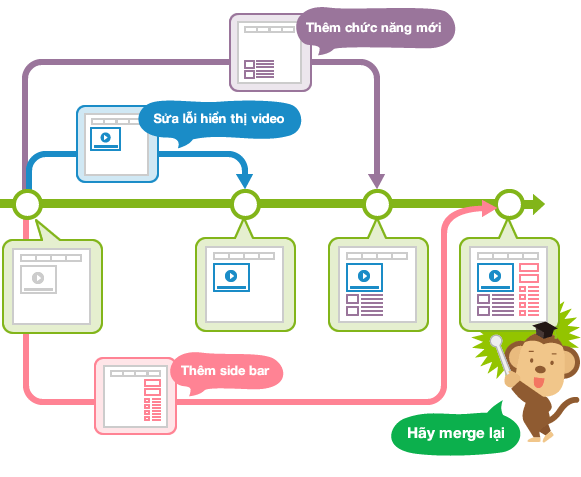
1. Khái niệm cơ bản

* Để phân nhánh và ghi lại luồng lịch sử. Brand đã phân nhánh sẽ không ảnh hưởng đến các brand khác nên có thể dễ dàng tiến hành nhiều thay đổi đồng thời trong cùng một repository
* Brand đã phân nhánh có thể chỉnh sửa tổng hợp lại thành 1 brand bằng cách meger với brand khác



* Sau khi hoàn thành công việc, sẽ merge vào nhánh chính để đưa các thay đổi của mình vào nhánh chính
* ADVANTAGE:
  + Không ảnh hưởng đến công việc các thành viên khác
  + Trace lịch sử thay đổi khi phát sinh lỗi
* Sẽ có 1 nhánh chính, các thành viên sẽ tự tạo ra nhánh của riêng mình để không ảnh hưởng đến công việc của các thành viên khác
* Khi tiến hành commit lần đầu trong repo thì git sẽ tạo 1 brand có tên là matter. Những lần commit sau sẽ được thêm vào brand matter này cho đến khi chuyển đổi brand
* 2 Phương pháp sử dụng để tạo brand hiệu quả: Brand tích hợp và brand chủ đề:
* Intergration Brand ( Brand tích hợp):

1. **Các lệnh cơ bản trong git**
   1. INIT

* git init: khởi tạo git
  1. CLONE
* Git clone <url>: clone 1 project trên server về thưc mục
  1. BRANCH:
* Git branch : kiểm tra branch hiện tại\
* Git branch -r/-l: lấy danh sách các nhánh trên romote/local
* git branch <name\_branch> : tạo branch mới
* git branch -d <name\_branch> : Xóa branch
* git merge <message> : ghép nhánh
  1. Để đưa subsidiary vào branch master, thì trước hết sẽ di chuyển đến branch master.
     + git checkout master
     + git branch new-branch
     + git checkout new-branch
     + git add .
     + git commit –m "first-commit"
     + git checkout master
     + git merge new-branch
  2. CHECK OUT:
  + git checkout <name\_branch> : checkout 1 nhánh
  + git checkout -b <name\_branch> : tạo nhánh mới và check out nhánh đó
  + git checkout master: quay lại nhánh master
  1. ADD
* khi thay đổi source code (thêm , xóa, sửa..) cần cập nhật lên staging area
* git add . : cập nhật tất cả các file
* git add . <file name> : cập nhật theo file
  1. STATUS:
* kiểm tra trạng thái của working dictory và staging area
  + git status
  1. COMMIT
* ghi lại việc thay đổi và đẩy thông tin lên local repository
  + git commit -m 'comment…'
  1. PUSH
* cập nhật từ local repository lên server
  + git push origin <name\_branch> : cập nhật lên 1 branch
  + git remote add origin <remote\_url> : add mới 1 remote
  1. FETCH
* git fetch <name\_branch> : lấy source từ server về local reposiotry mà không làm thay đổi local Repository
* Nên sử dụng khi có nhiều người cùng làm việc trên cùng 1 nhánh
* Tổ chức và lưu giữ các commit của git:
  + Git lưu trữ tất cả các commit, local, branch trong thư mục ./.git/objects
  + Git lưu trữ các local branch commit và remote branch commit tách biệt rõ ràng thông qua việc sử dụng *branch ref.* Ref tham chiếu tới các local branch được lưu trữ trong ./.git/refs/heads/. Khi thực hiện lệnh git branch -l nó sẽ show tất cả các local branch
  + Remote branch cũng giống như local branch nhưng chúng sẽ ánh xạ tới các commit từ khu lưu trữ của người khác, để show các remote branch ta sử dụng lệnh git branch -r
  + Ta có thể kiểm tra remore branch giống như một local branch tuy nhiên điều này sẽ đặt ta vào trạng thái Detached HEAD (giống như kiểm tra một commit cũ) hoắc sử dụng các lệnh git checkout và git log
  1. PULL
  + git pull origin master : tải xuống nội dung remote và thực hiện 1 lệnh git merger để ghép và commit các nội dung mới của remote Repository
  + Nên sử dụng khi chỉ có 1 người làm việc trên 1 nhánh
  1. OTHER
  + git log : xem lịch sử sourcecode
  + git diff <local branch> <remote>/<remote branch> : hiển thị thay đổi giữa nhánh local và remote

1. **Git Experiment**
   1. *Đẩy project lên remote repository(gitlab, github):*
   * git init
   * git add .
   * git commit -m"finsh-project"
   * git remote add origin < link github của bạn >
   * git push -u origin master
   1. *Tạo và đẩy branch mới trên remote*

* Tạo một local branch và check out:
  + git checkout -b <branch-name>
* Khi đẩy lên remote sẽ tự động tạo branch mới :
  + git push --set-upstream <remote-name> <local-branch-name>
  1. *Connect other repository:*
  + git remote add <alisas name> < other remote\_url> // *thêm repository*
  + git remote set-url <alias name> <other remote\_url> // *set alias cho repository*
  + git fetch <alias name>
  + git checkout <local\_name> <alias\_name>/<remote branch name> // *tạo branch local tương ứng với 1 branch remote khác repo*
* merge từ 1 nhánh remote (repo khác) vào nhánh local hiện tại
  + git checkout <local branch>
  + git merge template/master --allow-unrelated-histories