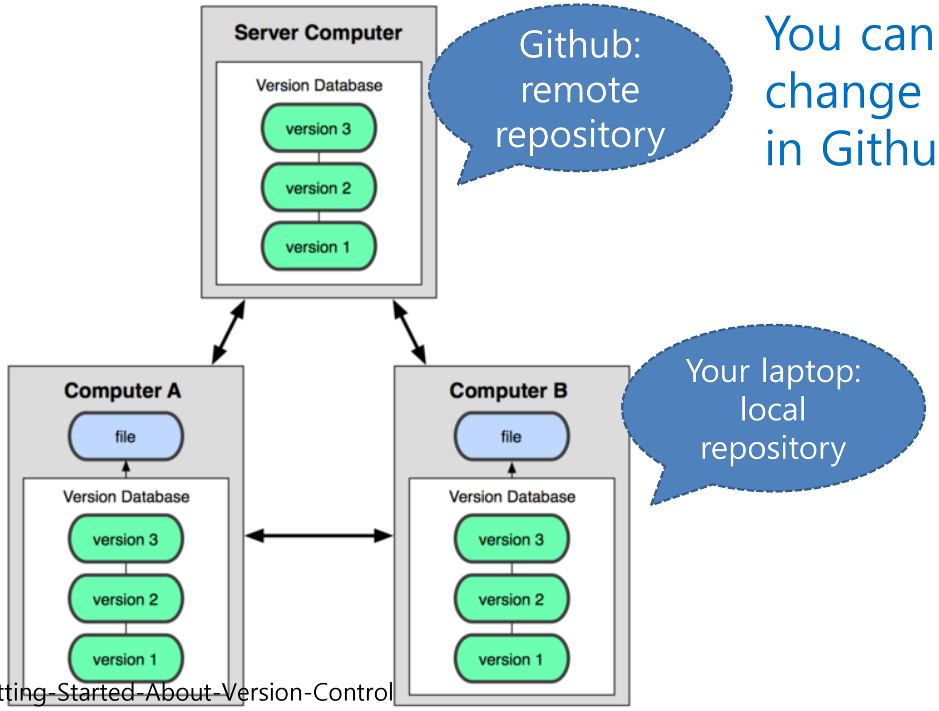
Git & Github

Younghoon Kim
nongaussian@hanyang.ac.kr

What is Git and GitHub?

- Github: Web-based hosting service for version control using git
- Git: Difference to traditional version control systems (e.g., SVN and CVS)
  - Distributed version control and source code management



You cannot change files directly in Github, generally

You can work in local repo. and manage them using git

https://git-scm.com/book/en/v1/Getting-Started-About-Version-Control

Benefit of Distributed VCS

- Advantage of cloning an entire repository into your workstation to get a local repository:
  - All operations (except push & pull) are very fast because the tool only needs to access the hard drive, not a remote server.
  - Since every contributor has a full copy of the project repository, they can share changes with one another if they want to get some feedback before affecting changes in the main repository.
  - If the central server gets crashed at any point of time, the lost data can be easily recovered from any one of the contributor's local repositories.

* From https://www.edureka.co/blog/what-is-git/

Installation (Git)

- Ubuntu
  - $ sudo apt update
  - $ sudo apt upgrade
  - $ sudo apt install git
- Mac
  - Install the Xcode Command Line Tools
  - Or install using brew as
    $ brew install git
- Reference
  - https://git-scm.com/book/en/v2/Getting-Started-Installing-Git

Git for Windows

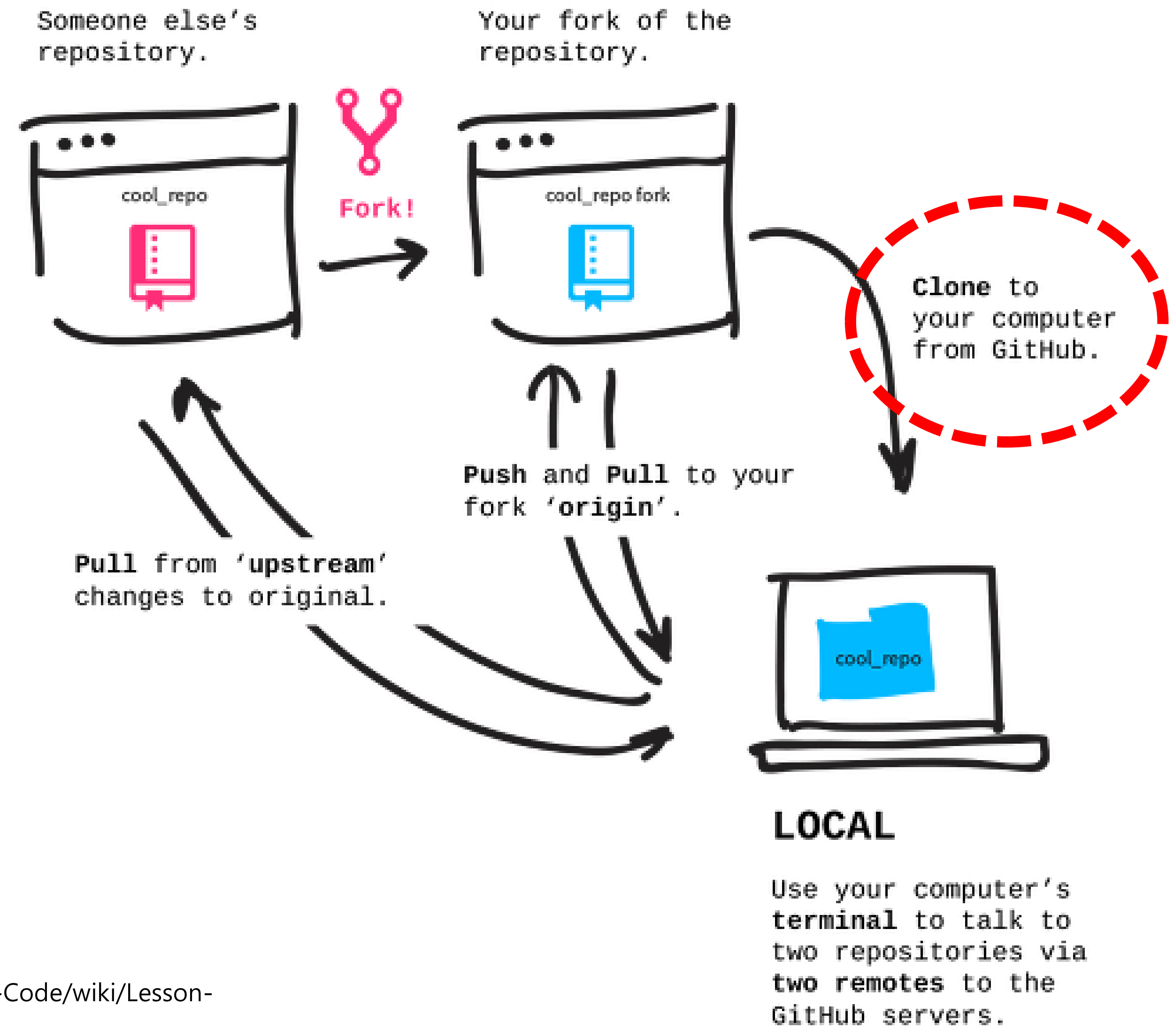
- gitforwindows.org

We bring the awesome **Git** SCM to Windows

FORK & CLONE

Cloning

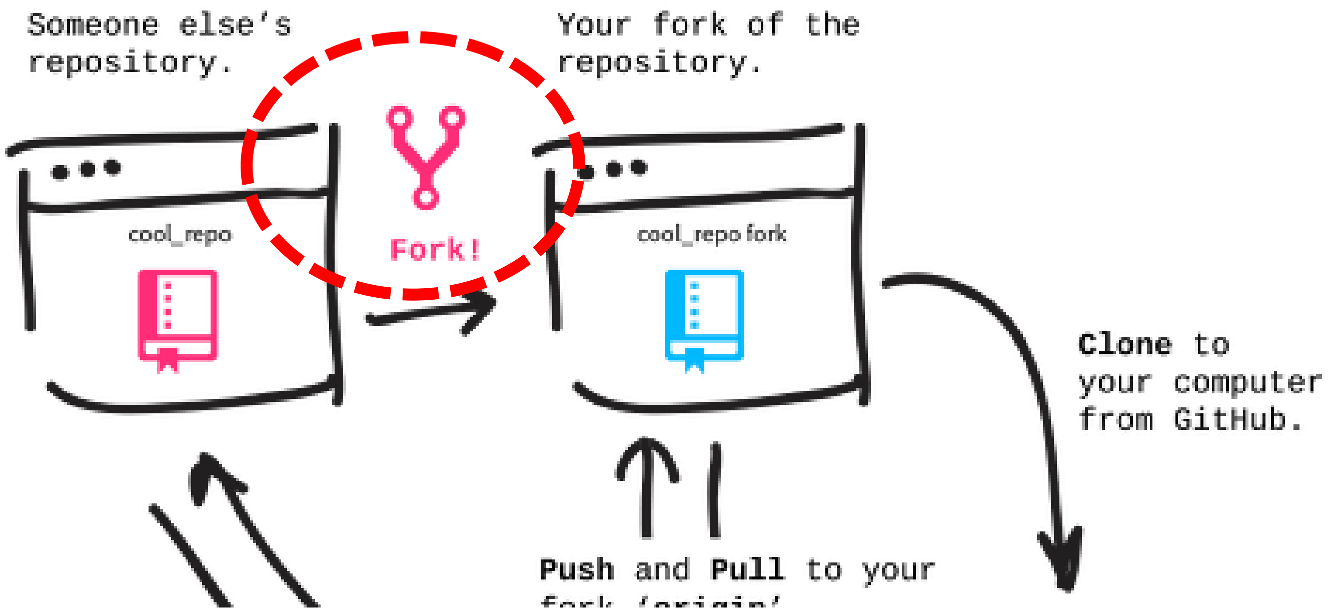
- to create a clone, or copy of the target repository on the local computer


Someone else's repository.
cool_repo
Fork!
Your fork of the repository.
cool_repo fork
Clone to your computer from GitHub.
Push and Pull to your fork 'origin'.
Pull from 'upstream' changes to original.
cool_repo
LOCAL
Use your computer's terminal to talk to two repositories via two remotes to the GitHub servers.

https://github.com/Rafase282/My-FreeCodeCamp-Code/wiki/Lesson-Save-your-Code-Revisions-Forever-with-Git

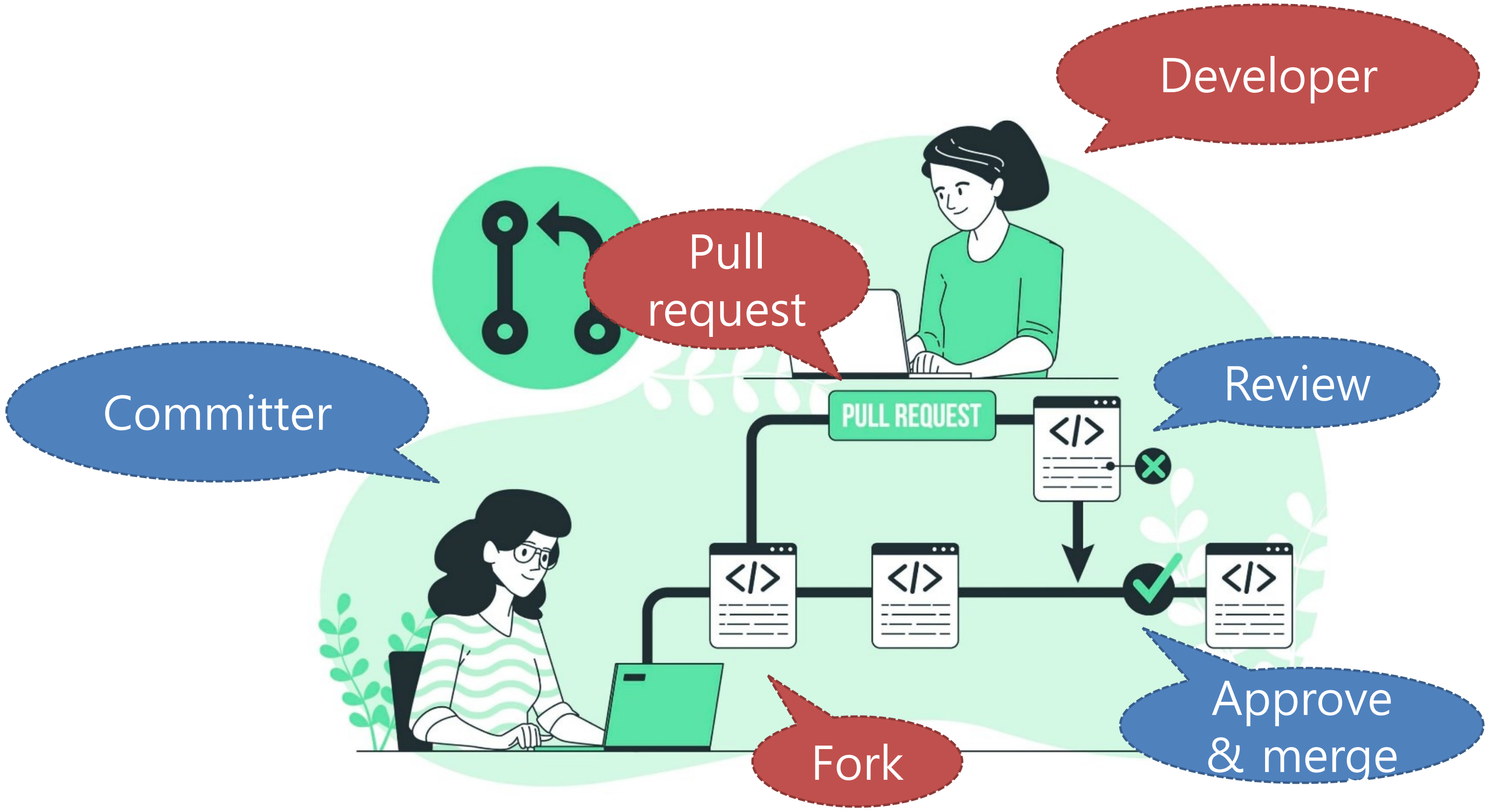
Forking

- A **fork** is a copy of repository.
- **Forking** a repository allows you to freely experiment with changes without affecting the original project.

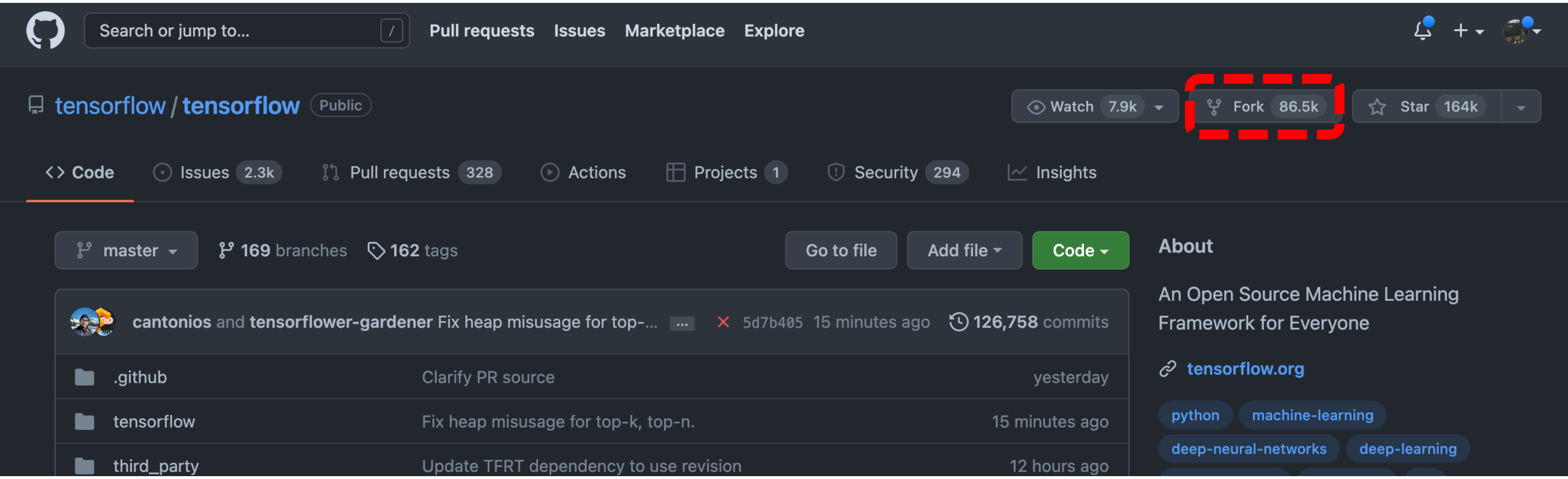

Someone else's repository.
Your fork of the repository.
cool_repo
Fork!
cool_repo fork
Clone to your computer from GitHub.
Push and Pull to your

https://github.com/Rafase282/My-FreeCodeCamp-Code/wiki/Lesson-Save-your-Code-Revisions-Forever-with-Git

Pull Request



Forking Tensorflow



Search or jump to... / Pull requests Issues Marketplace Explore

tensorflow / tensorflow Public

Watch 7.9k · Fork 86.5k · Star 164k

Code · Issues 2.3k · Pull requests 328 · Actions · Projects 1 · Security 294 · Insights

master · 169 branches · 162 tags · Go to file · Add file · Code

cantonios and tensorflower-gardener Fix heap misusage for top-... … ✕ 5d7b405 15 minutes ago 126,758 commits

.github	Clarify PR source	yesterday
tensorflow	Fix heap misusage for top-k, top-n.	15 minutes ago
third_party	Update TFRT dependency to use revision	12 hours ago

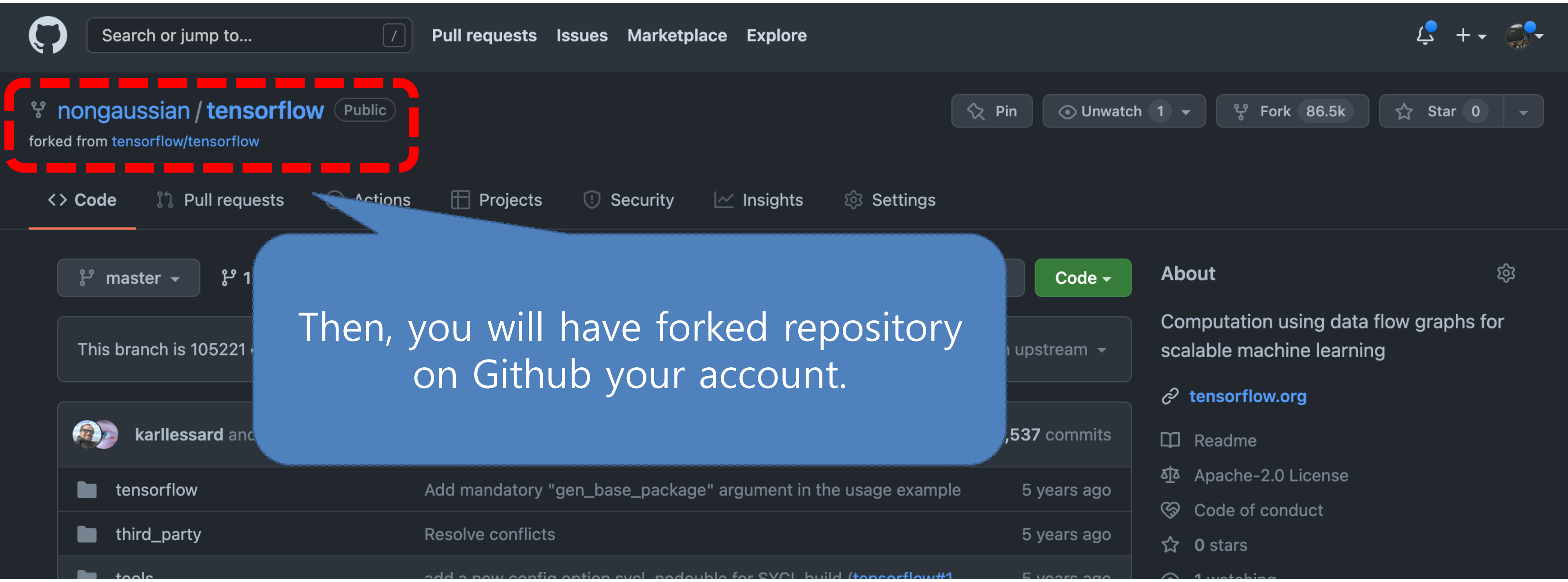
About

An Open Source Machine Learning Framework for Everyone

tensorflow.org

python · machine-learning · deep-neural-networks · deep-learning

Forking Tensorflow in Github



Search or jump to... /
Pull requests Issues Marketplace Explore

nongaussian / tensorflow Public
forked from tensorflow/tensorflow

Pin | Unwatch 1 | Fork 86.5k | Star 0

Code | Pull requests | Actions | Projects | Security | Insights | Settings

master | 1 | Code

This branch is 105221 … upstream

karllessard and … ,537 commits

tensorflow	Add mandatory "gen_base_package" argument in the usage example	5 years ago
third_party	Resolve conflicts	5 years ago
tools	add a new config option sycl_nodouble for SYCL build (tensorflow#1	5 years ago

About

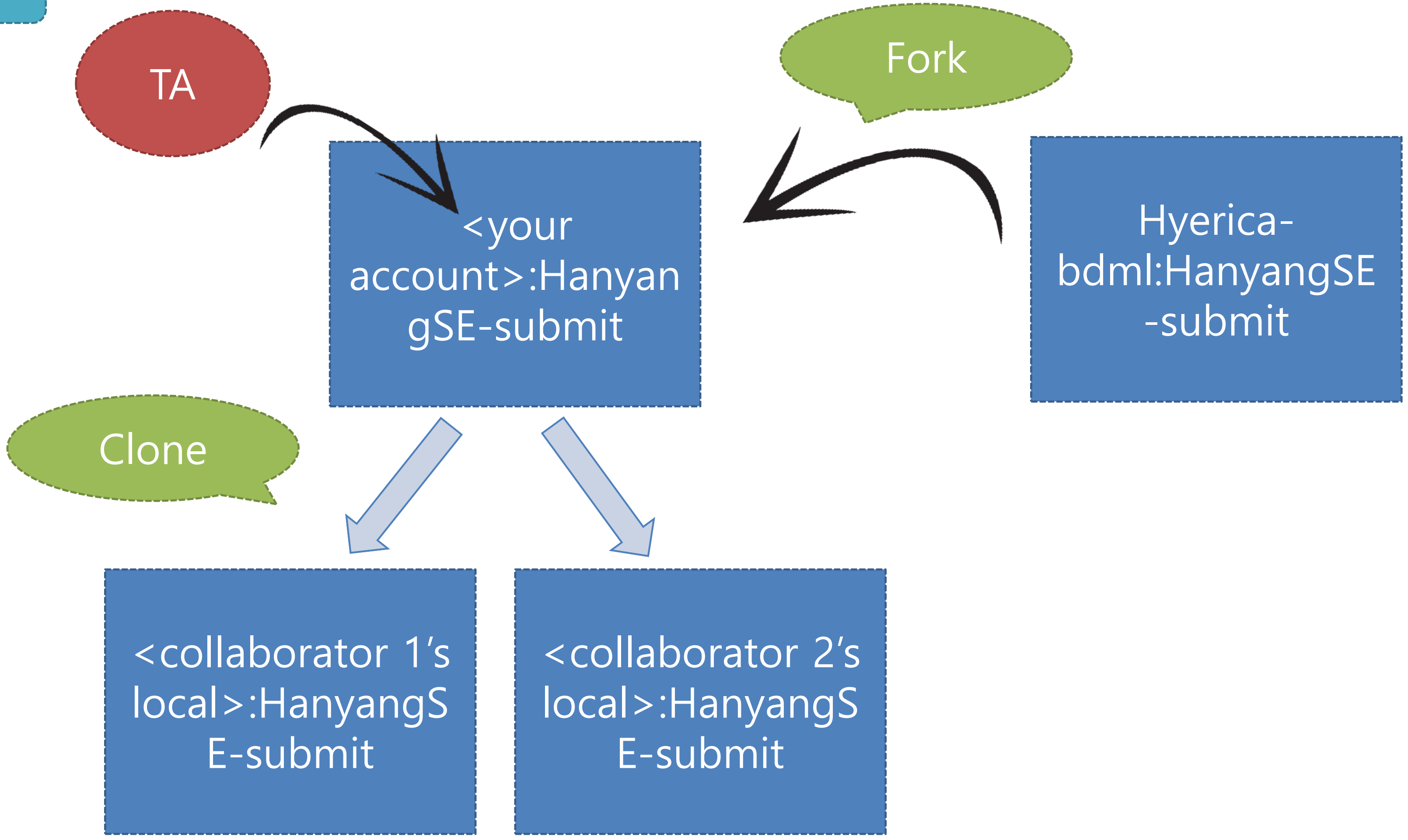
Computation using data flow graphs for scalable machine learning

tensorflow.org

Readme
Apache-2.0 License
Code of conduct
0 stars
1 watching

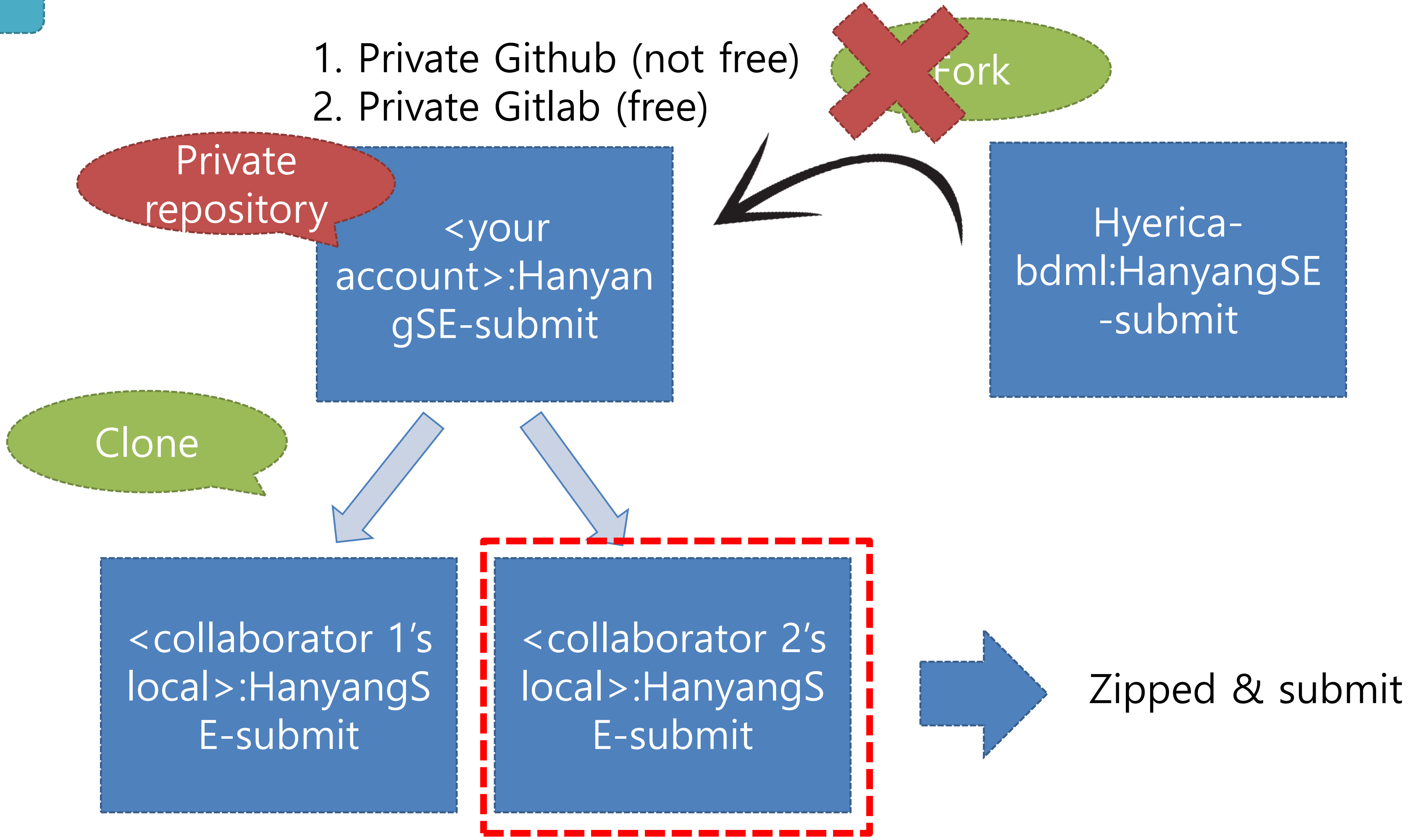
Then, you will have forked repository on Github your account.

A Typical Process


TA
Fork
<your account>:HanyangSE-submit
Hyerica-bdml:HanyangSE-submit
Clone
<collaborator 1's local>:HanyangSE-submit
<collaborator 2's local>:HanyangSE-submit

Using Git For Our Project

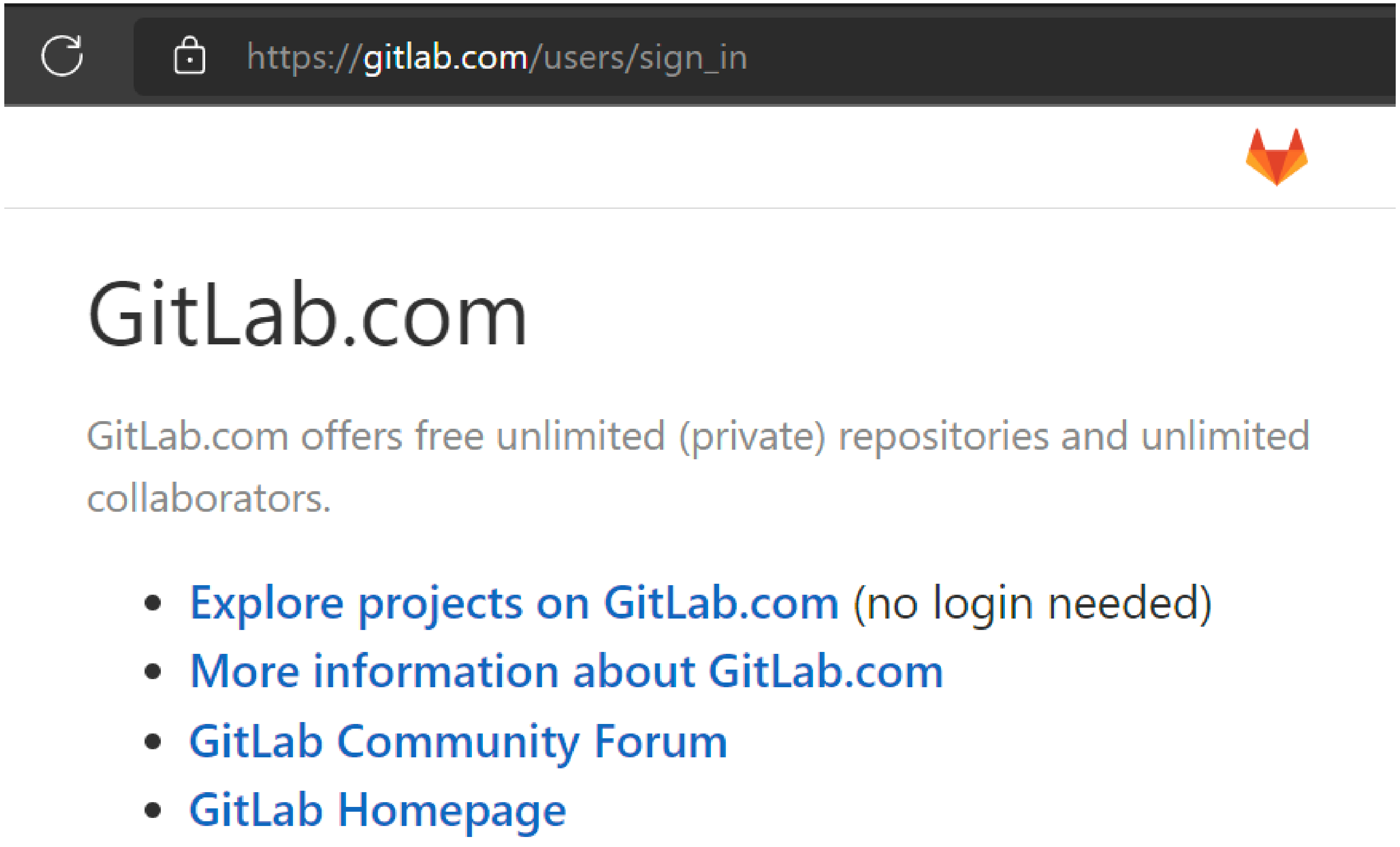
1. Private Github (not free)
2. Private Gitlab (free)


Fork
Private repository
<your account>:HanyangSE-submit
Hyerica-bdml:HanyangSE-submit
Clone
<collaborator 1's local>:HanyangSE-submit
<collaborator 2's local>:HanyangSE-submit

Zipped & submit

Use GitLab Instead of Github

- https://gitlab.com/



https://gitlab.com/users/sign_in

GitLab.com

GitLab.com offers free unlimited (private) repositories and unlimited collaborators.

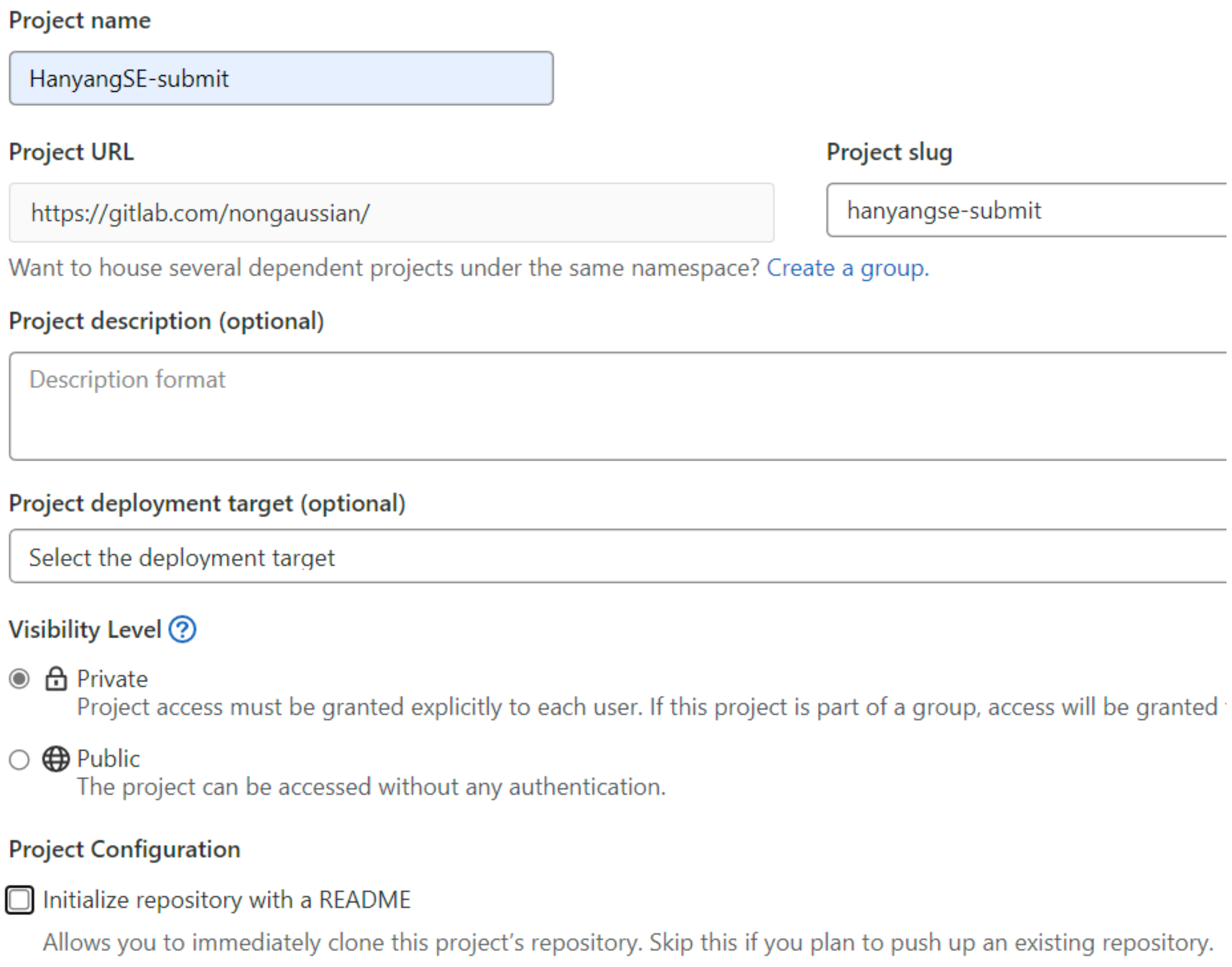
- Explore projects on GitLab.com (no login needed)
- More information about GitLab.com
- GitLab Community Forum
- GitLab Homepage

Steps

- 1. Create a blank repository in Gitlab
- 2. Clone the blank one into your local
- 3. Add the official HanyangSE-submit repo. in Github to your local repo. as a remote
- 4. Pull the code from the repo. of Github
- 5. Push them into the repo. of Gitlab
- 6. Teamwork & submit the zipped directory

1. Create A Blank Repository in Gitlab

- Project name ➔ HanyangSE-submit



Project name

HanyangSE-submit

Project URL

https://gitlab.com/nongaussian/

Project slug

hanyangse-submit

Want to house several dependent projects under the same namespace? Create a group.

Project description (optional)

Description format

Project deployment target (optional)

Select the deployment target

Visibility Level

[✓ Private]
Project access must be granted explicitly to each user. If this project is part of a group, access will be granted

Public
The project can be accessed without any authentication.

Project Configuration

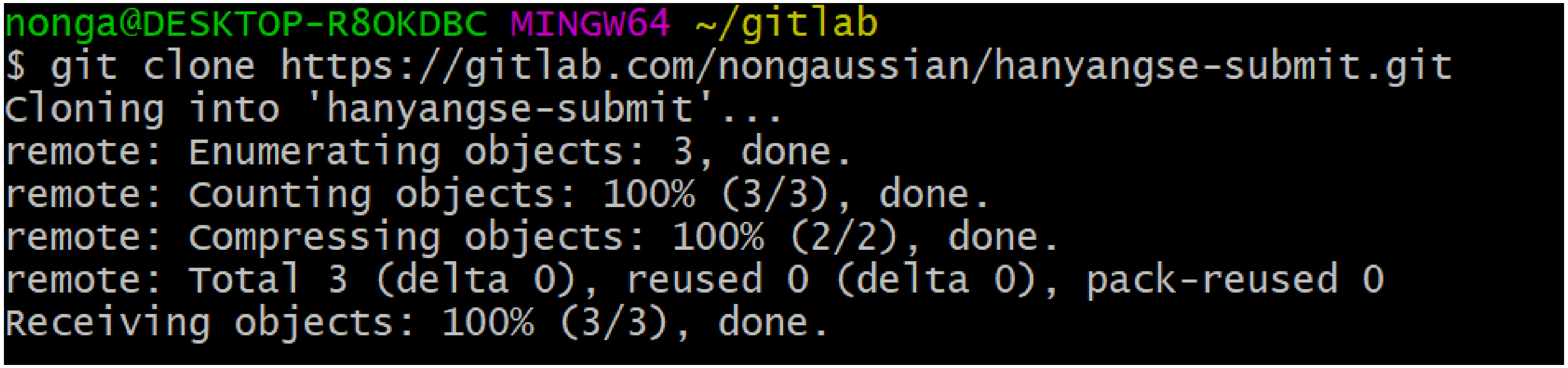
☐ Initialize repository with a README
Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.

Private

NO README

2. Clone the Blank Repository of Gitlab

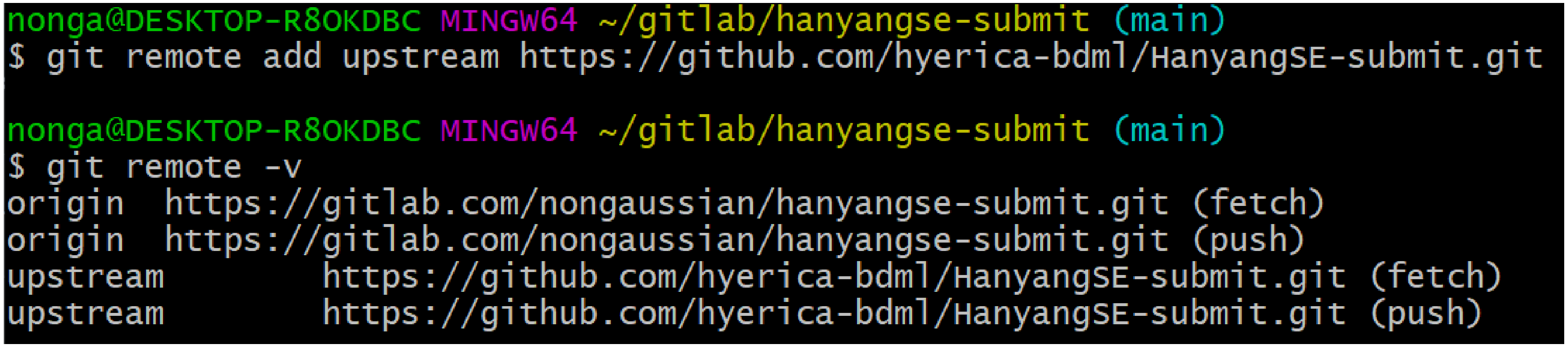
- Clone the blank repository HanyangSE-submit into your local computer
  - i.e., `git clone https://gitlab.com/<your-ID>/hanyangse-submit.git`



```
nonga@DESKTOP-R8OKDBC MINGW64 ~/gitlab
$ git clone https://gitlab.com/nongaussian/hanyangse-submit.git
Cloning into 'hanyangse-submit'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

3. Add the Github Repo. As A Remote On Your Local Repo.

- In the root directory of your local blank repository cloned from Gitlab
  - Add the HanyanSE-submit repository in Github as a remote upstream
  - i.e., `$ git remote add upstream https://github.com/hyerica-bdml/HanyangSE-submit.git`

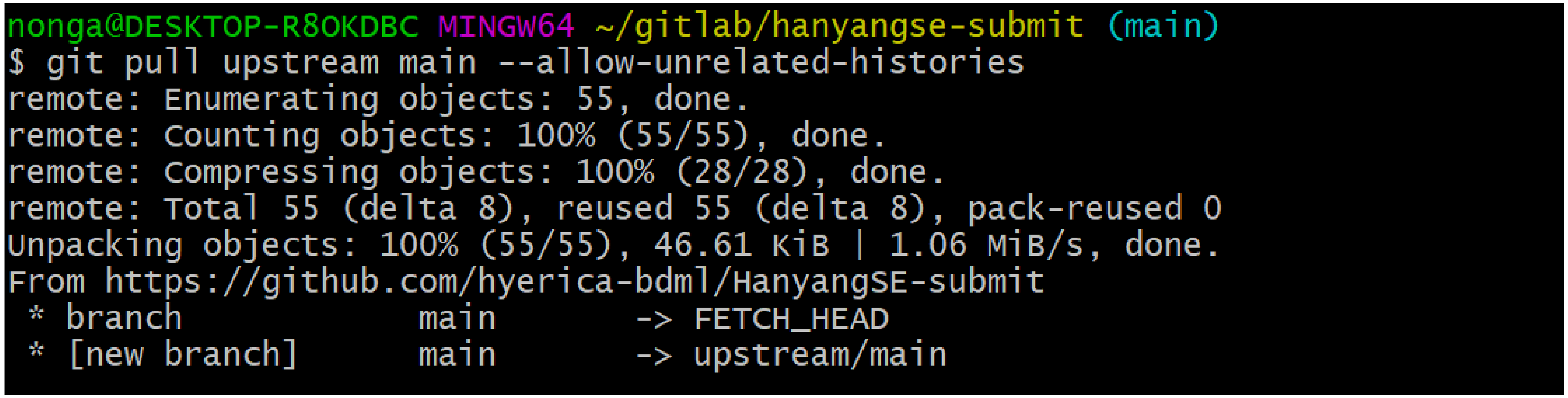


```
nonga@DESKTOP-R8OKDBC MINGW64 ~/gitlab/hanyangse-submit (main)
$ git remote add upstream https://github.com/hyerica-bdml/HanyangSE-submit.git

nonga@DESKTOP-R8OKDBC MINGW64 ~/gitlab/hanyangse-submit (main)
$ git remote -v
origin  https://gitlab.com/nongaussian/hanyangse-submit.git (fetch)
origin  https://gitlab.com/nongaussian/hanyangse-submit.git (push)
upstream        https://github.com/hyerica-bdml/HanyangSE-submit.git (fetch)
upstream        https://github.com/hyerica-bdml/HanyangSE-submit.git (push)
```

4. Pull The Repository of Github

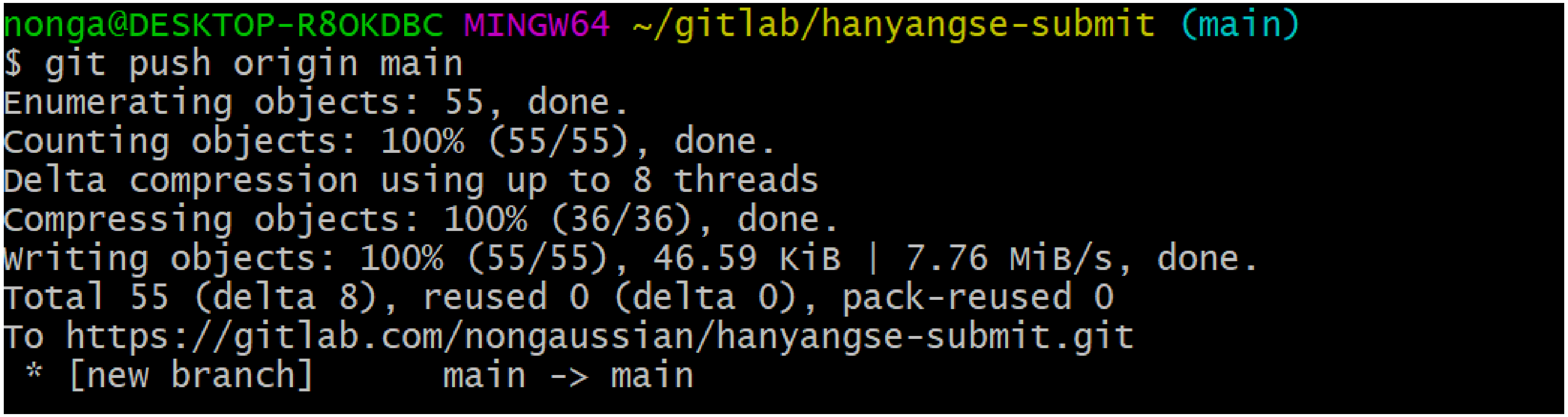
- Pull down the repository of Github on your local computer
  - i.e., `$ git pull upstream main --allow-unrelated-histories`



```
nonga@DESKTOP-R8OKDBC MINGW64 ~/gitlab/hanyangse-submit (main)
$ git pull upstream main --allow-unrelated-histories
remote: Enumerating objects: 55, done.
remote: Counting objects: 100% (55/55), done.
remote: Compressing objects: 100% (28/28), done.
remote: Total 55 (delta 8), reused 55 (delta 8), pack-reused 0
Unpacking objects: 100% (55/55), 46.61 KiB | 1.06 MiB/s, done.
From https://github.com/hyerica-bdml/HanyangSE-submit
 * branch            main       -> FETCH_HEAD
 * [new branch]      main       -> upstream/main
```

5. Push The Repository Into Gitlab

- Push the code pulled from Github into your Gitlab repository
  - i.e., `$ git push origin main`

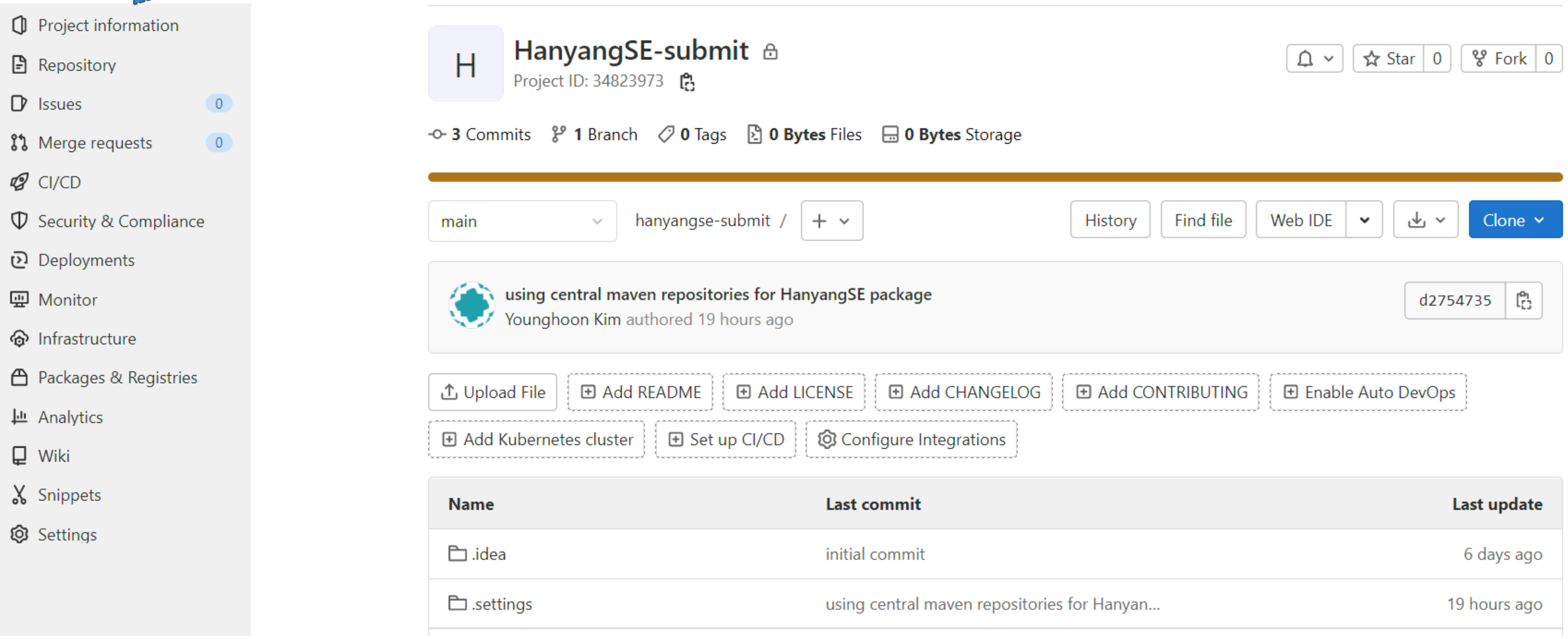


```
nonga@DESKTOP-R8OKDBC MINGW64 ~/gitlab/hanyangse-submit (main)
$ git push origin main
Enumerating objects: 55, done.
Counting objects: 100% (55/55), done.
Delta compression using up to 8 threads
Compressing objects: 100% (36/36), done.
Writing objects: 100% (55/55), 46.59 KiB | 7.76 MiB/s, done.
Total 55 (delta 8), reused 0 (delta 0), pack-reused 0
To https://gitlab.com/nongaussian/hanyangse-submit.git
 * [new branch]      main -> main
```

5. Push The Repository Into Gitlab

- Then, your team can work with the Gitlab's repository

Menu to add a member



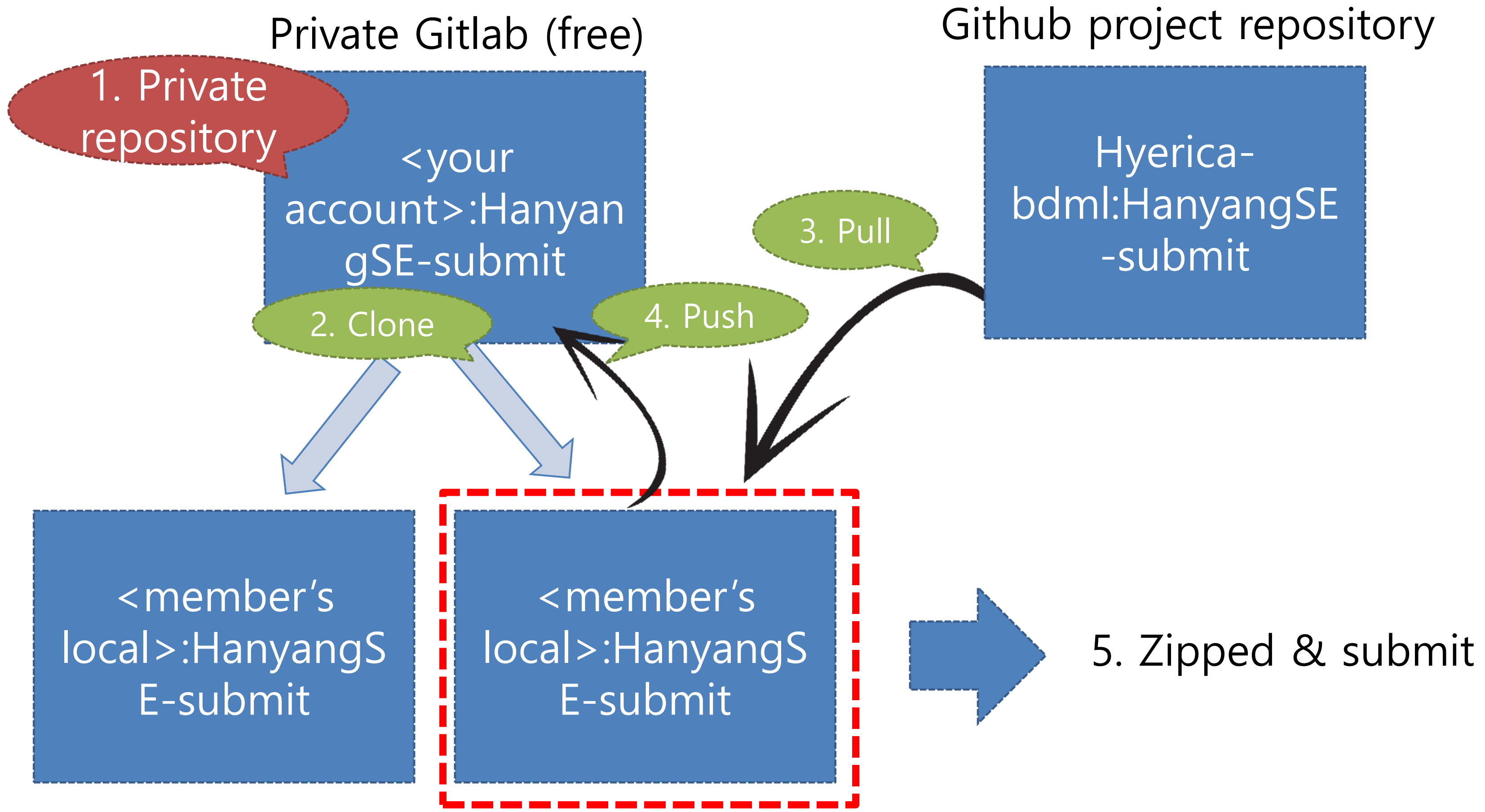
HanyangSE-submit
Project ID: 34823973

3 Commits · 1 Branch · 0 Tags · 0 Bytes Files · 0 Bytes Storage

using central maven repositories for HanyangSE package
Younghoon Kim authored 19 hours ago
d2754735

Name	Last commit	Last update
.idea	initial commit	6 days ago
.settings	using central maven repositories for Hanyan...	19 hours ago

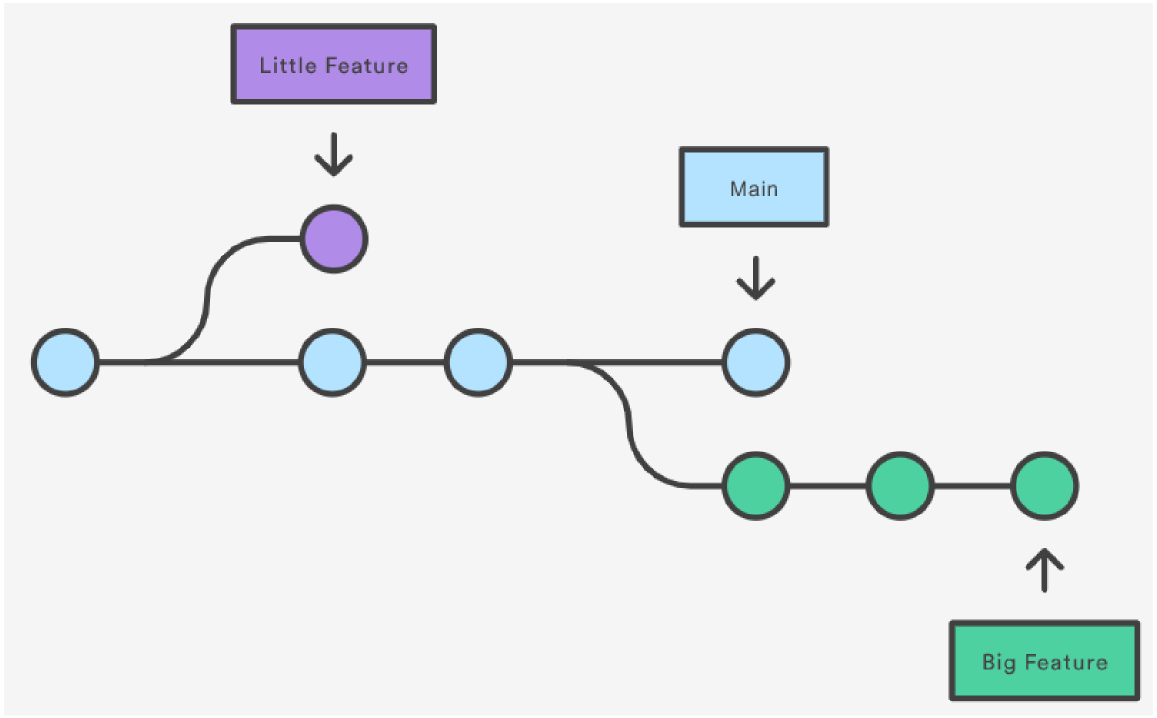
Using Private Github


Private Gitlab (free)
Github project repository
1. Private repository
<your account>:HanyangSE-submit
Hyerica-bdml:HanyangSE-submit
3. Pull
2. Clone
4. Push
<member's local>:HanyangSE-submit
<member's local>:HanyangSE-submit
5. Zipped & submit

BASIC CONCEPTS & COMMANDS

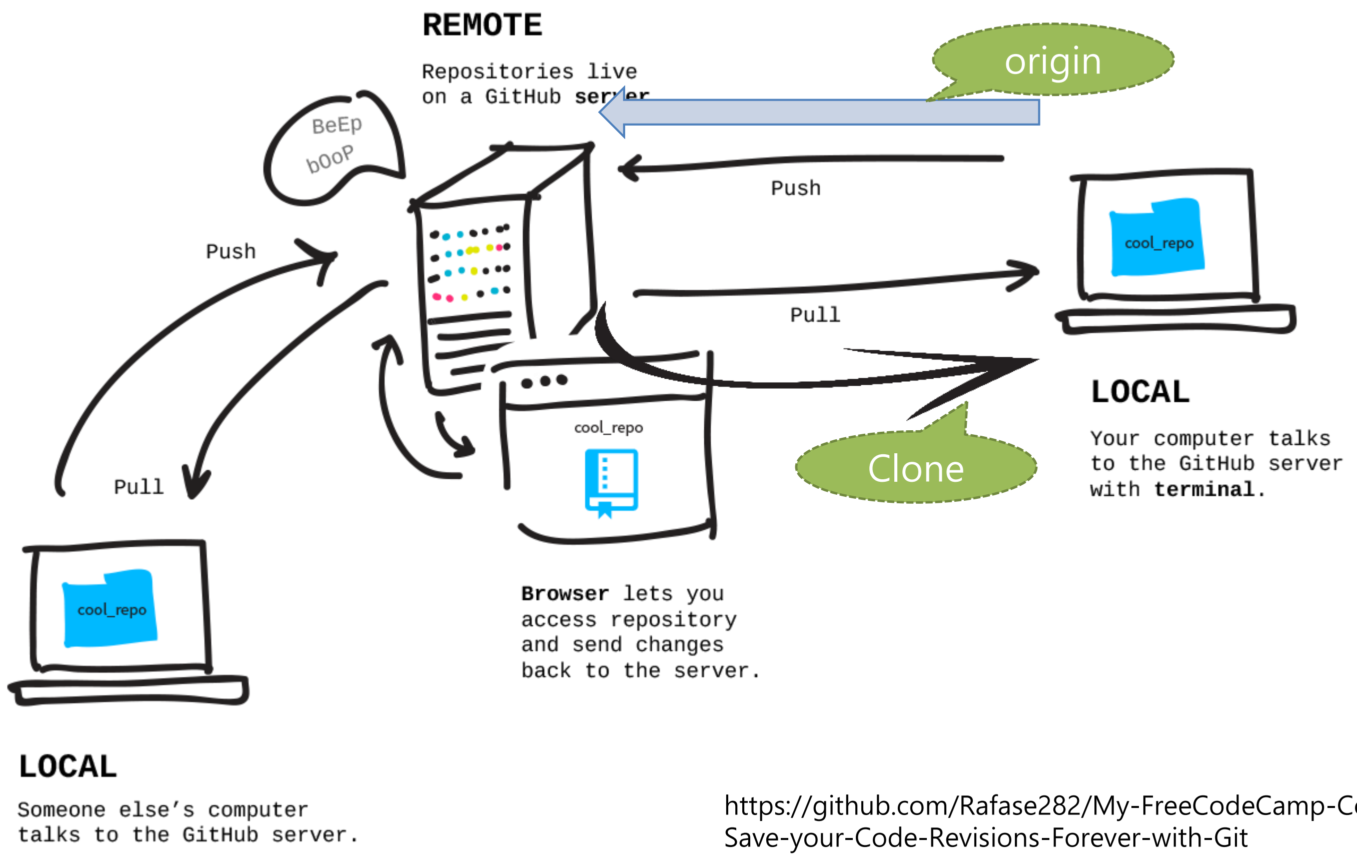
Branch

- Git branch
  - When you want to add a new feature or fix a bug—no matter how big or how small—you spawn a new branch
  - The default branch is “main” in Github and Gitlab (used to be “master”)



Remote

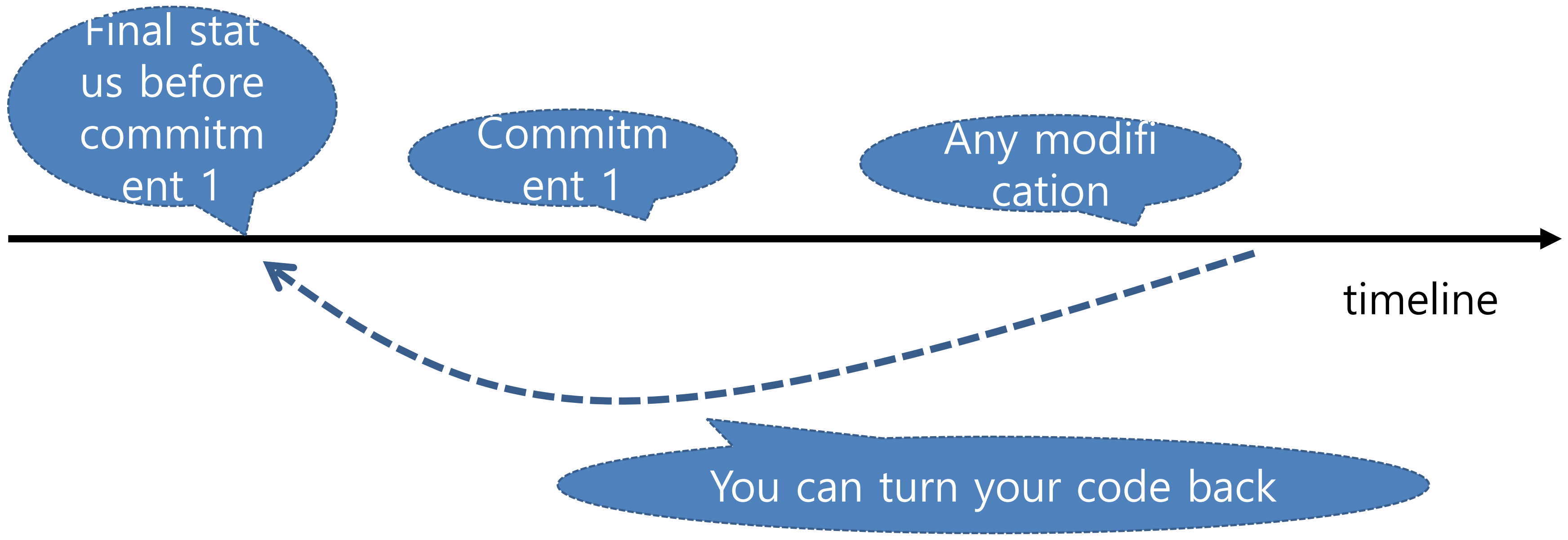
- A common repository that all team members use to exchange their changes


REMOTE
Repositories live on a GitHub server
origin
BeEp
bOoP
Push
Push
Pull
Pull
cool_repo
cool_repo
cool_repo
Clone
LOCAL
Your computer talks to the GitHub server with terminal.
Browser lets you access repository and send changes back to the server.
LOCAL
Someone else's computer talks to the GitHub server.

https://github.com/Rafase282/My-FreeCodeCamp-Code/wiki/Lesson-Save-your-Code-Revisions-Forever-with-Git

Commit

- The "commit" command is used to save your changes to the (local) repository.



Commit

- Ex 1) change or add files
  - $ git add file1
  - $ git commit -m "update file 1"
- Ex 2) Remove files
  - $ git rm file2
  - $ git commit -m "remove file 2"

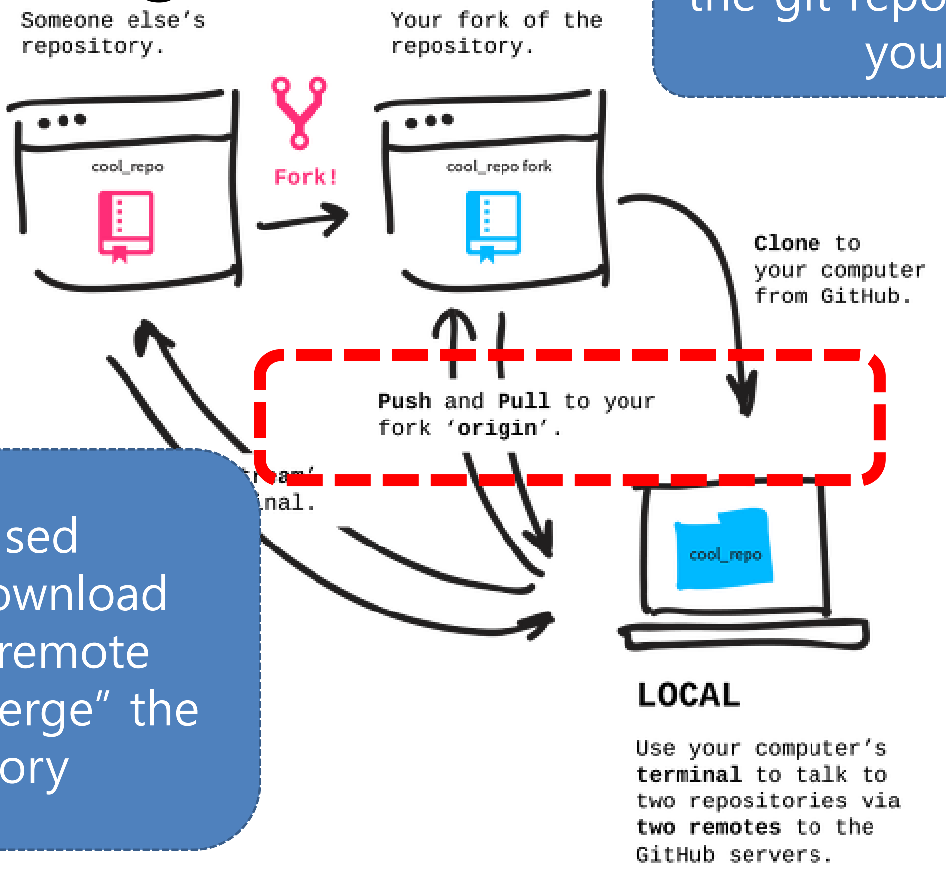
Push & Pull

Remote name

Branch name

- $ git push origin main
- $ git pull origin main

"push" is used to upload on the git repository server what you "commit"

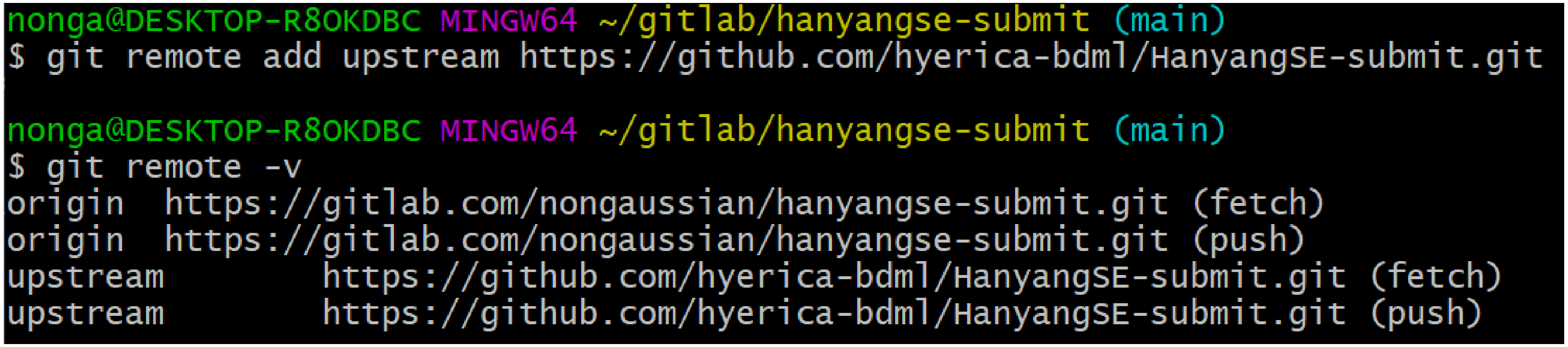

Someone else's repository.
Your fork of the repository.
cool_repo
Fork!
cool_repo fork
Clone to your computer from GitHub.
Push and Pull to your fork 'origin'.
ream'
nal.
cool_repo
LOCAL

Use your computer's **terminal** to talk to two repositories via **two remotes** to the GitHub servers.

"git pull" is used to "fetch" and download content from a remote repository and "merge" the local repository

Merge *upstream* 's changes into the local repository

- Add a remote
  - $ git remote add <remote-name> <upstream-url>



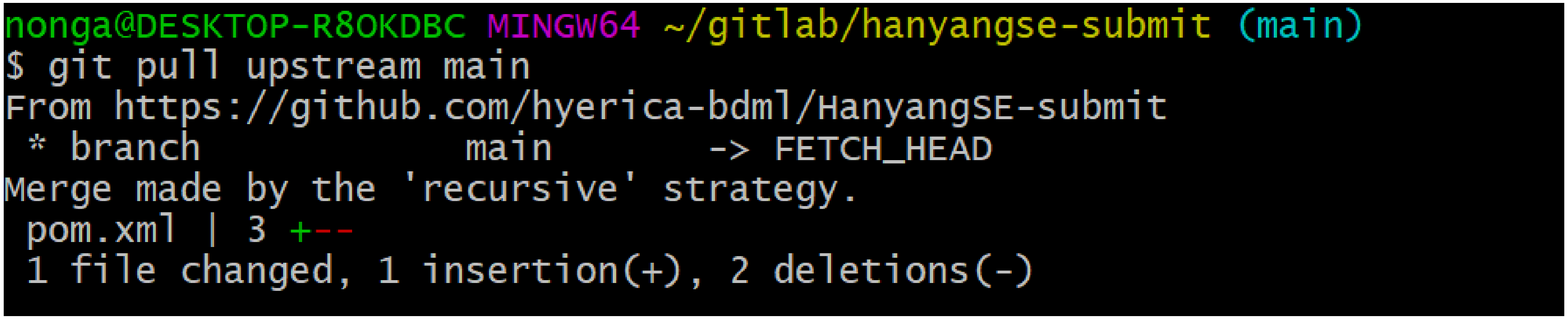
```
nonga@DESKTOP-R8OKDBC MINGW64 ~/gitlab/hanyangse-submit (main)
$ git remote add upstream https://github.com/hyerica-bdml/HanyangSE-submit.git

nonga@DESKTOP-R8OKDBC MINGW64 ~/gitlab/hanyangse-submit (main)
$ git remote -v
origin  https://gitlab.com/nongaussian/hanyangse-submit.git (fetch)
origin  https://gitlab.com/nongaussian/hanyangse-submit.git (push)
upstream        https://github.com/hyerica-bdml/HanyangSE-submit.git (fetch)
upstream        https://github.com/hyerica-bdml/HanyangSE-submit.git (push)
```

"upstream" usually refer to the original repo that you have forked from

Merge *upstream* 's changes into the local repository

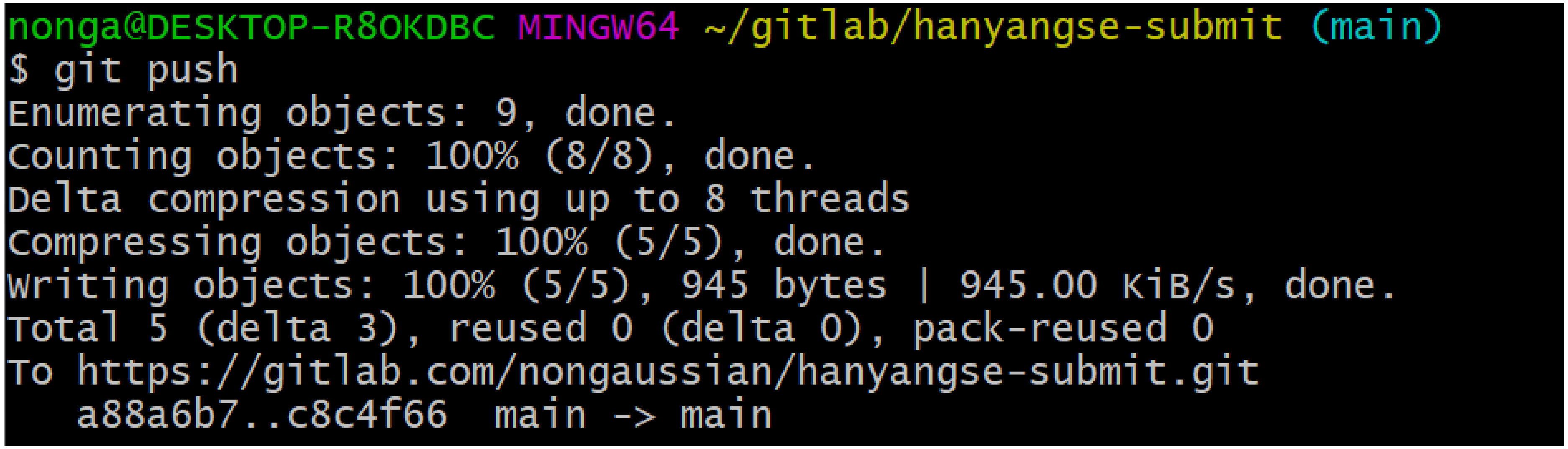
- Pull from upstream
  - $ git pull upstream main



```
nonga@DESKTOP-R8OKDBC MINGW64 ~/gitlab/hanyangse-submit (main)
$ git pull upstream main
From https://github.com/hyerica-bdml/HanyangSE-submit
 * branch            main       -> FETCH_HEAD
Merge made by the 'recursive' strategy.
 pom.xml | 3 +--
 1 file changed, 1 insertion(+), 2 deletions(-)
```

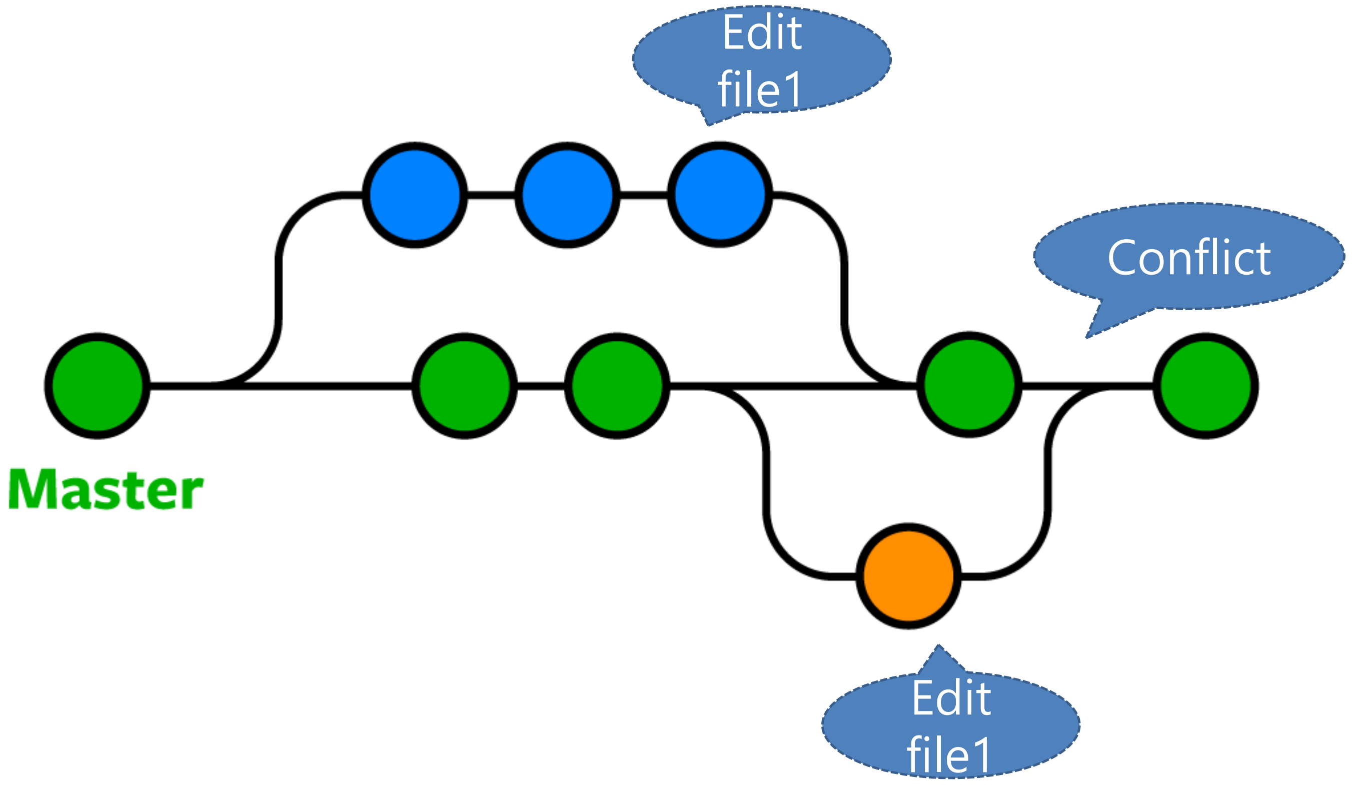
Merge *upstream* 's changes into the origin repository

- Push to origin
  - $ git push origin main



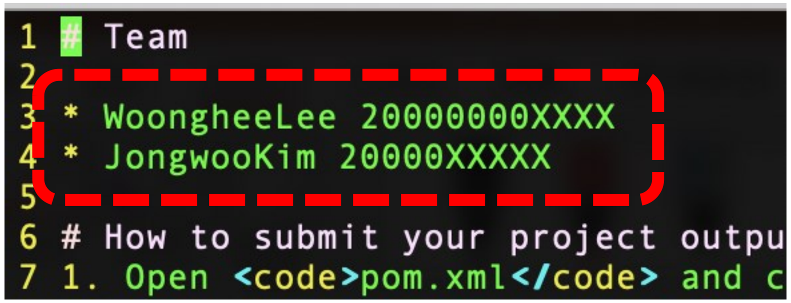
```
nonga@DESKTOP-R8OKDBC MINGW64 ~/gitlab/hanyangse-submit (main)
$ git push
Enumerating objects: 9, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 945 bytes | 945.00 KiB/s, done.
Total 5 (delta 3), reused 0 (delta 0), pack-reused 0
To https://gitlab.com/nongaussian/hanyangse-submit.git
   a88a6b7..c8c4f66  main -> main
```

How To Handle Conflicts



How To Handle Conflicts

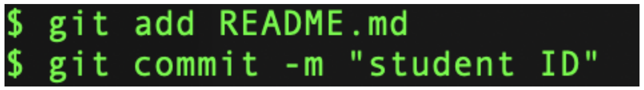
- Scenario:
  - You change “README.md” file on your repository & commit



```
# Team

* WoongheeLee 20000000XXXX
* JongwooKim 20000XXXXX

# How to submit your project outpu
1. Open <code>pom.xml</code> and c
```

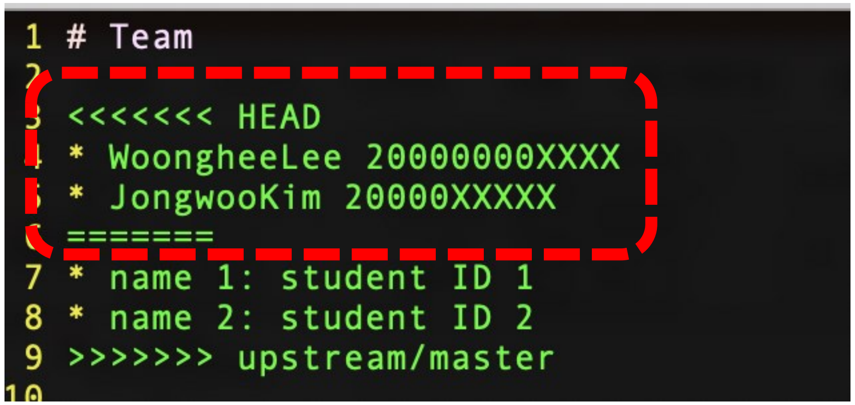


```
$ git add README.md
$ git commit -m "student ID"
```

  - TA changes “README.md” on the upstream repository
  - You try to pull upstream into your local but fail due to conflict

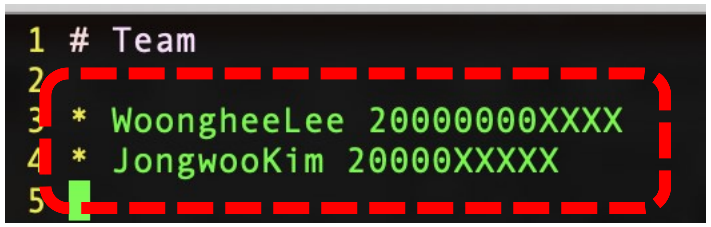
How To Handle Conflicts

- Open the conflict file with text editor



```
# Team

<<<<<<< HEAD
* WoongheeLee 20000000XXXX
* JongwooKim 20000XXXXX
=======
* name 1: student ID 1
* name 2: student ID 2
>>>>>>> upstream/master
```

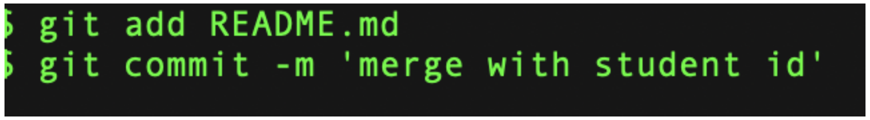


```
# Team

* WoongheeLee 20000000XXXX
* JongwooKim 20000XXXXX
```

Check every single conflicts

- Add and commit it



```
$ git add README.md
$ git commit -m 'merge with student id'
```