可以看到剛剛新增的檔案出現在untracked files ****
4. Add file and commit  
   **laptop $ git add [laptop\_hostname]  
   laptop $ git status  
     
   laptop $ git commit -m "add file [laptop\_hostname] for git tutorials"**commit 後面打的是說明你這次改這些檔案的功能以及目的  
   behind the commit command you will put the description of your commit’s purpose

**laptop $ git status**



1. Push the file to github. Before pushing, you have to git pull first.  
   上傳檔案到github伺服器，上傳之前要先git pull 更新到最新狀態  
   **laptop $ git pull**  
   **laptop $ git push origin master**  
   go to the website: <https://github.com/Sensing-Intelligent-System/git_tutorial>   
   you can see the file you push  
   你可以看到commit & push的檔案

### Task 3: git branch, git checkout

When multiple people work on the same project, code maintaining becomes hard if everyone push their code to the same branch. To make it easier, use **git checkout** to create your own branch and push code to that branch.

1. Check current branch

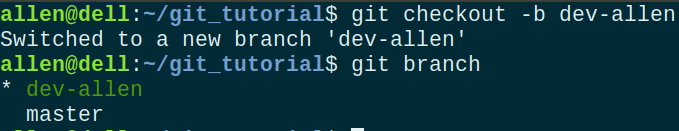
**laptop $ git branch**



1. Create your branch

**laptop $ git checkout -b [branch name] (e.g. git checkout -b dev-[your name])**

**laptop $ git branch**

****

1. Add file and commit

Please redo task2 again, but push the file to your branch this time

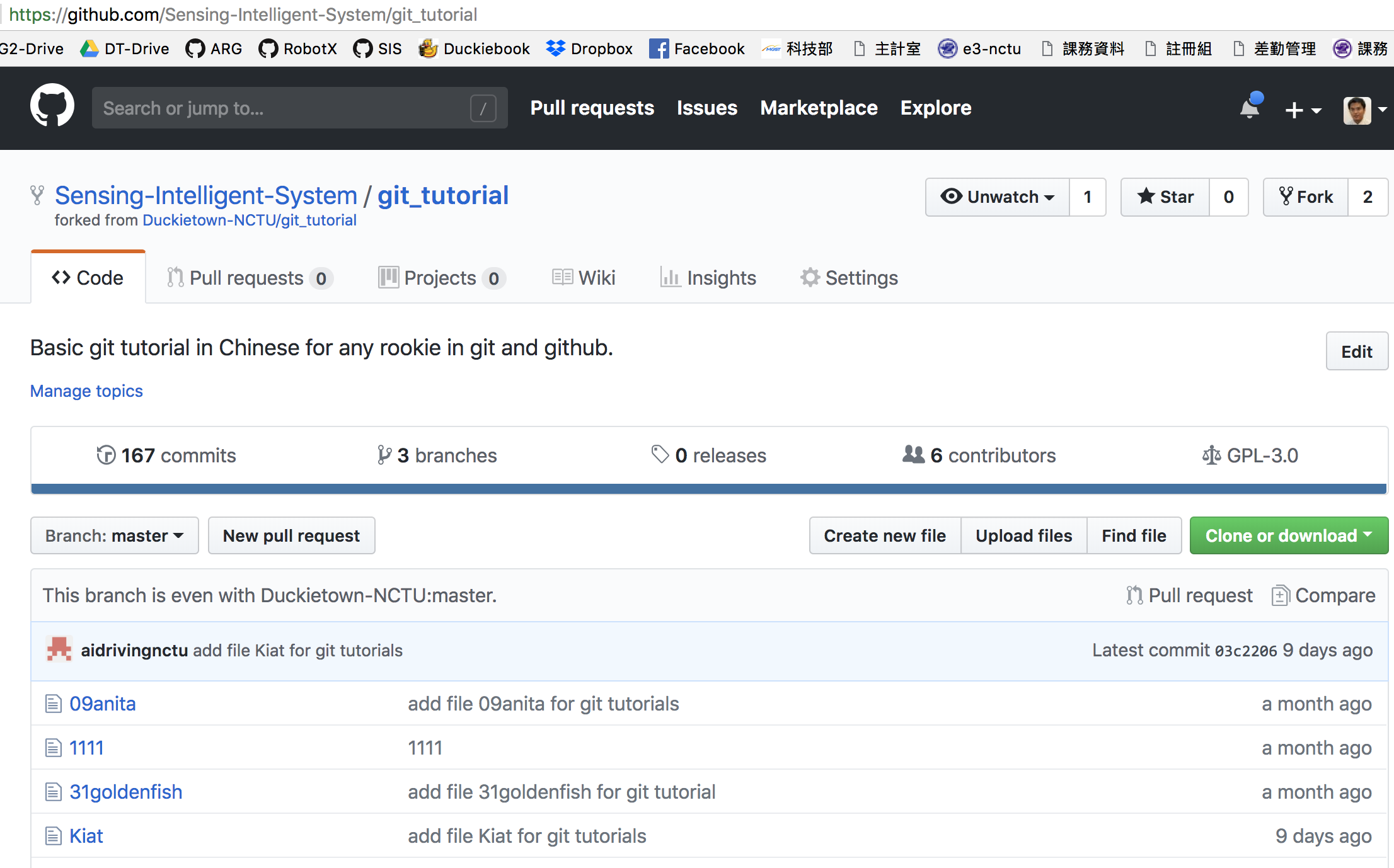
### Task 4: Join Slack

Slack will be used for announcements and discussions. Please join the Sensing-int-sys-2020 workspace through the link below. Don’t forget to upload your profile after joining it.

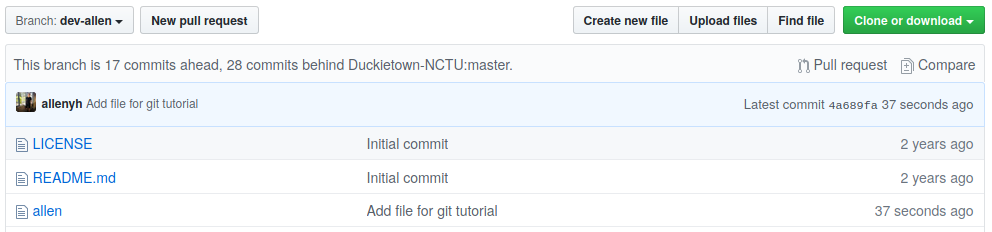
<https://join.slack.com/t/sensing-int-sys-2020/shared_invite/enQtOTgyOTY0Njk4NTUxLTc3MGM1ZTM1ODVhZTViMGEwY2IyMWNjMzE3ZDgzMjg5MmM1YTQ3NjZkODM4ZTA1MWYxYmJlODY4MTAyNjg3NDM>

### 1.4 Check points

Show the file you push in github repo, in master branch and in your own branch

ex: (in master branch)  


(in your own branch)



## Guidelines:

### 2.1 You Should

* Write clear/descriptive comments for the commits so that your teammates know what you did.
* Commit frequently whenever you finish something
* Test before commit.
  + Baseline: Can successfully build and pass all unit tests.
  + Strongly prefer: Test each new feature and confirm that they can work.
* Follow the same coding standard.
  + A tip to make your commit clear & descriptive: Make sure the diff of each commit readable, and each commit is just about the new features/fixes, not formatting!
  + The default values of your IDE are preferred. It does not matter whether using 2 spaces, 4 spaces, or tabs for indents. Just make sure everyone uses the same rules. Don’t need to create your own. Don’t need to be fancy. Just make sure that it is consistent among all teammates.
  + Otherwise, your diff will contain too many useless lines about formatting.

### 2.2 You Should Not

* Commit generated files -> add them to .gitignore
* Commit files that can break the build.
  + This is the first indicator that the commit is not tested, and this is the first bad habit to avoid, no matter you are researchers, programmers, or engineers.
* Commit large binary files, except figures in papers -> video, data, etc. try git-lfs (optional)