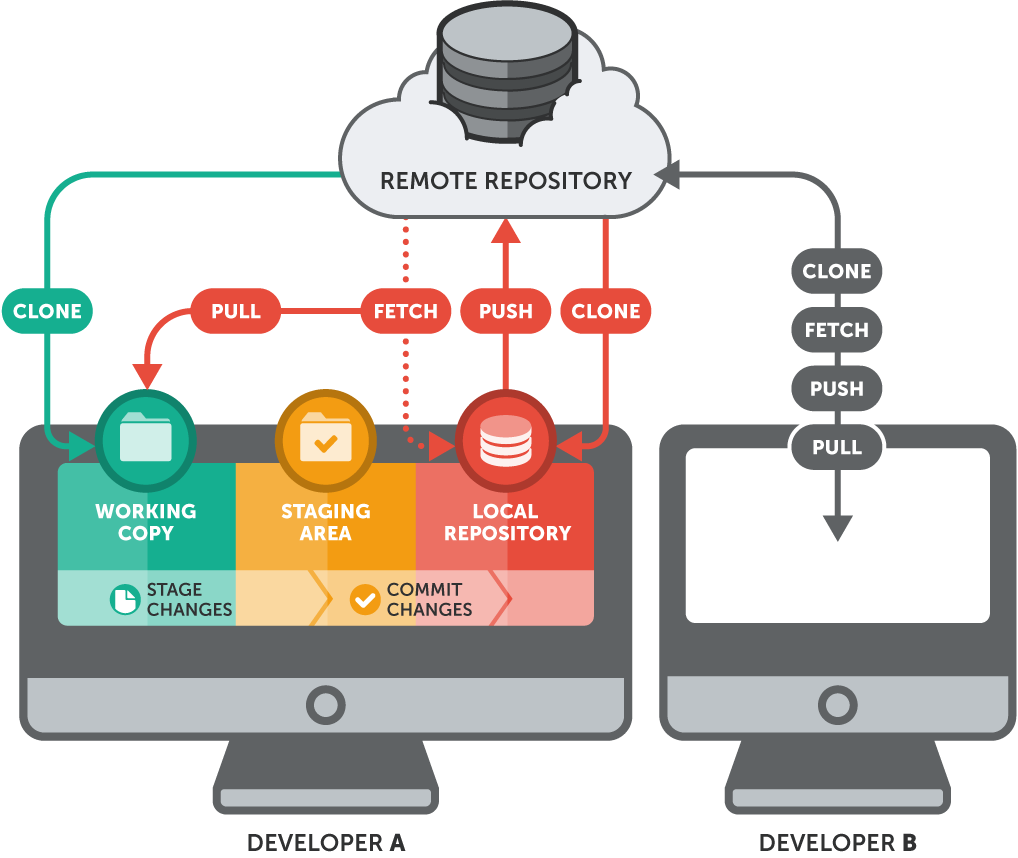
# SỬ DỤNG GIT CƠ BẢN

Git là một hệ thống quản lí phiên bản (Version Control System) kiểu phân tán. Hệ thống quản lí phiên bản là hệ thống theo dõi sự thay đổi tập hợp các file và nội dung trong file của dự án. Kiểu phân tán nghĩa là mã nguồn của dự án ko tập trung ở một server (SVN là hệ thống đặc trưng cho kiểu tập trung này), mà bất cứ máy tính nào đều có thể coi như một server con, và các server con ấy đồng bộ với một server đám mây. Một số dịch vụ server đám mây hỗ trợ Git như GitHub, BitBucket,… Đa phần chúng ta dùng GitHub.



Nơi lưu trữ trên mây gọi là “remote repository”, còn nơi lưu trữ ngay tại máy tính cá nhân người dùng gọi là “local repository”.

Một dự án được chia làm 3 vùng: vùng làm việc (working area), vùng đánh dấu (staging area) và vùng kho lưu trữ cục bộ (local repository).

Các file mà chúng ta trực tiếp chỉnh sửa nằm ở vùng làm việc. Để Git theo dõi phiên bản của các file, chúng ta thêm các file đó vào ***Vùng đánh dấu*** bằng hành động “add”. Khi nào cần lưu phiên bản hiện tại của dự án, chúng ta tạo bản sao của cả dự án vào vùng kho lưu trữ cục bộ bằng hành động “commit”. Khi nào chắc chắn phiên bản cuối cùng là ổn định, chúng ta sẽ đẩy toàn bộ các commit lên kho đám mây bằng hành động “push”. Đó là 3 hành động cơ bản khi sử dụng Git cho bất cứ dự án nào.

### Cấu hình ban đầu cho hệ thống Git

### Các câu lệnh cơ bản

-----------------------------------------------------------------------------

Clone một Repository từ Remote:

|  |
| --- |
| git clone <url> |

-----------------------------------Branch------------------------------------

1. Tạo một Branch Mới:

|  |
| --- |
| git branch <tên-branch> |

-----------------------------------------------------------------------------

Chuyển Đổi Giữa Các Branch:

|  |
| --- |
| git checkout <tên-branch> |

-----------------------------------------------------------------------------

Tạo và Chuyển Đến Một Branch Mới Ngay Lập Tức:

|  |
| --- |
| git checkout -b <tên-branch-mới> |

-----------------------------------------------------------------------------

Kiểm tra Trạng Thái (Xem Các File Đã Thay Đổi):

|  |
| --- |
| git status |

-----------------------------------------------------------------------------

Thêm 1 File đã thay đổi vào Staging Area:

|  |
| --- |
| git add [Tên file] |

Thêm nhiều File đã thay đổi vào Staging Area:

|  |
| --- |
| git add [Tên\_File1 Tên\_File2 …] |

Thêm tất cả các file đã thay đổi vào Staging Area:

|  |
| --- |
| git add . |

-----------------------------------------------------------------------------

Commit (Lưu Trạng Thái Hiện Tại):

|  |
| --- |
| git commit -m "Nội dung commit" |

Lệnh git commit sẽ đánh dấu phiên bản cho tất cả các file, bao gồm cả các file bạn vừa mới add.

-----------------------------------------------------------------------------

Đẩy Các Thay Đổi Lên Remote Repository:

|  |
| --- |
| git push origin <tên-branch> |

-----------------------------------------------------------------------------

Lấy Các Thay Đổi Mới Từ Remote Repository:

|  |
| --- |
| git pull origin <tên-branch> |

-----------------------------------------------------------------------------

Xem Lịch Sử Các Commit:

|  |
| --- |
| git log |

-----------------------------------------------------------------------------

Xem Các Branch Hiện Tại và Branch Remote:

|  |
| --- |
| git branch -a |

-----------------------------------------------------------------------------

Tạo Tag cho Một Commit Cụ Thể:

|  |
| --- |
| git tag -a <tên-tag> -m "Phiên bản x.x.x" |

---------------------------------Xóa File------------------------------------

Xóa File từ Trạng Thái Staging và Trên Remote:

|  |
| --- |
| git rm <tên-file> git commit -m "Xóa file" git push origin <tên-branch> |

Xoá file ra khỏi Git:

|  |
| --- |
| git rm --cached [tên\_file] |

Để xoá sự quản lý của Git đối với file đó, hay nói cách khác, là ngược lại với add, hành động này chuyển trạng thái file đã được đánh dấu (staged) sang trạng thái ko theo dõi (untracked). Lưu ý rằng dù bạn xoá đi file trong vùng làm việc, thì “ảnh” của nó vẫn còn được giữ trong vùng đánh dấu, tức là vẫn trong sự quản lý của Git, vì thế “ảnh” của nó vẫn đi vào commit. Lệnh git rm –cached mới thực sự gỡ bỏ sự theo dõi file.

Nếu bỏ tham số –cached, file ko những bị xoá khỏi Git mà còn biến mất khỏi thư mục làm việc.

Git commands we used at work:

𝟭. 𝗴𝗶𝘁 𝗱𝗶𝗳𝗳: Show file differences not yet staged.

𝟮. 𝗴𝗶𝘁 𝗰𝗼𝗺𝗺𝗶𝘁 -𝗮 -𝗺 "𝗰𝗼𝗺𝗺𝗶𝘁 𝗺𝗲𝘀𝘀𝗮𝗴𝗲": Commit all tracked changes with a message.

𝟯. 𝗴𝗶𝘁 𝘀𝘁𝗮𝘁𝘂𝘀: Show the state of your working directory.

𝟰. 𝗴𝗶𝘁 𝗮𝗱𝗱 𝗳𝗶𝗹𝗲\_𝗽𝗮𝘁𝗵:Add file(s) to the staging area.

𝟱. 𝗴𝗶𝘁 𝗰𝗵𝗲𝗰𝗸𝗼𝘂𝘁 -𝗯 𝗯𝗿𝗮𝗻𝗰𝗵\_𝗻𝗮𝗺𝗲: Create and switch to a new branch.

𝟲. 𝗴𝗶𝘁 𝗰𝗵𝗲𝗰𝗸𝗼𝘂𝘁 𝗯𝗿𝗮𝗻𝗰𝗵\_𝗻𝗮𝗺𝗲: Switch to an existing branch.

𝟳. 𝗴𝗶𝘁 𝗰𝗼𝗺𝗺𝗶𝘁 --𝗮𝗺𝗲𝗻𝗱:Modify the last commit.

𝟴. 𝗴𝗶𝘁 𝗽𝘂𝘀𝗵 𝗼𝗿𝗶𝗴𝗶𝗻 𝗯𝗿𝗮𝗻𝗰𝗵\_𝗻𝗮𝗺𝗲: Push a branch to a remote.

𝟵. 𝗴𝗶𝘁 𝗽𝘂𝗹𝗹: Fetch and merge remote changes.

𝟭𝟬. 𝗴𝗶𝘁 𝗿𝗲𝗯𝗮𝘀𝗲 -𝗶: Rebase interactively, rewrite commit history.

𝟭𝟭. 𝗴𝗶𝘁 𝗰𝗹𝗼𝗻𝗲: Create a local copy of a remote repo.

𝟭𝟮. 𝗴𝗶𝘁 𝗺𝗲𝗿𝗴𝗲: Merge branches together.

𝟭𝟯. 𝗴𝗶𝘁 𝗹𝗼𝗴 --𝘀𝘁𝗮𝘁: Show commit logs with stats.

𝟭𝟰. 𝗴𝗶𝘁 𝘀𝘁𝗮𝘀𝗵: Stash changes for later.

𝟭𝟱. 𝗴𝗶𝘁 𝘀𝘁𝗮𝘀𝗵 𝗽𝗼𝗽: Apply and remove stashed changes.

𝟭𝟲. 𝗴𝗶𝘁 𝘀𝗵𝗼𝘄 𝗰𝗼𝗺𝗺𝗶𝘁\_𝗶𝗱: Show details about a commit.

𝟭𝟳. 𝗴𝗶𝘁 𝗿𝗲𝘀𝗲𝘁 𝗛𝗘𝗔𝗗~𝟭: Undo the last commit, preserving changes locally.

𝟭𝟴. 𝗴𝗶𝘁 𝗳𝗼𝗿𝗺𝗮𝘁-𝗽𝗮𝘁𝗰𝗵 -𝟭 𝗰𝗼𝗺𝗺𝗶𝘁\_𝗶𝗱: Create a patch file for a specific commit.

𝟭𝟵. 𝗴𝗶𝘁 𝗮𝗽𝗽𝗹𝘆 𝗽𝗮𝘁𝗰𝗵\_𝗳𝗶𝗹𝗲\_𝗻𝗮𝗺𝗲: Apply changes from a patch file.

𝟮𝟬. 𝗴𝗶𝘁 𝗯𝗿𝗮𝗻𝗰𝗵 -𝗗 𝗯𝗿𝗮𝗻𝗰𝗵\_𝗻𝗮𝗺𝗲: Delete a branch forcefully.

𝟮𝟭. 𝗴𝗶𝘁 𝗿𝗲𝘀𝗲𝘁: Undo commits by moving branch reference.

𝟮𝟮. 𝗴𝗶𝘁 𝗿𝗲𝘃𝗲𝗿𝘁: Undo commits by creating a new commit.

𝟮𝟯. 𝗴𝗶𝘁 𝗰𝗵𝗲𝗿𝗿𝘆-𝗽𝗶𝗰𝗸 𝗰𝗼𝗺𝗺𝗶𝘁\_𝗶𝗱: Apply changes from a specific commit.

𝟮𝟰. 𝗴𝗶𝘁 𝗯𝗿𝗮𝗻𝗰𝗵: Lists branches.

𝟮𝟱. 𝗴𝗶𝘁 𝗿𝗲𝘀𝗲𝘁 --𝗵𝗮𝗿𝗱: Resets everything to a previous commit, erasing all uncommitted changes.