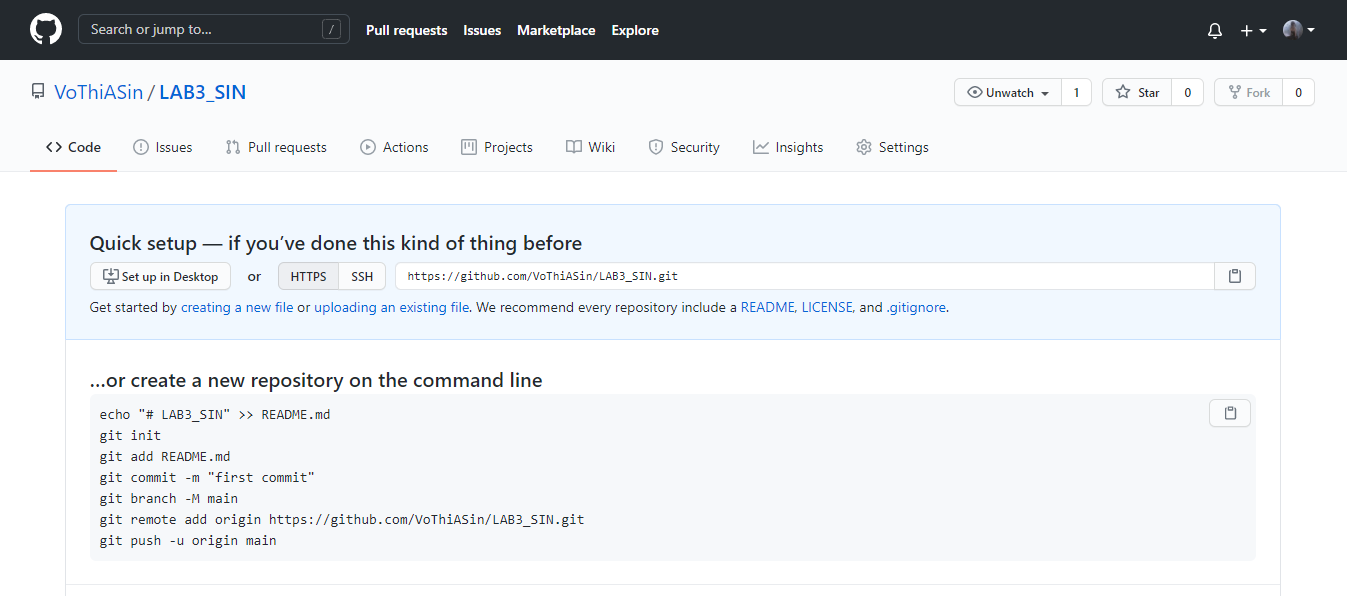
**Thực Hành Lab-03**

Họ Và Tên: Võ Thị A Sin – 1711061697

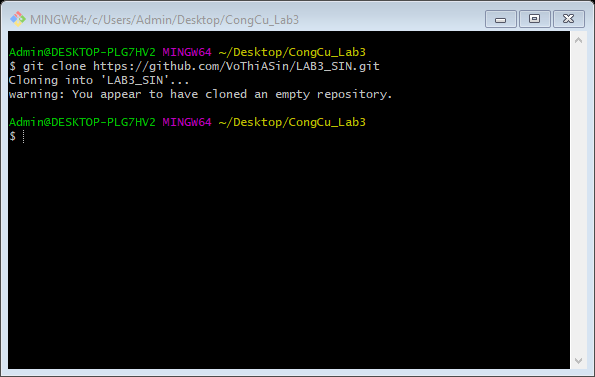
BÀI TẬP

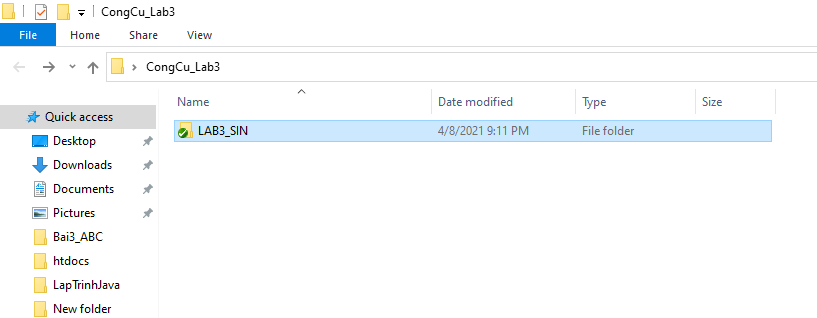
1. Cấu hình thiết lập cơ bản, và tạo một repo mới

* Tạo 1 Repo mới trên github.com có tên LAB3\_SIN

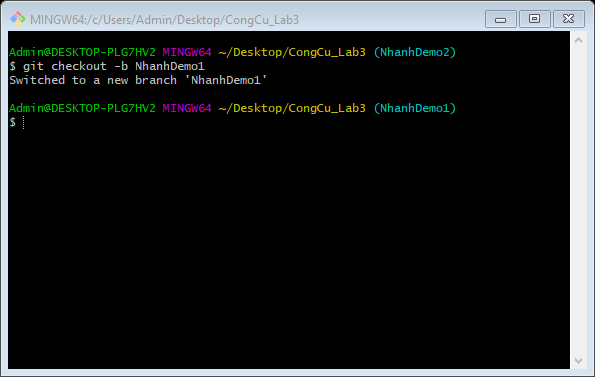


1. Lệnh clone một remote repository

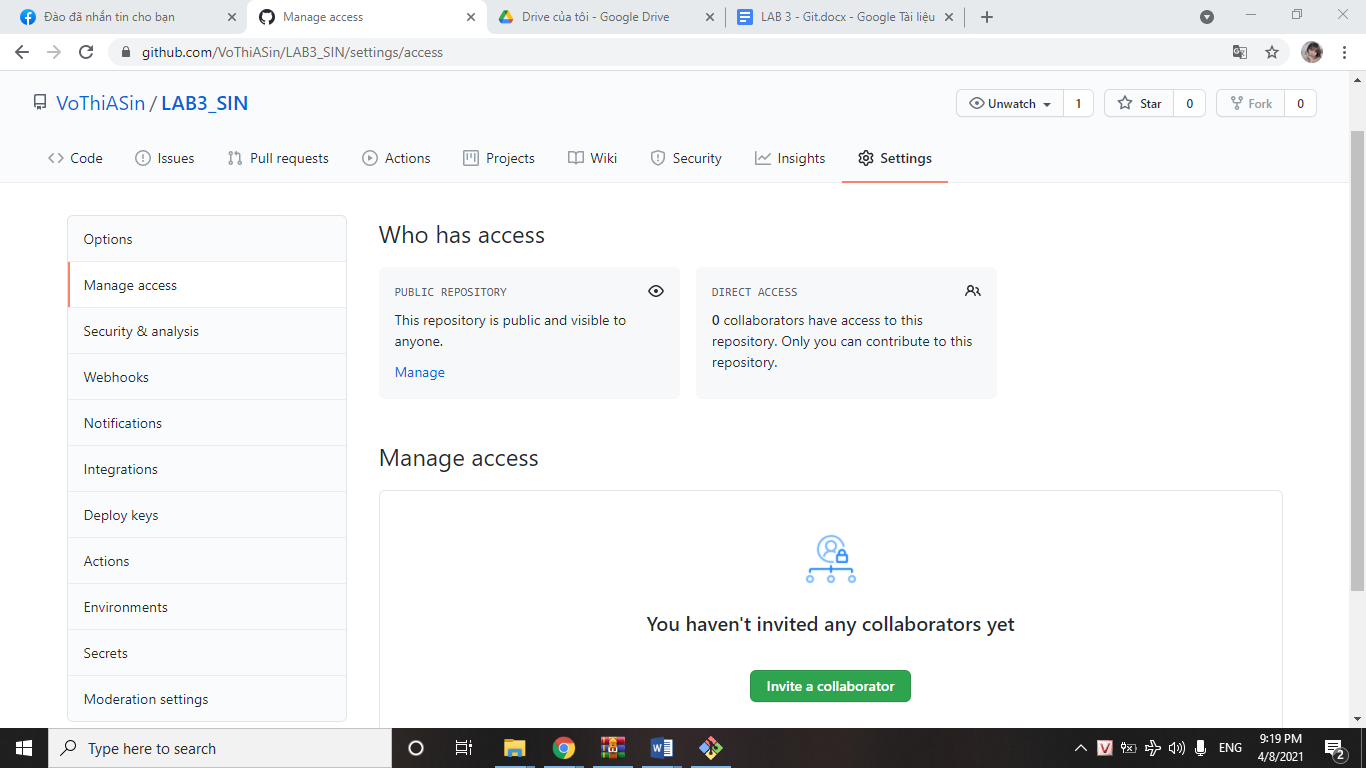


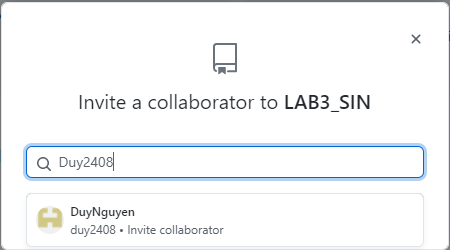


1. Tạo mới một nhánh có tên là[NhanhDemo1]

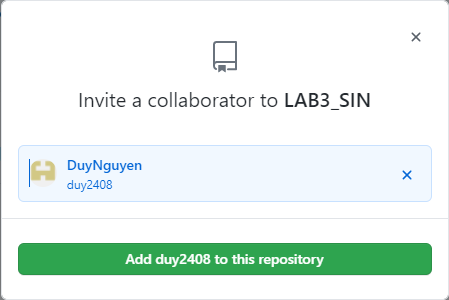


1. Mời người khác vào chung project.

* Vào project muốn mời trên github chọn mục: Setting>Manage access> Invite a collaborator
* 
* Điền username hoặc email của người cần mời vào ô tìm kiếm



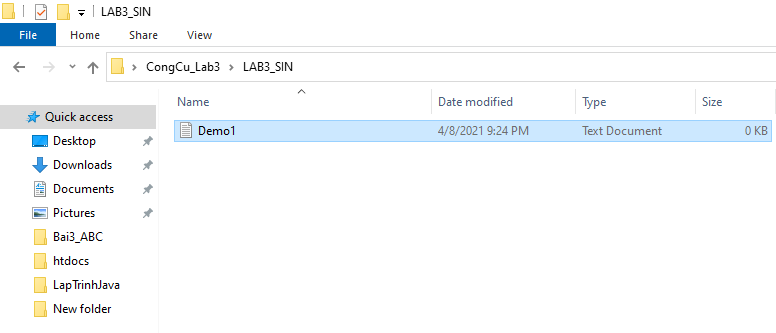
* Chọn Add để mời.



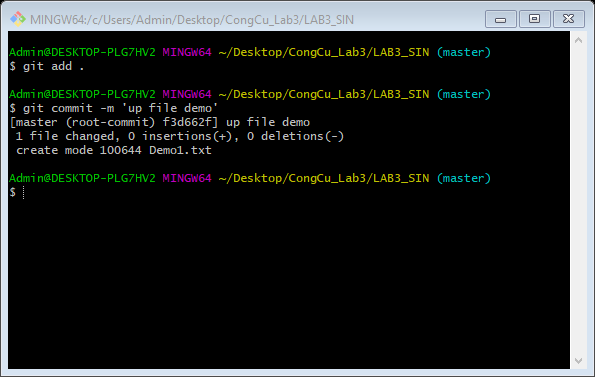
* Người được mời sẽ nhận được 1 email, chỉ cần click invite là sẽ tham gia project

1. Lệnh commit file

* Tạo một file mới có tên Demo1.txt trong thư mục của Project

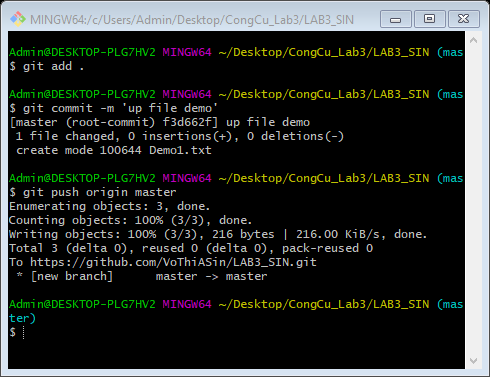


* Thực hiện commit file demo.txt vừa tạo lên



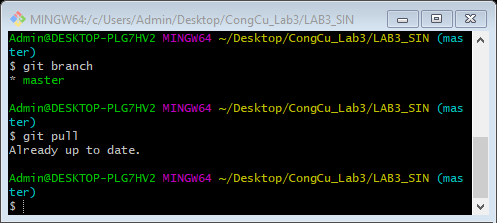
1. Push lên remote reposition

* Dùng lệnh “git push origin <branch>” để push lên remote repository



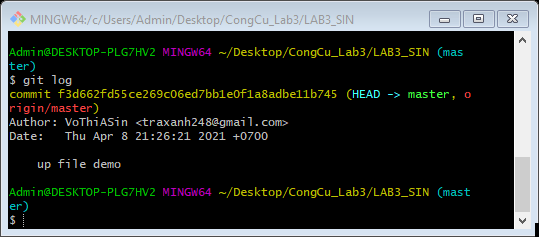
1. Pull từ remote repository

* Dùng lệnh “git pull” để pull từ repo



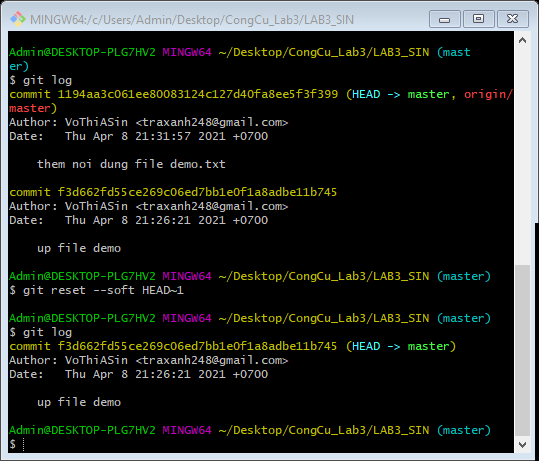
1. Lệnh xem lịch sử các phiên bản.

* Dùng lệnh “git log ” để xem lại các commit



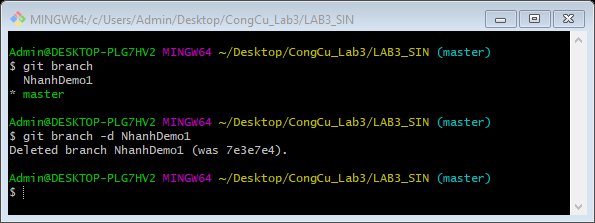
1. Lệnh phục hồi về phiên bản trước.

* Dùng lệnh git reset để phục hồi về phiên bản trước



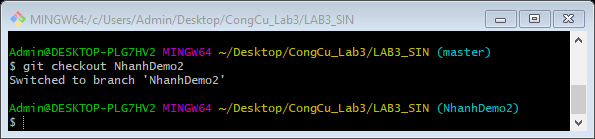
1. Xóa nhánh

* Dùng lệnh “git branch –d <branch name>” để xóa branch muốn xóa



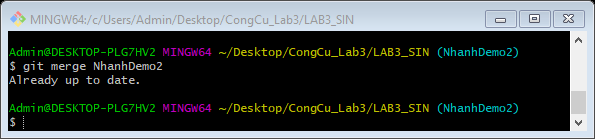
1. Chuyển qua làm việc tại khác nhánh khác nhau

* Dùng lệnh “git checkout <branch name>” để chuyển qua lại giữa các branch



1. Lệnh merge từ các nhánh

* Dùng lệnh “git merge”



Định Nghĩa:

* $ git config --global user.name “[name]”: chạy câu lệnh này để thiết lập tên người dùng.
* $ git config --global user.email “[email address]”: chạy câu lệnh này để thiết lập địa chỉ email
* $ git config --global color.ui auto:

color.ui: Biến này xác định giá trị mặc định cho các biến như color.diff và color.grep điều khiển việc sử dụng màu trên mỗi họ lệnh. Phạm vi của nó sẽ mở rộng khi nhiều lệnh tìm hiểu cấu hình hơn để đặt mặc định cho tùy chọn --color. Đặt nó thành always nếu bạn muốn tất cả đầu ra không dành cho tiêu thụ máy sử dụng màu, thành true hoặc auto nếu bạn muốn đầu ra như vậy sử dụng màu khi được ghi vào thiết bị đầu cuối hoặc false hoặc never nếu bạn không sử dụng lệnh trừ khi được bật rõ ràng với một số cấu hình khác hoặc tùy chọn --color.

* $ git init [project name]: Nếu như bạn muốn theo dõi một dự án cũ trong Git, bạn cần ở trong thư mục của dự án đó. Lệnh này sẽ tạo một thư mục mới có tên .git, thư mục này chứa tất cả các tập tin cần thiết cho kho chứa.
* $ git clone [url]: Thực hiện lệnh git clone để copy, sao chép về các repo từ server, từ dịch vụ lưu trữ git repo, hay từ máy này sang máy khác, thư mục này sang thư mục khác.
* $ git status: Lệnh git status hiện thị thông tin khác nhau (do thêm mới, xóa đi, sửa đổi các file) giữa các file trong các trường hợp:

1. Khác nhau giữa các file trong vùng staging (chỉ mục) và commit tại con trỏ HEAD (Thường HEAD ở vị trí commit cuối): thông tin này bạn có thể thực hiện lệnh commit để lưu staging vào dữ liệu Git

2. Khác nhau giữa các file trong thư mục làm việc và trong staging bạn có thể chạy git add rồi commit

3. Khác nhau giữa thư mục làm việc và những file chưa được giám sát bởi Git bạn có thể chạy git add rồi commit

* $ git diff: Lệnh git diff hiện thị thông tin thay đổi giữa thư mục làm việc và vùng index (staging) hoặc với commit cũ, thông tin thay đổi giữa index(staging) và commit, thông tin thay đổi giữa hai nhánh ...
* $ git add [file]:

Lệnh git add sử dụng để đánh chỉ mục (index) các nội dung mới, mới cập nhật trong thư mục làm việc, nó chuẩn bị nội dung sắp xếp cho lần commit tiếp theo. Khái niệm đánh chỉ mục ở trên có nghĩa là lưu lại ảnh chụp (snapshot) thông tin thay đổi của thư mục làm việc so với lần commit trước (hoặc so với snapshot chưa commit), snapshot lưu ở khu vực gọi là staging (sắp xếp, chuẩn bị). Bạn có thể thực hiện lệnh git add nhiều lần để tạo tạo ra một snapshot cuối cùng trước khi thực hiện commit.

* $ git diff --staged: Kiểm tra sự thay đổi của index (staging) với commit cuốiS
* $ git reset [file]: Nếu muốn hủy một file nào đó trong vùng staging chứ không phải toàn bộ thì dùng lệnh này.
* $ git commit –m “[mesage]”: Để lưu lại những gì đã làm vào branch thì bạn sử dụng lệnh commit với cú pháp này. Trong đó message là tin nhắn bạn muốn lưu cho lần commit đó.
* $ git branch: Git Branch dùng để kiểm soát các phiên bản của ứng dụng trong khi vẫn tiếp tục phát triển nó. Cơ bản, development branch là một hỗn hợp của tình trang code, có thể tạo một hướng phát triển mới cho nó. Nó có thể xong xong với những Git branch khác. Nó cũng có thể không tích hợp tính năng mới của code vào theo thứ tự chính xác và hướng nhất định. Git branch giúp ta có thể xem toàn bộ các branch.
* $ git branch [branch name]: tạo một branch mới.
* $ git checkout [branch name]: chuyển qua lại giữ các nhánh dùng câu lệnh này, với branch name là tên nhánh muốn chuyển đến.
* $ git merge [branch]: Lệnh git merge sử dụng để gộp nhánh, gộp nhánh này vào nhánh khác. Khi gộp nhánh git thường căn cứ vào 3 commit, để tạo ra một commit gộp, nếu có xung đột cần xử lý.
* $ git branch –d [branch name]: Để xóa local branch, bạn có thể dùng một trong các Git commands:

git branch -d branch\_name: xóa local branch nếu bạn đã push và merge nó lên branch remote.

git branch -D branch\_name: xóa local branch bất kể bạn có push và merge nó lên remote branch hay chưa.

• $ git rm [file]: xóa file khỏi thư mục làm việc

• $ git rm –cached[file]: Để xoá sự quản lý của Git đối với file đó, hay nói cách khác, là ngược lại với add, hành động này chuyển trạng thái file đã được đánh dấu (staged) sang trạng thái ko theo dõi (untracked). Lưu ý rằng dù bạn xoá đi file trong vùng làm việc, thì “ảnh” của nó vẫn còn được giữ trong vùng đánh dấu, tức là vẫn trong sự quản lý của Git, vì thế “ảnh” của nó vẫn đi vào commit. Lệnh git rm –cached mới thực sự gỡ bỏ sự theo dõi file. Nếu bỏ tham số –cached, file ko những bị xoá khỏi Git mà còn biến mất khỏi thư mục làm việc.

• $ git mv [ten file ban dau] [ten file muon doi] : Đổi tên tập tin thì cũng giống như là việc bạn gỡ bỏ tên cũ và đặt vào nó cái tên mới.

\*.log

Build/

Temp-\* : Tệp văn bản có tên .gitignore ngăn chặn việc tạo phiên bản ngẫu nhiên của

các tệp và đường dẫn phù hợp với các paterns được chỉ định

• $ git ls-files-other-ignored-exclude-standard: Liệt kê tất cả các tệp bị bỏ qua trong dự án này.

• $ git stash : được sử dụng khi muốn lưu lại các thay đổi chưa commit, thường rất hữu dụng khi bạn muốn đổi sang 1 branch khác mà lại đang làm dở ở branch hiện tại. Khi này branch đã trở nên "sạch sẽ" và git status sẽ cho thấy bạn có thể chuyển sang branch tuỳ thích.

• $ git stash pop: xoá stash đó khỏi đống stash của bạn.

• $ git stash list : xem danh sách stash

• $ git stash drop : lấy lại thay đổi và xoá nội dung thay đổi lưu trong stack.

• $ git log: Lệnh git log giúp bạn xem lại thông tin lịch sự commit, nhằm giám sát sự thay đổi của dự án. Lệnh git log có nhiều tham số để xuất ra, định dạng các thông tin hiện thị theo cách mong muốn. Bạn có thể định dạng cách các thông tin mỗi commit được in ra khi xem, cũng như có thể lọc thông tin nào đó. Mặc đinh thi hành git log nó liệt kê các commit theo thứ tự từ mới nhất đến cũ nhất, mỗi commit có các thông tin gồm: mã hash của commit, dòng thông báo, người tạo commit và ngày tạo commit.

• $ git log –follow[file] : Liệt kê lịch sử phiên bản cho một tệp, bao gồm cả tên

• $ git diff [nhanh a]…[nhanh b]: Lệnh git diff hiện thị thông tin thay đổi giữa thư mục làm việc và vùng index (staging) hoặc với commit cũ, thông tin thay đổi giữa index(staging) và commit, thông tin thay đổi giữa hai nhánh a và b ...

• $ git show [commit]: Xem thông tin cụ thể của một commit

• $ git reset [commit] : được dùng để quay về một điểm commit nào đó, đồng thời xóa lịch sử của các commit trước nó.

• $ git reset –hard[B]: Sẽ đưa HEAD về vị trí commit B, đồng thời loại bỏ tất cả các thay đổi tại B.

• $ git fetch [bookmark]: Lệnh git fetch tải về dữ liệu từ Remote Repo (các dữ liệu như các commit, các file, refs), dữ liệu tải về để bạn theo dõi sự thay đổi của remote, tuy nhiên tải về bằng git fetch nó chưa tích hợp thay đổi ngay local repository của bạn, mục đích để theo dõi các commit người khác đã cập nhật lên server, để có được thông thông tin khác nhau giữa remote và local.

• $ git merge [bookmark]/[branch]: ử dụng để gộp nhánh, gộp nhánh này vào nhánh khác. Khi gộp nhánh git thường căn cứ vào 3 commit, để tạo ra một commit gộp, nếu có xung đột cần xử lý

• $ git push[alias][brach]: Lệnh git push được sử dụng để chuyển hoặc đẩy cam kết, được thực hiện trên một nhánh cục bộ trong máy tính của bạn đến một kho lưu trữ từ xa như GitHub. Lệnh được sử dụng để đẩy lên GitHub được đưa ra bên dưới.

• $ git pull: Lệnh git pull là sự kết hợp của git fetch tìm nạp các cam kết gần đây trong kho lưu trữ cục bộ và hợp nhất git, sẽ hợp nhất nhánh từ điều khiển từ xa với chi nhánh cục bộ cũng 'remote\_name' là tên kho lưu trữ và 'branch\_name' là tên của chi nhánh cụ thể.