**GIỚI THIỆU VỀ GIT VÀ GITHUB**

Icon

Description automatically generated

Nhóm QWERTY

*Thành viên nhóm:*

*1/Cấn Đức Quang (nhóm trưởng)*

*2/Dương Minh Thái*

*3/Trương Đức Thiện*

*4/Nguyễn Duy Hào*

*5/Huỳnh Nguyễn Vân Khánh*

**MỤC LỤC**

**Chương 1: Giới thiệu về Git, Github và giao diện cơ bản 2**

Một vài thông tin về Github và Git 2

Tính năng của Github va Git 3

Giao diện cơ bản của Github và 3

**Chương 2: Các tập lệnh cơ bản trong Git 5**

Một số thuật ngữ cần nhớ 5

Các lệnh cơ bản được sử dụng trong Git 8

**Chương 3: Tổng kết 11**

**CHƯƠNG 1: GIỚI THIỆU VỀ GIT, GITHUB VÀ GIAO DIỆN CƠ BẢN**

**1/Một số thông tin về Github và Git:**

* Git là tên gọi là một hệ thống quản lý phiên bản phân tán phổ biến nhất hiện nay **(Distributed Version Control System – DVCS**).
  + **DVCS** nghĩa là hệ thống giúp mỗi máy tính có thể lưu trữ nhiều phiên bản khác nhau của một mã nguồn được nhân bản **(clone)** từ một kho chứa mã nguồn **(repository)**, mỗi thay đổi vào mã nguồn trên máy tính sẽ có thể ủy thác **(commit)** rồi đưa lên máy chủ nơi đặt kho chứa chính. Và một máy tính khác (nếu họ có quyền truy cập) cũng có thể clone lại mã nguồn từ kho chứa hoặc clone lại một tập hợp các thay đổi mới nhất trên máy tính kia.
* Còn GitHub có đầy đủ những tính năng của Git, ngoài ra nó còn bổ sung những tính năng về social để các developer tương tác với nhau.

**2/Tính năng của Git và Github:**

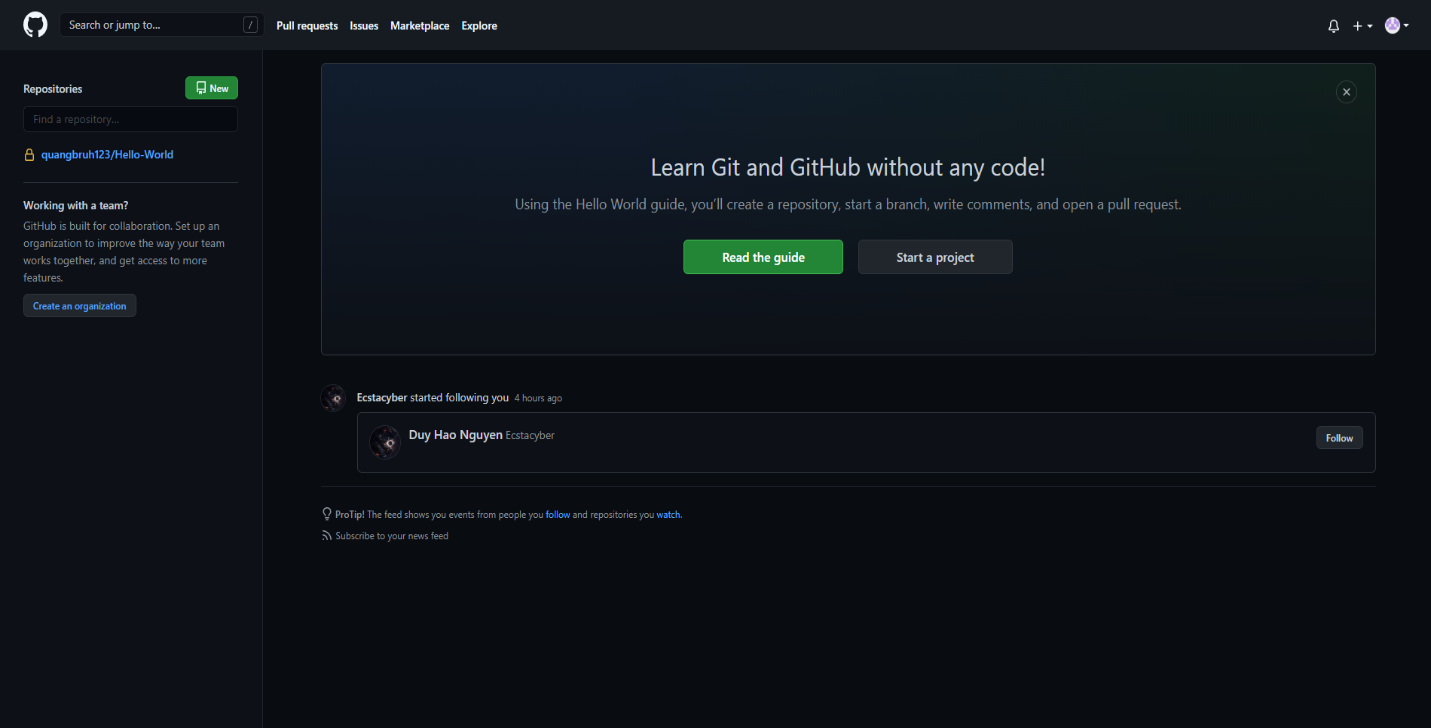
-Về cơ bản Git và Github có các tính năng như nhau:

**+ GitHub** được coi là một mạng xã hội dành cho lập trình viên lớn nhất và dễ dùng nhất với các tính năng cốt lõi như:

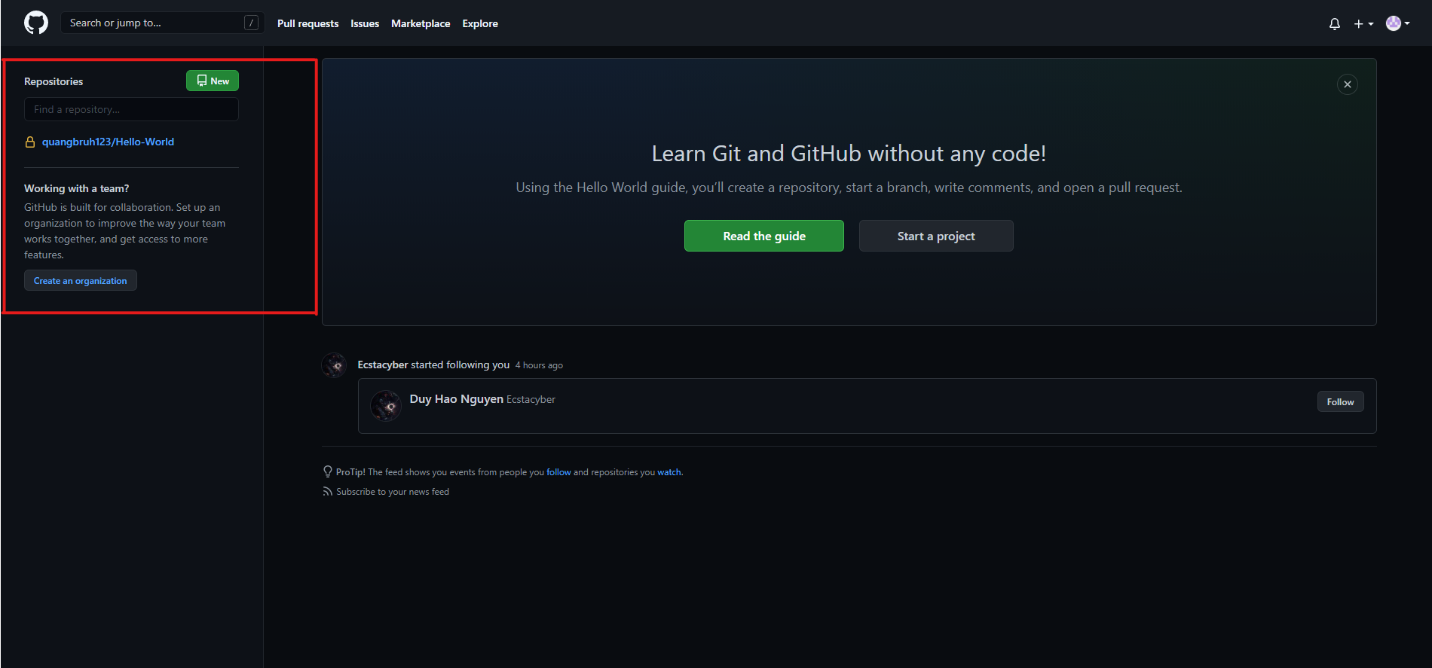
1. Wiki, issue, thống kê, đổi tên project, project được đặt vào namespace là user.
2. Watch project: theo dõi hoạt động của project của người khác. Xem quá trình người ta phát triển phầm mềm thế nào, project phát triển ra sao.
3. Follow user: theo dõi hoạt động của người khác.

Có 2 cách tiếp cận GitHub: Tạo project của riêng mình Contribute cho project có sẵn: fork project có sẵn của người khác, sửa đổi, sau đó đề nghị họ cập nhật sửa đổi của mình **(tạo pull request).**

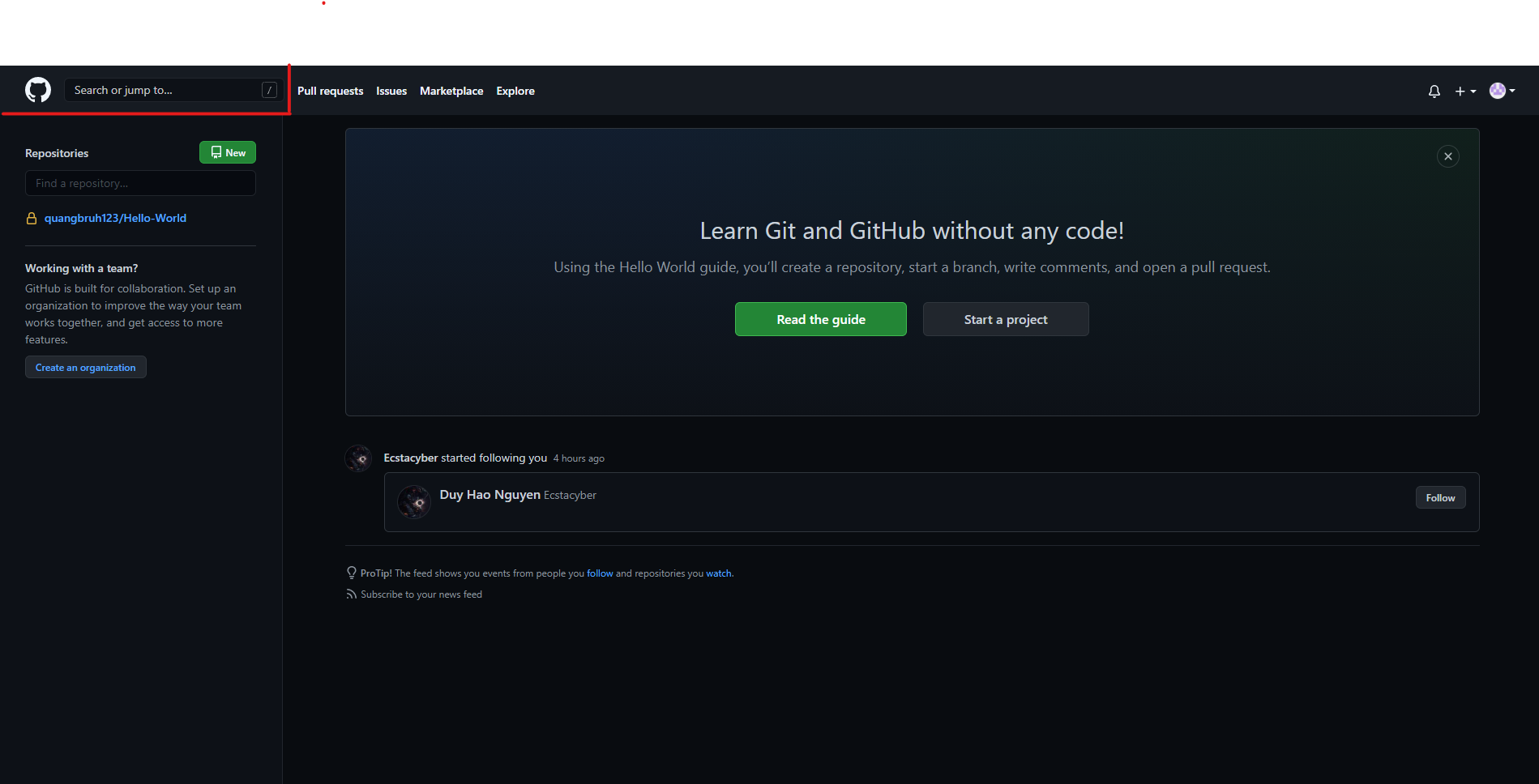
**3/Giao diện cơ bản của Github:**

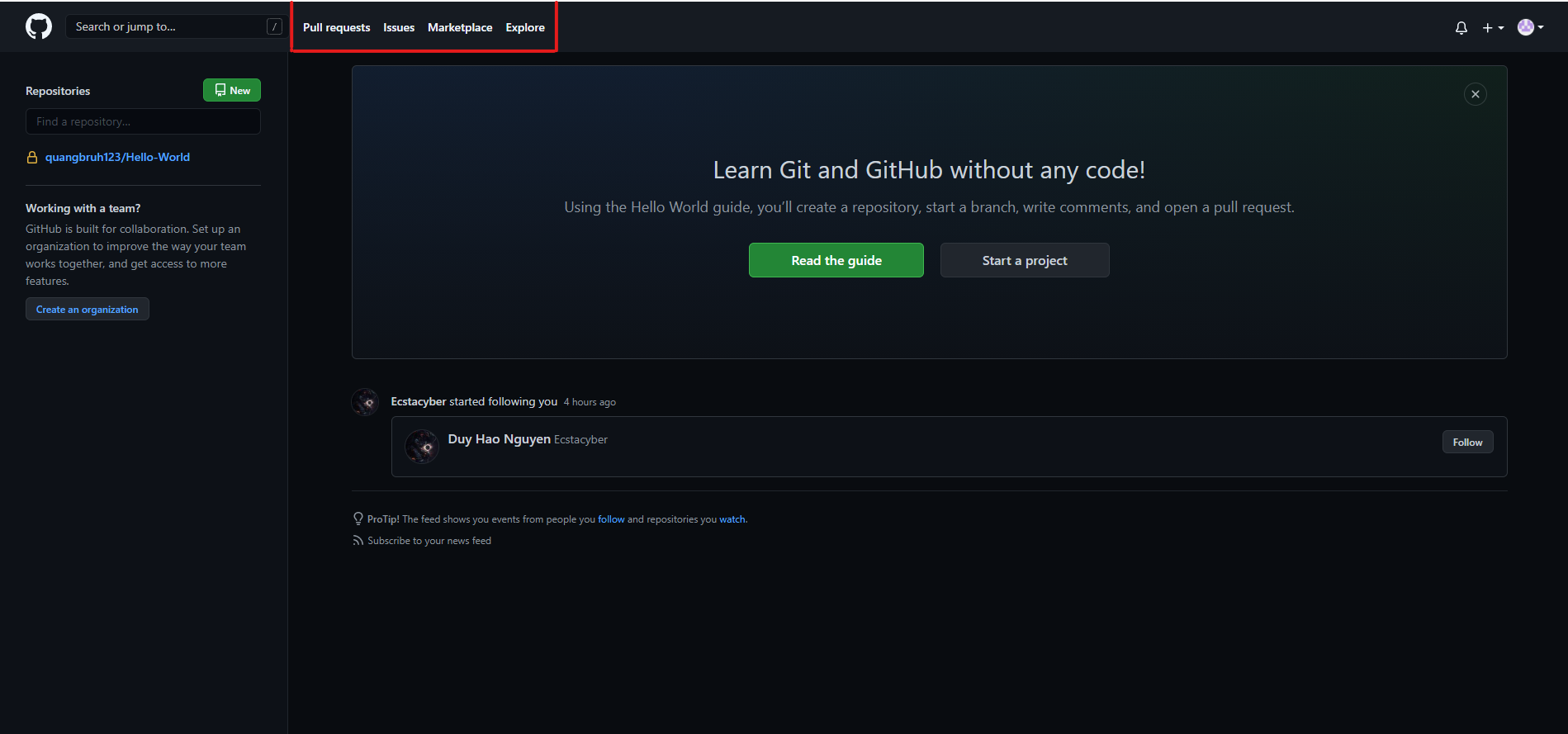


Đây là giao diện cơ bản của Github, gồm có các phần chính như sau:

1/Danh mục các repository (kho chứa mã nguồn) của bạn:

2/**Thanh tìm kiếm (search bar)** làm nhiệm vụ tìm kiếm mọi thứ liên quan đến các thao tác của bạn trên Github.



3/Các thẻ (tab) của Github bao gồm: Pull requests, Issues, Marketplace, Explore.

**CHƯƠNG 2: CÁC TẬP LỆNH CƠ BẢN TRONG GIT**

**1/Một số thuật ngữ cần nhớ:**

Việc làm việc với các lệnh ở trong Git sẽ dễ dàng hơn nếu bạn có thể nắm được một số thuật ngữ quan trọng sau đây:

**1. Branch**

Các Branch (nhánh) đại diện cho các phiên bản cụ thể của một kho lưu trữ tách ra từ project chính của bạn.

Branch cho phép bạn theo dõi các thay đổi thử nghiệm bạn thực hiện đối với kho lưu trữ và có thể hoàn nguyên về các phiên bản cũ hơn.

**2. Commit**

Một commit đại diện cho một thời điểm cụ thể trong lịch sử dự án của bạn. Sử dụng lệnh commit kết hợp với lệnh git add để cho git biết những thay đổi bạn muốn lưu vào local repository.

**3. Checkout**

Sử dụng lệnh git checkout để chuyển giữa các branch. Chỉ cần nhập git checkout theo sau là tên của branch bạn muốn chuyển đến hoặc nhập git checkout master để trở về branch chính (master branch).

**4. Fetch**

Lệnh git fetch tìm nạp các bản sao và tải xuống tất cả các tệp branch vào máy tính của bạn. Sử dụng nó để lưu các thay đổi mới nhất vào kho lưu trữ của bạn. Nó có thể tìm nạp nhiều branch cùng một lúc.

**5. Fork**

Một fork là một bản sao của một kho lưu trữ (repository). Các lập trình viên thường tận dụng lợi ích của fork để thử nghiệm các thay đổi mà không ảnh hưởng đến dự án chính.

**6. Head**

Các commit ở đầu của một branch được gọi là head. Nó đại diện cho commit mới nhất của repository mà bạn hiện đang làm việc.

**7. Index**

Bất cứ khi nào bạn thêm, xóa hoặc thay đổi một file, nó vẫn nằm trong chỉ mục cho đến khi bạn sẵn sàng commit các thay đổi. Nó như là khu vực tổ chức (stagging area) cho Git. Sử dụng lệnh git status để xem nội dung của index của bạn.

Những thay đổi được tô sáng bằng màu xanh lá cây đã sẵn sàng để được commit trong khi những thay đổi màu đỏ thì chưa.

**8. Master**

Master là nhánh chính của tất cả các repository của bạn. Nó nên bao gồm những thay đổi và commit gần đây nhất.

Nhánh Master

**9. Merge**

Lệnh git merge kết hợp với các yêu cầu kéo (pull requests) để thêm các thay đổi từ nhánh này sang nhánh khác.

**10. Origin**

Origin là phiên bản mặc định của repository. Origin cũng đóng vai trò là bí danh hệ thống để liên lạc với nhánh chính.

Lệnh git push origin master để đẩy các thay đổi cục bộ đến nhánh chính.

**11. Pull**

Pull requests thể hiện các đề xuất thay đổi cho nhánh chính. Nếu bạn làm việc với một nhóm, bạn có thể tạo các pull request để yêu cầu người bảo trì kho lưu trữ xem xét các thay đổi và hợp nhất chúng.

Lệnh git pull được sử dụng để thêm các thay đổi vào nhánh chính.

**12. Push**

Lệnh git push được sử dụng để cập nhật các nhánh từ xa với những thay đổi mới nhất mà bạn đã commit.

**13. Rebase**

Lệnh git rebase cho phép bạn phân tách, di chuyển hoặc thoát khỏi các commit. Nó cũng có thể được sử dụng để kết hợp hai nhánh khác nhau.

**14. Remote**

Một Remote (kho lưu trữ từ xa) là một bản sao của một chi nhánh. Remote giao tiếp ngược dòng với nhánh gốc (origin branch) của chúng và các Remote khác trong kho lưu trữ.

**15. Repository**

Kho lưu trữ Git chứa tất cả các tệp dự án của bạn bao gồm các branch, tags và commit.

**16. Stash**

Lệnh git stash sẽ loại bỏ các thay đổi khỏi chỉ mục của bạn và xóa stashes chúng đi sau.

Nó có ích nếu bạn muốn tạm dừng những gì bạn đang làm và làm việc khác trong một khoảng thời gian. Bạn không thể đặt stash nhiều hơn một bộ thay đổi ở cùng một thời điểm.

**17. Tags**

Tags cung cấp cho bạn một cách để theo dõi các commit quan trọng. Các tags nhẹ chỉ đơn giản đóng vai trò là con trỏ trong khi các tags chú thích được lưu trữ dưới dạng các đối tượng đầy đủ.

**19. Upstream**

Trong ngữ cảnh của Git, upstream đề cập đến nơi bạn push các thay đổi của mình, thường là nhánh chính (master branch).

**2/Các lệnh cơ bản được sử dụng trong Git:**

Ở đây chúng ta sẽ đề cập tới một số lệnh thông dụng nhất và thường gặp trong công việc:

**1) git config**

Tác dụng : Git cung cấp sẵn git config cho phép bạn xem hoặc chỉnh sửa các biến cấu hình để quản lý toàn bộ các khía cạnh của Git như giao diện hay hoạt động. Có

Cách xài : -Việc đầu tiên bạn nên làm khi cấu hình Git là chỉ định tên tài khoản và địa chỉ e-mail. Điều này rất quan trọng vì mỗi Git sẽ sử dụng chúng cho mỗi lần commit, những thông tin này được gắn bất di bất dịch vào các commit:

git config -- global user.name và git config -- global user.email.

-Ngoài ra git còn cung cấp một số lệnh config khác liên quan tới trình soạn thảo (dùng để soạn thảo các dòng lệnh), kiểm tra các cấu hình cài đặt của git trên máy bạn.

**2) git init**

Tác dụng : Khởi tạo 1 .git ở trong thư mục gốc ở local, là một thư mục ẩn và chứa các file liên quan đến việc liên kết với repository ở trên server

Cách xài: git init trong thư mục gốc của dự án.

**3) git clone**

Tác dụng: Là lệnh thực hiện việc tạo ra một bản sao của repository về máy tính đang local. Và bằng cách đó bạn sẽ có được tất cả lịch sử đã xảy ra trên repository mà server đang lưu trữ và hoàn toàn có thể quay lại , undo lại từ bất kỳ thời điểm commit nào.

Bạn có thể sử dụng clone để copy từ thư mục này sang thư mục khác, hoặc copy một repository thông qua một Url (VD: Github)

Cách xài: git clone <:clone git url:>

**5) git add**

Tác dụng: Sau khi bạn thay đổi source code: thêm mới, sửa, xoá files,… Bạn cần phải cập nhật lên Staging Area để chuẩn bị cho việc commit. Để cập nhật hết các files:

Cách xài: git add

**6) git commit**

Tác dụng: - Commit là thao tác báo cho hệ thống biết bạn muốn lưu lại trạng thái hiện hành, ghi nhận lại lịch sử các xử lý như đã thêm, xóa, cập nhật các file hay thư mục nào đó trên repository.

Khi thực hiện commit, trong repository sẽ ghi lại sự khác biệt từ lần commit trước với trạng thái hiện tại. Các commit ghi nối tiếp với nhau theo thứ tự thời gian do đó chỉ cần theo vết các commit thì có thể biết được lịch sử thay đổi trong quá khứ.

* Khi bạn thực hiện commit, hệ thống đều yêu cầu bạn phải nhập vào commit message để ghi chú tóm tắt là trong lần commit này là bạn đã thực hiện những thay đổi nào, có ý nghĩa thế nào,…

Cách dùng: git commit -m ”Đây là message, bạn dùng để note những thay đổi để sau này dễ dò lại”

**7)git clone**

Tác dụng: sao chép một kho chứa đã tồn tại

Cách dùng: $ git clone https://github.com/user/repository.git

Câu lệnh trên sẽ tạo một thư mục mới có tên giống trên của repo.

**8)git branch**

Khi sử dụng Git, bạn có thể tạo ra nhiều nhánh (branch) khác nhau. Câu lệnh Git này dùng để kiểm tra branch hiện tại:

$ git branch

Để tạo mới một branch:

$ git branch <name\_branch>

Để chuyển và tạo mới:

$ git branch -b <name\_branch>

**9)Gộp nhánh**

Sau một thời gian cập nhật các file và push lên git trên branch mới, bây giờ mình cần ghép (merge) code lại vào nhánh gốc (master). Trước tiên, cần phải checkout ra khỏi branch hiện tại cần gộp để vào branch master, sau đó thì dùng lệnh merge để ghép branch mới vào master:

$ git checkout master

$ git merge <new\_branch>

**10)Xem lại lịch sử commit**

$ git log

Lệnh git log sẽ cho bạn biết về người commit, ngày giờ, message của những lần commit đó.

**11)Xem thay đổi trước khi push**

$ git diff

Lệnh này giúp bạn biết những gì đã được thay đổi giữa nhánh hiện tại và nhánh trước nó.

**12)Gộp commit**

$ git rebase -i HEAD~

Sau dấu ~ là số commit bạn muốn gộp. Sau khi gõ lệnh này một cửa sổ trình soạn thảo hiện ra. Thay đổi ký tự pick của dòng các dòng sau dòng đầu thành s rồi lưu lại/kết thúc. Khi đó, trình soạn thảo để chỉnh sửa giải thích commit thiết lập cho commit sau khi đã tổng hợp sẽ được hiển thị, nên hãy chỉnh sửa lưu lại/kết thúc.

**13)Pull từ remote repository**

$ git pull origin master

Lệnh trên sẽ gộp những thay đổi mới kéo về từ máy chủ từ xa với nhánh hiện tại trên máy local

**CHƯƠNG 3: TỔNG KẾT**

Có thể nói rằng, Git được sử dụng rộng tãi trong các công ty ngày nay. Nhờ các điểm vượt trội so với SVN trước đây mà Git rất được ưu ái bởi các lập trình viên trên toàn thế giới.

Link tài liệu tham khảo:

Tài liệu tham khảo git:

[***https://try.github.io/***](https://try.github.io/)

***https://www.codecademy.com/learn/learn-git***

[***https://topdev.vn/blog/github-la-gi/#tinh-nang-cua-github***](https://topdev.vn/blog/github-la-gi/#tinh-nang-cua-github)

***https://topdev.vn/blog/git-la-gi/***