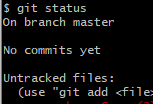
**Git Cơ Bản**

Để 1 dự án được trở thành một kho chứ Git thì chúng ta thực hiện lệnh: **git init**



Là nơi lưu trữ toàn bộ dữ liệu về Git cho dự án

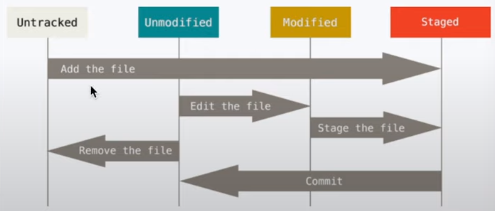
* Để kiểm tra trạng thái thư mục làm việc thì dung lệnh: **git status**
* Mặc định khi tạo git nó sẽ tạo ra 1 nhánh mặc định , gọi là “**master**”



* Một file mà git theo dõi gồm có **3 trạng thái**

được theo dõi bởi git, ở trạng thái này là đánh dấu lưu vào csdl

được theo dõi bởi git, nhưng không có sự thay đổi gì so với dự liệu đã từng lưu



**2: Lưu data**

**1**

được theo dõi bởi git, nhưng có sự thay đổi so với dữ liệu cuối

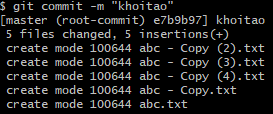
Không được theo dõi bởi git

**git add**: có 2 chức năng:

Đưa file đã bị sửa ”modified” vào “Staged”

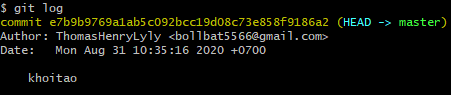
Đưa file từ “Untracked” vào “Staged”

**git commit -m “<nội dung commit>”**: lưu vào cơ sở dữ liệu của git để lưu trữ lâu dài, dữ liệu là nio56 dung trong “**Staged**”



Mã commit dài 40 kí tự nhưng ở đây chỉ hiện 7 kí tự để theo dõi

**git log**: kiểm tra trong nhánh có những commit nào,

**git log --oneline**: cần những thông tin vắn tắt

Người thực hiện commit

Ngày giờ thực hiện commit

Nội dung commit

id commit



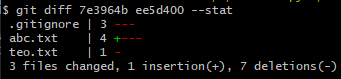
**git restore** <tên file> hoặc “\*” để phục hồi tất cả file: phục hồi lại file đã mất từ cái commit cuối cùng đã thực hiện trước đó

**git diff**: Cho biết là đã sửa gì trong file đó (chưa commit)

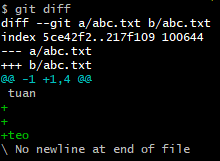
**git diff --staged**: So với commit cuối trước đó vùng “staged”, nếu đã commit rồi

**git diff <id commit 1> <id commit 2>** : so sánh sự thay đổi của 2 commit

**git diff <id commit 1> <id commit 2> --stat**: thống kê sự thay đổi của 2 commit



Nằm trong thư mục làm việc



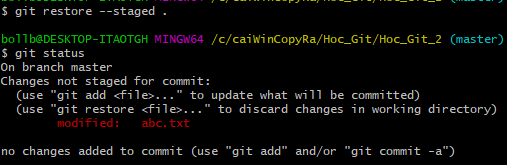
Nằm trong commit cuối

**git checkout <id của commit> --** <nếu dùng < . > Sẽ sửa hết các file, nếu muốn sửa file nào thì nhập tên file đó vào>



**sau khi phục hồi thì chúng ta cần commit**

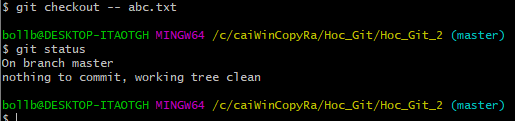
**git restore --staged** . : nếu không muốn file này được lưu trữ trong “stage” trở về modifile



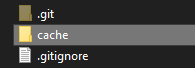
Khi chúng ta thay đổi 1 file (lần 1) và đã đưa lên trạng thái “staged”, nhưng lúc đó chúng ta lại thay đổi nội dung file 1 lần nữa (lần 2)và muốn trở về (lần 1) thì dùng: **git restore -- <tên file or . là tất cả file>**

* Tiếp tục chạy commit

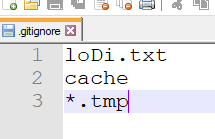
**Trở về trạng thái cuối cùng của commit**



**Để cho git lờ đi , không theo dõi các file không mong muốn, thì cần tạo một file tên là “.gitignore”, trong file đó sẽ chứa tên các thư mục các file cần lờ đi**



Nơi chứa tên các file, thư mục cần lờ

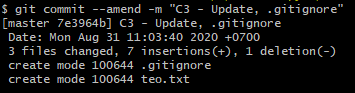


Tên file

Tên thư mục

Loại những tên có đuôi : “tmp”

Để cập nhật lại mà không tạo mới cái commit, muốn muốn lại commit trước đó, vì file đơn giản hay chỉ cập nhật gì đó, dùng lệnh: **git commit –amend -m <“C3 – Update, .gitignore”>**





Vẫn giữ nguyên commit cũ

**git reset –soft HEAD~1:**

+ xóa nội dung thay đổi trong commit không bị xóa hẳn đưa vào vùng “staged” để có nhu cầu thi chúng ta xem xét và đưa vào 1 commit khác, chú ý: Xóa commit cuối cùng

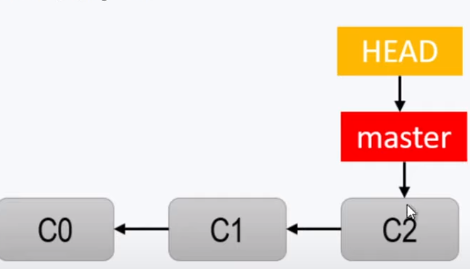
+ Có cơ hội phục hồi lại commit: git commit -m “< >”

**git reset –hard HEAD~1:**

+ Chú ý: Xóa hẳn commit và không có cơ hội phục hồi được và xóa commit cuối cùng

git reset -- <tên file hoặc là . >: Khi file sửa đổi và được “add” vào và chúng ta không muốn commit nó, xóa ra khỏi trạng thái “**staged**”

**Branch**

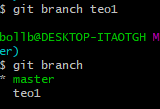


**git branch**: Kiểm tra trong git đang có những nhánh nào



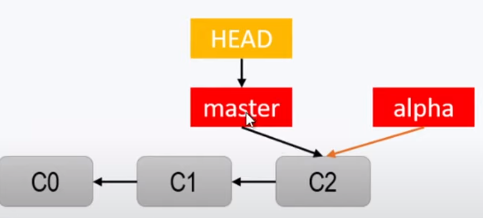
Có 1 nhánh master, “\*” thể hiện đang làm việc cho nhánh đó

**git branch <tên nhánh>** : tạo ra nhánh mới

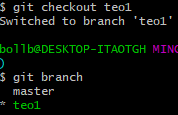


Tạo nhánh “tèo”

nhánh “tèo” được tạo



git checkout <tên nhánh>: chuyển qua làm việc cho nhánh tương ứng

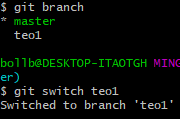


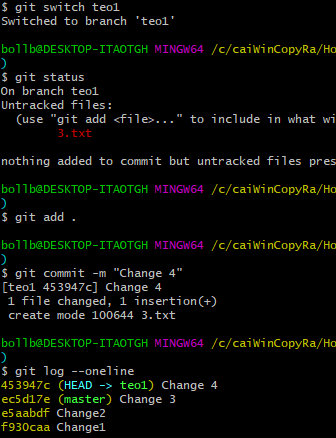
Chuyển nhánh “teo1”

Đã chuyển, “\*” thể hiện đang làm việc ở nhánh đó

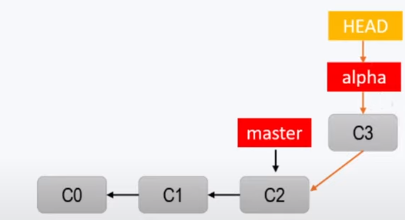


git switch <tên nhánh> hoặc git checkout <tên nhánh>: chuyển qua làm việc cho nhánh tương ứng

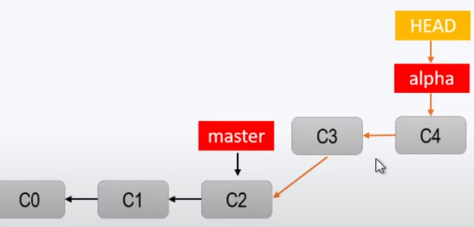


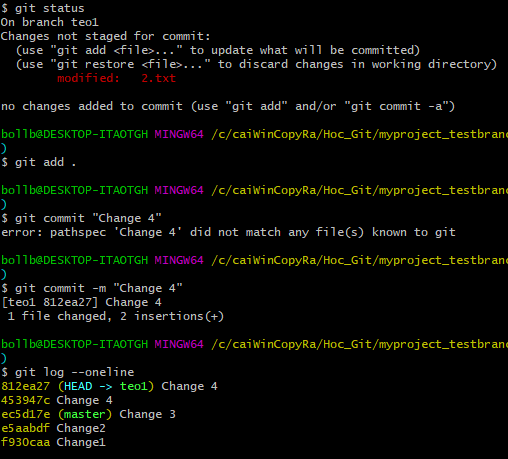


Commit của nhánh “teo1”

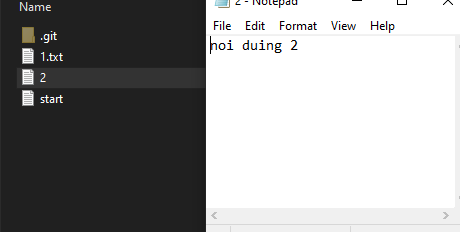


Thay đổi nội dung trong 1 file và commit nó ở 1 nhánh khác, khi chuyễn qua các nhánh khác nhau thì nội dung của file sẽ khác nhau

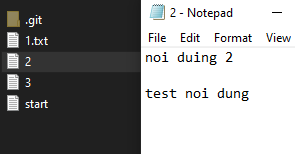




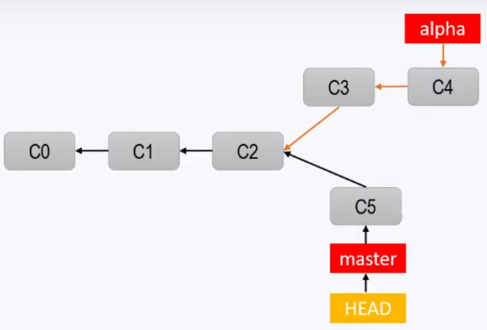
Khi ở nhánh master:



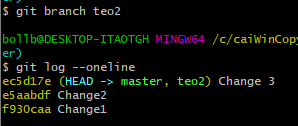
Khi ở nhánh teo1:



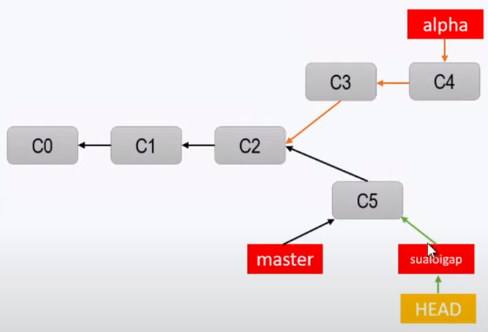
Tạo thêm nhiều commit cho nhánh



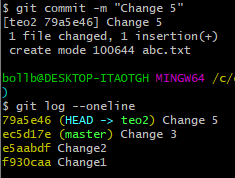
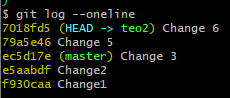
Tạo thêm nhiều nhánh nữa trong nhánh:



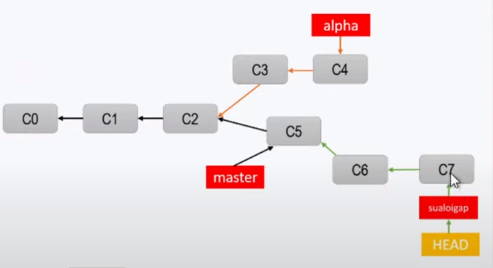
Nhánh teo2 và master có chung 1 commit Change 3 và thừa hưỡng các commit cũ



Tạo thêm nhiều commit cho teo2

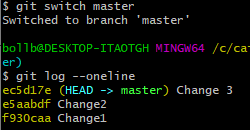


Nhánh teo2 đang ở xa nhánh master

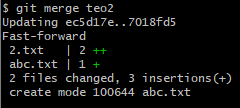
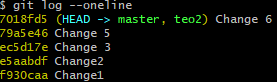


**Sau khi tạo nhánh và kiểm tra kỹ r, chúng ta có thể gộp với nhánh master**

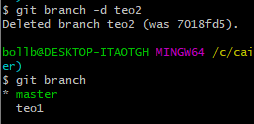
+ chuyển vể nhánh master



+ git merge <tên nhánh cần gộp với master>:

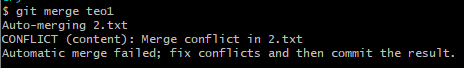


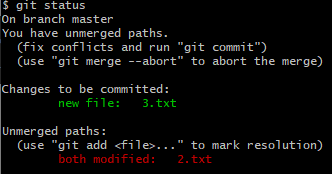
+ Sau khi gộp nhánh, nếu chúng ta ra k cần nhánh đó nữa thì tiến hành xóa nhánh:

 **git branch -d <tên nhánh cần xóa>**

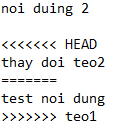
Chú ý: Khi gộp các nhánh lại sẽ có lúc xung đột vì nếu 1 file được sửa bởi 2 nhánh khác nhau khi gộp sẽ bị xung đột dữ liệu vì git không biết chọn nhánh nào để cập nhật, vì thế chúng ta cần phải lựa chọn sửa giữ lại data của nhánh nào

Bị xung đột ở file “2.txt”





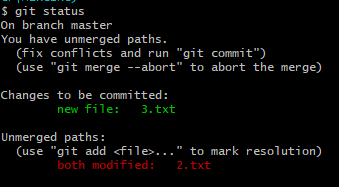
Bị lổi vì git đã sửa trong file đó



Sửa bởi git , nội dung này do nhánh teo2 sửa đổi, đáng lẻ là nhánh master do nhánh teo2 đã gọp với master

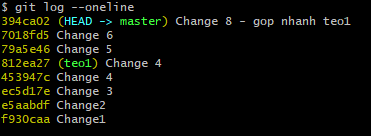
Sửa bởi git , nội dung này do nhánh teo1 sửa đổi

NẾu lấy nội dung nào thì xóa nội dung của nhánh khác đi, còn nếu lấy cả hai thì vẫn giữ, xóa mấy cái HEAD đi

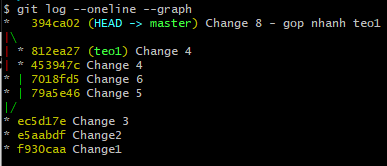


Đã xử lý xung đột

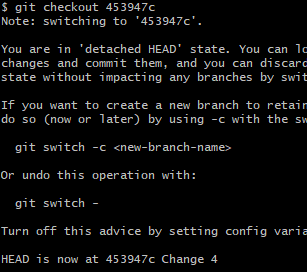
Sau đó chúng ta dùng “git.add . “ và “git commit -m <> “ như bình thường



Để xem sẽ đồ thị nhánh thì chúng ta dùng : **git log --oneline --graph**

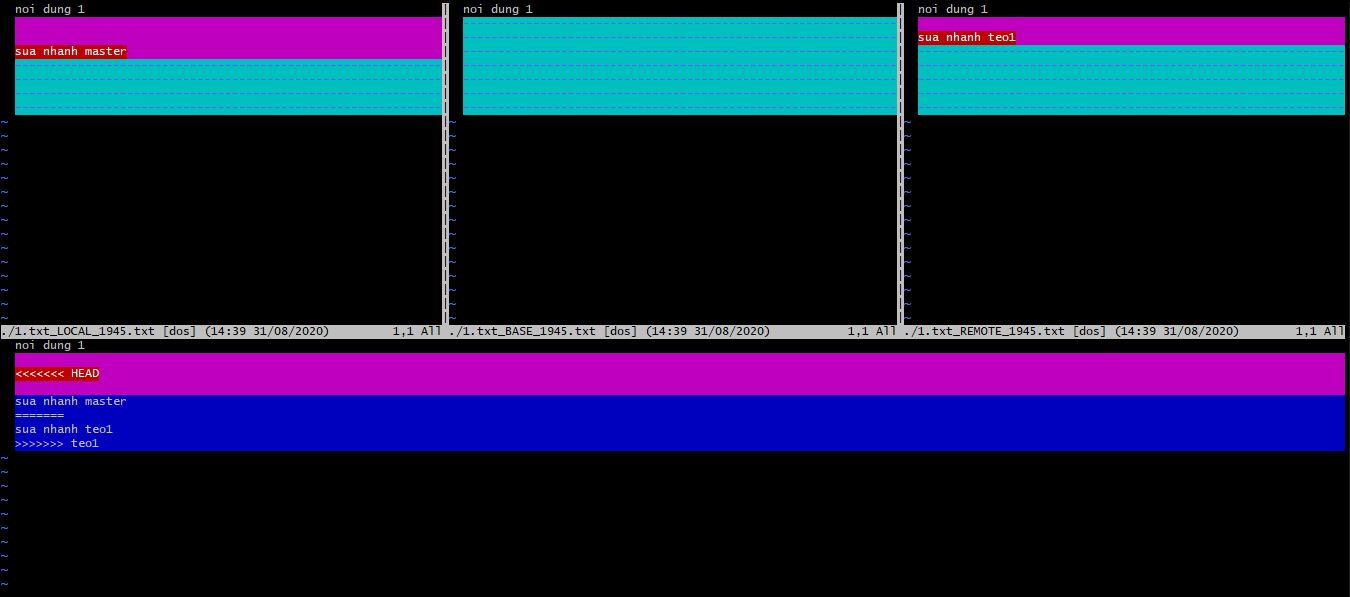


Để chuyển con trỏ làm việc về 1 commit bất kỳ, ta dùng: **git checkout <id commit>**



Con trỏ về commit Change 4

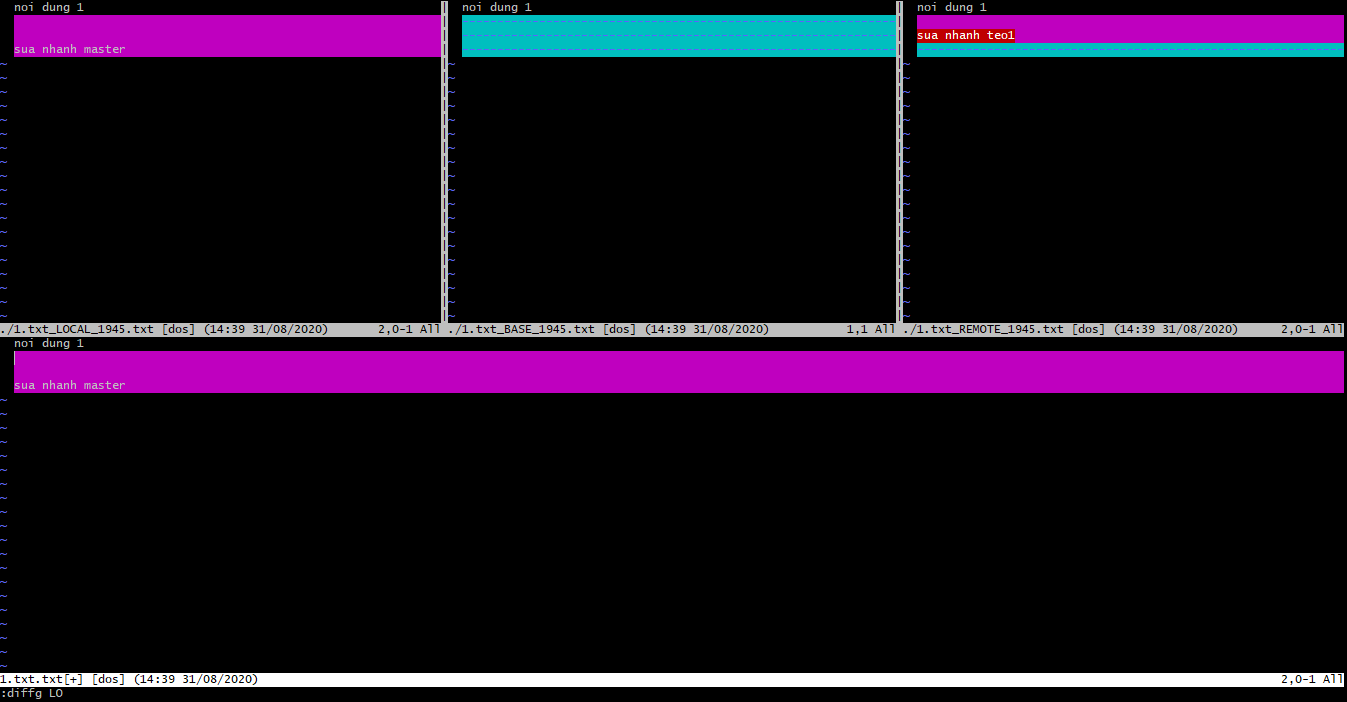
Git có 1 công cụ để sửa lỗi là : **git mergetool**



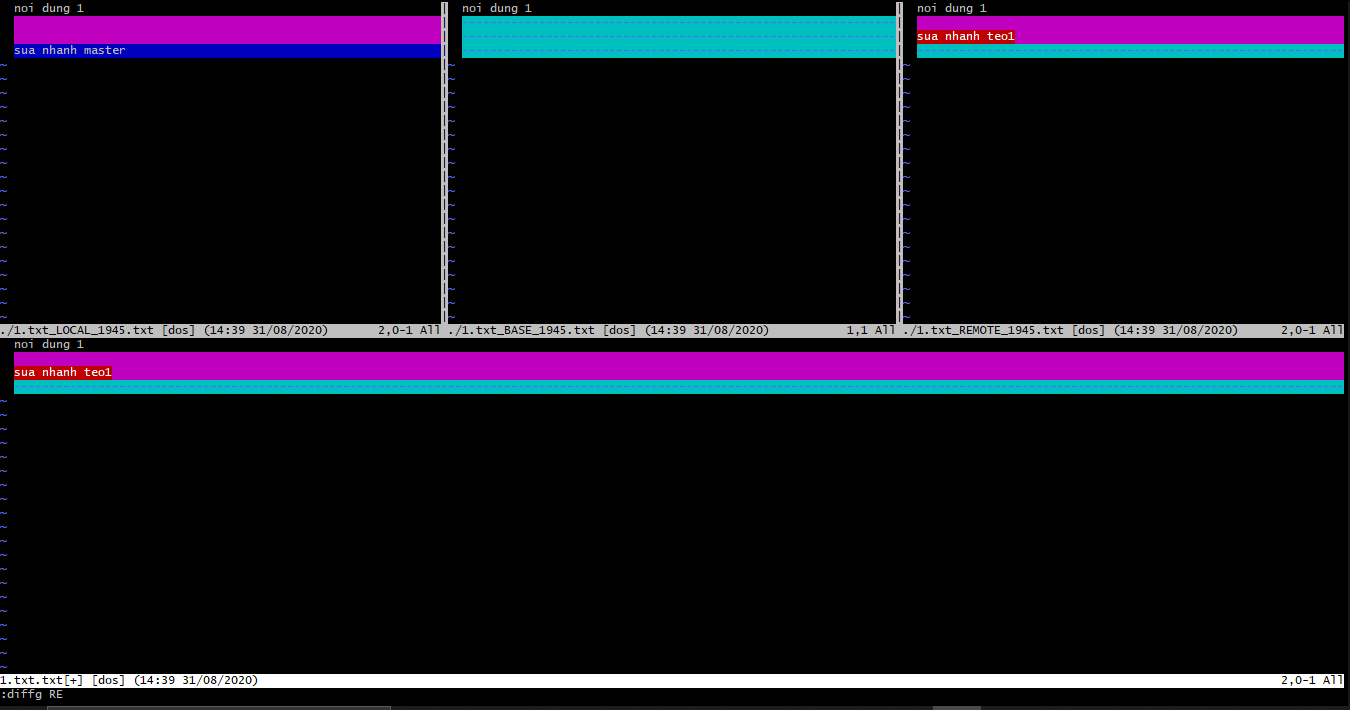
Nếu chúng ta muốn giữ lại master (bên phải) là local vì đang ở trong nhánh master

Gõ trực tiếp vào terminal: dấu “ : “ và gõ

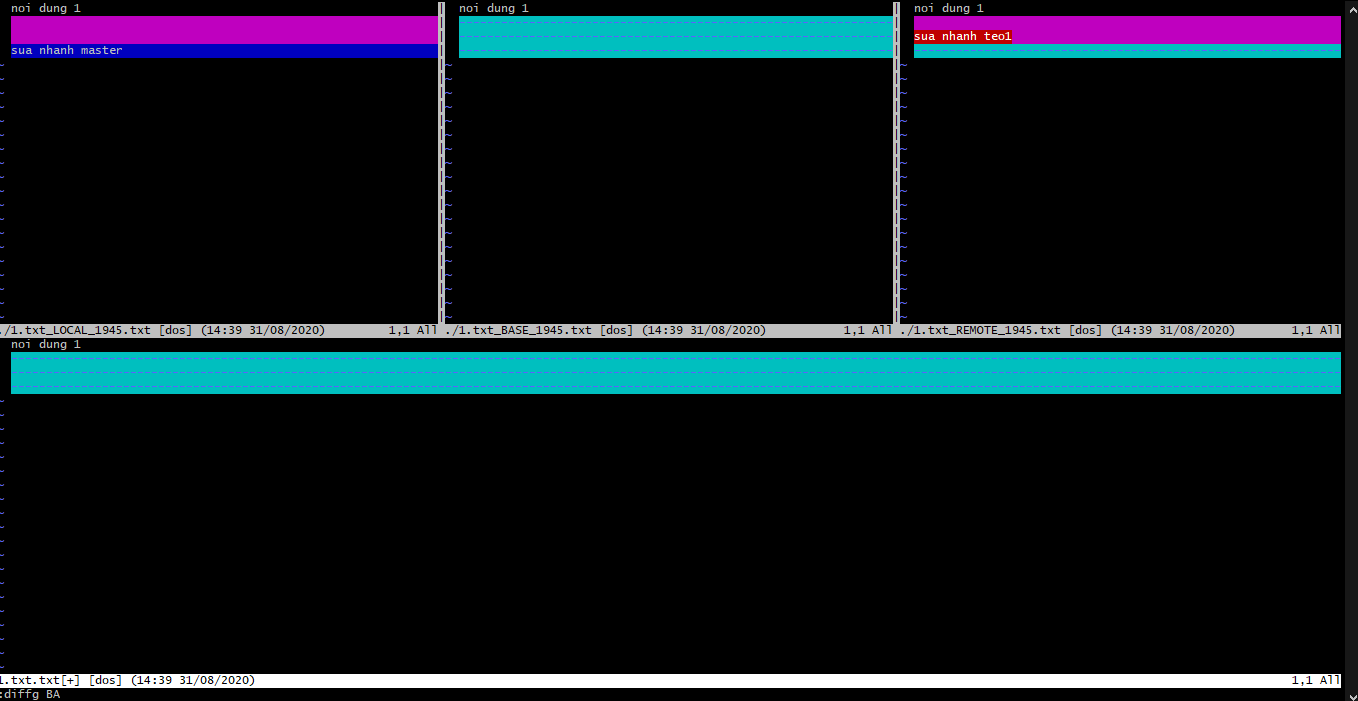
+ diffg LO : lấy local nghĩa là lấy master



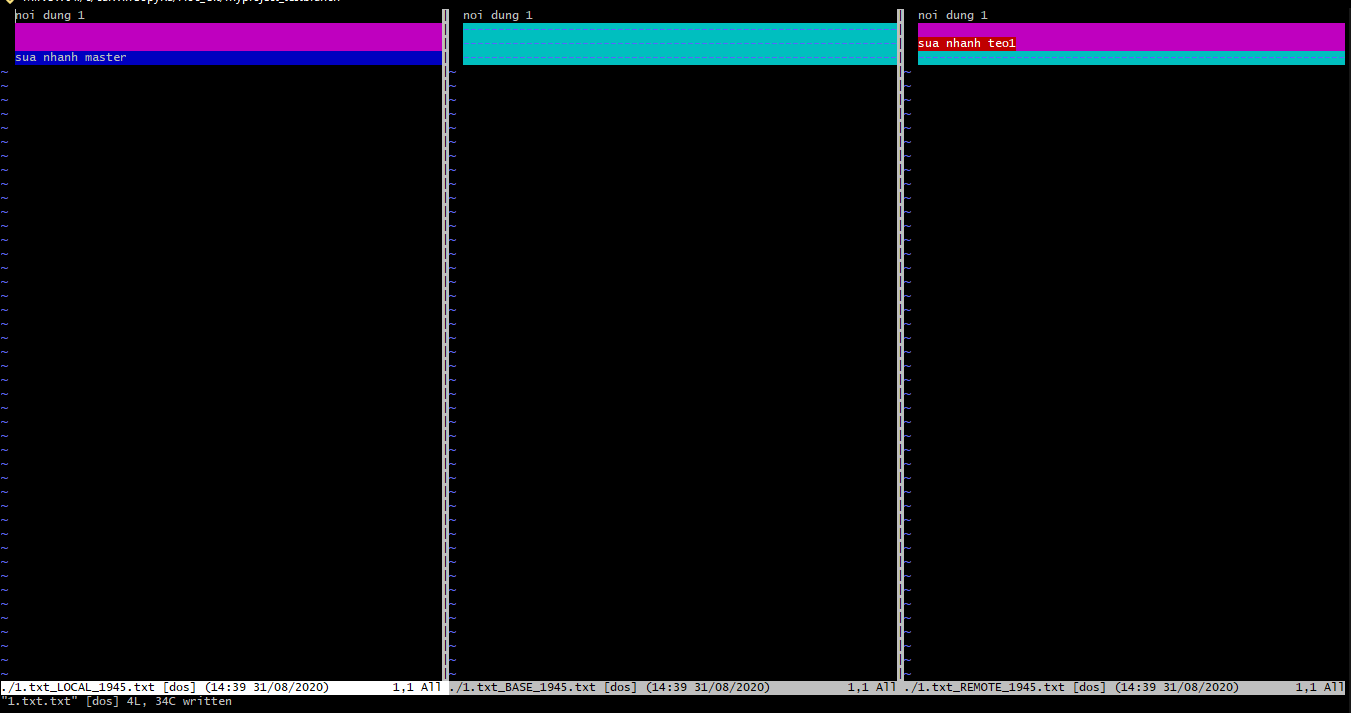
+ diffg RE: lấy nhánh remote



+ diffg BA: giữ lại nội dung gốc của file đó



Sau khi chọn được nội dung thì nhập : wq -> Enter đến khi nào thoát ra thì thôi





Xóa file xung đột đi

Thực hiện lại các bước như trên



**Sử dụng gitHub và gitLab**