# CHƯƠNG 1: GIỚI THIỆU

## Quản lý phiên bản:

Quản lý phiên bản - Version Control System (VCS) là một hệ thống ghi nhận và lưu lại sự thay đổi của các file theo thời gian, từ hệ thống đó một file có thể phục hồi quay về trạng thái (phiên bản) ở một thời điểm trước đó. Ngoài ra bạn có thể theo dõi sự thay đổi của một file theo thời gian, ai đã thay đổi, thay đổi vào lúc nào .... Có nhiều hệ thống VCS mà bạn có thể chọn sử dụng như: Concurrent Versions System, Subversion, Git, Mercurial

## Lập trình cộng tác:

### Khái niệm lập trình cộng tác:

**Lập trình cộng tác:** Là một cách tiếp cận để phát triển phần mềm trong đó các cặp hoặc nhóm lập trình viên làm việc cùng nhau trong một dự án bằng cách mã hóa ý tưởng của nhau hoặc chỉnh sửa mã của nhau.

* Làm việc theo cặp hoặc theo nhóm giúp mã đã hoàn thành chứa ít lỗi và lỗi hơn, đồng thời điều này dẫn đến chất lượng mã tốt hơn và các dự án được hoàn thành nhanh hơn.
* Nó cũng cho phép gỡ lỗi nhanh hơn và khả năng phục hồi dự án lớn hơn, vì thiết lập này giúp các nhà phát triển khác tiếp quản dễ dàng hơn trong trường hợp một trong các nhà phát triển phải rời khỏi dự án.

### Phân loại lập trình cộng tác:

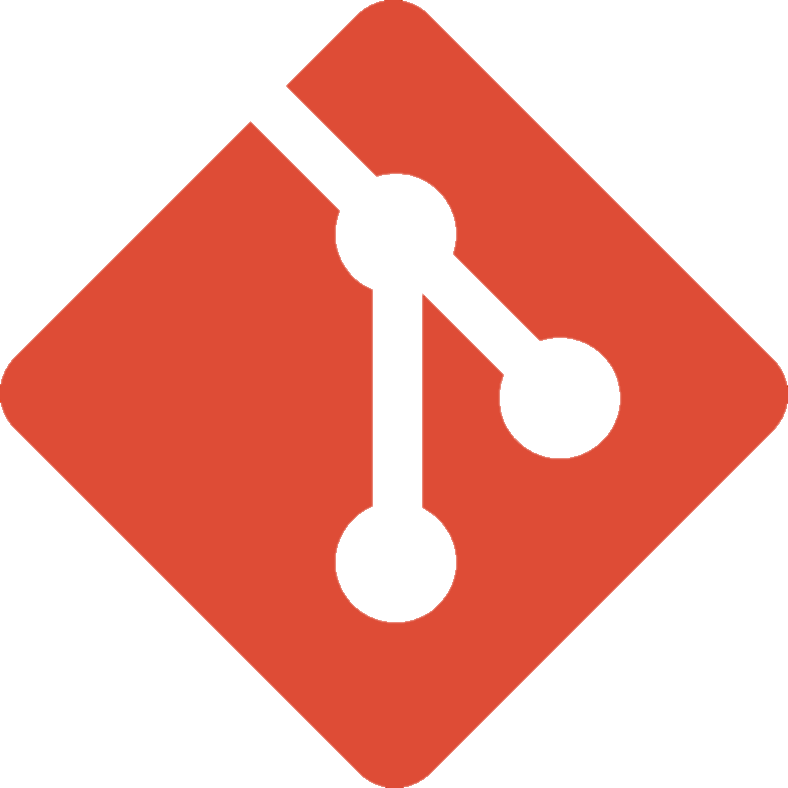
### Gồm có 3 thiết lập chung:

* **Lặp trình cặp:** Là liên quan đến hai nhà phát triển làm việc tại một máy trạm duy nhất. Một nhà phát triển đóng vai trò là người điều khiển và người kia đóng vai trò là người điều hướng.
* **Lặp trình mob:** Ý tưởng đằng sau lập trình mob tương tự như lập trình cặp, ngoại trừ nó liên quan đến nhiều hơn hai người. Tại bất kỳ thời điểm nào, một nhà phát triển trình điều khiển đang ở máy trạm trong khi các nhà lập trình điều hướng khác hướng dẫn trình điều khiển viết mã.
* **Chia sẽ mã:** Không giống như lập trình cặp và lập trình mob, chia sẻ mã liên quan đến thiết lập truyền thống của mỗi nhà phát triển làm việc trên máy của họ để viết mã của riêng họ. Tuy nhiên, với thiết lập chia sẻ mã, các thành viên của nhóm phát triển có cơ hội xem xét, sửa đổi, gỡ lỗi và thêm vào mã của nhau.

## Git:

### Giới thiệu về Git:

* **Git** viết tắt của từ Global Information Tracker. Là một hệ thống quản lý phiên bản phân tán (Distributed Version Control System) xuất hiện vào năm 2005. Hay để dễ hiểu, Git được dùng để quản lý mã nguồn dự án của chúng ta. Git được biết đến là một trong những hệ thống phổ biến nhất trong giới lập trình, nhờ vào những khả năng quản lý và tính hữu dụng của nó trong công việc.



Hình 1: Logo của Github

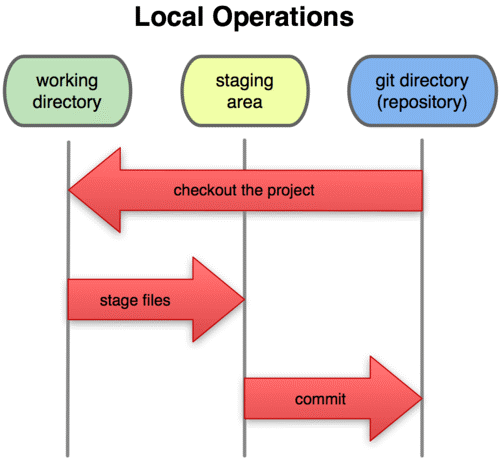
* Cơ chế lưu trữ phiên bản của Git là nó sẽ tạo ra một snapshot (lưu lại nội dung tất cả các tập tin, thư mục tại thời điểm đó) trên mỗi tập tin sau khi commit, từ đó nó có thể cho phép ta tái sử dụng lại một ảnh chụp nào đó mà bạn có thể hiểu đó là một phiên bản. Đây cũng chính là lợi thế của Git so với các DVCS khác khi nó không"lưu cứng" dữ liệu mà sẽ lưu với dạng snapshot.

### Tính năng của Git:

* Cập nhật và lưu lại những phiên bản khác nhau của source code.
* Khôi phục lại source code tử một phiên bản bất kỳ.
* Dễ dàng so sánh những thay đổi từ các phiên bản khác nhau.
* Phát hiện ai đã sửa phần nào, đã làm phát sinh lỗi.
* Khôi phục lại tập tin bị đặt
* Phối hợp với các đồng nghiệp một cách hiệu quả.

### Quy trình sử lý công việc (workflow):

Khi sử dụng Git bạn nên lưu ý là dùng Git giúp bạn quản lý mã nguồn mà không thể chỉnh code trong Git. Bạn có thể thực hiện các công việc của mình trên môi trường làm việc với các chức năng tiện ích trên IDE của ngôn ngữ lập trình khác. Quy trình xử lý công việc trên Git sẽ như sau:



Hình 2: Quy trình xử lý các công việc của Git

### Lợi ích của Git:

* Khi không có kết nối internet, chúng ta vẫn có thể làm việc và commit.
* Thời gian trẻ thấp do hầu hết dữ liệu đều được sao chép ở máy cục bộ.
* Làm việc dễ dàng hơn với branch.
* Dễ dàng trong việc deployment sản phẩm

### Lịch sử về Git:

* **Git** ra đời từ một chút của sự kế thừa có tính sáng tạo của Linus Torvalds, cụ thể:
* Nhân của Linux là một dự án phần mềm mã nguồn mở của một phạm vi khá lớn. Trong phần lớn thời gian bảo trì của nhân Linux (1991–2002), các thay đổi của phần mềm được truyền đi dưới dạng các văn bản và các tập tin lưu trữ. Vào năm 2002, dự án nhân Linux bắt đầu sử dụng một DVCS độc quyền có tên là Bitkeeper.
* Vào năm 2005, sự hợp tác giữa cộng đồng phát triển nhân Linux và công ty thương mại phát triển BitKeeper bị phá vỡ, và công cụ đó không còn được cung cấp miễn phí nữa. Chính điều này đã thúc đẩy cộng đồng phát triển Linux (chính xác hơn là Linus Torvalds, người sáng lập ra Linux) phát triển công cụ của riêng họ. dựa trên những bài học tử việc sử dụng BitKeeper.

### Khái niệm Repository:

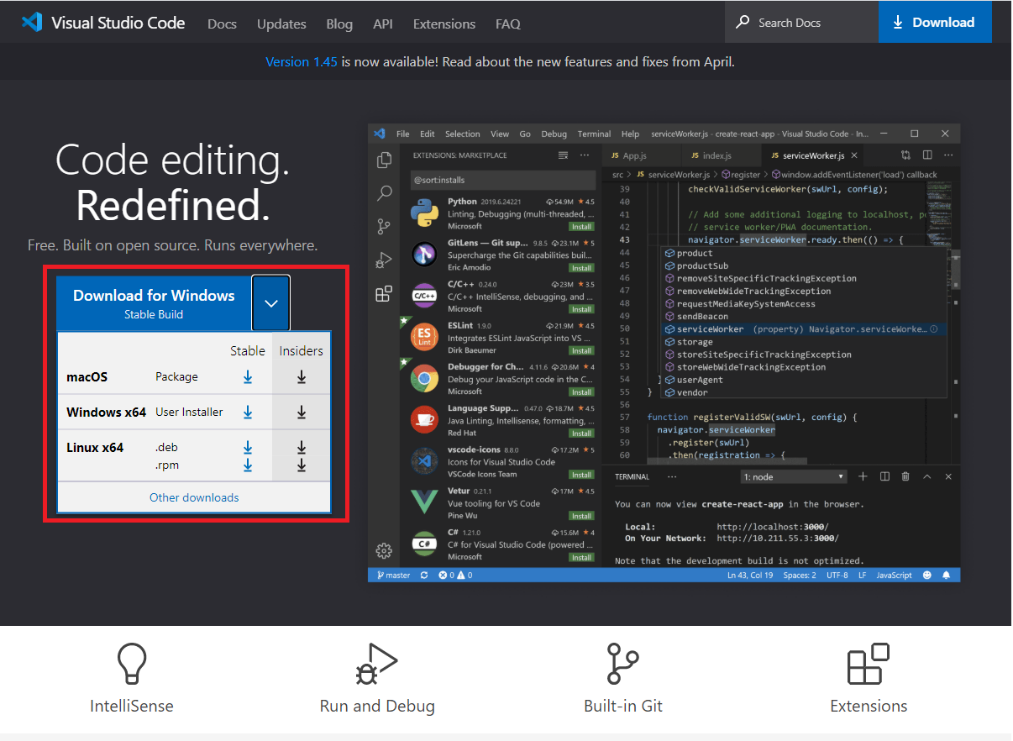
**Repository** là nơi sẽ ghi lại trạng thái của thư mục và file. Trạng thái được lưu lại đang được chứa như là lịch sử thay đổi của nội dung. Bằng việc đặt thư mục muốn quản lý lịch sử thay đổi dưới sự quản lý của repository, có thể ghi chép lại lịch sử thay đổi của thư mục và file trong thư mục đó.

* + Đầu tiên, repository của Git được phân thành 2 loại là remote repository và localmrepository.
* **Remote repository:** Là repository để chia sẻ giữa nhiều người và bố trí trên server chuyên dụng.
* **Local repository:** Là repository bố trí trên máy của bản thân mình, dành cho một người dùng sử dụng.

### Một số thao tác:

* Cài đặt VSCode.

Các bạn vào trang chủ vscode: https://code.visualstudio.com/, chọn phiên bản phù hợp với thiết bị của các bạn và tải về.

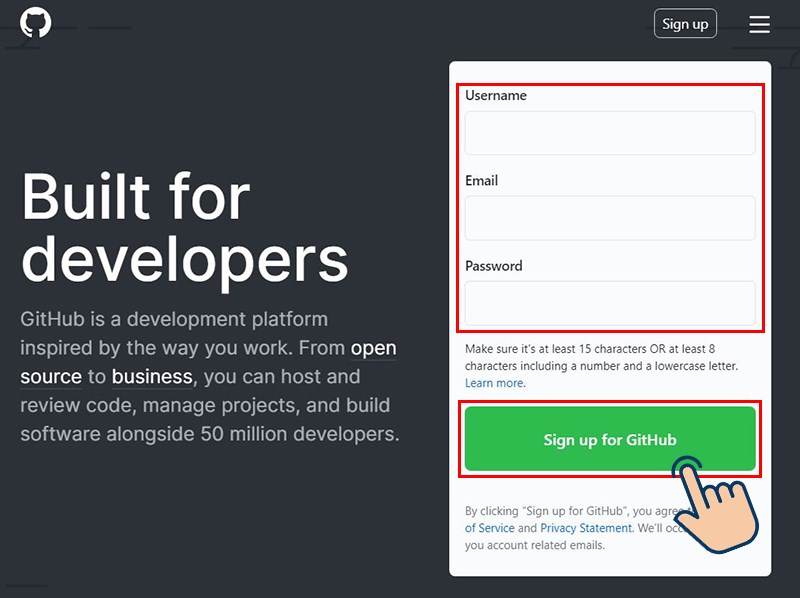


Hình 3: Trang cài đạt Visual Studio Code

Sau khi tài về, tiến hành chạy file cài đặt. Việc cài đặt rất đơn giản, chỉ cần Next – Next – Next là xong.

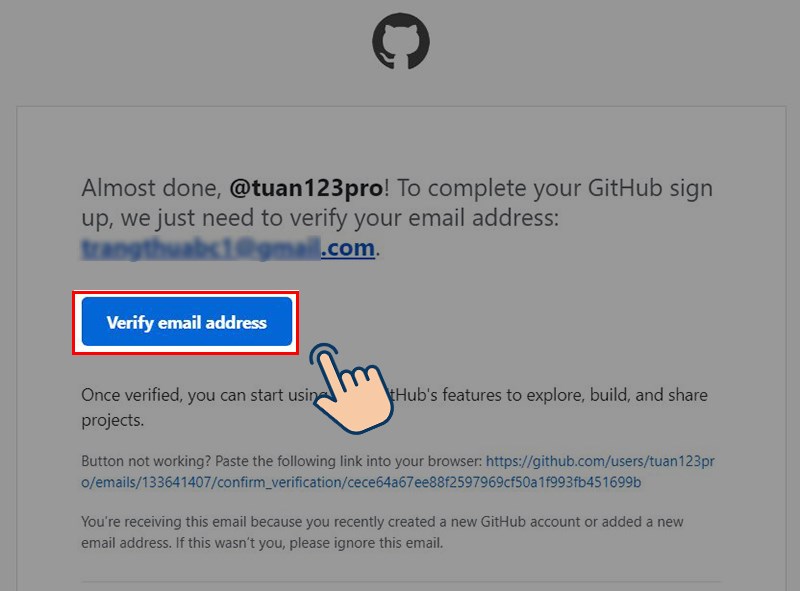
* Cài đặt Git.
* Đi đến trình cài đặt Git dành cho Windows và tải xuống phiên bản mới nhất.
* Khi trình cài đặt đã bắt đầu, hãy làm theo các hướng dẫn được cung cấp trong màn hình hướng dẫn **Cài đặt Git** cho đến khi quá trình cài đặt hoàn tất.
* Mở **command prompt** (hoặc **Git Bash**) rồi gõ git version để xác định Git đã được cài đặt thành công cùng phiên bản của nó.
* Đăng kí tài khoản Github/đăng nhập tài khoản có sẵn.

**Bước 1:** Đầu tiên bạn hãy vào trang web Gihub. Sau đó nhập đầy đủ **thông tin tài khoản** của bạn và nhấn **Sign up for Github.**



Hình 4: Nhập đầy đủ thông tin tài khoản của bạn và nhấn Sign up for Github

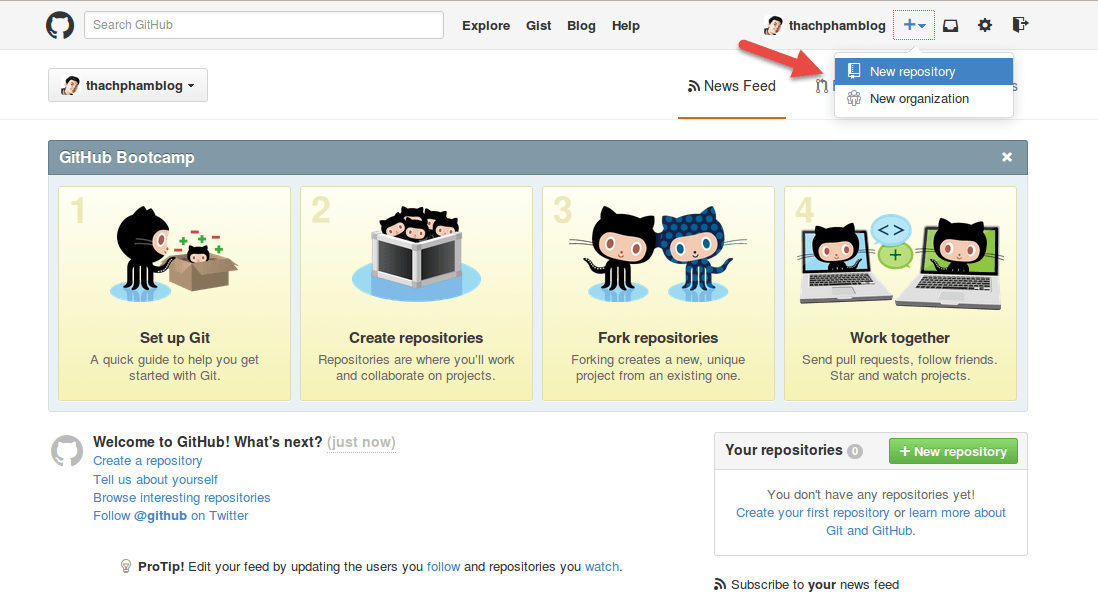
**Bước 2:** Lúc này bạn hãy vào **Email để kích hoạt tài khoản** của mình nhé.



Hình 5: Vào Email để kích hoạt tài khoản của mình

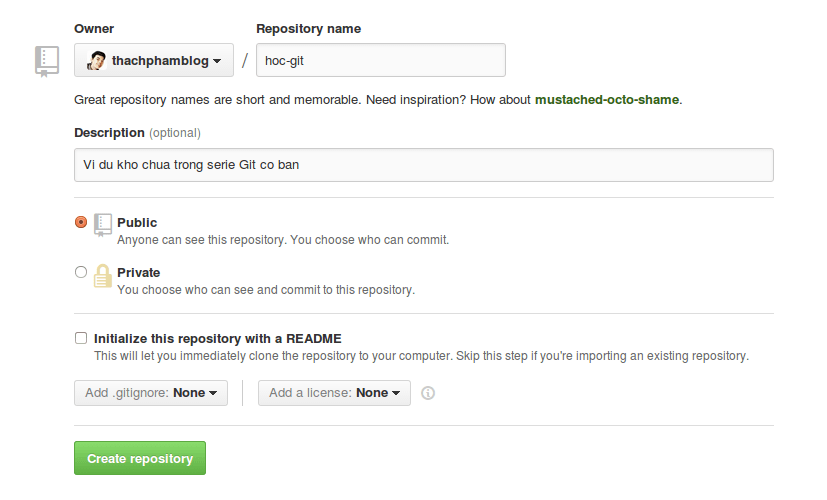
* Tạo Repository đầu tiên.

Trước tiên bạn cần đăng nhập vào Github, sau đó ấn vào dấu + trên menu và chọn New repository.

[](https://thachpham.com/wp-content/uploads/2015/04/github-create-repository.png)

Hình 6: Trang chủ Github

Bạn sẽ cần đặt tên cho kho chứa của bạn. Bạn có thể chọn loại kho chứa là **Public** (ai cũng có thể clone) và **Private** (chỉ có những người được cấp quyền mới có thể clone).

[](https://thachpham.com/wp-content/uploads/2015/04/github-create-repository-02.png)

Hình 7: Điền thông tin khi tạo Repository trên Github

Khi tạo xong nó sẽ dẫn bạn tới trang hướng dẫn làm việc với kho chứa vừa tạo. Và kho chứa của bạn bây giờ sẽ có địa chỉ là https://github.com/$user-name/$repository, ví dụ https://github.com/thachphamblog/hoc-git.

Clone cái kho chứa này về máy của mình bằng lệnh git clone địa\_chỉ.

**$** git clone https://github.com/thachphamblog/hoc-git

Cloning into 'hoc-git'...

warning: You appear to have cloned an empty repository.

Checking connectivity... done

Truy cập vào thư mục working tree (thư mục vừa clone repository về) và thử tạo ra một file tên là README.md, sau đó dùng lệnh git add để đưa file này vào Staging Area.

**$** cd hoc-git

**$** echo "# Huong dan Git co ban" > README.md

**$** git add README.md

**$** git commit -m "First commit on Github"

[master (root-commit) 6e729a4] First commit on Github

1 file changed, 1 insertion(+)

create mode 100644 README.md

Tuy nhiên sau khi commit xong, tập tin đã được commit sẽ vẫn không thể xuất hiện trong kho chứa trên Github mà bạn phải làm thêm một việc nữa đó là dùng lệnh git push để đẩy các tập tin đã được commit lên Github. Lưu ý rằng **bạn sẽ cần nhập tài khoản và mật khẩu Github**.

**$** git push origin master

Counting objects: 3, done.

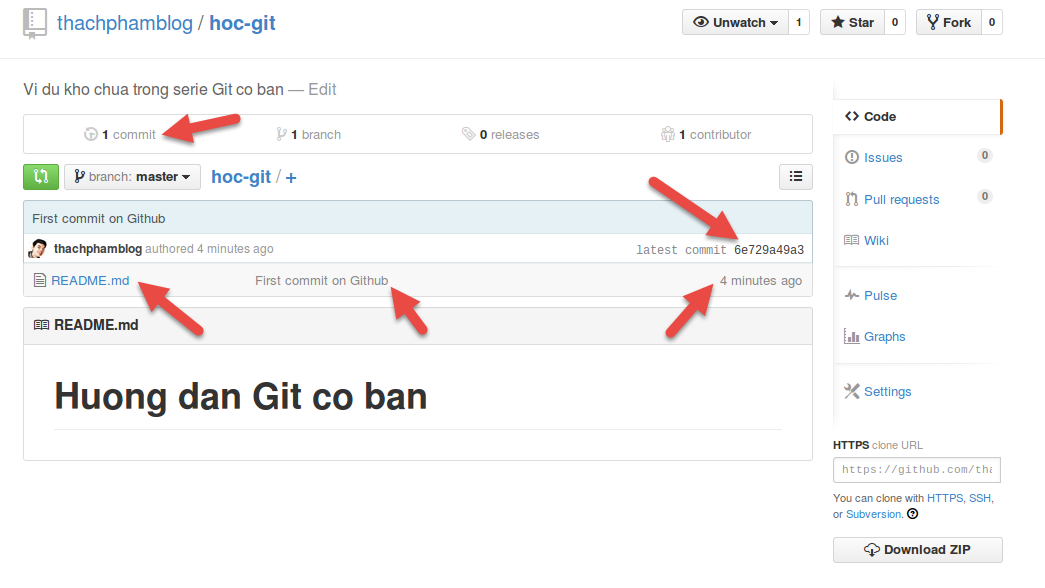
Writing objects: 100% (3/3), 244 bytes | 0 bytes/s, done.

Total 3 (delta 0), reused 0 (delta 0)

To https://github.com/thachphamblog/hoc-git

\* [new branch] master -> master

Trong đó origin nghĩa là tên remote và master là tên branch, hai cái này mình sẽ giải thích kỹ hơn ở bài riêng của nó. Bây giờ bạn có thể kiểm tra kho chứa của bạn trên Github.

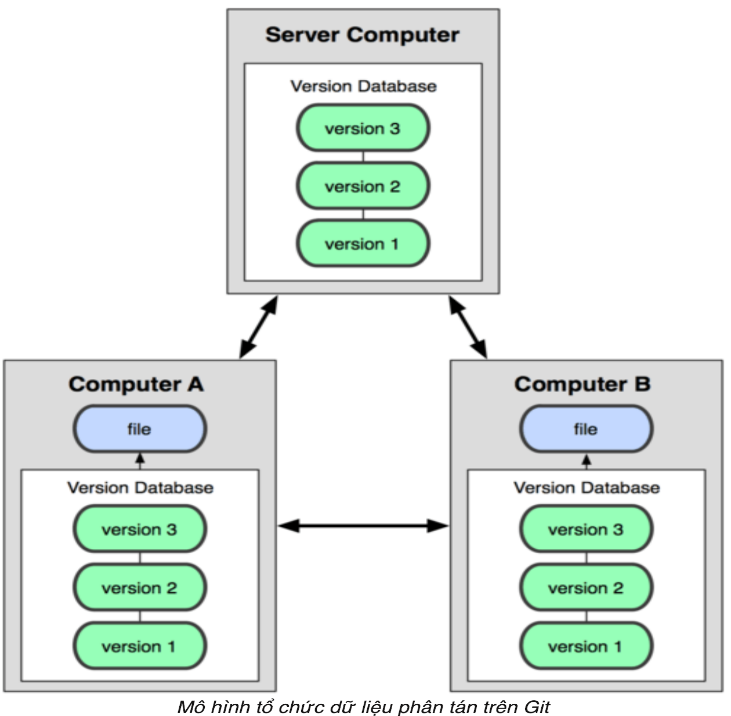
[](https://thachpham.com/wp-content/uploads/2015/04/github-create-repository-03.png)

Hình 8: Kết quả sau khi push mã nguồn đã được commit lên Github

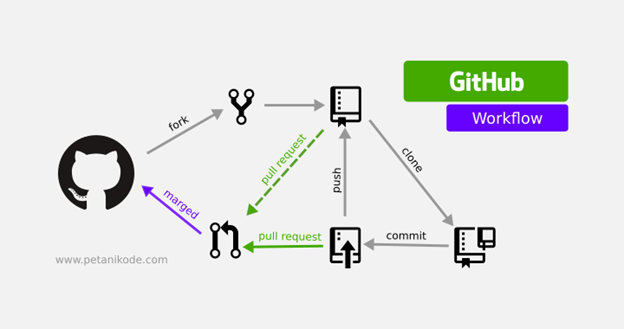
Có thể bạn sẽ thấy mỗi khi push lên Github thì nó hiển thị ra thông báo quá dài dòng, bạn có thể cho ẩn các thông báo đi mà chỉ hiển thị dòng khai báo username và password trên Github thì có thể thiết lập với lệnh này.

git config --global push.default simple

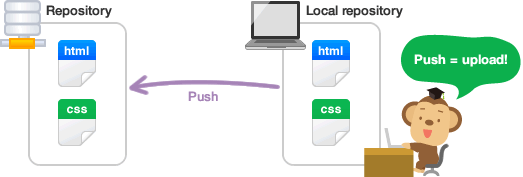
5. Kết nối thư mục mã nguồn với Repository.



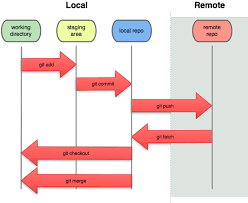
6. Kết nối với repository từ xa.



7. Push thay đổi từ local lên remote repository.



8. Kéo tất cả thay đổi từ remote repository về local.



### Một số lệnh Git cơ bản:

* **Git config:** Để set user name và email của bạn trong main configuration file.
* **Git init:** Là câu lệnh được sử dụng khi chúng ta bắt đầu một project mới. Khi thực hiện câu lệnh, nó sẽ giúp chúng ta bắt đầu một repository mới, sau đó nó sẽ là nơi để quản lý source code của các developers.
* **Git add:** Tất cả các file code mới hoặc file được chỉnh sửa sẽ được thêm vào repository.
* **Git commit:** Là câu lệnh được sử dụng phổ biến nhất. Khi được thực thi, câu lệnh này sẽ giúp chúng ta lưu những sự thay đổi ở các file trong staging area xuống repository.
* **Git branch:** Dùng để tạo ra nhánh mới.
* **Git remote:** Để kết nối kho lưu trữ cục bộ với kho lưu trữ từ xa. Một kho lưu trữ từ xa có thể được đặt tên để tránh phải nhớ URL của kho lưu trữ.
* **Git clone:** Để tạo bản sao làm việc cục bộ của kho lưu trữ từ xa hiện có, hãy sử dụng git clone để sao chép và tải kho lưu trữ về máy tính. Sao chép tương đương với git init khi làm việc với kho lưu trữ từ xa. Git sẽ tạo một thư mục cục bộ với tất cả các tệp và lịch sử kho lưu trữ.
* **Git pull:** Để tải phiên bản mới nhất của kho lưu trữ, hãy chạy git pull. Thao tác này kéo các thay đổi từ kho lưu trữ từ xa sang máy tính cục bộ.
* **Git push:** Gửi các commit cục bộ đến kho lưu trữ từ xa. git push yêu cầu hai tham số: kho lưu trữ từ xa và tên nhánh.
* **Git log:** Giúp bạn xem lại thông tin lịch sự commit, nhằm giám sát sự thay đổi của dự án.

## GitHub:

### Giới thiệu về GitHub.

**GitHub** là một dịch vụ cung cấp [kho lưu trữ mã nguồn](https://vi.wikipedia.org/w/index.php?title=Kho_l%C6%B0u_tr%E1%BB%AF_m%C3%A3_ngu%E1%BB%93n&action=edit&redlink=1) [Git](https://vi.wikipedia.org/wiki/Git_(ph%E1%BA%A7n_m%E1%BB%81m)) dựa trên nền web cho các dự án phát triển phần mềm. GitHub cung cấp cả phiên bản trả tiền lẫn miễn phí cho các tài khoản. Các dự án [mã nguồn mở](https://vi.wikipedia.org/wiki/Ph%E1%BA%A7n_m%E1%BB%81m_ngu%E1%BB%93n_m%E1%BB%9F) sẽ được cung cấp kho lưu trữ miễn phí.

### Lịch sử GitHub:

Một trong những cách tốt nhất để chia sẻ những gì bạn đã học được với người khác là đưa mã của bạn lên GitHub. Github được phát triển vào ngày 19 tháng 10 năm 2007. Sau đó trang web này được đưa ra vào tháng 4 năm 2008 bởi các nhà nhà phát triển Tom Preston-Werner Chns Wanstrath, và PJ Hyett. GitHub vừa là một trang web vừa là một dịch vụ hỗ trợ phát triển phần mềm bằng cách cho phép bạn lưu trữ mã của mình trong các vùng chứa, được gọi là kho lưu trữ và bằng cách theo dõi các thay đổi được thực hiện đối với mã của bạn. Ngoài ra, nó cung cấp dịch vụ lưu trữ và các công cụ để xây dựng, kiểm tra và triển khai mã.

### Tính năng của GitHub:

* Wiki, issue, thống kê, đổi tên project, project được đặt vào namespace là user.
* Watch project: theo dõi hoạt động của project của người khác. Xem quá trình người ta phát triển phầm mềm thế nào, project phát triển ra sao.
* Follow user: theo dõi hoạt động của người khác.

### Lợi ích của GitHub đối với lập trình viên:

* Quản lý source code dễ dàng.
* Github giúp cải thiện kỹ năng code.
  + Github là một kho tài nguyên tuyệt vời.