- Git là một một hệ thống quản lý phiên bản mã nguồn mở theo hướng phân tán (Distributed Version Control Systems), ra đời vào năm 2005 và hiện được sử dụng rộng rãi, gồm kho ở máy chủ (remote) và máy cá nhân (local). Những kho lưu trữ ở phía cá nhân là một bản sao của kho lưu trữ chung, trường hợp kho chứa ở máy chủ trục trặc hoặc phát sinh những sự cố không mong muốn thì kho chứa ở máy cá nhân có thể sao chép ngược trở lại máy chủ để khôi phục toàn bộ hệ thống.

- Version Control System (VCS) là một loại hệ thống giúp chúng ta quản lý các thay đổi trong source code theo dấu thời gian một cách dễ dàng. Một cách dễ hiểu, VCS lưu trữ source code và mọi thay đổi ta thực hiện trên đó dựa theo dòng thời gian, từ đó có thể đưa code quay trở về trạng thái cũ, hoặc kiểm tra những thay đổi đã thực hiện để dễ dàng sửa lỗi, nâng cấp,...

- VCS có hai loại:

* Centralized Version Control Systems (CVCS) hay còn gọi là VCS tập trung. Tất cả các folder, file của dự án sẽ nằm tập trung tại 1 server lớn duy nhất, tất cả các thành viên tham gia dự án có thể lấy code về, chỉnh sửa, sau đó đẩy code trở lại server. VCS tập trung cũng có hạn chế, nếu không may xảy ra sự cố, mà các sao lưu dự phòng chưa được tạo ra tính đến thời điểm đó, bạn sẽ mất toàn bộ lịch sử của dự án đó, ngoại trừ những phiên bản cục bộ mà người dùng có được trên máy tính cá nhân.
* Khắc phục nhược điểm của VCS thì một loại VCS nữa được đề xuất đó là Distributed Version Control Systems (DVCS) hay Hệ thống quản lý phiên bản phân tán. Nó giống với CVCS là có một máy chủ, ở đó có Database lưu giữ các phiên bản của file, tuy nhiên khác biệt đó là các máy client (các developer) kết nối vào thì nó không chỉ lấy file mà nó lấy luôn cả hệ thống Database. Điều này có nghĩa là khi Server bị ngắt, các máy client vẫn làm việc bình thường trên Database ở máy trạm, sau đó commit (chuyển) lên Server sau, hoặc Database ở Server bị lỗi thì bất kỳ máy client nào đều có thể phục hồi lại cho Server.

- Tác dụng chính của VCS :

* Lưu lại lịch sử các version của bất kỳ thay đổi nào của dự án. Giúp xem lại các sự thay đổi hoặc khôi phục (revert) lại sau này.
* Việc chia sẻ code trở nên dễ dàng hơn, các lập trình viên có thể chia sẻ public cho bất kỳ ai, hoặc private chỉ cho một số người có thẩm quyền để truy cập vào và lấy code về.

- Vốn là một VCS nên Git cũng ghi nhớ lại toàn bộ lịch sử thay đổi của source code trong dự án. Lập trình viên nào sửa file, thêm dòng code tại đâu, xóa dòng code ở hàng nào…đều được Git ghi nhận và lưu trữ lại.

Sự khác biệt chính giữa Git và bất kỳ VCS nào khác là cách Git nghĩ về dữ liệu của nó.

* Về mặt khái niệm, hầu hết các hệ thống khác đều lưu trữ thông tin dưới dạng danh sách các thay đổi dựa trên file. Các hệ thống này (CVS, Subversion, Perforce, Bazaar, v.v.) xem thông tin chúng lưu giữ dưới dạng như là một tập hợp các file và những thay đổi được thực hiện đối với các file tại các khoảng thời gian khác nhau.
* Tuy nhiên, Git không nghĩ đến việc lưu trữ dữ liệu của mình như các VCS khác. Thay vào đó, Git coi thông tin được lưu trữ như là một tập hợp các snapshot – ảnh chụp toàn bộ nội dung tất cả các file tại các thời điểm khác nhau.
* Mỗi khi bạn “commit”, Git sẽ “chụp” và tạo ra một snapshot, rồi dùng một tham chiếu dẫn tới snapshot đó. Để hiệu quả, nếu các tệp không thay đổi, Git sẽ không lưu trữ lại file — mà chỉ là một liên kết đến tệp giống file trước đó mà nó đã lưu trữ.
* Đây là điểm khác biệt quan trọng giữa Git và gần như tất cả các VCS khác. Điều này làm cho Git giống như một hệ thống tệp nhỏ với một số công cụ cực kỳ mạnh mẽ được xây dựng trên nó, thay vì chỉ đơn giản là một VCS.

- Khi làm việc với Git, chúng ta sẽ có các lợi ích như:

* Quản lý code tốt hơn: nếu bạn vô tình (hoặc cố ý) nghịch ngợm và tạo ra một đàn bug nhưng không thể Ctrl Z, bạn sẽ phải làm gì? Hệ thống phiên bản của Git sẽ giúp bạn quay lại phiên bản trước đó mà không phải sao chép file theo hướng thủ công.
* Làm việc nhóm dễ dàng: Trước khi biết đến Git, có phải bạn đã từng share code cho đứa bạn thân bằng cách copy từng đoạn code rồi dán vào khung chat, hay upload từng file lên drive rồi share cho người khác? Với Git, việc chia sẻ code trở nên dễ hơn bao giờ hết, bạn có thể đẩy (push) code lên máy chủ (remote) và người khác kéo (pull) code về máy của họ (local). Cách làm này ít rủi ro và tiêu tốn ít tài nguyên cũng như thời gian của bạn so với cách truyền thống.
* Mở rộng tính code nhưng không ảnh hưởng đến phiên bản gốc: Bạn là người sáng tạo, bạn muốn tạo ra một tính năng mới nhưng sợ nó sẽ làm hỏng cả chương trình. Chức năng phân nhánh của Git cho phép bạn làm việc tự do ở một nhánh (branch) riêng mà không ảnh hưởng đến chương trình gốc, nếu chức năng mới hoạt động đúng như yêu cầu, bạn có thể gộp nhánh của bạn vào nhánh chính và chương trình gốc sẽ có tính năng bạn vừa tạo ra.
* Git miễn phí: bạn chỉ cần download và cài đặt Git vào máy tính là có thể sử dụng được rồi. Vì thế, Git tiện dụng không những cho các lập trình viên lành nghề, mà còn rất ư là phù hợp cho các bạn sinh viên nữa đấy.

Git quả là một vị cứu tinh của các developer. Các bạn hãy theo dõi  fanpage WebDev Studios để có thể biết thêm những kiến thức bổ ích về Git vào những bài học sau.

**- Tạo project local và đẩy lên github:**

1. Tạo 1 github repo

2. git init

3. Viết code

Git status

Git add

Git commit -m “mes….”

Git remote add origin http://...

Git push origin master

**- Tạo project trên github:**

1. Tạo 1 github repository (folder) – nơi chứa code

2. Clone project (Hay chính là copy project):

Git clone link\_your\_repo

3. Viết code

Git status (Trạng thái các file)

Git add

Git commit -m “mes….”

Git push origin master

**- Những câu lệnh làm việc một mình:**

1. git status

2. git add

3. git commit

4. git push

**- Những câu lệnh làm việc nhóm:**

1. git pull

2. git merge

3. git conflic

4. git checkout branch/commit