**I.TÌM HIỂU VỀ GIT**

**1.Giới thiệu chung**

Git là tên gọi của một hệ thống quản lý phiên bản phân tán (Distributed Version Control System – DVCS) là một trong những hệ thống quản lý phiên bản phổ biến nhất hiện nay. DVCS nghĩa là hệ thống giúp mỗi máy tính có thể lưu trữ nhiều phiên bản khác nhau của một mã nguồn được nhân bản (Clone) từ một kho chứa mã nguồn (Repository), mỗi thay đổi mã nguồn trên máy tính sẽ có thể ủy thác (commit) rồi đưa lên máy chủ nơi đặt kho chứa. Và một máy tính khác (nếu có quyền truy cập) cũng có thể clone lại mã nguồn từ kho chứa hoặc clone lại một tập hợp các thay đổi mới nhất trên máy tính kia.

*Phân biệt Git với Github:*

Git là tên gọi của một mô hình hệ thống còn github chính là một dịch vụ máy chủ repository công cộng, mỗi người có thể tạo tài khoản trên đó để tạo ra các kho chứa của riêng mình để làm việc. Ngoài tính năng nói chung của hệ thống Git, github còn hỗ trợ người dùng những tính năng quan trọng thông qua API sau:

-API to update the repository via HTTP: cho phép người dùng edit file source code trên web browser thông qua HTTP – POST

-API to Access Compare Views: cho phép người dùng review và so sánh code của dự án thông qua việc xem các commit, comments, các dòng khác nhau giữa 2 version của file code… Tính năng này cũng thông qua HTTP – POST, người dùng có thể thực hiện trên web browser.

-Còn nhiều API hữu ích khác mà chúng ta có thể tìm hiểu thêm tại <http://develop.github.com>

Git hỗ trợ 2 giao diện là **GitHub** (giao diện thân thiện với người dùng) và **GitShell** (giao diện code là chủ yếu, với khả năng truy cập sâu vào cơ chế hoạt động của Git hơn).

**2.Cách làm việc của nhóm:**

Khi tạo nhóm , thiết lập và phân quyền tài khoản thì gặp phải vấn đề:

- Nếu muốn phân quyền và dự án private thì phải đóng tiền mua bản quyền của git nên nhóm ko thể làm được.

- Để public và phân quyền tài khoản ngang hàng thì trong quá trình làm việc rất dễ gặp conflict, người này sửa, xóa code của người kia, trùng lặp code…

Chính vì vậy thay vì để public, tạo tài khoản và phân quyền, nhóm chúng tôi đã chọn phương án là chia các branch con để làm việc độc lập nhưng có 1 branch master để đảm bảo việc pull/push là hợp lệ và ít xảy ra conflict nhất và cũng để thuận tiện trong việc sử dụng giao diện GitHub thay cho GitShell

**3.Cài đặt Git**

Chúng ta có thể tải về Git thông qua trang chủ chính thức của Git là “git-scm.com/downloads”. Trong bài báo cáo này, nhóm chúng tôi sử dụng phiên bản Git dành cho hệ điều hành Windows. Khi cài đặt, bộ cài đặt tự động cài cả GitHub lẫn GitShell để tiện cho người sử dụng.

**4.Quy trình làm việc**

Quy trình làm việc là rất quan trọng và cần được rèn luyện nếu người sử dụng muốn thành thạo trong việc sử dụng Git. Quy trình làm việc của Git được biểu diễn bằng sơ đồ sau:

|  |
| --- |
| Add Commit Push  Index  Remote repository  Local repository  Workspace  Fetch    Pull |

Các thành phần trong quy trình

**Workspace**: trạng thái làm việc hiện tại. Các thao tác làm thay đổi mã nguồn, thêm hoặc xóa tài nguyên, … sẽ được lưu trữ ở đây.

**Index**: trạng thái khi các thay đổi đã được lưu lại (bằng các lệnh add)

**Local repository**: sau khi lưu lại, bạn thực hiện thao tác commit, đưa toàn bộ mã nguồn vào local repository. Mỗi commit sẽ có 1 tag (nhãn) riêng để có thể phục hồi lại bất cứ lúc nào.

**Remote repository**: bằng thao tác push, toàn bộ mã nguồn sẽ được đưa lên remote repository và có thể chia sẻ với các thành viên khác thông qua internet.

**Head:** trỏ đến commit gần nhất

Các thao tác

**Khởi tạo git:** Khai báo tên và địa chỉ email để khi commit , git sẽ nhận diện đây là commit của ai, từ đâu .

Cú pháp: git config --global user.name “tên của tài khoản”

git config --global user.email “địa chỉ email của tài khoản”

**Clone**: khi trên remote sever đã có sẵn thư mục và các file mã nguồn, việc cần làm là lấy các thông tin dữ liệu đó về máy tính cá nhân. Lệnh clone sẽ giúp chúng ta thực hiện việc đó.

*Cú pháp: git clone /đường dẫn nơi muốn clone/repository/*

*Ví dụ: git clone https://github.com/KurtLyndon/TTN-TH248*

**Commit**: sau khi thực hiện xong các thay đổi, chúng ta sẽ đưa các thay đổi này vào index và sau đó là Local repository. Trước khi commit chúng ta sẽ sử dụng lệnh add để cập nhật tình trạng các file vào index.

*Cú pháp:*

*git all –all (tự động thêm index tất cả các tập tin thay đổi)*

*git add <tên tập tin> (thêm vào index của tập tin đó)*

*git add \* (thêm vào index tất cả tập tin thay đổi)*

Sau đó thì commit bằng lệnh

*git commit –m “ghi chú commit”=> Head sẽ trỏ đến commit này*

**Push**: sau khi commit, các thay đổi sẽ được lưu trữ tại Local repository, chúng ta sử dụng lệnh push để đưa toàn bộ dữ liệu lên Remote repository.

*Cú pháp: git push origin branch\_name, trong đó branch\_name là tên của nhánh là việc hiện tại, hoặc có thể là master để đẩy trực tiếp lên repository chủ*

**Pull**: là hành động kéo dữ liệu từ trên Remote repository về máy tính cá nhân. Chúng ta có thể kéo toàn bộ dữ liệu hoặc chỉ định một nhánh cụ thể nào đó sau đó tiến hành trộn (merge).

*Cú pháp: git pull <branch\_name>*

**Remote:** Nếu chưa clone một repository hiện có và muốn kết nối repository của hiện tại đến máy chủ remote.

*Cú pháp: git remote add origin <máy-chủ>*

**Fetch**: khi xảy ra lỗi, nếu ta muốn trở lại commit gần nhất để bắt đầu lại công việc, lệnh fetch sẽ giúp ta thực hiện điều đó. Nó sẽ source mới nhất trên sever về và đè lên source hiện tại trên máy local.

*Cú pháp: git fetch <branch\_name>*