* **git init** : Khởi tạo một kho lưu trữ Git mới trong thư mục hiện tại.
* **git clone <repository\_url>**: Sao chép một kho lưu trữ từ xa vào máy tính của bạn.
* **git add <file>** hoặc **git add .** Đánh dấu các tệp hoặc thư mục để theo dõi (staging) trong phiên làm việc hiện tại.
* **git commit -m "message"**: Tạo một commit với các thay đổi đã được đánh dấu (staged) và kèm theo một thông điệp để mô tả commit đó.
* **git log**: Hiển thị danh sách các commit trong kho lưu trữ, bao gồm thông tin về tác giả, thời gian và thông điệp commit.
* **git branch**: Liệt kê tất cả các nhánh (branch) có trong kho lưu trữ.
* **git checkout -b <branch\_name>:** Tạo nhánh và chuyển sang nhánh mới
* **git branch <branch\_name>:** Tạo nhánh không chuyển sang nhánh mới
* **git merge <branch>**: Hợp nhất các thay đổi từ một nhánh vào nhánh hiện tại. git merge -m "Gộp nhánh feature vào master" feature
* **git pull**: Cập nhật các thay đổi từ kho lưu trữ từ xa và tự động hợp nhất (merge) vào nhánh hiện tại.
* **git push**: Đẩy các commit đã tạo trên nhánh hiện tại lên kho lưu trữ từ xa.
* **git remote add <name> <repository\_url>**: Thêm một kho lưu trữ từ xa (remote) vào danh sách các remote đã được cấu hình.
* **git fetch <remote>**: Tải các thay đổi từ kho lưu trữ từ xa vào máy tính của bạn mà không thực hiện hợp nhất
* **git diff**: Hiển thị sự khác biệt giữa các phiên bản của tệp.
* **git branch -d <branch\_name>:** Xóa nhánh đã hoàn thành (đã hợp nhất vào nhánh hiện tại), lệnh này sẽ cố gắng xóa nhánh có tên **<branch\_name>**. Tuy nhiên, nếu có các thay đổi chưa được hợp nhất từ nhánh này, Git sẽ từ chối xóa nó và cung cấp một cảnh báo.
* **git branch -D <branch\_name>:** Xóa nhánh mà không quan tâm đến các thay đổi chưa được hợp nhất, Lệnh này sẽ xóa nhánh **<branch\_name>** mà không kiểm tra các thay đổi chưa được hợp nhất. Sử dụng lựa chọn này nếu bạn chắc chắn rằng bạn muốn xóa nhánh đó và không cần lưu giữ các thay đổi chưa được hợp nhất.

**3 Cấp độ reset**

* **git reset –soft <commit>**: Đặt lại HEAD để trỏ đến một commit cụ thể và giữ INDEX và thư mục làm việc không thay đổi. Thay đổi trong commit được chỉ định trở thành thay đổi chưa được đánh dấu (unstaged).( **<commit>** đại diện cho commit mà bạn muốn đặt lại HEAD tới. Bạn cần thay thế **<commit>** bằng mã hash của commit hoặc sử dụng các tùy chọn khác để chỉ định commit cụ thể mà bạn muốn sử dụng.)
* **git reset --mixed** (mặc định): Đặt lại HEAD để trỏ đến một commit cụ thể, loại bỏ tất cả thay đổi khỏi INDEX (staging area), nhưng không làm thay đổi thư mục làm việc. Thay đổi trong commit được chỉ định trở thành thay đổi chưa được đánh dấu (unstaged).
* **git reset --hard**: Đặt lại HEAD để trỏ đến một commit cụ thể, loại bỏ tất cả thay đổi khỏi INDEX và cả thư mục làm việc, trở về trạng thái của commit đó. Thay đổi chưa được lưu trữ (unstaged changes) và thay đổi không được lưu trữ (untracked changes) sẽ bị mất mát. Sử dụng cẩn thận.

**Các trạng thái**

* Trạng thái Modified (Sửa đổi): Trạng thái này xảy ra khi bạn đã thay đổi một tệp trong thư mục làm việc và Git phát hiện sự thay đổi đó, nhưng bạn chưa thêm nó vào vùng chờ (staging area) bằng lệnh git add. Trạng thái này có nghĩa rằng tệp đã được chỉnh sửa nhưng chưa được chuẩn bị để commit.
* Trạng thái Staged (Đã thêm vào vùng chờ): Khi bạn sử dụng lệnh git add, bạn đang di chuyển các thay đổi từ trạng thái "Modified" vào vùng chờ (staging area). Trạng thái này cho biết rằng các thay đổi đã được chuẩn bị để commit.
* Trạng thái Committed (Đã commit): Khi bạn sử dụng lệnh git commit, bạn tạo ra một bản ghi của các thay đổi đã được thêm vào vùng chờ. Các thay đổi này đã được lưu trữ trong kho lưu trữ Git và có thể được xem lại bất kỳ lúc nào trong tương lai. Trạng thái này cho biết rằng các thay đổi đã được lưu trữ và lịch sử commit đã được cập nhật.

**Gộp nhánh bằng cách git rebase**

* **git rebase <nhánh-khác>**: Sử dụng lệnh này để thực hiện rebase một nhánh khác vào nhánh hiện tại
* **git rebase --continue**: Sử dụng lệnh này sau khi giải quyết xung đột trong quá trình rebase. Nó cho phép bạn tiếp tục quá trình rebase sau khi đã commit các thay đổi đã giải quyết xung đột.
* **git rebase --abort**: Sử dụng lệnh này để hủy bỏ quá trình rebase nếu gặp vấn đề hoặc muốn quay lại trạng thái trước đó.
* **git rebase -I <commit bắt đầu>**: Sử dụng lệnh này để mở một giao diện tương tác cho việc rebase. Bạn có thể chọn và sắp xếp lại các commit, chỉnh sửa commit message và thậm chí drop commit trong danh sách.
* **git rebase -p <nhanhkhac>**: Sử dụng lệnh này để thực hiện rebase và duy trì các thông tin về các commit cha trong trường hợp bạn đang làm việc với nhiều nhánh.
* **git pull –rebase origin <nhanhkhac>**: Sử dụng lệnh này để thực hiện pull từ một nhánh từ xa và rebase thay vì merge. Điều này giúp duy trì lịch sử commit tại máy tính của bạn trong trường hợp có thay đổi từ xa.