VIỆN CÔNG NGHỆ THÔNG TIN T3H

Đào tạo chuyên sâu - Trải nghiệm thực tế

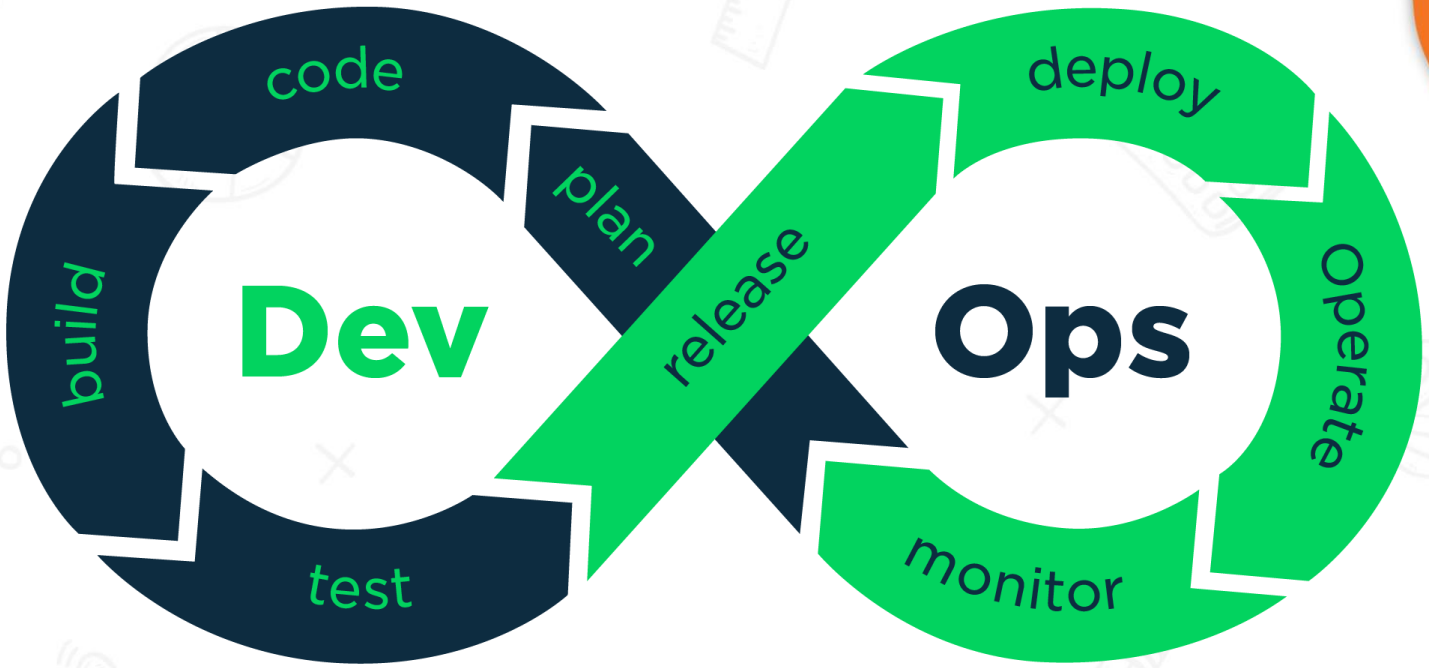
Môn học

DevOps

Giảng viên

Tan Do

0868880797



Nội dung buổi 08

- Git là gì? Các khái niệm cơ bản
- Cài đặt Git
- Các lệnh git cơ bản
- Đăng ký GitHub
- CICD là gì?
- Lợi ích và quy trình triển khai CICD
- Jenkins là gì? Hướng dẫn cài đặt & tạo job đầu tiên
- Sử dụng Jenkins CLI và xóa job

Git là gì?

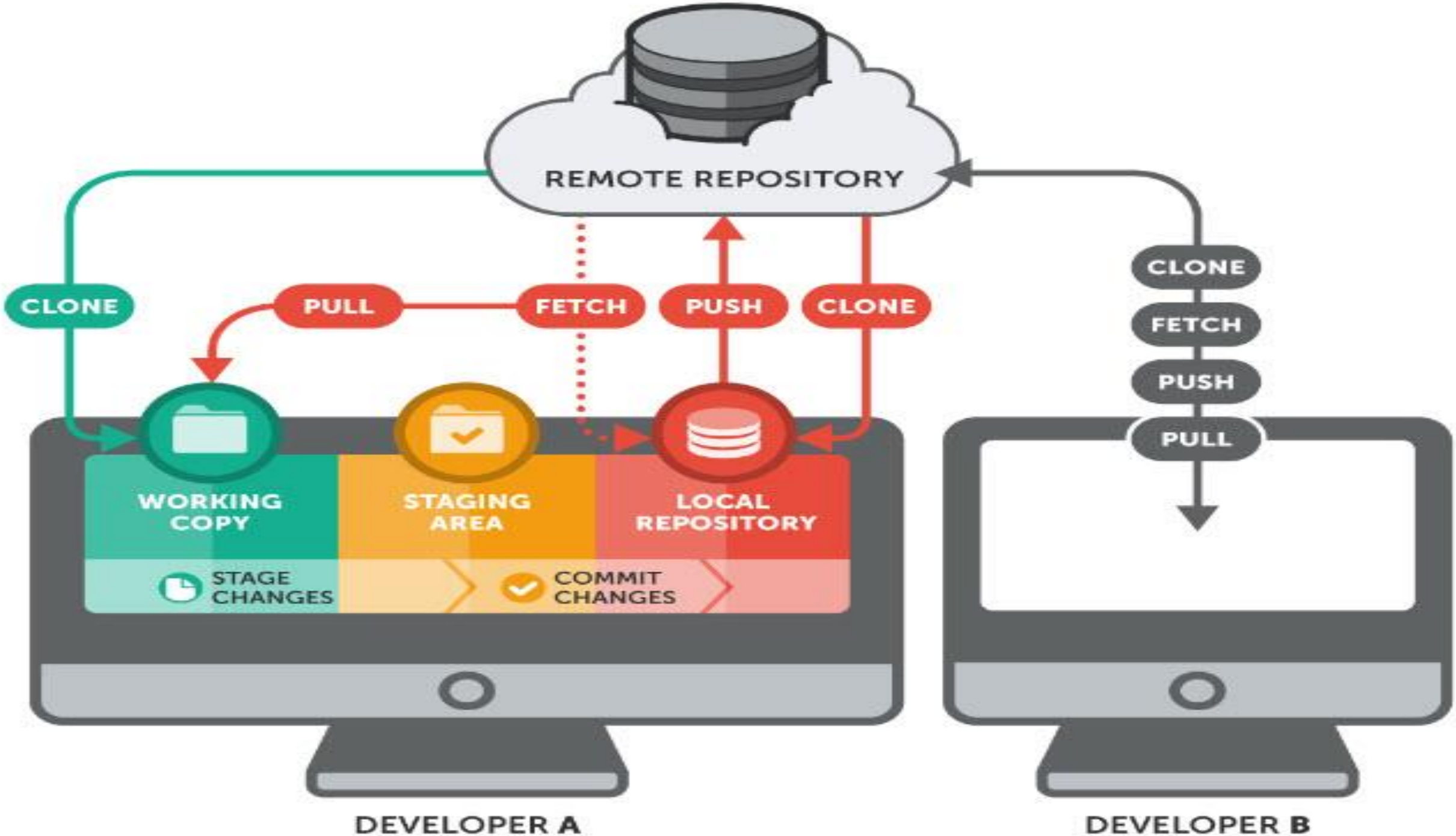
- Là một hệ thống quản lý phiên bản phân tán (Distributed Version Control System – DVCS)
- Đảm bảo không có xung đột mã giữa các nhà phát triển
- Là một trong những hệ thống quản lý phiên bản phân tán phổ biến nhất hiện nay
- Cung cấp cho mỗi lập trình viên kho lưu trữ (repository) riêng chứa toàn bộ lịch sử thay đổi
- Cho phép quay lại phiên bản cũ hơn của mã

Các khái niệm cơ bản trong Git

- Repository (Repo)
- Branch

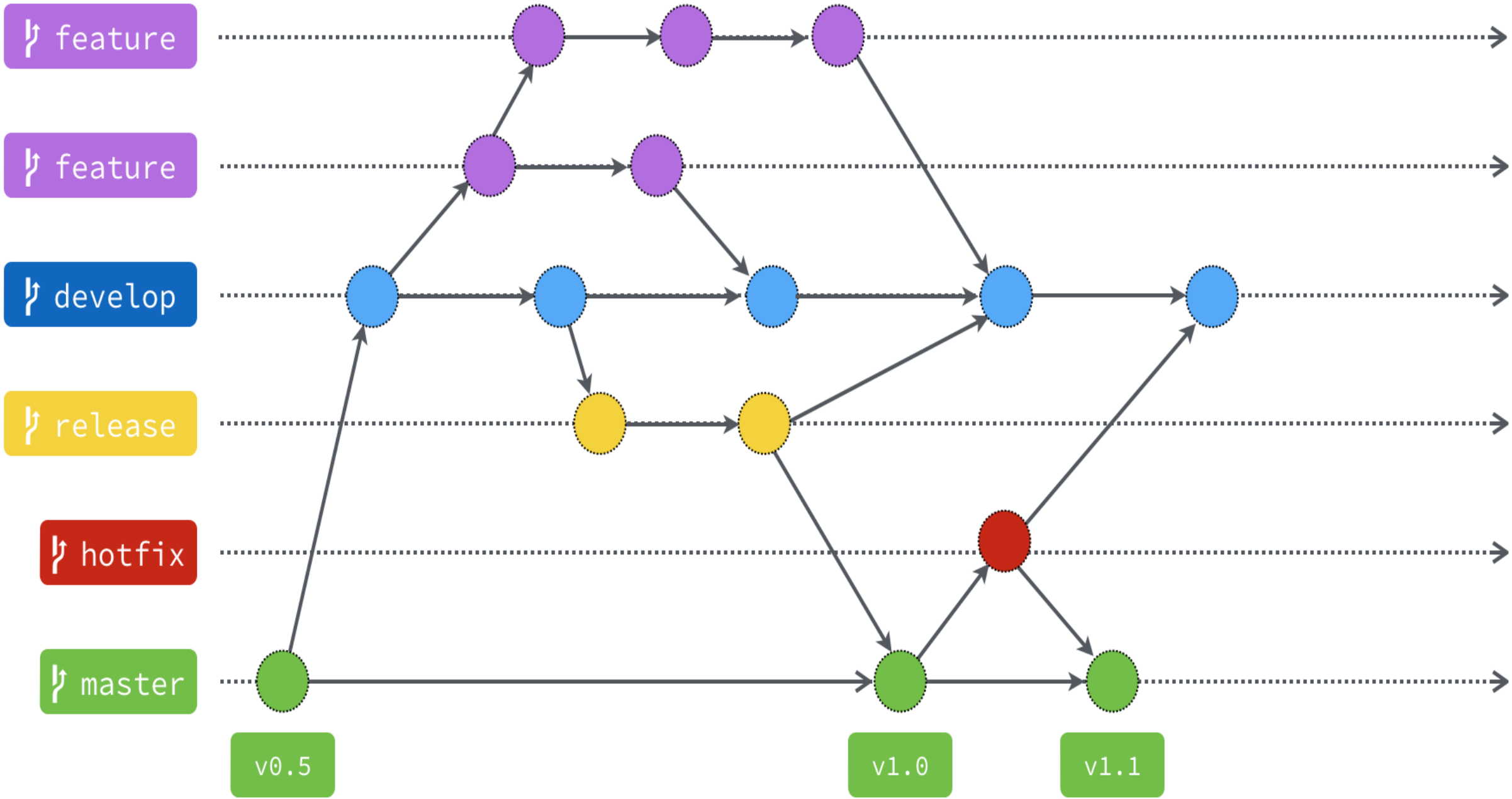
Các khái niệm cơ bản trong Git

- REPOSITORY (Repo)
  - Là nơi chứa tất cả mã nguồn cho một dự án được tạo bởi Git (khai báo thư mục chứa dự án)
  - Có hai cấu trúc dữ liệu chính đó là Object store và Index được lưu trữ ẩn trong thư mục .git
  - Có hai loại repository là local repo và remote repo
    - Local repository: Là repo được cài đặt trên máy tính của lập trình viên, repo này sẽ đồng bộ hóa với remote bằng các lệnh của Git
    - Remote repository: Là repo được cài đặt trên server chuyên dụng, điển hình hiện nay là Github, Bitbucket


REMOTE REPOSITORY
CLONE
PULL
FETCH
PUSH
CLONE
WORKING COPY
STAGING AREA
LOCAL REPOSITORY
STAGE CHANGES
COMMIT CHANGES
DEVELOPER A
CLONE
FETCH
PUSH
PULL
DEVELOPER B

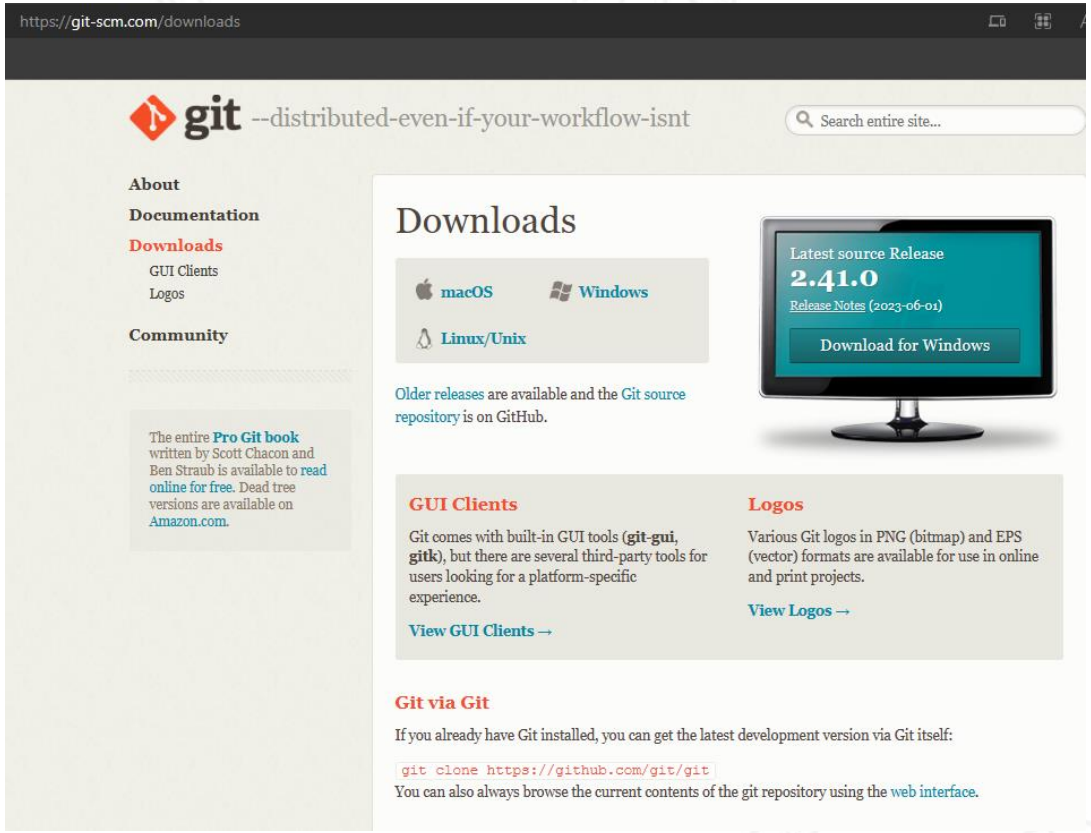
Các khái niệm cơ bản trong Git

- BRANCH
  - Là những nhánh ghi lại luồng thay đổi của lịch sử
  - Các hoạt động trên mỗi branch sẽ không ảnh hưởng lên các branch khác nên có thể tiến hành nhiều thay đổi đồng thời trên một repository giúp giải quyết nhiều nhiệm vụ cùng lúc
  - Khi tạo một repository thì Git sẽ thiết lập branch mặc định là master, nghĩa là nó sẽ tự tạo một branch master và mọi hoạt động của ban lúc này đều nằm trên branch master


feature
feature
develop
release
hotfix
master
v0.5
v1.0
v1.1

Cài đặt Git

https://git-scm.com/downloads



Cài đặt Git

Config lần đầu:

- git config --global user.name “Tan Do“
- git config --global user.email tan@ix.com

Kiểm tra thông tin:

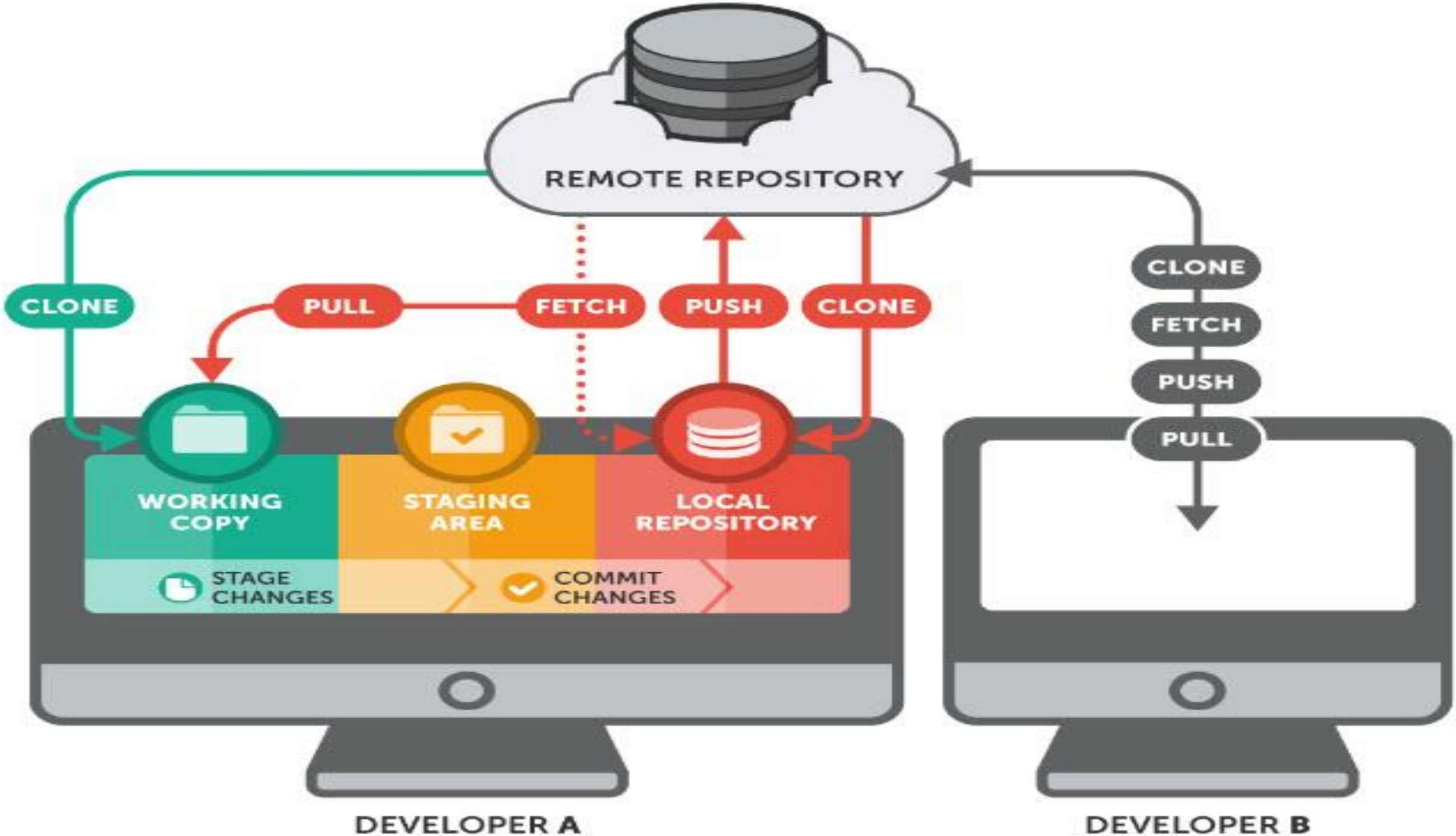
- git config –list
- git config --list --show-origin
- git config --get remote.origin.url

Các lệnh git cơ bản

- git init
  - Khởi tạo 1 git repository 1 project mới hoặc đã có
  - Run git init trong thư mục gốc của dự án
- git clone
  - Copy 1 git repository từ remote source.
  - git clone <:clone git url:>
- git status
  - Check trạng thái của những file đã thay đổi trong thư mục làm việc

Các lệnh git cơ bản

- git add
  - Để đưa một tập tin vào Staging Area
  - Cách dùng:
    - git add tên_file
    - Thêm hết file của thư mục thì:
      - git add all
      - git add –a
      - git add .
- git commit
  - Lưu lại một snapshot của các sự thay đổi trong thư mục làm việc, các thay đổi nằm trong Staging Area
  - Mỗi lần commit nó sẽ được lưu lại lịch sử chỉnh sửa của code kèm theo tên và địa chỉ email của người commit
  - Có thể khôi phục lại tập tin trong lịch sử commit của nó để chia cho một branch khác, nên dễ dàng khôi phục lại các thay đổi trước đó
  - git commit -m “message”


REMOTE REPOSITORY
CLONE
PULL
FETCH
PUSH
CLONE
WORKING COPY
STAGING AREA
LOCAL REPOSITORY
STAGE CHANGES
COMMIT CHANGES
DEVELOPER A
CLONE
FETCH
PUSH
PULL
DEVELOPER B

Các lệnh git cơ bản

- git push
  - Đẩy các thay đổi lên remote
  - git push <:remote:> <:branch:>
- git pull
  - Kéo các thay đổi từ remote về local
  - git pull <:remote:> <:branch:>

Các lệnh git cơ bản

- git branch
  - Liệt kê tất cả các branch (nhánh).
  - git branch hoặc git branch -a
- git checkout
  - Chuyển sang branch khác
    - git checkout <: branch:>
  - Tạo và chuyển sang branch khác
    - git checkout -b <: branch:>

Các lệnh git cơ bản

- git stash
  - Lưu thay đổi mà không muốn commit ngay lập tức
  - git stash trong thư mục làm việc
- git merge
  - Merge 2 branch lại với nhau
  - git merge <:branch_muon_merge:>
- git reset
  - Khi đã đưa một tập tin nào đó vào Staging Area nhưng bây giờ muốn loại bỏ nó ra khỏi đây để không phải bị commit theo
  - git reset HEAD tên_file

Các lệnh git cơ bản

- git remote
  - Để check remote/source đang có hoặc add thêm remote
    - git remote #để kiểm tra và liệt kê
      - git remote -v
    - git remote add <: remote_url:> # để them
      - git remote add origin https://github.com/xxx
      - git push origin master
    - git remote remove remote_name # để remove
      - git remote remove origin
    - git remote rename ten_cu ten_moi # để rename
    - git remote show origin # để xem remote origin

Đăng ký GitHub

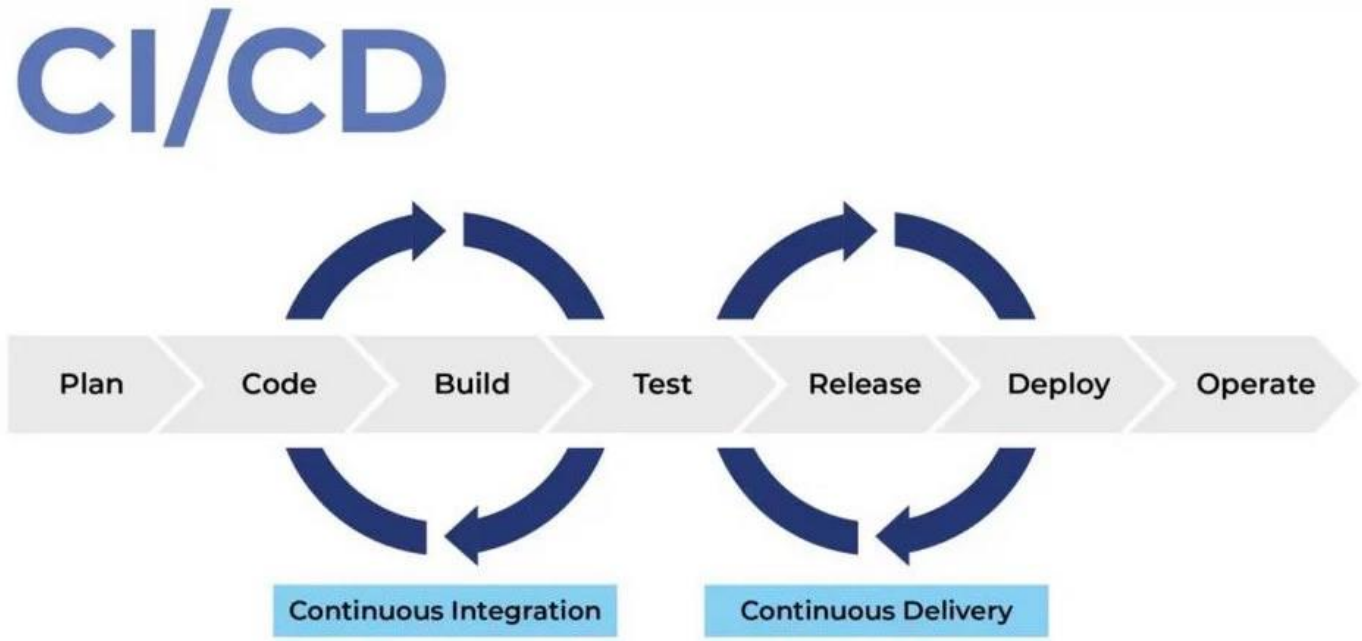
- Tạo tài khoản trên GitHub
  - https://github.com/
- Thực hành tạo repo trên GitHub và push code
- Sử dụng trong thực tế các dự án

Quy trình làm việc

- Đầu tiên, các thành viên trong team dev sẽ bắt đầu pull code mới nhất từ repo về branch để thực hiện các yêu cầu chức năng nhất định
- Tiếp đó là quá trình lập trình và test code để đảm bảo chất lượng của chức năng cũng như toàn bộ source code
- Thành viên code xong thì sẵn sàng cho việc commit vào branch develop của team
- Thành viên cập nhật code mới từ repo về local repo
- Merge code và giải quyết conflict
- Build và đảm bảo code pass qua các tests dưới local
- Commit code lên repo
- Máy chủ CI lắng nghe các thay đổi code từ repository và có thể tự động build/test, sau đó đưa ra các thông báo (pass/failure) cho các thành viên

CICD là gì?

- Continuous Integration/Continuous Deployment
- Tích hợp liên tục/ Triển khai liên tục
- Là một phương pháp cơ bản thường được áp dụng trong quy trình phát triển phần mềm



CICD là gì?

- Continuous Integration (CI)
  - Đây là quá trình tích hợp mã nguồn từ nhiều nhà phát triển vào một kho lưu trữ chung hàng ngày. Sau đó, hệ thống tự động xây dựng, kiểm thử và kiểm tra tích hợp. Mục tiêu của CI là giảm thiểu xung đột và lỗi hệ thống đo tích hợp mã nguồn
- Continuous Deployment (CD)
  - CD là quá trình tự động hóa việc triển khai các phiên bản mới của phần mềm vào môi trường sản phẩm hoặc thử nghiệm một cách nhanh chóng và tin cậy. Mục tiêu của CD là tối ưu hóa quá trình triển khai phần mềm, từ việc xây dựng, kiểm thử đến việc đưa phần mềm vào sử dụng thực tế
- CI/CD giúp:
  - Tăng cường tính ổn định và tin cậy của việc phát triển và triển khai phần mềm
  - Đồng thời giảm thiểu thời gian và rủi ro trong quá trình phát triển sản phẩm

Lợi ích khi ứng dụng CI/CD

- Tăng tính ổn định: CI/CD giúp phát hiện sớm và giảm thiểu lỗi do tích hợp mã nguồn, từ đó tăng cường tính ổn định của hệ thống
- Tối ưu hóa quy trình phát triển: CI/CD tự động hóa việc kiểm tra tích hợp, xây dựng và triển khai, giúp tối ưu hóa quy trình phát triển phần mềm. Từ đó giảm thiểu thời gian và công sức cần thiết cho các công việc lặp đi lặp lại
- Đảm bảo chất lượng: Tích hợp liên tục và triển khai liên tục giúp đảm bảo chất lượng sản phẩm thông qua việc tự động hóa các bước kiểm tra, giúp giảm thiểu lỗi và rủi ro liên quan đến triển khai sản phẩm
- Đẩy nhanh quá trình triển khai: CI/CD cho phép triển khai các phiên bản mới của phần mềm nhanh chóng và tin cậy. Nền tảng tăng cường khả năng thích ứng và phản hồi nhanh chóng với yêu cầu thay đổi từ người dùng và thị trường

Quy trình ứng dụng CI/CD

- Tích hợp liên tục (Continuous Integration - CI)
  - Tích hợp mã nguồn: Nhóm phát triển code và tích hợp mã nguồn vào hệ thống quản lý phiên bản, ví dụ như Git
  - Kiểm tra tương quan: Mã nguồn được tự động kiểm tra liên tục để đảm bảo tính toàn vẹn và tính tương thích
  - Xây dựng tự động: Tự động xây dựng ứng dụng từ mã nguồn đã tích hợp để tạo ra phiên bản có thể triển khai

Quy trình ứng dụng CI/CD

- Triển khai liên tục (Continuous Deployment - CD)
  - Kiểm thử tự động: Sau khi xây dựng, phiên bản ứng dụng sẽ trải qua các bước kiểm thử tự động để đảm bảo chất lượng
  - Triển khai tự động: Phiên bản ứng dụng đã kiểm thử sẽ được triển khai tự động vào một môi trường thử nghiệm hoặc sản phẩm
  - Kiểm tra và giám sát: Sau khi triển khai, hệ thống sẽ tự động thực hiện các bước kiểm tra và giám sát để đảm bảo rằng ứng dụng hoạt động đúng như kỳ vọng
  - Quy trình này đảm bảo rằng mọi thay đổi trong mã nguồn sẽ được tích hợp, kiểm tra, triển khai một cách tự động và liên tục. Nền tảng góp phần tối ưu hóa quy trình phát triển và đảm bảo chất lượng của sản phẩm phần mềm

Các công cụ CI/CD phổ biến

- Jenkins
- Gitlab CI/CD
- GitHub Action
- Azure DevOps
- TeamCity
- CircleCI
- Travis CI

Jenkins là gì?

- Là công cụ mã nguồn mở được sử dụng để thực hiện tích hợp liên tục
- Cho phép các nhà phát triển tự động hóa quá trình kiểm tra mã nguồn và thực hiện xây dựng sản phẩm, giúp đảm bảo rằng mã nguồn được tích hợp và kiểm tra một cách liên tục và đồng nhất
- Cung cấp hàng loạt tính năng linh hoạt cho việc lập lịch tác vụ tự động, quản lý nguồn mã, xây dựng sản phẩm và triển khai mã nguồn
- Thông qua việc cảnh báo khi có lỗi, Jenkins có tác dụng tăng cường chất lượng phần mềm và giảm thiểu thời gian phát triển
- Tích hợp môi trường đơn giản để tạo và quản lý các quy trình CI/CD và liên tục cung cấp phản hồi về chất lượng mã nguồn, tiến độ dự án

Hướng dẫn cài đặt Jenkins

- Bước 1: Yêu cầu hệ thống
  - Jenkins có thể chạy trên nhiều hệ điều hành như Windows, macOS, hoặc Linux
  - Xác định hệ điều hành đang sử dụng và đảm bảo hệ thống đáp ứng các yêu cầu cần thiết
- Bước 2: Cài đặt Java
  - Jenkins chạy trên Java, vì vậy cần cài đặt Java Development Kit (JDK) trên hệ thống nếu chưa có
- Bước 3: Cài đặt Jenkins
  - Truy cập trang web chính thức của Jenkins (https://www.jenkins.io/download/) để tải xuống phiên bản Jenkins phù hợp với OS
- Bước 4: Khởi động Jenkins
  - Sau khi cài đặt xong, khởi động Jenkins thông qua trình duyệt web bằng cách truy cập địa chỉ localhost
  - Địa chỉ IP máy chủ với cổng mặc định (http://localhost:8080)
- Bước 5: Hoàn thiện quá trình cài đặt
  - Khi truy cập lần đầu, Jenkins sẽ yêu cầu nhập "Admin password“
  - Sẽ tìm thấy mật khẩu này trong các tệp tin cài đặt của Jenkins (hoặc docker logs container_id)
- Bước 6: Cài đặt plugins
  - Sau khi hoàn tất bước trên, sẽ được yêu cầu cài đặt các plugin cần thiết cho Jenkins
  - Có thể chọn cài đặt các plugin theo mặc định hoặc cài đặt thủ công theo nhu cầu cụ thể của dự án
- Bước 7: Tạo tài khoản quản trị
  - Cuối cùng, tạo tài khoản quản trị cho Jenkins

Hướng dẫn cài đặt Jenkins bằng Docker

- Tham khảo
  - https://hub.docker.com/r/jenkins/jenkins
  - https://github.com/jenkinsci/docker/blob/master/README.md

```
docker run -p 8080:8080 -p 50000:50000 --restart=on-failure jenkins/jenkins:lts
```

```
docker run -p 8080:8080 -p 50000:50000 --restart=on-failure -v jenkins_home:/var/jenkins_home -d --name jenkins jenkins/jenkins:lts
```

Hướng dẫn cài đặt Jenkins bằng Docker

```
services:
  jenkins:
    container_name: jenkins
    image: jenkins/jenkins:lts
    privileged: true
    user: root
    ports:
    - 8080:8080
    - 50000:50000
    volumes:
      - jenkins_home:/var/jenkins_home
volumes:
  jenkins_home:
```

Hướng dẫn cài đặt Jenkins bằng Docker

- Run Jenkins container với user root
  - Dockerfile

```
FROM jenkins/jenkins:lts
USER root
RUN curl -sSL https://get.docker.com/ |sh && \
    usermod -aG docker jenkins && \
    usermod -aG root jenkins
USER jenkins
```

  - docker-compose

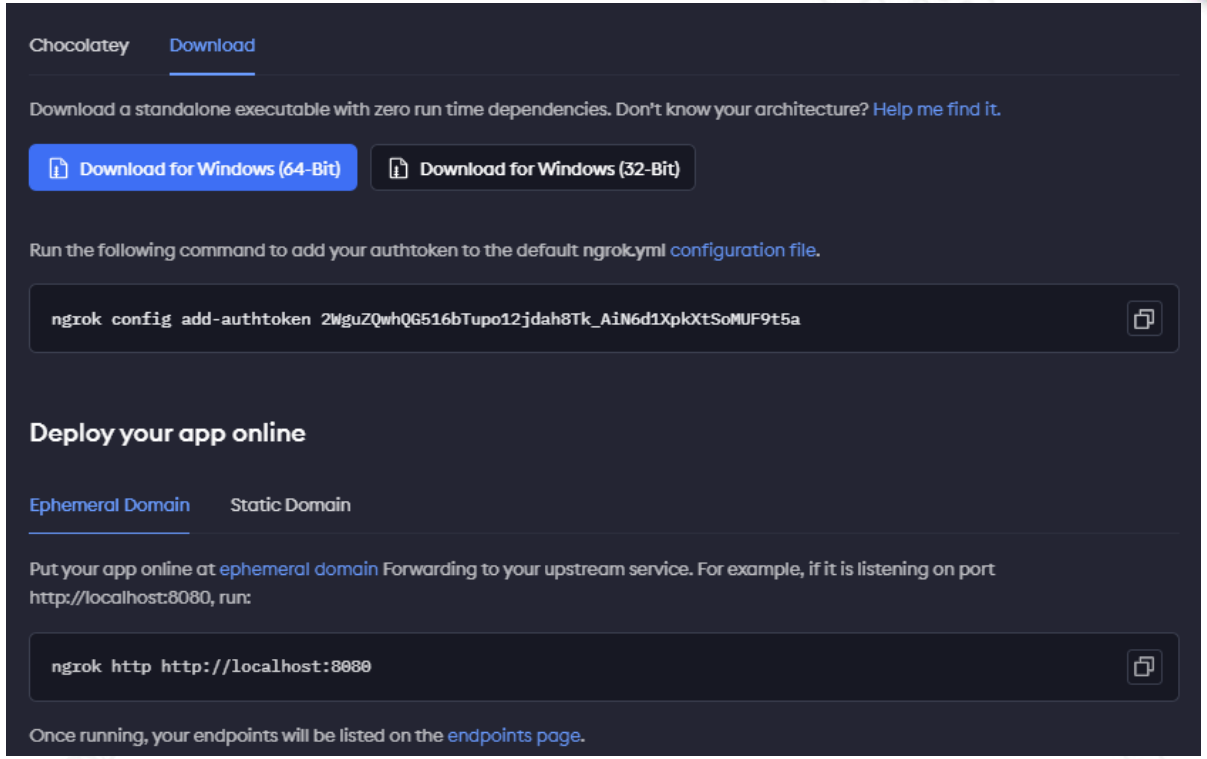
```
name: jenkins
services:
  jenkins:
    container_name: jenkins
    build:
      context: .
      dockerfile: ./Dockerfile
    ports:
      - 8080:8080
      - 50000:50000
    volumes:
      - jenkins_home:/var/jenkins_home
      - /var/run/docker.sock:/var/run/docker.sock

volumes:
  jenkins_home:
```

  - docker exec -it jenkins docker version
  - docker exec -it jenkins ssh -V

Hướng dẫn cài đặt ngrok

- Đăng ký tài khoản và login trên ngrok
- https://ngrok.com/download
- Giải nén và run lệnh ngrok config
- Sau khi cài đặt xong run lệnh
  - ngrok http 8080
  - Kiểm tra url public mới



Tạo job đầu tiên với Jenkins

- Click “Create a job” hoặc “New Item”
  - Nhập tên job. Vd “job1”
  - Chọn Freestyle project
  - Click OK
- Click Build steps > Add build step
  - Chọn Execute shell (nếu chạy trên Linux)

```
echo “Hello World”
echo `date`
echo $(date)
```

  - Save > Build Now
- Kiểm tra thông tin trong Console Output

Tạo job đầu tiên với Jenkins

- Run lệnh dưới
  - docker exec -it Jenkins bash
  - NAME=Tan Do
  - echo $NAME
  - date
  - cat /tmp/info
- Đưa lệnh trên vào job
  - Chọn Execute shell (nếu chạy trên Linux)

    NAME=Tan Do
    echo “Hello $NAME, now is $(date)”
    echo “Hello $NAME, now is $(date)” > /tmp/info
  - Save > Build Now

Tạo job đầu tiên với Jenkins

- Run lệnh dưới
  - docker exec -it Jenkins bash
  - vi script.sh
    - FNAME=$1
    - LNAME=$2
    - echo “Hello $FNAME, $LNAME”
  - ./script.sh
  - chmod +x ./script.sh
  - ./script.sh
  - cat script.sh
  - ./script.sh Tan Do
  - docker cp script.sh jenkins:/tmp/script.sh
- Đưa lệnh trên vào job
  - Chọn Execute shell (nếu chạy trên Linux)
    /tmp/script.sh Tan Do
  - Save > Build Now
- Đưa lệnh trên vào job
  - Chọn Execute shell (nếu chạy trên Linux)
    FNAME=Tan
    LNAME=Do
    tmp/script.sh $FNAME $LNAME
  - Save > Build Now

Tạo job đầu tiên với Jenkins

- Chỉnh sửa configure của job
  - General > This project is parameterized > Add Parameter
    - Add LNAME, FNAME as String Parameter
  - Đưa lệnh trên vào job
    - Chọn Execute shell (nếu chạy trên Linux)

      tmp/script.sh $FNAME $LNAME
    - Save > Build Now

Sử dụng Jenkins CLI và xóa job

- Sử dụng Jenkins CLI
  - Truy cập Manage Jenkins > Jenkins CLI
  - Download file jenkins-cli.jar từ hướng dẫn
    - java -jar jenkins-cli.jar -s http://localhost:8080/ help
  - Lấy thông tin xác thực
    - Truy cập User (Admin) lấy thông tin Jenkins User ID (admin)
    - Truy cập User (Admin) > Configure > API Token > Tạo mới token
  - Truy cập vào Jenkins bằng CLI
    - Mở màn hình dòng lệnh CLI

```
export JENKINS_USER_ID=admin
export JENKINS_API_TOKEN=11a863918810891530c11fb45ce3325766
```

- Xóa job bằng CLI
  - Tại nơi chứa file jenkins-cli.jar, mở CLI

```
java -jar jenkins-cli.jar -s http://localhost:8080/ version
java -jar jenkins-cli.jar -s http://localhost:8080/ delete-job job1
```

  - Kiểm tra thông tin job đã xóa

T3H
IT-Institute

VIỆN CÔNG NGHỆ THÔNG TIN T3H

Đào tạo chuyên sâu - Trải nghiệm thực tế

THANK YOU

www.t3h.edu.vn