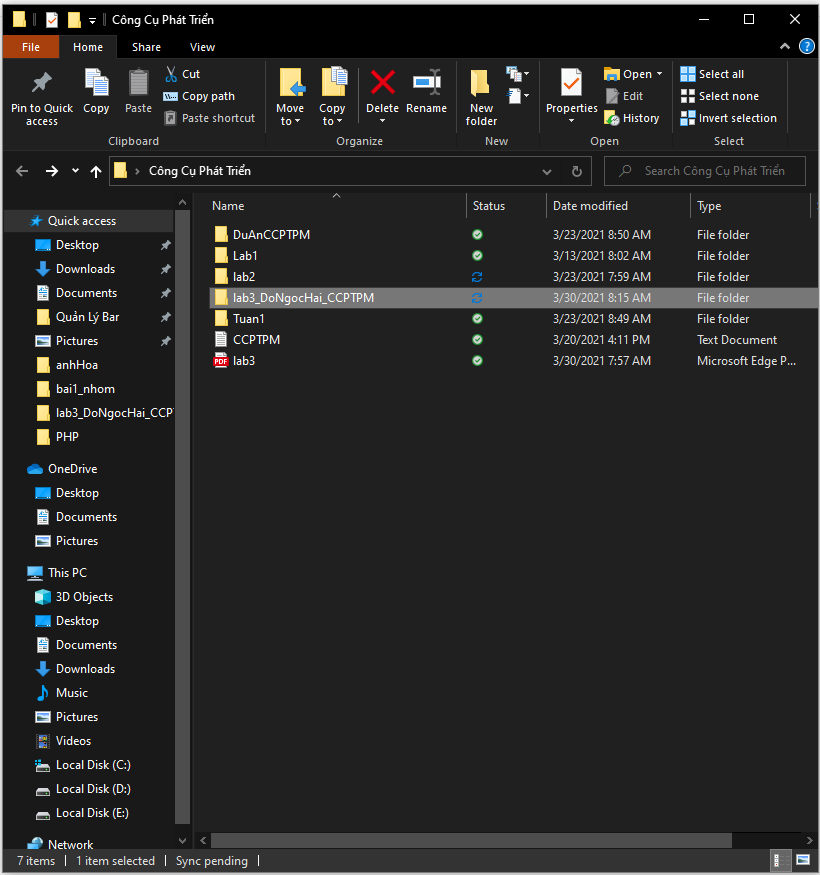
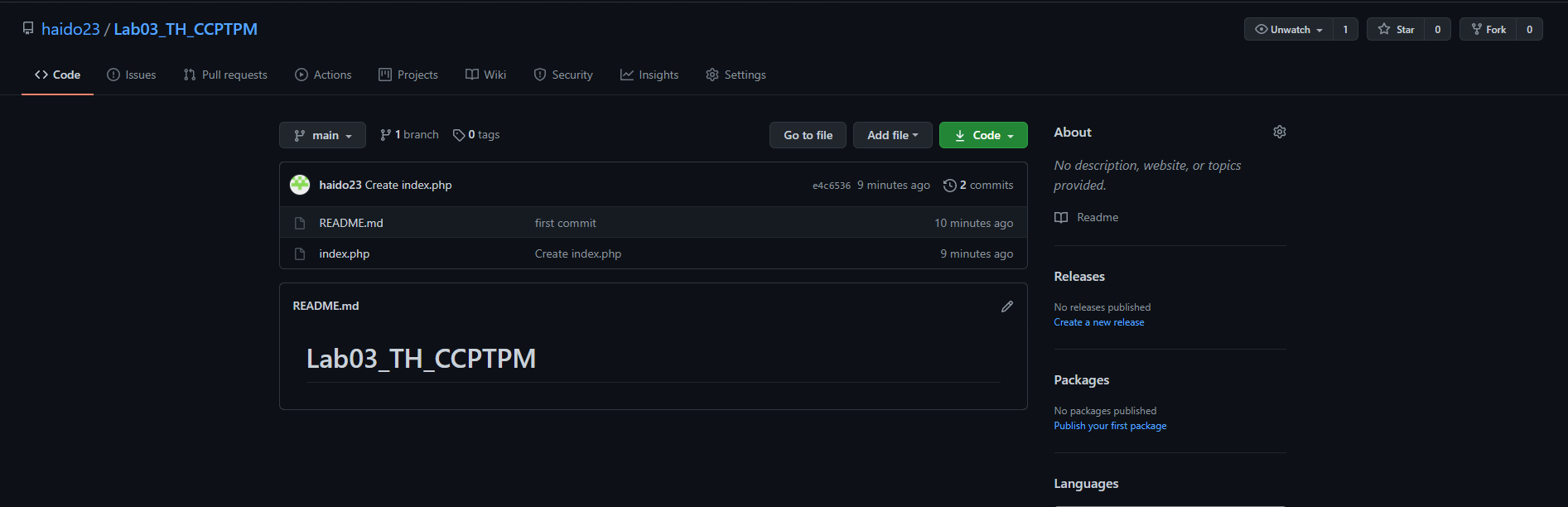
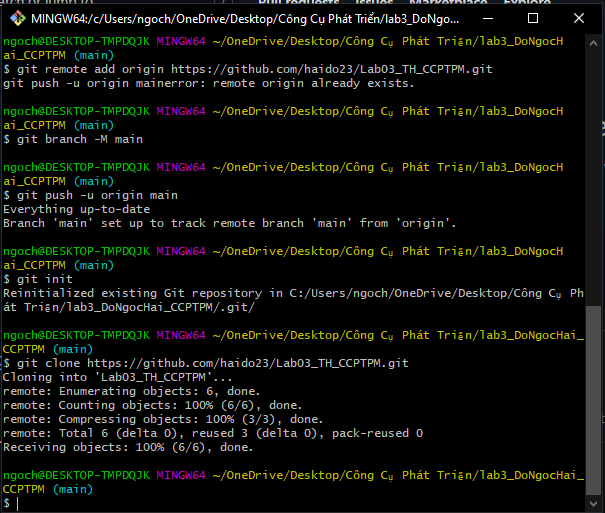
Đỗ Ngọc Hải - 1711062320

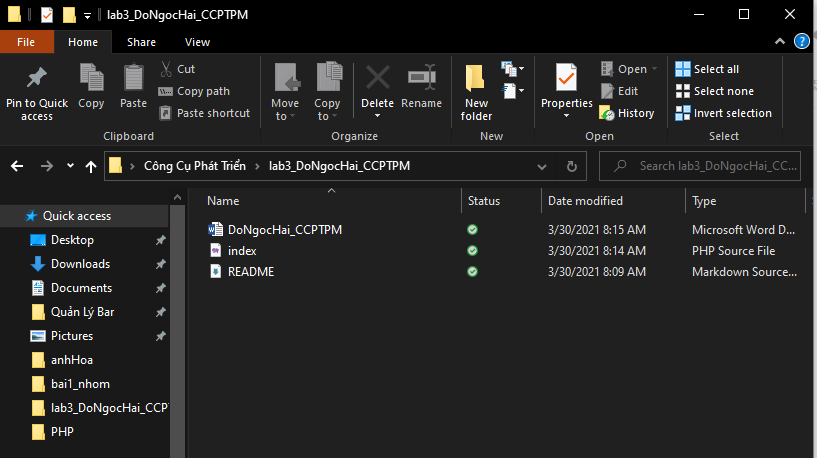
Xử dụng git bash để làm công việc sau và chụp ảnh lại kết quả ở file word (làm trên local và làm cá nhân) - Tạo folder Project trong ổ D bằng câu lệnh



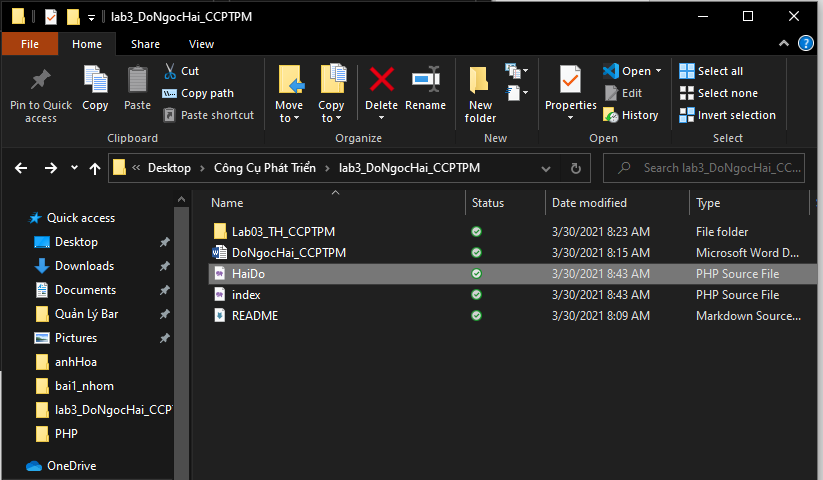
- Cd đến và khởi tạo Git



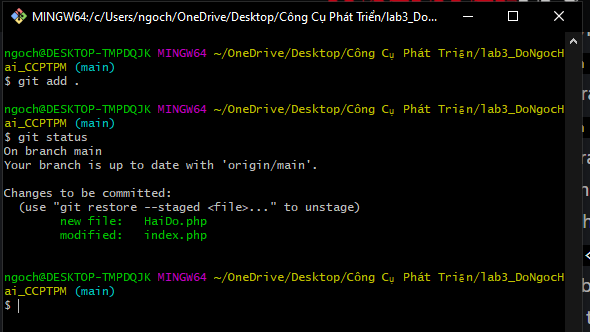




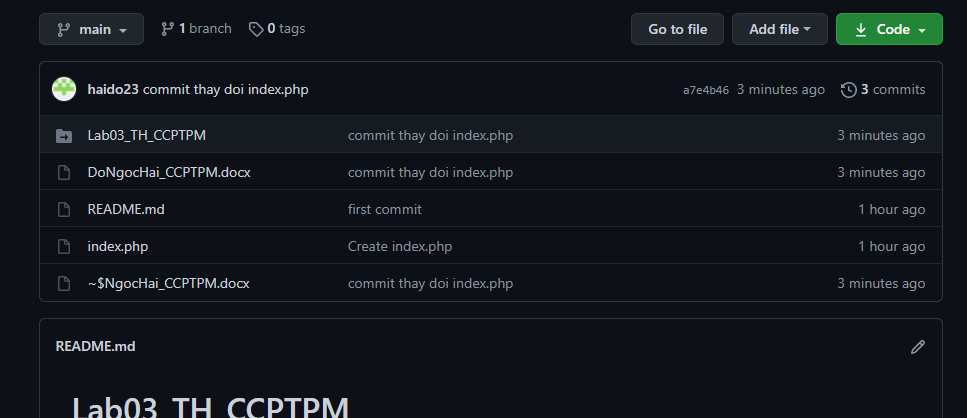
- Thêm một project MVC/ Winform

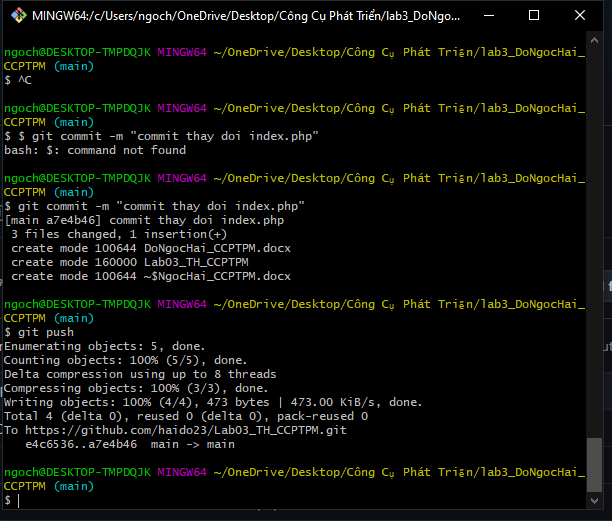


-add project vào staging area

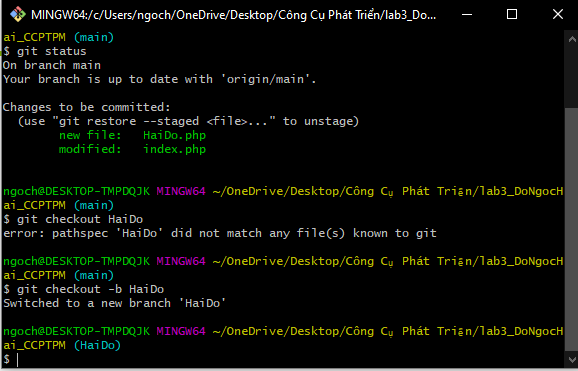


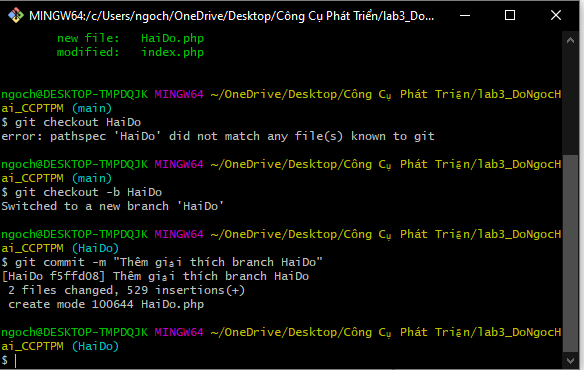
- Commit project lên local repository



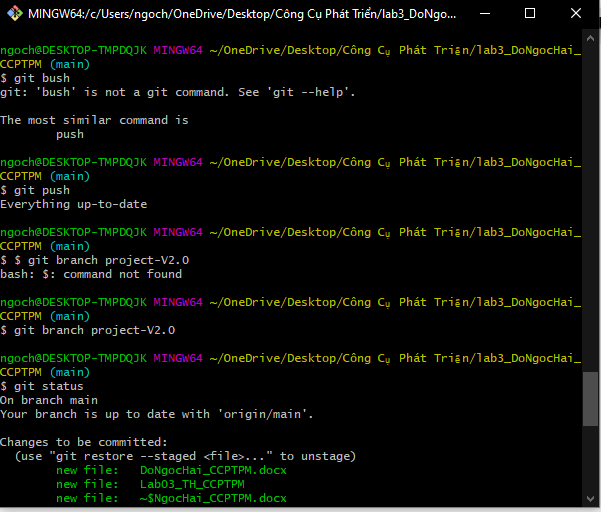


-Checkout project sang thư mục khác





-Tạo một Branch mới đặt tên là project-V2.0



Yêu cầu : Chụp lại kết quả và giải thích hoạt động của các lệnh sau bằng tiếng việt :

**- Init**

Lệnh git init tạo dự án Git mới

Sử dụng lệnh git init để khởi tạo một local repo mới hoặc repo lưu trữ ở server với git init –bare

Lệnh **git init** được sử dưng để tạo, khởi tạo một kho chứa Git mới (Git Repo) ở local. Khi đang trong thư mục dự án chạy lệnh git init nó sẽ tạo ra một thư mục con (ẩn) tên .git, thư mục này chứa tất cả thông tin mô tả cho kho chứa dự án (Repo) mới - những thông tin này gọi là metadata gồm các thư mục như *objects, refs ...*Có một file tên HEAD cũng được tạo ra - nó trỏ đến commit hiện tại.

Lệnh git init nhanh chóng tạo ra quản lý phiên bản của dự án dạng none bare mà bạn không cần có ngay một server để lưu Repo từ xa, không yêu cầu bạn phải nạp file dữ liệu nào. Tất cả phải làm là vào thư mục dự án cần khởi tạo và thi hành lệnh sau để khởi tạo: Git init

Sau lệnh này bạn có một Repo ở local và bắt đầu thực thi được các lệnh khác của Git.

**- Add**

# Lệnh git add lưu thông tin chuẩn bị commit

Thực hiện lệnh tạo snapshot, lưu thông tin thay đổi vào vùng staging để chuẩn bị cho lệnh commit

Lệnh git add sử dụng để **đánh chỉ mục (index)** các nội dung mới, mới cập nhật trong **thư mục làm việc**, nó chuẩn bị nội dung sắp xếp cho lần commit tiếp theo.

Khái niệm **đánh chỉ mục** ở trên có nghĩa là lưu lại **ảnh chụp** (snapshot) thông tin thay đổi của thư mục làm việc so với lần commit trước (hoặc so với snapshot chưa commit), snapshot lưu ở khu vực gọi là **staging** (sắp xếp, chuẩn bị)

Bạn có thể thực hiện lệnh git add nhiều lần để tạo tạo ra một snapshot cuối cùng trước khi thực hiện commit.

**- Commit**

# Sử dụng lệnh git commit

Thực hiện câu lệnh git commit để lưu dữ liệu vào hệ thống Git, tùy chọn các tham số như tự động staging, cập nhật vào commit cuối

Lệnh git commit thực hiện lưu vào CSDL Git toàn bộ nội dung chứa trong index (vùng staging) và kèm theo nó là một đoạn text thông tin (log) mô tả sự thay đổi của của commit này so với commit trước. Sau khi commit con trỏ HEAD tự động dịch chuyển đến commit này (Trong nhánh hiện tại)

**- Pull**

Lệnh git pull lấy về thông tin từ remote và cập nhật vào các nhánh của local repo.

**- Push**

Lệnh git push đẩy dữ liệu từ local lên server

Thực hiện lệnh git push để tạo upstream, đẩy dữ liệu từ local repo đến remote repo, push các nhánh, xóa các nhánh của remote

Lệnh git push được sử dụng để đẩy các commit mới ở máy trạm (local repo) lên server (remote repo). Nguồn để đẩy lên là nhánh mà con trỏ HEAD đang trỏ tới (nhánh làm việc).

Đích mà nó đẩy lên (ghi vào nhánh nào) có thể chỉ định trong tùy chọn của lệnh, tuy nhiên cũng không cần chỉ định nếu có thiết lập liên hệ giữa nguồn và đích trước. Lệnh git push cũng có thể xóa đi các nhánh của remote

Một số tham số hay dùng như:

* --all đẩy tất cả các nhánh lên server
* --tags đẩy tất cả các tag lên server
* --delete xóa một nhánh chỉ ra trên server
* -u đẩy và tạo một upstream (luồng upload tương ứng với nhánh của local), hay sử dụng cho lần đầu đẩy lên server

- **Branch**

Branch là cái dùng để phân nhánh và ghi lại luồng của lịch sử. Branch đã phân nhánh sẽ không ảnh hưởng đến branch khác nên có thể tiến hành nhiều thay đổi đồng thời trong cùng 1 repository.

**- Checkout**

Lệnh git checkout

Lệnh git checkout được dùng để chuyển nhánh hoặc để phục hồi file trong thư mục làm việc từ một commit trước đây ...

Từ phiên bản git 2.23 còn có thể 2 lệnh với chức năng tương ứng là: git switch và git restore

**-Status**

# Lệnh git status xem trạng thái của Repo

Sử dụng lệnh git để biết thông tin trạng thái sửa đổi, thêm mới, xóa các file trước khi thực hiện commit

## Lệnh git status

Lệnh git status hiện thị thông tin khác nhau (do thêm mới, xóa đi, sửa đổi các file) giữa các file trong các trường hợp:

1. Khác nhau giữa các file trong vùng staging (chỉ mục) và commit tại con trỏ HEAD (Thường HEAD ở vị trí commit cuối):

thông tin này bạn có thể thực hiện lệnh commit để lưu staging vào dữ liệu Git

1. Khác nhau giữa các file trong thư mục làm việc và trong staging

bạn có thể chạy git add rồi commit

1. Khác nhau giữa thư mục làm việc và những file chưa được giám sát bởi Git

bạn có thể chạy git add rồi commit

- **Log**

# Sử dụng lệnh git log xem lại lịch sử các commit trên git

Tìm kiếm, xem lại, lọc các thông tin log lịch sử commit trên git bằng lệnh git log với các thiết lập khác nhau, như pretty, hiện thị đồ thị log

## Lệnh git log

Lệnh git log giúp bạn xem lại thông tin lịch sự commit, nhằm giám sát sự thay đổi của dự án. Lệnh git log có nhiều tham số để xuất ra, định dạng các thông tin hiện thị theo cách mong muốn. Bạn có thể định dạng cách các thông tin mỗi commit được in ra khi xem, cũng như có thể lọc thông tin nào đó.

Mặc đinh thi hành git log nó liệt kê các commit theo thứ tự từ mới nhất đến cũ nhất, mỗi commit có các thông tin gồm: mã hash của commit, dòng thông báo, người tạo commit và ngày tạo commit : git log

**- Merge**

## Lệnh git merge

Lệnh git merge sử dụng để gộp nhánh, gộp nhánh này vào nhánh khác. Khi gộp nhánh git thường căn cứ vào 3 commit, để tạo ra một commit gộp, nếu có xung đột cần xử lý .

**- Rebase**

## Lệnh git rebase

Lệnh git rebase cũng gộp các commit từ nhánh này vào nhánh khác, bằng cách xây dựng lại các commit base kế thừa từ nhánh khác và viết lại lịch sử commit sau các commit cơ sở mới.