Docker dùng cho đội back end chủ yếu

Docker cho phép cài đặt Linux lên trên host OS

Public Docker Repository

-------------- Docker commands -----------

docker version //Lấy thông số Docker

docker ps //Lấy danh sách các container đang chạy

docker ps -a //Lấy tất cả các container đang chạy hoặc đang ngủ

docker pull <image\_name:tag>

docker rm <container\_name>

docker image ~ class

docker container ~ object/instance

Khởi động docker container

docker run --name <new\_container\_name> -it <container\_image> /bin/sh

docker run --name teo -it alpine:latest /bin/sh

docker start <container\_name>

docker stop <container\_name>

docker exec teo -it /bin/sh

-i interactive

-t terminal

---------------- Linux command -----------

# login as super user or root

$ login as normal user

ls liệt kê file thư mực hiện thời

ls -a liệt kê file thư mực hiện thời kể cả ẩn

whoami trả về current login user

root: là user có quyền cao nhất

clear xóa màn hình console

history liệt kê các lệnh trong lịch sử chúng ta vừa gõ

exit thoát khỏi terminal

pwd in ra thư mục hiện thời ~ print working directory

/usr/bin # là dấu nhắc - prompt

cd <destination\_dir> change directory

/ root directory

~ user directory

. current directory

.. go parent directory

../.. go grand parent directory

mkdir <directory\_name>

touch <file\_name> tạo file rỗng

cat <file\_name> in nội dung file ra console

vi <file\_name> mở trình vi để sửa file

ESC + I: chuyển sang mode Insert để nhập dữ liệu

ESC chuyển về mode Normal để

:wq ~ write + quit

:w ~ write mà không quit

:q! ~ quit mà không lưu

Alpine là một biến thể Linux rất nhẹ, chuyển để build Docker image

apk update để cập nhật danh sách phần mềm cho Alpine

apk upgrade nâng cấp các phần mềm đã cài

apk add <package\_name> cài đặt gói phần mềm mới

nano là một công cụ edit file tốt hơn, thân thiện hơn vi

nó có thể high light syntax nếu cài đặt thêm bộ nhận dạng ngôn ngữ lập trình

Ctrl + X để thoát, muốn lưu lại thì gõ tiếp Y, không muốn lưu lại thì gõ N

Ctrl + W để ghi thay đổi

Ctrl + K để xóa một dòng hiện thời

2,git

Root(#) admin

Normal user($)

Esc chi doc

I them sua xoa

Cat os-release thong tin he dieu hanh

$ git log --all -- decorate --oneline –graph : xem re nhanh

**Cài đặt ubuntu**

*docker run --name ubuntu -it ubuntu:latest /bin/bash*

**Đăng nhập git**

$ git config --global user.name "techmaster"

$ git config --global user.email "cuong@techmaster.vn"

Khi làm việc với một local git repo thực chất lập trình viên chỉ sửa đổi file / folder trong thư mục làm việc. Từ thư mục làm việc cho đến khi code được chốt hạ (lưu phiên bản) thành commit, còn có một bước đệm trung gian gọi là staging area.

1. Dev sửa đổi code trong working directory
2. Dev sử dụng lệnh git add để chọn ra nhóm những file/folder chuẩn bị được commit đẩy lên vùng staging area
3. Dev có thể thêm / bớt các file/folder trong staging area
4. Dev dùng lệnh git commit  để lưu các thay đổi ở staging area thành một commit



**Tạo thư mục**

mkdir myweb

cd myweb

git init tạo thư mục .git ẩn

git init: tạo thư mục .git ẩn

**Tạo file**

Touch tênfile

Vd:

touch ReadMe.md

Kiểm tra trạng thái :

git status

chuyển file sang steging area

git add tên file

git add index.html

chuyển file sang repository

git commit –m “nội dung commit”

git commit -m "First commit"

xem lich sử

git log

git log –oneline

.gitignore được dùng để developer chỉ ra file / folder nào sẽ bị bỏ qua trong quá trình dev sửa đổi nội dung working directory. Những file này không được đẩy lên staging area trong lệnh **git add** hay không được liệt kê trong lệnh **git status**

Để không nhập lại, chúng ta cần phải lưu lại credential

git config credential.helper store

user name và password sẽ được lưu ở ~/.git-credentials. Bạn có thể xem lại bằng lệnh

$ cat ~/.git-credentials

https://gituser:password@github.com

Để xóa user name/pass gõ lệnh

git config credential.helper clear

Đẩy code lên git

Git push <tên rút gọn> <SH> <tên\_nhánh>

Vi du:

git push abc githttps://github.com/khuong02101997/demo-git-Vinid.git master

⬄ git push abc master

git fetch tải về danh sách các commites

git pull ban đầu giống với git fetch là lấy những thay đổi trên remote repository về, nhưng sau đó git pull thực hiện tiếp git merge để tái hợp code

// Tạo một branch

$ git branch dev

// Liệt kê branch ở local repository

$ git branch

dev

\* master

// Liệt kê branch ở remote repository

$ git branch -r

origin/master

// Chuyển sang branch mới

$ git switch dev

// Tạo một brach mới rồi chuyển sang

$ git checkout -b new\_branch

git switch chỉ chuyển giữa các branch

git checkout làm được nhiều việc hơn cụ thể

Xóa một branch ở remote repo

$ git push -d <remote\_name> <branch\_name>

Xoa branch(nhanh)

*$ git branch -d <branch\_name>*

unstage những file cần undo change. Ví dụ file demo.py  
*$ git restore --staged demo.py*

Unstage toàn bộ file, thư mục vừa được đẩy lên bằng lệnh git add   
*git rm -r –cached*

Tình huống: bạn sửa đổi một số file mã nguồn, bạn chưa đẩy những file này lên staging area và tất nhiên là chưa commit. Giờ bạn muốn hủy tất cả những sửa đổi và đưa working directory giống nội dụng của latest commit.

**Cách 1:**  
*$ git reset --hard*  
Lệnh này có thể chạy ở mọi phiên bản git, undo các thay đổi: sửa và xóa đối với những mã nguồn đang được quản lý trong git. Nếu có vài file mới thêm, chưa commit thì lệnh này sẽ không ảnh hưởng

**Cách 2:**  
*git restore .*  
Lệnh này khá mới, ở các phiên bản git cũ có thể chưa có

**Cách 3:**  
git checkout -- .  
Lệnh này lấy nội dung trong lần commit cuối cùng đè lên những thay đổi trong working directory

**git checkout -- SpecificFileName**  
Lệnh này lấy nội dung file cụ thể SpecificFileName trong lần commit cuối cùng đè lên những thay đổi trong working directory

git stash save <tên\_file> để lưu một bản nháp mới và đặt tên cho nó

git stash list để liệt kê các nháp đã lưu, các bản nháp sẽ được đánh số thứ tự 0-1-2-3-4.. 0 là gần đây nhất.

**dasjhkfh**

./scenario/\*\_test.js