**Trường Cao Đẳng**

Kỹ Thuật Cao Thắng

-----🙞🙜🕮🙞🙜-----

**BÀI BÁO CÁO**

Đề tài: Công cụ quản lý mã nguồn GIT

**MÔN: CC & MTPTPM**



Lớp : CĐTH17PMB

Giáo viên : Tôn Long Phước

Thành viên :

* Trần Huy Dức - 0306171128
* Lai Dương Phúc Bảo - 0306171110
* Trần Thái Đạt - 0306171126

**MỤC LỤC**

[**I.** **Giới thiệu** 2](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437173)

[1. GIT là gì ? 2](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437174)

[2. Quy trình xử lý công việc trên GIT ? 3](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437175)

[**II.** **Các khái niệm cơ bản** 4](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437176)

[1. Repository - Kho lưu trữ 4](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437177)

[2. Snapshot 5](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437178)

[3. Commit 5-6](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437179)

[4. Clone 6](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437180)

[5. Nhánh 7](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437181)

[6. Working tree và index 8](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437182)

[7. Tracked và Untracked 8-9](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437182)

[8. Conflict 9](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437182)

[**III.Cách sử dụng GitHub** 9](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437183)

[1. Đăng ký tài khoản GitHub 9-10](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437184)

[2. Tạo GitHub Repository 10-11](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437185)

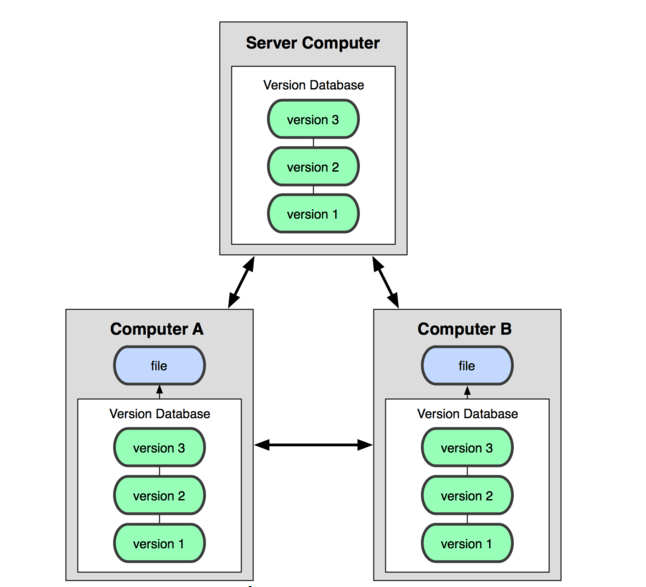
[3. Download và cài đặt GitHub Desktop 12-15](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437186)

[4. Kết nối GitHub và GitHub Desktop 15-20](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437187)

[**IV.** **Bảng phân công** 20](file:///C:\Users\trant\Downloads\TranHuyDuc_TranThaiDat_LaiDuongPhucBao.docx#_Toc18437191)

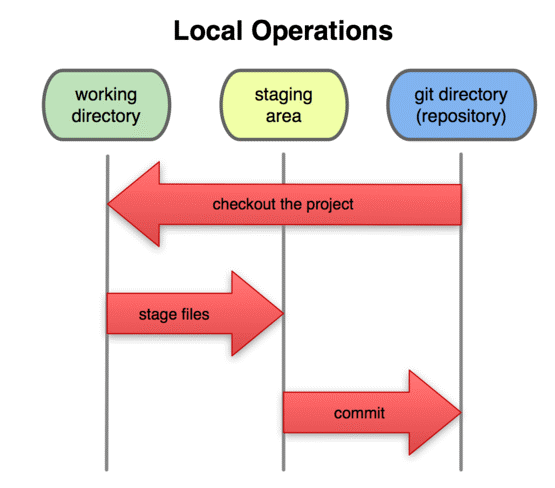
1. **Giới thiệu**
   1. *GIT là gì ?*

Git là một hệ thống quản lý phiên bản phân tán ( Distributed Version Control System – DVCS ) ra đời vào năm 2005 và hiện được dùng phổ biến. So với các hệ thống quản lý phiên bản tập trung khi tất cả mã nguồn và lịch sử thay đổi chỉ được lưu một nơi là máy chủ thì trong hệ thống phân tán, các máy khách không chỉ “ check out “ phiên bản mới nhất của các tập tin mà là sao chép ( mirror ) toàn bộ kho mã nguồn ( repository ). Như vậy, nếu như máy chủ ngừng hoạt động, thì bạn hoàn toàn có thể lấy kho chứ từ bất kỳ máy khách nào để sao chép ngược trở lại máy chủ để khôi phục lại toàn bộ hệ thống. Mỗi checkout thực sự là một bản sao đầy đủ của tất cả dữ liệu của kho chứa từ máy chủ.

 **Hình 1.1**: Mô hình tổ chức dữ liệu phân tán trên Git

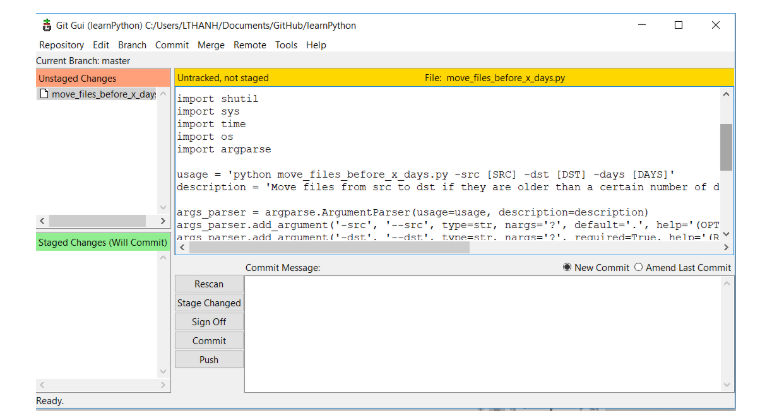
*1.2 Quy trình xử lý công việc ( workflow ) trên Git*

Khi chúng ta dùng Git để quản lý phiên bản mã nguồn thì không thể chỉnh được code trong Git. Chúng ta vẫn phải 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 mình đang làm việc. Thông thường, quy trình xử lý công việc trên Git sẽ như sau:



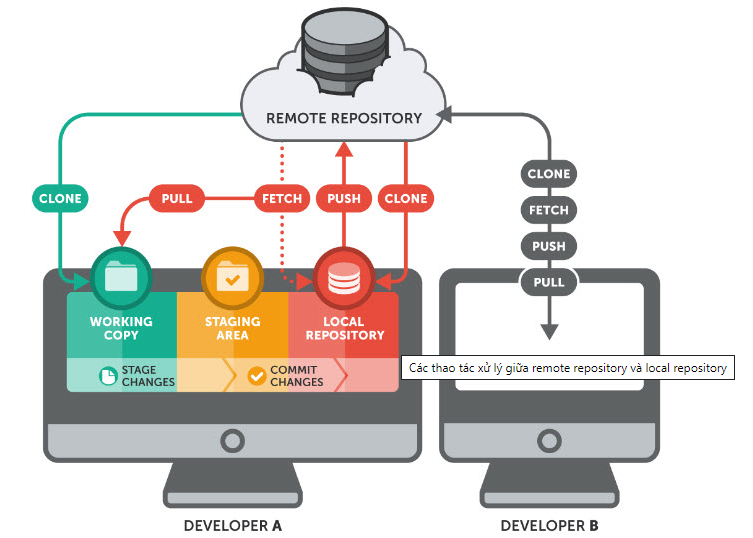
**Hình 1.2.1**: Quy trình xử lý công việc trên Git

Trong đó:

 **Hình 1.2.2**: Git cho phép bạn xem được trạng thái trong Working directory

1. **Các khái niệm cơ bản** 
   1. *Repository – Kho lưu trữ*

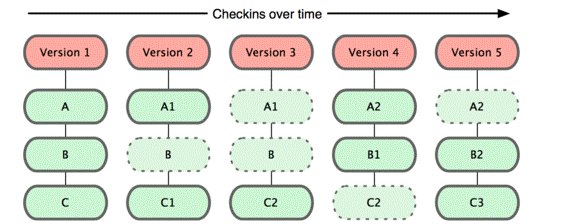
* Trong Git, Repository là nơi lưu trữ, quản lý tất cả những thông tin cần thiết ( thư mục, tập tin, ảnh, video, bảng biểu, dữ liệu … ) cũng như các sửa đổi và lịch sử của toàn bộ dự án. Khi tạo mới repository, chúng ta nên tạo thêm tập tin README hoặc một tập tin thông tin giới thiệu về dự án của mình.
* Chúng ta có có nhiều cách tổ chức cho repository, trong lập trình C# trên Visual Studio, chúng ta có thể lưu trữ một solution trong một kho, solution đó có thể chứa nhiều project.
* Có hai loại repository:
* Local repository: là repository dùng để chia sẽ giữa nhiều người và bố trí trên server chuyên dụng.
* Remote repository: là repository ở trên máy tính của chính bản thân mình, dành cho một người dùng sử dụng.



**Hình 2.1**: Các thao tác xử lý giữa remote repository và local repository

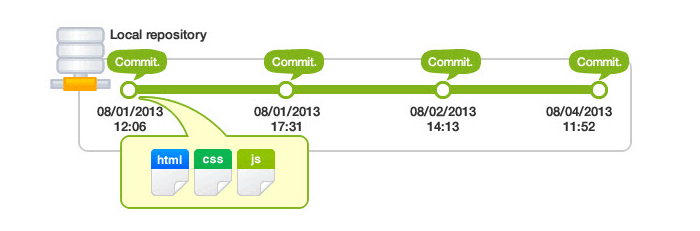
* 1. *Snapshot*

Cơ chế lưu trữ phiên bản của Git là sau mỗi lần bạn thực hiện lưu trạng thái ( commit ) sẽ tạo ra một “ ảnh chụp “ ( snapshot ) lưu lại nội dung tất cả các tập tin, thư mục tại thời điểm đó rồi tạo tham chiếu tới snapshot đó. Để hiệu quả hơn, nếu như tập tin không có thay đổi, Git không lưu trữ tập tin đó lại mà chỉ tạo liên kết tới tập tin gốc đã tồn tại trước đó. Sau đó khi cần bạn hoàn toàn có thể khôi phục và sử dụng lại một snapshot, hay còn gọi là phiên bản nào đó. Đây cũng chính là lợi thế của Git khi nó không lưu dữ liệu mà sẽ lưu dạng snapshot, giúp tiết kiệm không gian lưu trữ.



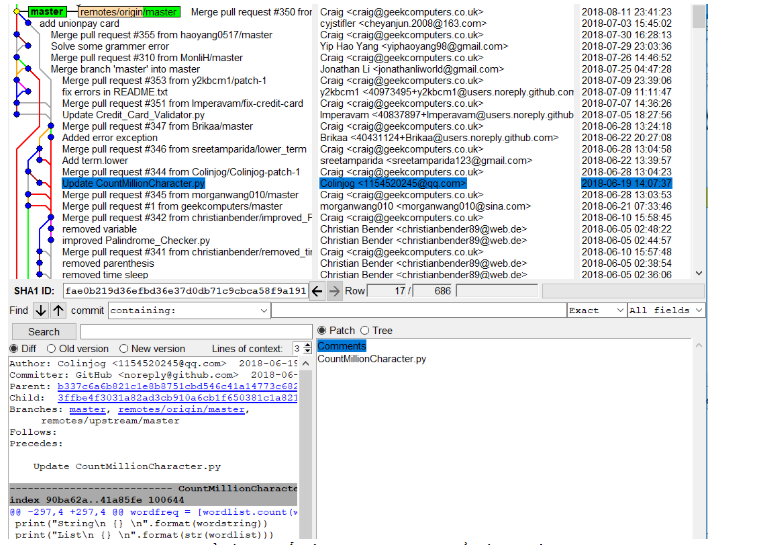
**Hình 2.2**: Git lưu trữ dữ liệu dưới dạng ảnh chụp của mã nguồn dự án

* 1. *Commit*
* Commit là thao tác báo cho hệ thống biết chúng ta 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.



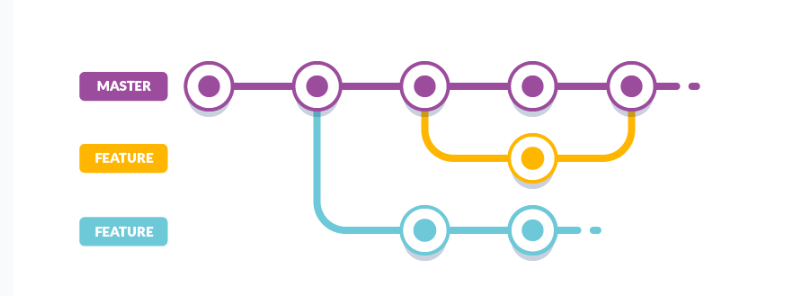
**Hình 2.3**: Thao tác commit trên Local repository

* Khi chúng ta 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 chúng ta đã thực hiện những thay đổi nào, có ý nghĩa thế nào,…
  1. *Clone*
* Clone là tạo ra một bản sao của gần như tất cả những gì của repository mà máy chủ đang lưu trữ. Chúng ta sẽ có được tất cả lịch sữ đã xảy ra trên repository và hoàn toàn có thể quay lại, undo lại từ bất kỳ thời điểm commit nào. Và một điểm nữa là nếu ổ cứng máy chủ bị hư, chúng ta có thể sử dụng bất kỳ bản sao trên bất kỳ máy khách nào để khôi phục lại trạng thái của máy chủ.
* Có thể clone từ bất kỳ kho chứa nào, nó sẽ sao chép luôn các thiết lập về repository và sẽ sự động tạo một master branch trên máy tính.



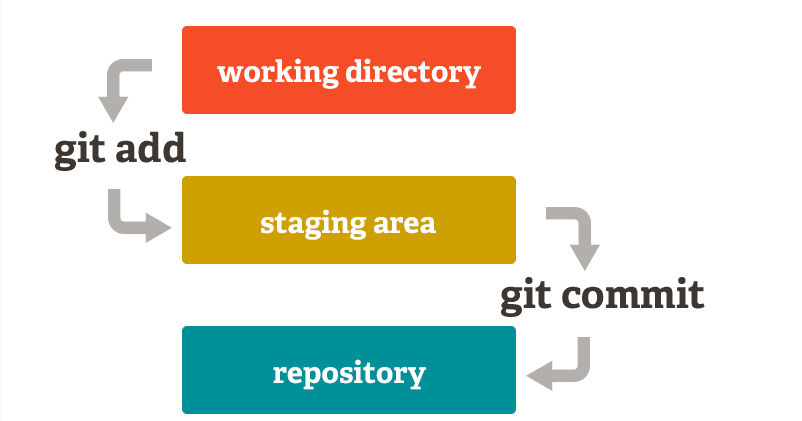
**Hình 2.4**: Lịch sử các thay đổi trên repository khi clone về máy cục bộ

* 1. *Nhánh ( Branch )*
* Nhánh có thể hiểu như là một không gian làm việc ( workspace ), Ví dụ khi chúng ta muốn tạo một tính năng A mới chúng ta sẽ tạo ra một nhánh mới để làm tính năng A. Đồng thời trong lúc chúng ta làm tính năng A thì chúng ta cũng có thể tạo ra một nhánh mới để sửa lỗi cho dự án của mình. Hai không gian làm việc này hoàn toàn không động đến nhau, nên dù tính năng A đã làm xong hay chưa đều không ảnh hưởng đến các nhánh ( không gian làm việc ) khác.
* Trong một Project sẽ luôn có một nhánh chính ( mặc định ) gọi là **master**. Tính năng được tạo ra trong các nhánh phụ sẽ được hợp nhất lại vào master khi đã làm xong, hành động này gọi là **merge.**



**Hình 2.5**: Các nhánh trong một Project

* 1. *Working tree và Index ( hoặc staging area )*
* Trên Git, những thư mục được đặt trong sự quản lý của Git, để mọi người thực hiện công việc trên đó, được gọi là **working tree.**
* Giữa repository và working tree tồn tại một nơi gọi là index hay staging area. Staging area là nơi để chuẩn bị cho việc commit vào repository.

****

**Hình 2.6**: Cách hoạt động của Working tree và Staging area

* 1. *Tracked và Untracked*
     1. *Untracked*
* Untracked là tập tin còn lại mà chúng ta sẽ không muốn làm việc với nó trong Git.
* Nếu chúng ta tạo ra hoặc thêm vào một tập tin mới vào trong thư mục làm việc của mình thì nó sẽ ở trạng thái Untracked, muốn cho nó trở thành tracked thì phải dùng lệnh “ git add “. Nếu tệp tin đó mới hoàn toàn thì nó sẽ rơi vào trạng thái staged và chúng ta có thể commit file đó.
  + 1. *Tracked*
* Tracked là tâp tin đã được đánh dấu theo dõi trong Git để bạn làm việc với nó. Và trạng thái Tracked nó sẽ có thêm các trạng thái phụ khác là Unmodified ( chưa chỉnh sửa gì ), Modified ( đã chỉnh sửa ) và Staged ( đã sẳn sàng để commit ).
* Một khi một tập tin đã được đưa về Tracked thì nó sẽ có thể thay đổi giữa 3 trạng thái khác nhau là Modified, Unmodified và Staged:
* **Unmodified** : File được Git quản lý nhưng không có thay đổi gì.
* **Modified** : File được Git quản lý nhưng đang bị thay đổi, cần dùng lệnh “ git add “ để đưa về trạng thái staged.
* **Staged** : File được git quản lý và sẵn sàng cho việc commit.
  1. *Conflict ( Xung đột )*

Conflict là trường hợp có 2 sự thay đổi trong cùng một hay nhiều dòng code trên một file mà Git không thể tự quyết định cách giải quyết cho “ đúng “ khi merge mã nguồn từ một nhánh khác về.

1. **Cách sử dụng GitHub**

Để làm việc với GitHub bạn cần:

1. Đăng ký 1 tài khoản GitHub và tạo 1 Reposity (GitHub Resposity).
2. Cài đặt GitHub Desktop, một công cụ trực quan quản lý Local Respository (Kho chứa dữ liệu địa phương).
3. Cấu hình để có thể đồng bộ hóa dữ liệu bằng GitHub Desktop lên Resposity server.
   1. *Đăng ký tài khoản GitHub*

* Trước hết bạn cần phải đăng ký miễn phí một tài khoản **GitHub**. Bạn có thể vào trang chủ của **GitHub** tại: <https://github.com>



**Hình 3.1:** Trang đăng ký GitHub

* Việc đăng ký một tài khoản là đơn giản,bạn chỉ cần nhập **username/password** và địa chỉ **email**. Sau khi đăng ký xong bạn cần vào Email kích hoạt tài khoản.
  1. *Tạo GitHub Repository*
* Sau khi đăng ký xong tài khoản GitHub, và đăng nhập vào. Bạn có thể tạo một GitHub Repository.



**Hình 3.2:** Cách tạo Repository



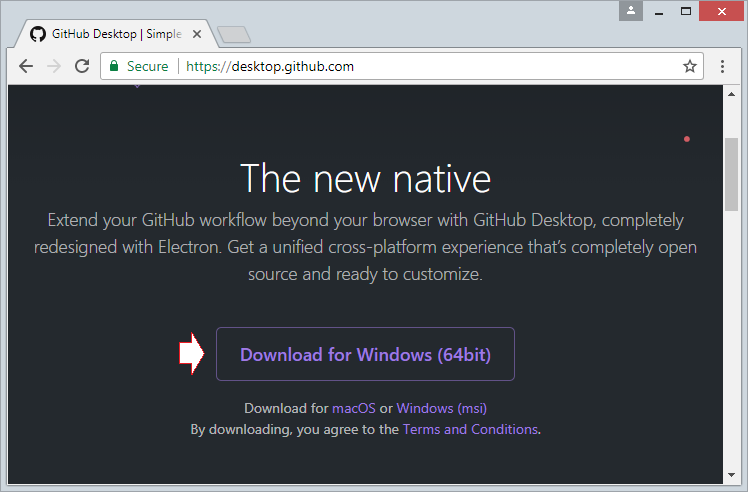
**Hình 3.3:** Đặt tên Repository



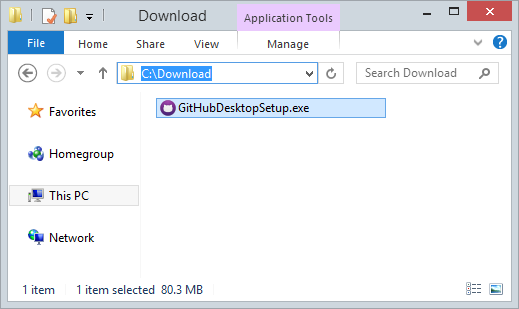
**Hình 3.4:** Kết quả

* 1. *Download và cài đặt GitHub Desktop*
* **GitHub Desktop** về bản chất là một công cụ trực quan cho phép bạn quản lý **Local Repository** (Kho chứa địa phương) trên máy tính của bạn.

Để download GitHub Desktop bạn vào địa chỉ: <https://desktop.github.com/>

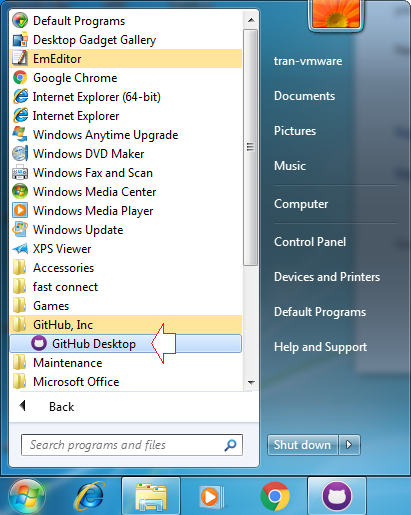


**Hình 3.5:** Nhấp vào để tiến hành download



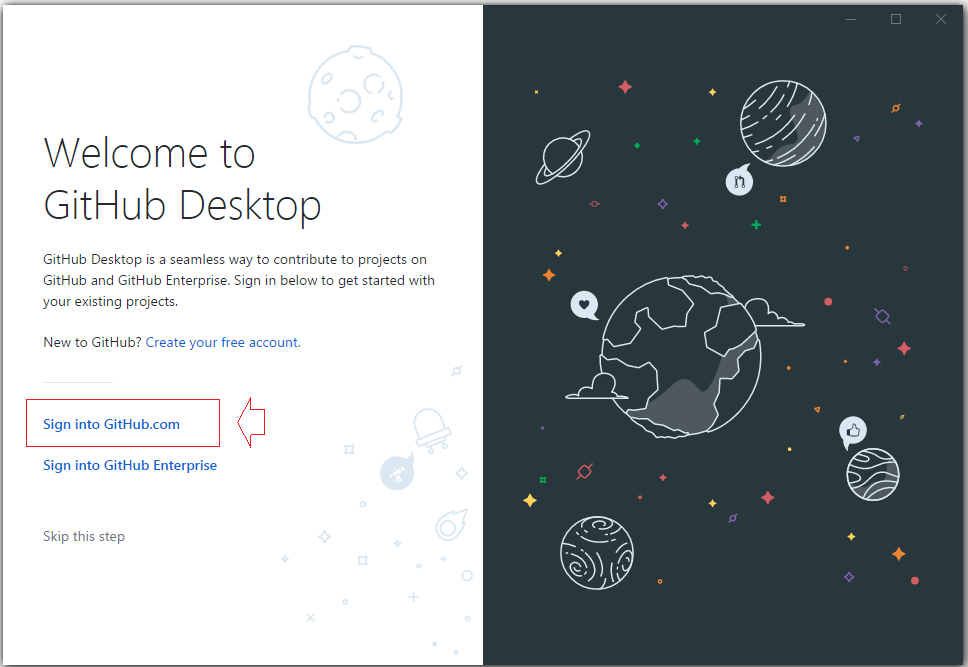
**Hình 3.6:** kết quả download

* Sau khi download xong, bạn cần cài đặt **GitHub Desktop** vào máy tính và sau đó ta tiến hành chạy **GitHub Desktop** như sau:

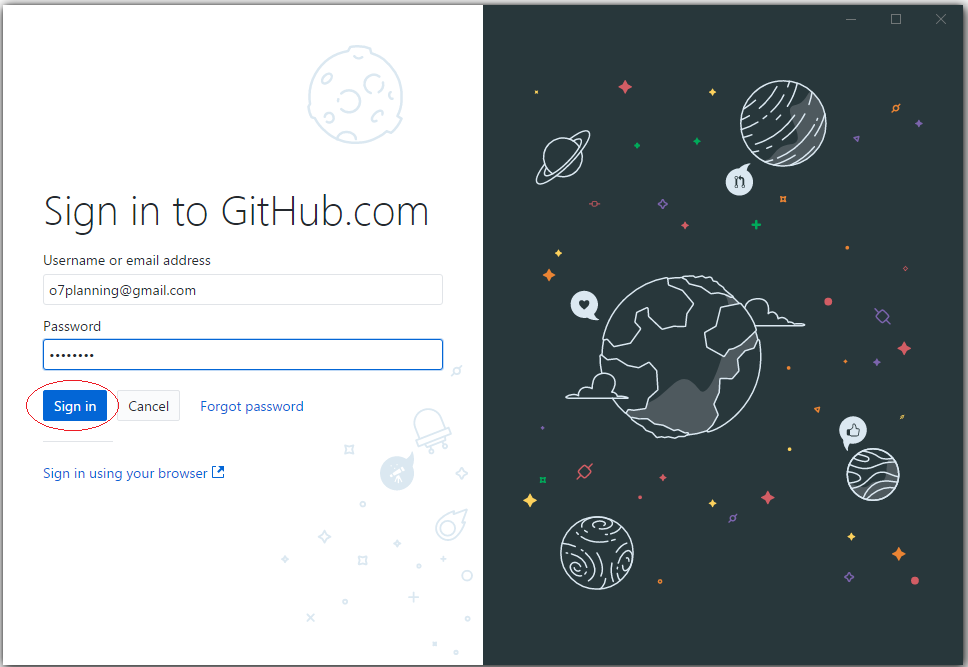


**Hình 3.7:** mở Startmenu tìm và mở GitHub Desktop

* Tiếp đó chúng ta sẽ đăng nhập trên **GitHub Desktop** để kết nối vào tài khoản **GitHub** của bạn.



**Hình 3.8:** chọn Sign into GitHub.com

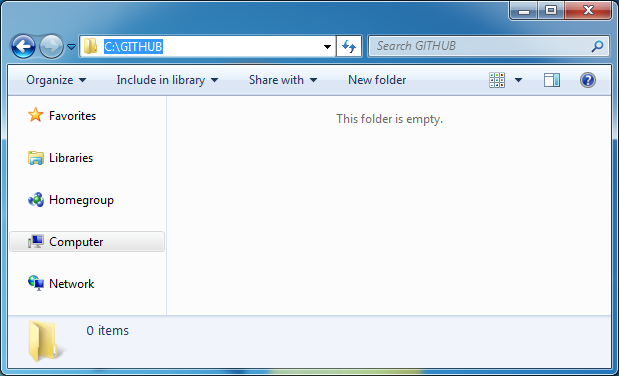


**Hình 3.9:** nhập tài khoản và mật khẩu



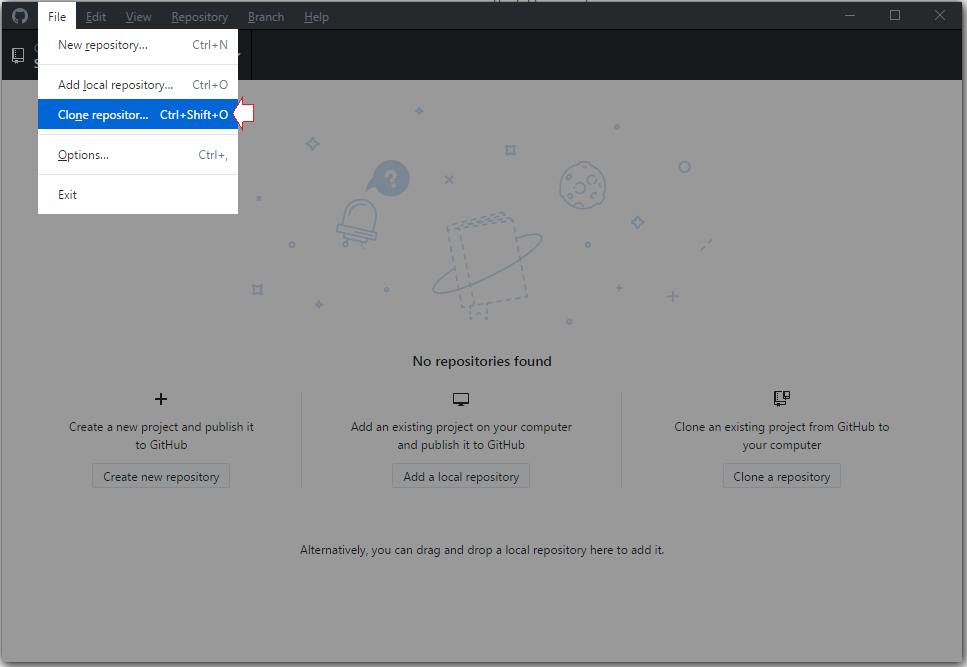
**Hình 3.10:** giao diện chính **GitHub Desktop**

* 1. *Kết nối GitHub và Github Desktop*
* Trước hết cần chọn một thư mục rỗng để làm vị trí chứa dữ liệu địa phương. Chẳng hạn: **C:/GITHUB**

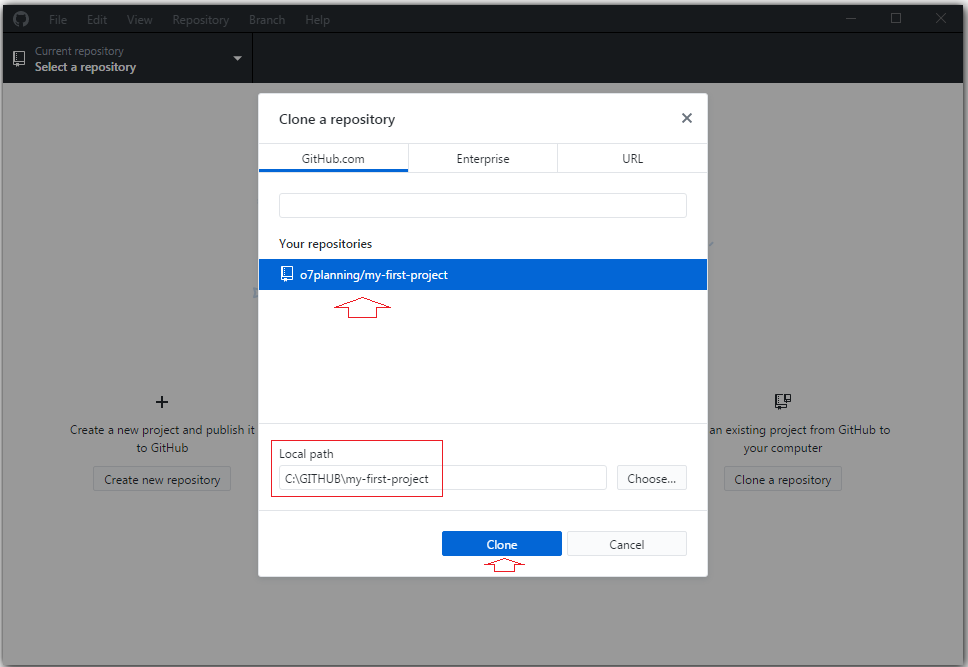


**Hình 3.11:** Tạo thư mục rỗng

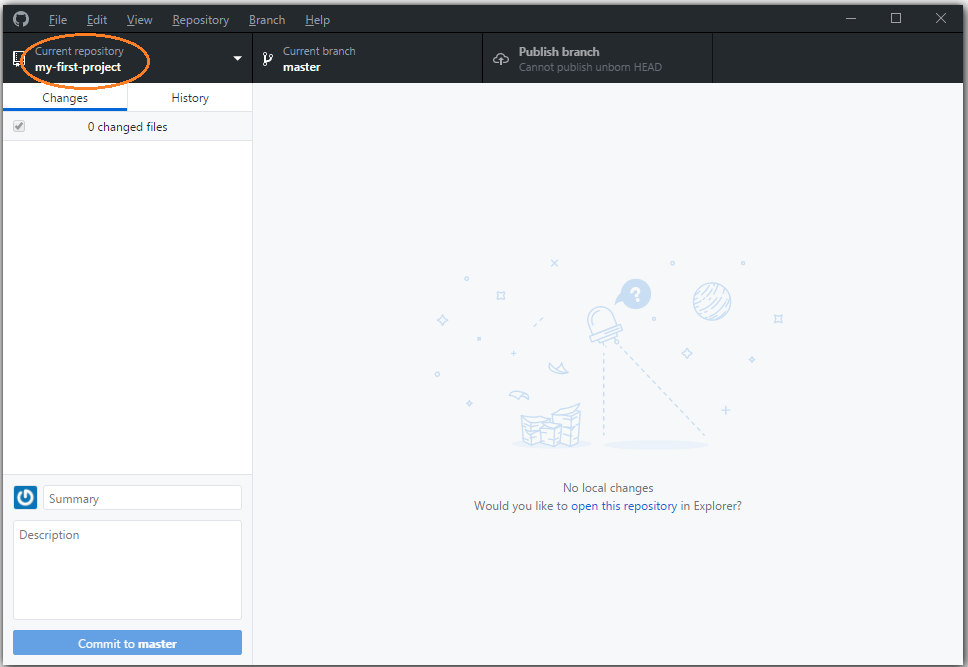
* Trên **GitHub Desktop**, lựa chọn một **Repository** bạn đã tạo trên **GitHub** để clone (Tạo bản sao chép) thành một bản ở máy tính địa phương của bạn.



**Hình 3.12**: Tạo clone

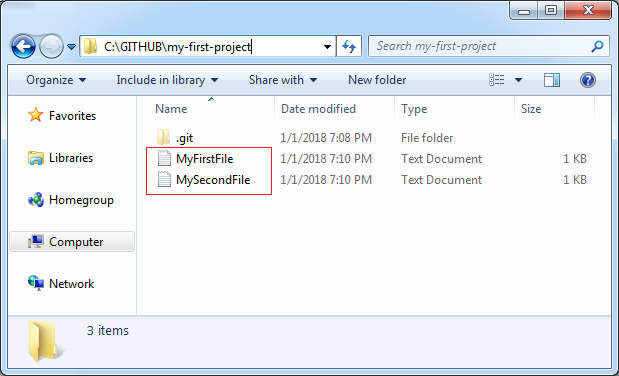


**Hình 3.13:** chọn repository để clone

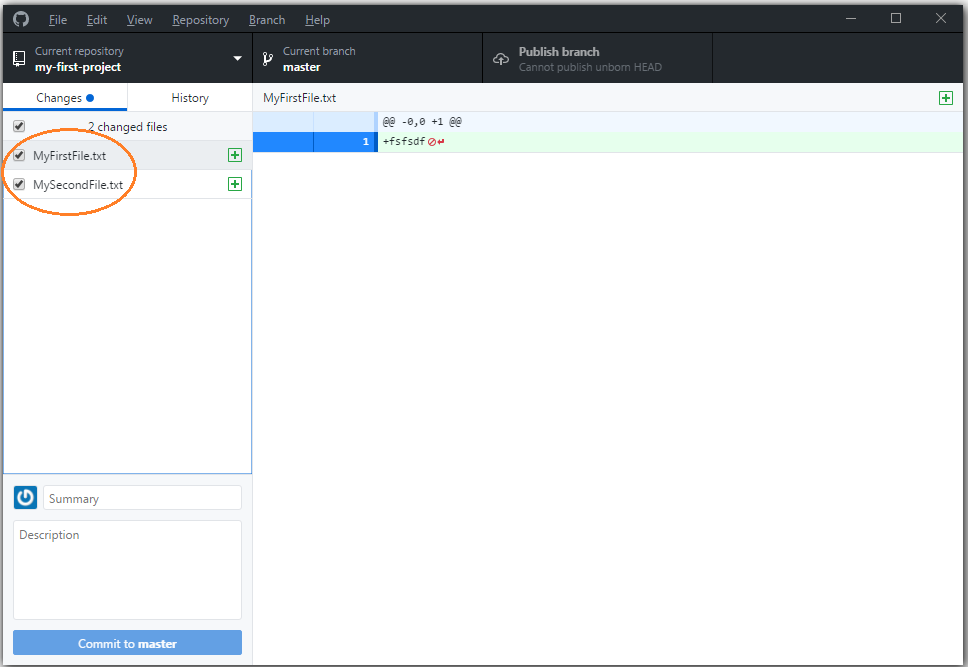


**Hình 3.14:** 1 Local Repository đã được tạo

* Lúc này chúng ta sẽ tiến hành copy dữ liệu vào **Local Repository**

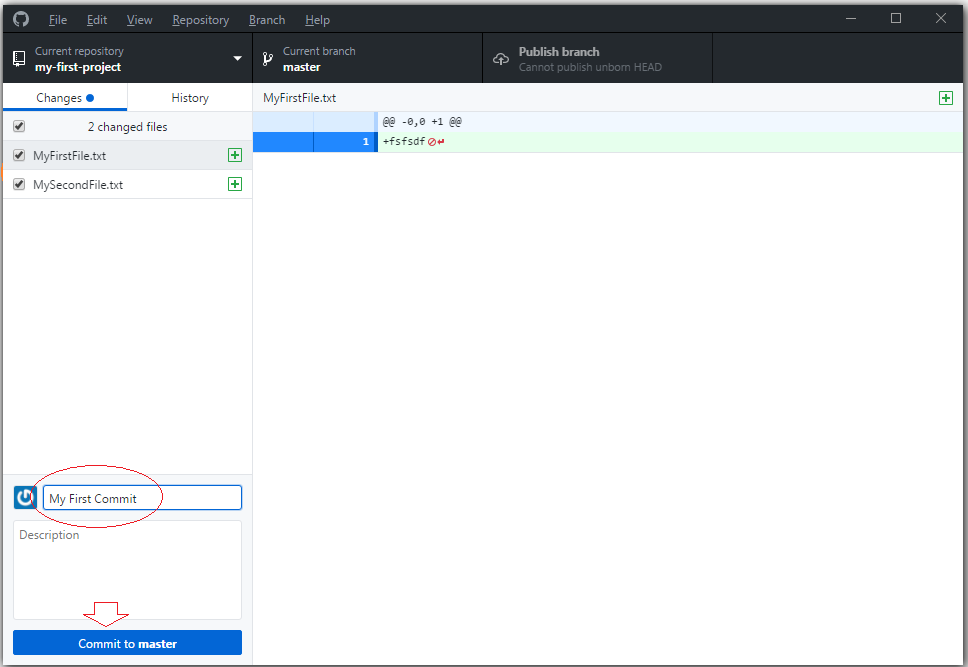


**Hình 3.15:** Copy dữ liệu



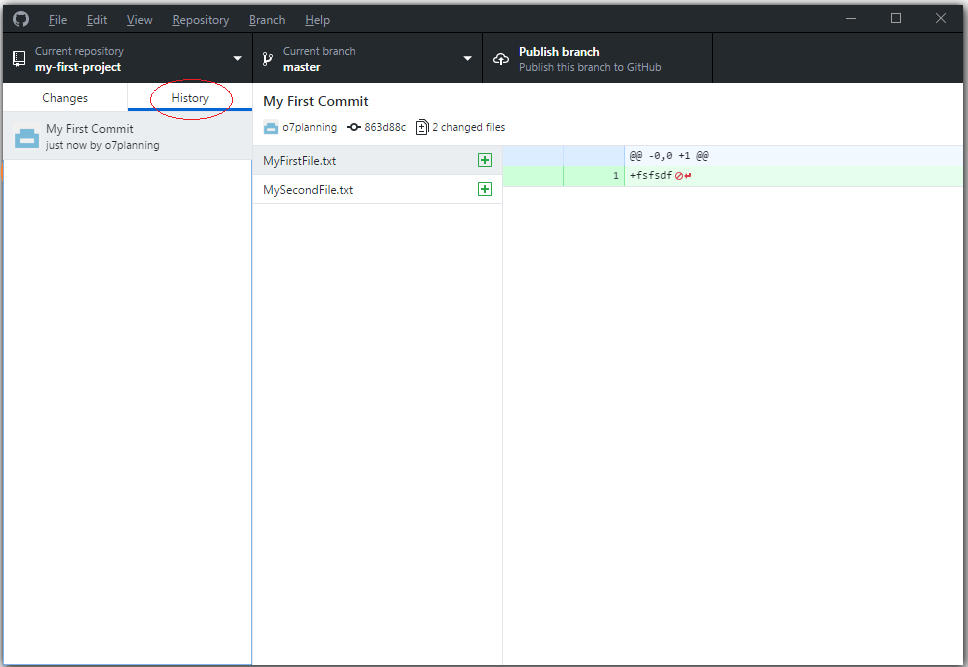
**Hình 3.16:** Dữ liệu đã được copy

* Chúng ta sẽ nhập thông tin ghi chú (Comment) và nhấn Commit dữ liệu.

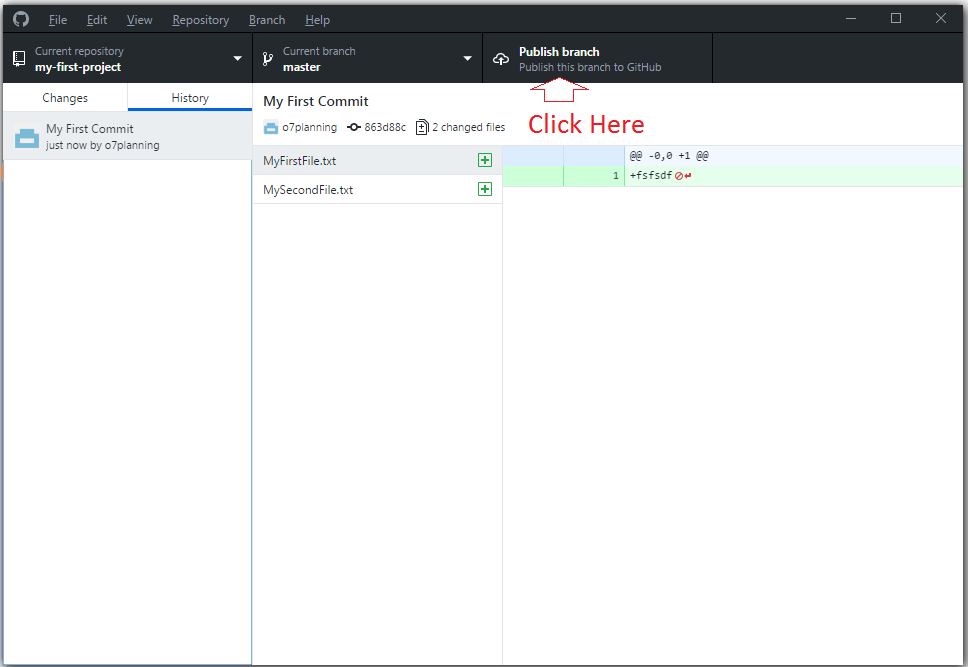


**Hình 3.17:** Nhập Comment r nhấp Commit

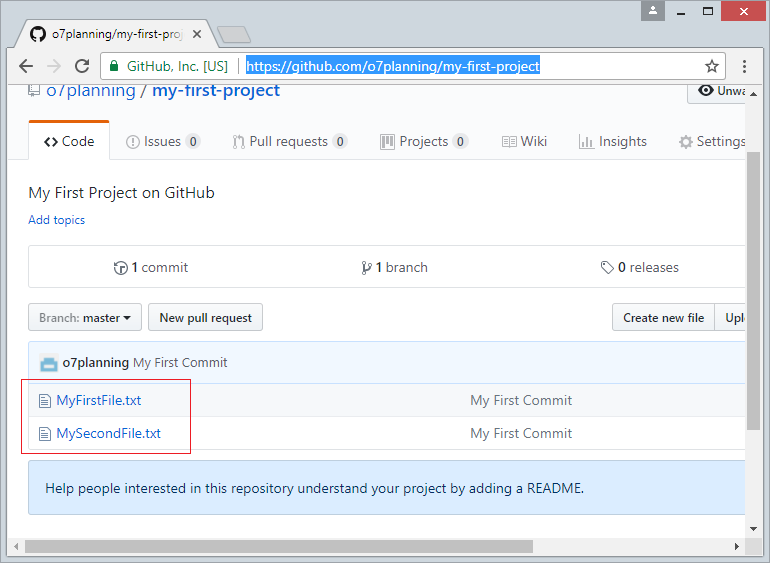
* Ngay sau đó bạn chuyển sang tab History r tiến hành công khai trên **Github** bằng cách nhấp vào Pulish branch



**Hình 3.18:** chuyển sang tab History



**Hình 3.19:** chọn Publish branch



**Hình 3.20:** Các file dữ liệu bạn có thể nhìn thấy trên Server.

|  |  |
| --- | --- |
| **BẢNG PHÂN CÔNG** | |
| Trần Huy Đức | * Làm PowerPoint   Tìm tài liệu phần:   * Giới Thiệu |
| Trần Thái Đạt | * Làm Word   Tìm tài liệu phần :   * Cách sử dụng GitHub |
| Lai Dương Phúc Bảo | * Làm Word   Tìm tài liệu phần:   * Các khái niệm cơ bản |