**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

🙤🙧🟍🙥🙦



**ĐỀ TÀI: GIT & GITHUB**

**MÔN HỌC: KĨ NĂNG NGHỀ NGHIỆP**

Nhóm thực hiện: nhóm 5

|  |
| --- |
| Sinh viên thực hiện:  Huỳnh Thị Mỹ Duyên 19521438  Nguyễn Huỳnh Thanh Trúc 19520314 Nguyễn Thị Thảo Hồng 20520192  Phạm Lê Dịu Ái 20520368 |

**TP. HỒ CHÍ MINH – 03/2021**

**MỤC LỤC**

CHƯƠNG 1: GIỚI THIỆU VỀ GIT & GITHUB2

**1.1 Git là gì 2**

**1.2 Github là gì 2**

**1.3 Sự khác nhau giữa git và github 2**

**1.4 Giao diện cơ bản github2**

CHƯƠNG 2: CÁC TẬP LỆNH CƠ BẢN6

CHƯƠNG 3: LỢI ÍCH CỦA GITHUB8

CHƯƠNG 4: TỔNG KẾT12

CHƯƠNG 5: TÀI LIỆU THAM KHẢO13

# CHƯƠNG 1: GIỚI THIỆU VỀ GIT & GITHUB

* 1. **Git là gì?**

Git là phần mềm quản lý mã nguồn phân tán được ra mắt vào năm 2005 bởi Linus Torvalds (cha đẻ của Linux). Hiện nay là phần mềm quản lí mã nguồn phổ biến nhất thế giới.

Mã nguồn phân tán: tức là các dữ liệu được lưu trữ trên server của Git và người dùng kết nối để lấy thao tác dữ liệu từ server đó.

Ví dụ: Các dự án thường có nhiều nhà phát triển làm việc cùng lúc với nhau, với Git họ sẽ tránh được xung đột code giữa các nhà phát triển, và họ có thể quay về phiên bản cũ hơn nếu phần mới có quá nhiều lỗi.

* 1. **Github là gì?**

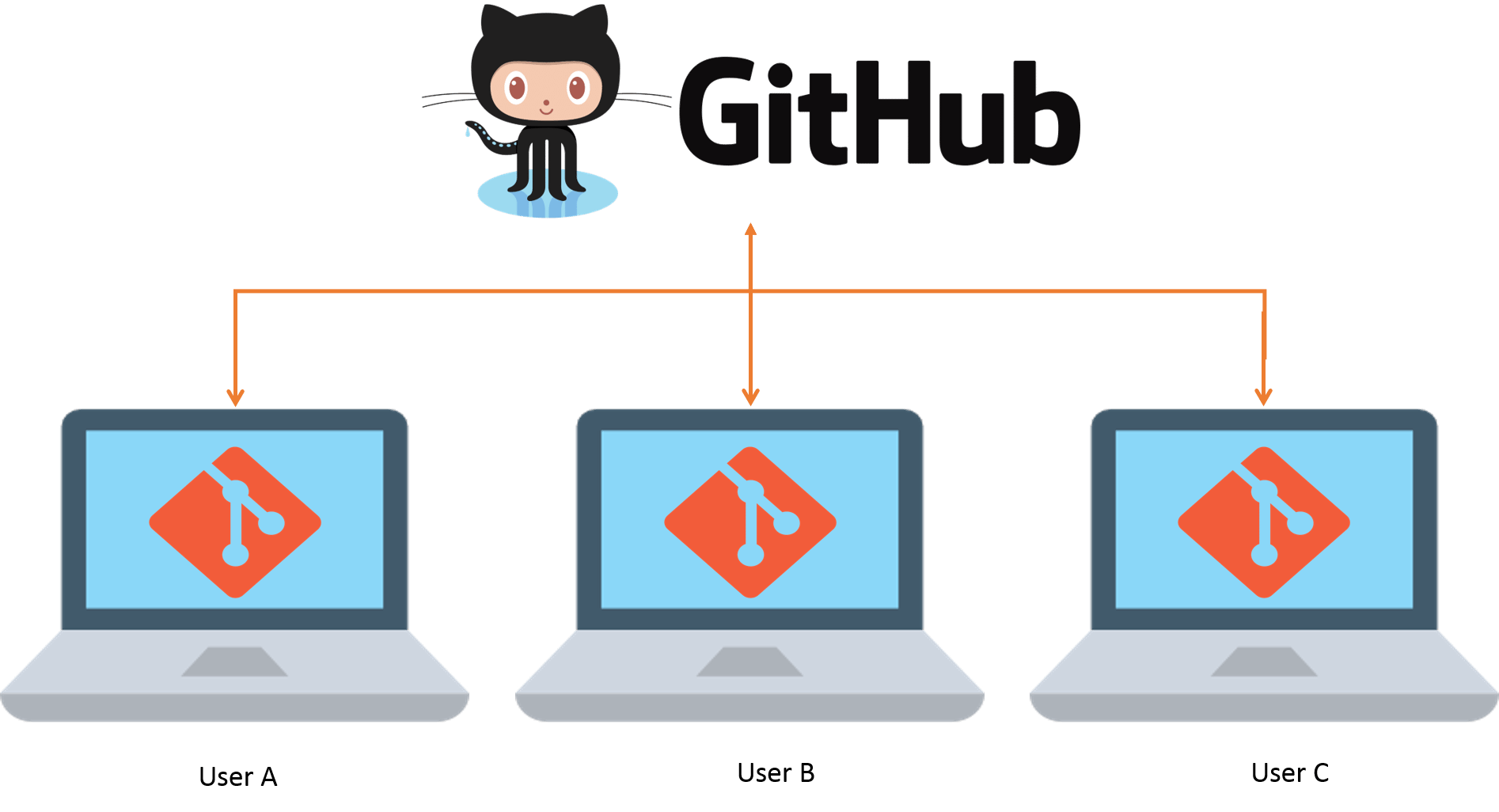
Github là dịch vụ cung cấp kho lưu trữ Git được phát triển trên nền Web ra mắt vào năm 2008 hiện thuộc sở hữu của Microsoft.

Git là phần mềm quản lí mã nguồn phân tán, Hub là trung tâm.

=> Github: trung tâm quản lí mã nguồn phân tán.

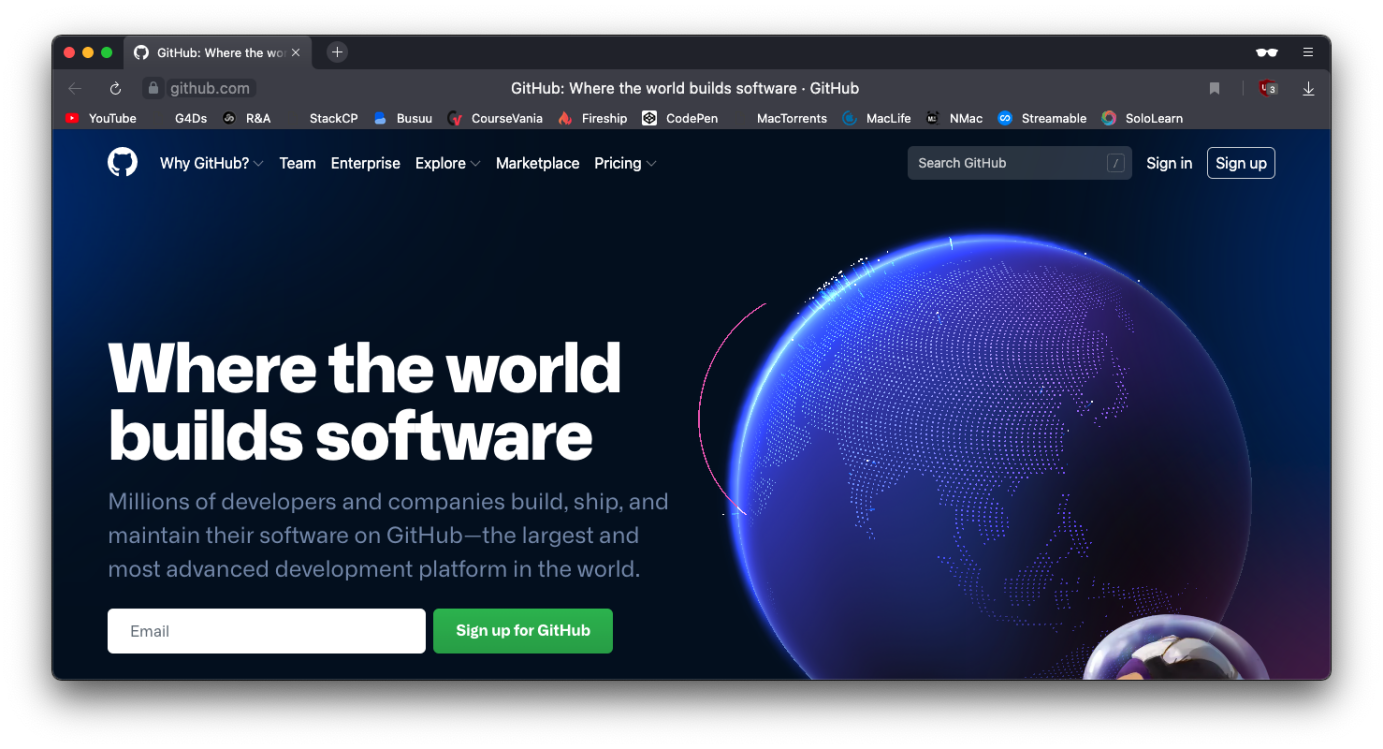
Github có đầy đủ tính năng của Git, ngoài ra nó còn bổ sung thêm nhiều tính năng như social networking, feeds, followers, discussion... giống như một mạng xã hội về các dự án mã nguồn mở.

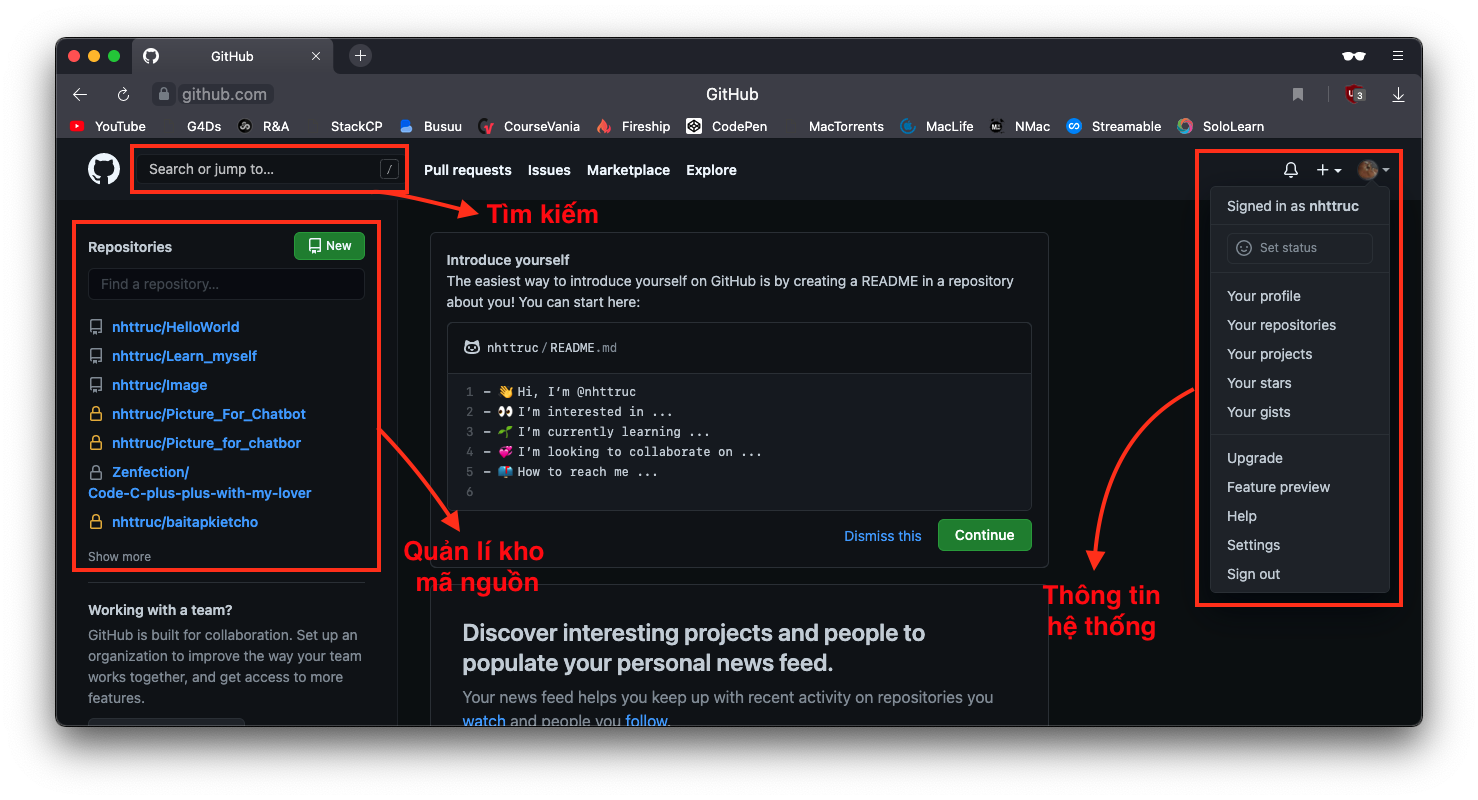
* 1. **Sự khác nhau giữa git và github:**

****Git là phần mềm quản lí mã nguồn phân tán, cụ thể hơn là một hệ thống kiểm soát phiên bản cho phép bạn theo dõi lịch sử mã nguồn của mình.

Github là một dịch vụ đám mây lưu trữ nhiều Git một cách trực quan, giúp ta quản lí nhiều Git tốt hơn.

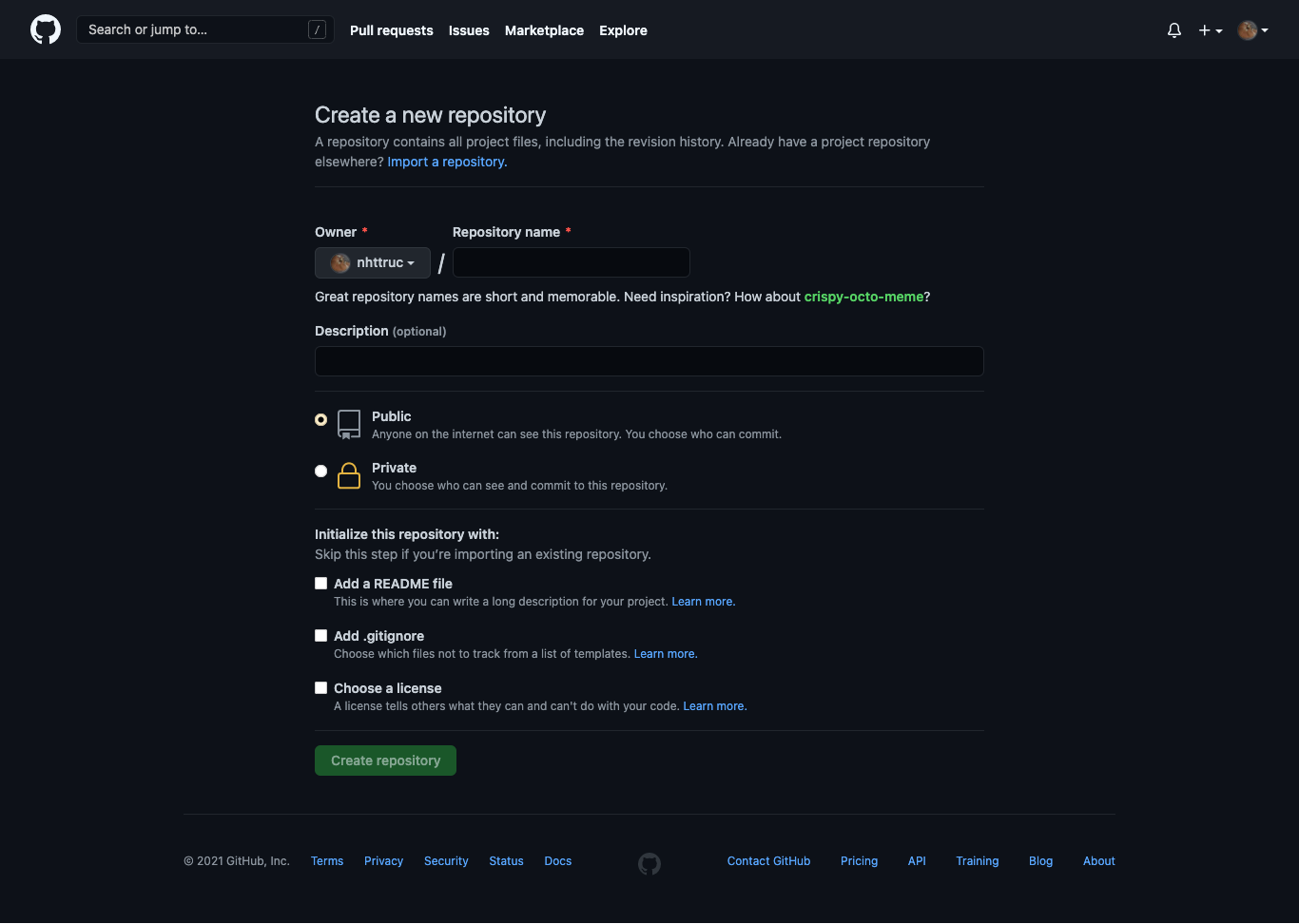
* 1. **Giao diện cơ bản của Github:**

Truy cập Github theo đường link: [github.com](https://github.com).

Sau đó đăng nhập vào tài khoản (Đăng ký nếu bạn chưa có tài khoản), và đây là màn hình chính của Github.

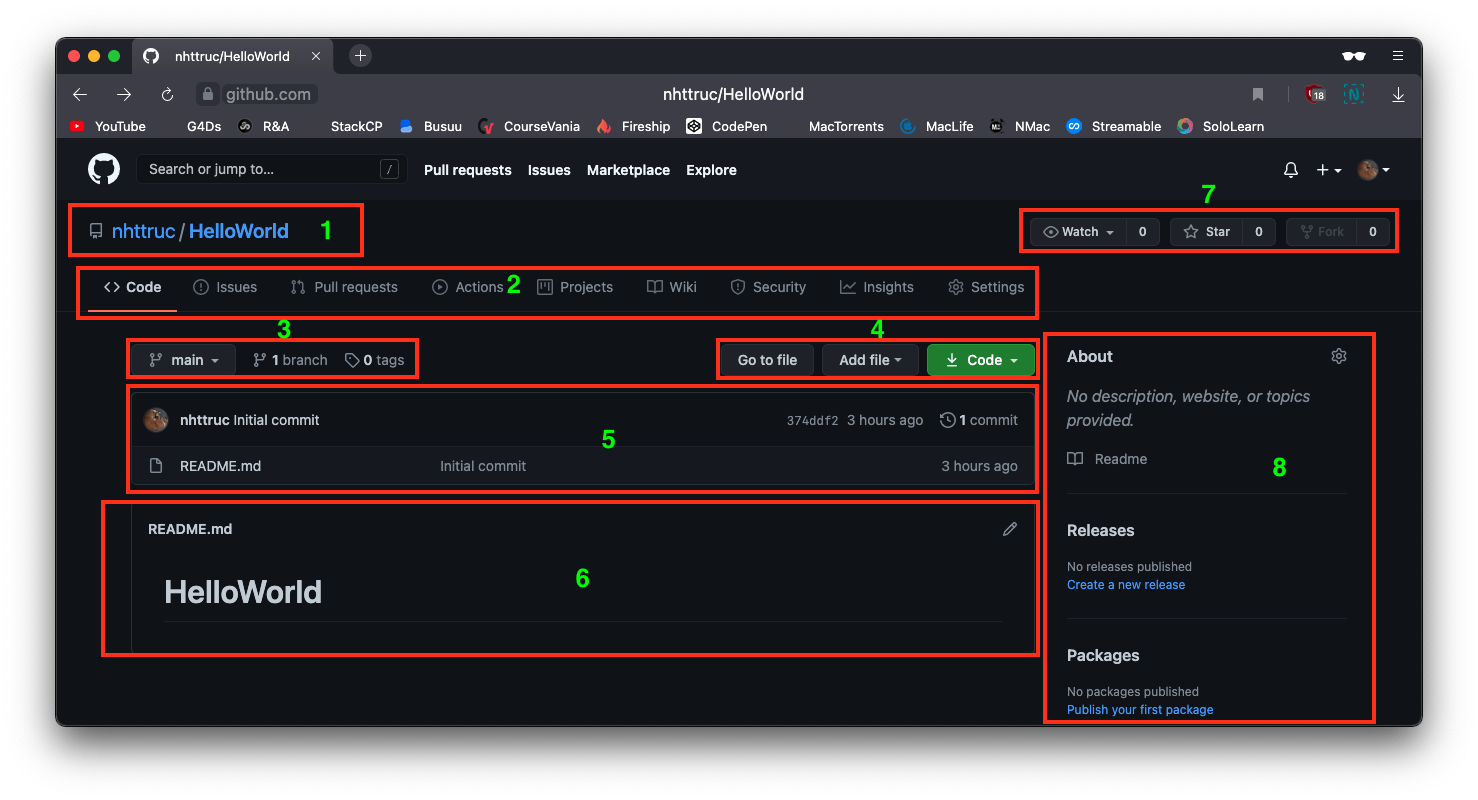
1. Tìm kiếm: Nơi tìm kiếm mọi dự án của tất cả mọi người dùng `Github`
2. Quản lý kho mã nguồn: Nơi tạo và truy cập các kho dự án cá nhân
3. Thông tin hệ thống: Nơi chỉnh sửa thông tin cá nhân và tuỳ chọn cài đặt

Để tạo kho dự án mới => ta bấm New trong khung Quản lí kho mã nguồn:



**💡Hướng dẫn sử dụng:**

* Owser: Tuỳ chọn người tạo ra kho dự án
* Repository name: Tên dự án (không được trùng tên các dự án hiện có)
* Description: Mô tả kho dự án (tuỳ chọn)
* Public & Private: Quyền truy cập dự án, nếu public ai cũng có thể xem, còn private chỉ bạn được xem
* Add a README file: thêm mặc định file README.md để hiển thị (như index.html của Web)
* Add .gitignore: thêm tuỳ chọn file không cần theo dõi.
* Choose a license: Chọn bản quyền cho kho dự án của bạn.

**Hướng dẫn sử dụng:**

1. Đường dẫn kho dự án của bạn
2. Các tính năng và tuỳ chọn của kho dự án

* Code: Cửa sổ quản lí mã nguồn
* Issues: Tạo các vấn đề để cộng đồng trao đổi
* Pull requests: Cho phép bạn nói với người khác về các thay đổi bạn đã đẩy lên kho Github
* Action: cho phép tạo workflows trên kho của bạn
* Project: Sắp xếp các vấn đề trong dự án của bạn
* Wiki: Viết nên lộ trình và miêu tả nhanh về dự án của bạn
* Security: Tuỳ chọn các vấn đề về bảo mật trong kho của bạn
* Insight: Theo dõi trực quan kho của bạn
* Settings: Tuỳ chọn cài đặt trong kho

1. Tạo và quản lí các nhánh của dự án
2. Thao tác dự án

* Go to file: Truy cập vào các file trong kho của bạn
* Add file: Thêm file vào kho của bạn
* Code: Clone hoặc tải dự án của bạn

1. Giao diện các file đang hiện có trong kho dự án
2. Hiển thị file `README.md` mặc định, viết bằng Markdown
3. Theo dõi kho dự án
4. Xuất bản phiên bản của bạn

* About: Miêu tả dự án của bạn
* Realease: Tải lên phiên bản sử dụng của bạn
* Package: Tải lên các gói của bạn

**CHƯƠNG 2**

**CÁC TẬP LỆNH CƠ BẢN CỦA GIT**

**2.1. Thiết lập chứng thực cá nhân**

$ git config --global user.name "User Name"

$ git config --global user.email "username@gmail.com"

Lưu ý: –global được sử dụng để áp dụng cho tất cả các projects. Nếu bạn ko sử dụng –global thì settings sẽ chỉ dùng cho riêng project đó.

**2.2. Tạo một kho chứa Git**

*$ git init*

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.

#### **2.3. Sao chép một kho chứa đã tồn tại**

*$ git clone https://github.com/user/repository.git*

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

**2.4. Nhánh trong git**

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>*

#### **2.5. Chuyển nhánh**

Trước khi muốn thay đổi source code, điều đầu tiên mà bạn cần phải làm là checkout một nhánh. Để checkout một nhánh, bạn dùng câu lệnh Git sau:

*$ git checkout <name\_branch>*

#### **2.6. Cập nhật thay đổi**

Sau khi bạn thay đổi source code: thêm mới, sửa, xoá files, … Bạn cần phải cập nhật lên Staging Area. Để cập nhật hết các files:

*$ git add.*

Sau lệnh add, bạn cần sử dụng câu lệnh Commit để đây thông tin thay đổi lên Local Respository:

*$ git commit -m "Message"*

#### **2.7. Cập nhật lên server**

Sau câu lệnh Commit, thông tin mới chỉ được cập nhật lên Local Repository. Nếu muốn cập nhật lên server thì bạn phải sử dụng câu lệnh push:

*$ git push origin <name\_branch>*

Ngoài ra, nếu chưa tồn tại remote trên server thì bạn cần phải add mới một remote trước rồi mới push:

*$ git remote add origin <remote\_url>*

*$ git push origin <name\_branch>*

#### **2.8. 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>*

#### **2.9. Xem lại lịch sử commit**

#### $ git log

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

#### **2.10. Xem thay đổi trước khi push**

*$ git diff*

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

#### **2.11. 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.

#### **2.12. 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: LỢI ÍCH CỦA GITHUB**

**3.1. Quản lí source code dễ dàng**

Khi bạn tạo một repo, toàn bộ source code của repo đó được lưu trên GitHub. Tại đây, bạn có thể coi lại quá trình mình đã làm việc thông qua các comment sau mỗi lần commit. Và cái hay ở đây, là nhiều người có thể cùng làm một repo.

Lợi ích đầu tiên, chính là bạn biết được ai đã commit và commit cái gì. Tiếp theo, source của bạn có thể phát triển theo nhiều nhánh. Nguyên tắc làm việc với các nhánh như thế này: Bạn có thể rẽ nhiều nhánh để phát triển project. Nhưng cuối cùng, bạn phải merge lại vào nhánh MASTER để ra được project hoàn chỉnh.

**3.2. Tracking sự thay đổi qua các version**

Khi có nhiều member cùng thực hiện một dự án thì khá là phức tạp để theo dõi revisons – ai thay đổi cái gì, lúc nào và mấy cái files đó được stored ở đâu. Đừng lo vì GitHub đã tính đến chuyện này giúp bạn, bằng cách luôn lưu lại những thay đổi bạn đã push lên repository. Cũng tương tự với Microsoft Word hay Google Drive, bạn có một lịch sử phiên bản để phòng trường hợp các phiên bản trước đó bị mất hay không được lưu.

**3.3. Markdown**

Markdown là một cách định dạng text trên web. Bạn có thể chỉnh sửa cách hiển thị của document, format từ như định dạng **in đậm**hay in nghiêng, thêm hình và tạo list những thứ bạn có thể làm với Markdown. Hầu hết, Markdown chỉ là đoạn text đơn thuần với những ký tự đặc biệt chèn vào, như # hay \*. Trong GitHub thì bạn có thể sử dụng Mardown ở những nơi: Git, Comments tại Issues và Pull Requests, các file có đuôi .md hay .markdown extension.

**3.4. Giúp chứng tỏ bạn là ai**

Chẳng thể phủ nhận những lời hay ý đẹp bạn viết trong CV là cần thiết. Nhưng Source code luôn là minh chứng tốt nhất để thể hiện bạn là developer thực thụ. Có thể nói, 1 phần GitHub “nho nhỏ” trong CV có thể đánh bóng vị trí của bạn, nổi bật hơn những ứng cử viên khác. Đối với nhà tuyển dụng, GitHub cũng giống như một chiếc máy liar-detech – phân biệt real developer với những kẻ “faker”.

Hãy đầu tư cho mình một tài khoản Github thật ấn tượng và đưa đường dẫn vào trong CV, chẳng nhà tuyển dụng nào lại dại dột mà bỏ qua bạn đâu.

Có rất nhiều công ty lớn trên thế giới xem đây là một yêu cầu trong quy trình tuyển dụng của họ. Nếu bạn có nhiều đóng góp cho cộng đồng hoặc có nhiều sản phẩm trên Github, sẽ là một lợi thế rất lớn so với các ứng viên khác. Vì bằng cách đăng tải các project của mình lên đây, bạn đã tạo cho mình một profile cá nhân vô cùng đáng tin cậy.

Vì khi nhìn vào đó, nhà tuyển dụng sẽ biết được ngay thế mạnh của bạn là gì, và khả năng coding của bạn thế nào.

### 3.5. Giúp cải thiện kỹ năng code, thậm chí là tracking bug

Có hàng ngàn hàng vạn cách để học, học trên Github sẽ là một ý kiến không tồi trong thời đại này. Với hàng vạn open source projects, hàng trăm ngàn contributors, hàng tỉ commit mỗi ngày thì chỉ bằng việc xem. So sánh, học tập từ những thay đổi đó đã đem lại cho bạn hàng tá điều hay để cải thiện kỹ năng code của bản thân mình.

“Bug tracking” là một tính năng được GitHub tích hợp vào để đơn giản hóa quá trình “tìm và diệt bọ”. Để hiểu được quy trình thì những gì bạn cần làm là mở dashboard của từng project lên và filter các thông tin. Sau đó, các câu hỏi sẽ được hệ thống, sắp xếp theo mức độ phổ biến, thời gian update hay tương tại. Phần mềm này cũng có giao diện khá mượt nên luôn được xếp hạng cao trong cộng đồng IT dev.

**3.6. Một kho tài nguyên tuyệt vời**

Với chức năng Explore, bạn có thể theo dõi, tìm kiếm những open source projects theo đúng technology pattern mà bạn ưa thích. Github hỗ trợ code search không kể nó ở dưới dạng một project riêng biệt hay là website. Ngoài ra, nền tảng này cũng có SEO khá tốt nên người dùng có thể tìm kiếm bất kỳ code string nào được chia sẻ public.

**3.7. Github Action**

### Trên server của Github có những workflow scripts chạy tự động. Dev có thể dùng chúng để phản hồi các events trên repositories hoặc thực hiện vài action. Ví dụ như tôi có viết một cái tiện ích nho nhỏ, [Autotagger – GitHub Marketplace](https://github.com/marketplace/actions/autotagger" \t "_blank" \o "github.com), sẽ tự động tạo git tafs khi mà số phiên bản của package.json thay đổi. Nhìn thì đây chỉ là hành động nhỏ nhưng sẽ có tác động rất lớn khi phải truy tìm code ngược về bản phát hành, và bớt đi một cơn “nhức đầu” cho các project maintainers đó chứ.

### **3.8. Học hỏi được nhiều kĩ năng mới**

GitHub có 1 kho tài liệu tuyệt vời giúp chúng ta học hỏi nhiều kỹ năng mới. Phần *“*[*help*](https://help.github.com/)*”* và *“*[*guides*](https://guides.github.com/)*”* có nhiều bài viết về hầu như tất cả chủ đề liên quan đến Git mà mọi người có thể nghĩ đến. Để lên trình code, chúng ta nên đọc open source trên Git (repositories), và phần *“*[*explore*](https://github.com/explore)*”* là phần dẫn đến những open source nổi tiếng.

* Bạn cần biết cách tạo ra SSH key? GitHub có một *[bài viết](https://help.github.com/articles/generating-ssh-keys/" \t "_blank)*.
* Bạn tự hỏi về quy trình làm việc của 1 Git tốt? GitHub có một *[hướng dẫn](https://guides.github.com/introduction/flow/" \t "_blank).*
* Nếu bạn đang tìm kiếm 1 gitignore mẫu cho dự án tiếp theo, GitHub có nhiều [ví dụ](https://github.com/github/gitignore" \t "_blank) cho hầu như mọi điều mà bạn có thể nghĩ đến.

Nếu bạn muốn đóng góp vào cộng đồng outsource (như xây dựng một gem trong Ruby, một plugin trong Java hay viết một module Javascript…), bạn có thể thoải mái fork một project của một team nào đó, rồi tạo pull request hoặc issues nếu như tìm được lỗi, cần support, hoặc là tạo project của riêng mình và chia sẻ với mọi người.

**CHƯƠNG 4: TỔNG KẾT**

**4.1. Kết quả đạt được**

Tìm hiểu tổng quan về công cụ Github và hướng dẫn sử dụng Github cho người dùng. So sánh Github với một vài công cụ quản lý như Git. Những lợi ích mà Github mang lại cho lập trình viên và nắm bắt những lệnh hay dùng trên Github.

GitHub là một nền tảng tuyệt vời đã thay đổi cách hoạt động của lập trình viên. Tuy nhiên, bất kỳ ai muốn quản lý dự án hữu hiệu cũng có thể tham gia vào GitHub và hợp tác làm chung dự án để đạt hiệu quả cao nhất.

Nếu đội ngũ của bạn làm trong một dự án cần cập nhật liên tục và cần giám sát toàn bộ các thay đổi, GitHub là dành cho bạn. Những giải pháp tương tự GitHub khác là [**GitLab**](https://about.gitlab.com/), **[BitBucket](https://bitbucket.org/" \t "_blank)**, nhưng chúng tôi vẫn khuyên bạn dùng GitHub.

**CHƯƠNG 5: TÀI LIỆU THAM KHẢO**

1.<https://topdev.vn/blog/github-la-gi/>

2.<https://itviec.com/blog/github-la-gi/>

3.[hocchudong/git-github-for-sysadmin: Hướng dẫn sử dụng git, github](https://github.com/hocchudong/git-github-for-sysadmin)

4.[Phân biệt về Git và GitHub (viblo.asia)](https://viblo.asia/p/phan-biet-ve-git-va-github-djeZ1g9o5Wz)

5.[GitHub là gì? Hướng dẫn sử dụng GitHub mới nhất 16/03/2021 (cryptoviet.com)](https://cryptoviet.com/github-la-gi)

**PHỤ LỤC PHÂN CÔNG NHIỆM VỤ**

|  |  |  |
| --- | --- | --- |
| **STT** | **Thành viên** | **Nhiệm vụ** |
| 1 | Huỳnh Thị Mỹ Duyên | Quay video minh họa các thao tác với tập lệnh của git và format bản báo cáo. |
| 2 | Nguyễn Huỳnh Thanh Trúc | Tìm hiểu về chương 1: Giới thiệu về git & github |
| 3 | Nguyễn Thị Thảo Hồng | Tìm hiểu về chương 2: Các tập lệnh cơ bản |
| 4 | Phạm Lê Dịu Ái | Tìm hiểu về chương 3: Lợi ích của github, chương 4: Tổng kết và format bản báo cáo. |