

**ĐẠI HỌC CÔNG NGHỆ GIAO THÔNG VẬN TẢI**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO BÀI TẬP**

**QUY TRÌNH CÔNG CỤ PHÁT TRIỂN PHẦN MỀM**

**ĐỀ TÀI : TÌM HIỀU CÔNG CỤ PHÁT TRIỂN PHẦN MỀM ỨNG DỤNG VÀO PROJECT**

**GIẢNG VIÊN HƯỚNG DẪN: Ths. NGUYỄN VĂN CƯỜNG**

**NHÓM SINH VIÊN SỐ 5: NGUYỄN PHÚ LONG 73DCTT23320**

**HOÀNG MẠNH HOÀN 73DCTT22265**

**NGUYỄN VĂN CHIẾN 73DCTT22223**

**ĐINH TUẤN AN 73DCTT23376**

**KIỀU MINH DŨNG 73DCTT22222**

**LỚP: 73DCTT21**

**Hà Nội, 2025**

**BẢNG PHÂN CÔNG NHIỆM VỤ**

|  |  |
| --- | --- |
| **Họ tên** | **Công việc** |
| Nguyễn Phú Long | Tìm hiểu nội dung |
| Nguyễn Văn Chiến | Tìm hiểu nội dung |
| Hoàng Mạnh Hoàn | Làm báo cáo |
| Đinh Tuấn An | Tìm hiểu nội dung |
| Kiều Minh Dũng | Tìm hiểu nội dung |

**MỤC LỤC**

[LỜI NÓI ĐẦU 5](#_Toc195386458)

[CHƯƠNG 1: TỔNG QUAN VỀ 1 SỐ PHẦN MỀM SỬ DỤNG ĐỂ QUẢN LÝ MÃ NGUỒN. 6](#_Toc195386459)

[1.1. Tổng quan về Git. 6](#_Toc195386460)

[1.1.1 Khái niệm về Git. 6](#_Toc195386461)

[1.1.2. Lý do việc sử dụng Git để kiểm soát mã nguồn. 6](#_Toc195386462)

[1.1.3. Lợi ích của việc sử dụng Git. 6](#_Toc195386463)

[1.1.4. Các Git hoạt động. 6](#_Toc195386464)

[1.1.5. Một số lệnh git cơ bản. 6](#_Toc195386465)

[1.2. Tổng quan về SVN. 7](#_Toc195386466)

[1.2.1. Khái niệm về SVN. 7](#_Toc195386467)

[1.2.2. Các thành phần của SVN. 7](#_Toc195386468)

[1.2.3. Lợi ích của SVN. 8](#_Toc195386469)

[1.2.4. Cách SVN hoạt động. 8](#_Toc195386470)

[1.2.5. Một số lệnh SVN. 9](#_Toc195386471)

[1.3. Tổng quan về SSH, FTP, SFTP. 9](#_Toc195386472)

[1.3.1. Tìm hiểu SSH. 9](#_Toc195386473)

[1.3.2. Tìm hiểu về SFTP. 10](#_Toc195386474)

[1.3.3. SFTP (SSH File Transfer Protocol). 10](#_Toc195386475)

[CHƯƠNG 2. ÁP DỤNG CÁC PHẦN MỀM ĐỂ THỰC HIỆN QUẢN LÝ MÃ NGUỒN. 12](#_Toc195386476)

[2.1. Tổng quan về project. 12](#_Toc195386477)

[2.1.1. Giới thiệu chung về project. 12](#_Toc195386478)

[2.1.2. Mục tiêu dự án. 12](#_Toc195386479)

[2.2.3. Công cụ và công nghệ sử dụng. 12](#_Toc195386480)

[2.2. Build Project trên VM remote access. 13](#_Toc195386481)

[2.3 Sử dụng công cụ Git để quản lý mã nguồn. 16](#_Toc195386482)

[2.4. Cấu hình truy cập ssh, ftp, sftp. 18](#_Toc195386483)

[2.4.1. Cấu hình ssh. 18](#_Toc195386484)

[2.4.2 Cấu hình ftp. 18](#_Toc195386485)

[2.4.2. Cấu hình sftp. 19](#_Toc195386486)

[KẾT LUẬN 22](#_Toc195386487)

# LỜI NÓI ĐẦU

Trong thời đại công nghệ phát triển không ngừng, việc xây dựng và triển khai các hệ thống web/app không chỉ dừng lại ở lập trình mà còn bao gồm việc thiết lập môi trường phát triển, quản lý mã nguồn hiệu quả và đảm bảo khả năng truy cập từ nhiều thiết bị. Đặc biệt, trong môi trường học tập và làm việc chuyên nghiệp, kỹ năng triển khai dự án trên máy ảo (Virtual Machine – VM), truy cập từ xa và sử dụng các công cụ hỗ trợ như Git, SVN, SSH… đóng vai trò rất quan trọng.

Báo cáo này được thực hiện nhằm mục tiêu tìm hiểu, xây dựng và triển khai một project web/app thực tế trên máy ảo, có khả năng truy cập từ máy tính khác trong cùng mạng nội bộ thông qua địa chỉ IP thật (192.168.x.x), thay vì chỉ sử dụng localhost.

Bên cạnh việc xây dựng hệ thống, báo cáo cũng trình bày quá trình sử dụng hệ thống quản lý mã nguồn như Git hoặc SVN để quản lý, đồng bộ và bảo trì mã nguồn một cách hiệu quả. Việc ứng dụng các công cụ phát triển hiện đại giúp sinh viên không chỉ nâng cao kỹ năng kỹ thuật mà còn hiểu rõ quy trình triển khai một dự án thực tế, từ môi trường phát triển đến triển khai và quản lý.

Báo cáo gồm hai phần chính: phần lý thuyết giới thiệu và giải thích các công cụ được sử dụng như Git, SSH, Virtual Machine,… và phần thực tế mô tả quá trình triển khai project, công cụ đã sử dụng, môi trường vận hành (Web Server, hệ điều hành, cơ sở dữ liệu,…), từ đó rút ra những kinh nghiệm thực tiễn trong việc phát triển và triển khai hệ thống phần mềm.

# CHƯƠNG 1: TỔNG QUAN VỀ 1 SỐ PHẦN MỀM SỬ DỤNG ĐỂ QUẢN LÝ MÃ NGUỒN.

## 1.1. Tổng quan về Git.

### 1.1.1 Khái niệm về Git.

Git là một hệ thống kiểm soát phiên bản phân tán để quản lý mã nguồn. Hệ thống kiểm soát phiên bản ghi lại và lưu các thay đổi khi bạn sửa đổi tệp, cho phép bạn khôi phục phiên bản trước đó của công việc mình làm bất kỳ lúc nào.

### 1.1.2. Lý do việc sử dụng Git để kiểm soát mã nguồn.

Nếu không có hệ thống kiểm soát phiên bản, lập trình viên thường lưu nhiều bản sao của tệp bằng cách đặt tên theo ngày hoặc số phiên bản (ví dụ: code\_02-12-2022.php, code\_03-12-2022.php). Cách làm này **không hiệu quả**, dễ gây nhầm lẫn và khó quản lý, đặc biệt khi làm việc nhóm.

Git giải quyết vấn đề này bằng cách:

* Ghi lại mọi thay đổi trên mã nguồn.
* Cho phép khôi phục phiên bản cũ một cách dễ dàng.
* Giúp nhiều người cùng làm việc mà không bị ghi đè mất dữ liệu.
* Hỗ trợ làm việc trên nhiều nhánh (branch) để phát triển các tính năng độc lập.

### 1.1.3. Lợi ích của việc sử dụng Git.

* Theo dõi lịch sử thay đổi mã nguồn.
* Làm việc nhóm hiệu quả, tránh xung đột dữ liệu.
* Hỗ trợ phân nhánh giúp phát triển linh hoạt.
* Hoàn toàn miễn phí và được sử dụng rộng rãi trong cộng đồng lập trình viên.

### 1.1.4. Các Git hoạt động.

Git hoạt động theo mô hình **phân tán**, trong đó mỗi lập trình viên có một bản sao đầy đủ của kho lưu trữ. Các thành phần chính trong Git bao gồm:

* **Repository (Kho lưu trữ)**: Chứa toàn bộ mã nguồn và lịch sử thay đổi.
* **Commit**: Ghi lại thay đổi tại một thời điểm cụ thể.
* **Branch (Nhánh làm việc)**: Hỗ trợ phát triển tính năng mới một cách độc lập.
* **Merge**: Kết hợp các thay đổi từ nhánh này vào nhánh khác.
* **Staging Area**: Khu vực tạm trước khi commit, giúp kiểm soát thay đổi

### 1.1.5. Một số lệnh git cơ bản.

|  |  |
| --- | --- |
| Lệnh Git | Chức năng |
| git init | Khởi tạo một Git repository mới trong thư mục hiện tại |
| git clone <url> | Sao chép một repository từ server về máy cục bộ |
| git status | Kiểm tra trạng thái hiện tại của các file |
| git add <file> | Thêm file vào khu vực staging để chuẩn bị commit |
| git add . | Thêm tất cả các file đã thay đổi vào staging |
| git commit -m "msg" | Tạo một bản ghi commit với nội dung mô tả |
| git log | Hiển thị lịch sử các commit |
| git branch | Liệt kê các nhánh hoặc tạo nhánh mới |
| git checkout <branch> | Chuyển sang nhánh khác |
| git merge <branch> | Gộp nhánh được chỉ định vào nhánh hiện tại |
| git pull | Tải và gộp các thay đổi từ remote repository |
| git push | Đẩy các thay đổi từ máy cục bộ lên remote repository |
| git remote -v | Hiển thị danh sách các remote hiện tại |
| git reset --hard <hash> | Quay lại trạng thái commit cụ thể (xóa luôn thay đổi sau đó) |
| git rm <file> | Xóa file khỏi Git và hệ thống |

## **1.2. Tổng quan về SVN.**

### 1.2.1. Khái niệm về SVN.

**SVN (Subversion)** là một phần mềm mã nguồn mở có khả năng kiểm soát và quản lý các phiên bản mã nguồn khác trong quá trình phát triển phần mềm. Có thể hiểu rằng, nó giúp người dùng có thể thực hiện các thao tác chỉnh sửa tệp và thư mục như những công cụ khác. Ngoài ra, bạn có thể xem lại những thay đổi và truy xuất các phiên bản trước của thông tin dữ liệu.

### 1.2.2. Các thành phần của SVN.

SVN (Subversion) bao gồm nhiều thành phần quan trọng giúp quản lý phiên bản mã nguồn hiệu quả. SVN là công cụ chính, cung cấp giao diện dòng lệnh để lập trình viên tương tác với kho lưu trữ. svnadmin là công cụ quản trị giúp tạo, sửa chữa và quản lý repository, trong khi svnserve hoạt động như một máy chủ độc lập, cho phép máy khách kết nối từ xa để truy cập và quản lý mã nguồn.

Ngoài ra, mod\_dav\_svn là một module giúp tích hợp SVN với máy chủ web Apache, hỗ trợ truy cập repository thông qua giao thức HTTP/HTTPS. Repository (kho lưu trữ) là nơi chứa toàn bộ mã nguồn và lịch sử thay đổi của dự án, còn working copy (bản sao làm việc) là phiên bản mã nguồn mà lập trình viên tải về máy để chỉnh sửa.

Lập trình viên có thể sử dụng client SVN như TortoiseSVN hoặc dòng lệnh SVN để thao tác với mã nguồn. Bên cạnh đó, SVN còn hỗ trợ hooks (móc nối) – các tập lệnh tự động thực thi khi có sự kiện như commit, update trong repository. Tất cả các thành phần này phối hợp với nhau để đảm bảo quá trình quản lý phiên bản diễn ra hiệu quả, đồng bộ và an toàn.

### 1.2.3. Lợi ích của SVN.

**Quản lý tập trung:** Dữ liệu được lưu trữ trên một máy chủ duy nhất, giúp dễ kiểm soát quyền truy cập và bảo mật mã nguồn.

**Theo dõi lịch sử thay đổi:** Mỗi commit ghi lại đầy đủ thông tin, cho phép theo dõi quá trình phát triển và khôi phục phiên bản trước dễ dàng.

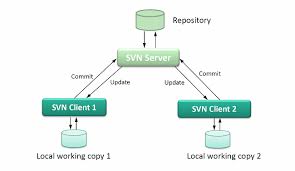
**Xử lý tệp tin lớn:** SVN được thiết kế để quản lý hiệu quả các tệp có kích thước lớn.

**Đồng bộ hóa nhóm:** Tạo điều kiện thuận lợi cho việc giám sát và quản lý các hoạt động của nhóm phát triển.

**Đáng tin cậy:** Phù hợp cho các tổ chức cần quản trị chặt chẽ và bảo mật cao.

### 1.2.4. Cách SVN hoạt động.

SVN hoạt động theo mô hình tập trung, nghĩa là toàn bộ mã nguồn và lịch sử thay đổi được lưu trữ trên một kho lưu trữ trung tâm. Khi làm việc với SVN, lập trình viên sẽ lấy bản sao (working copy) của kho lưu trữ về máy tính cá nhân, thực hiện thay đổi trên đó và sau đó commit các thay đổi trở lại kho lưu trữ trung tâm. Các thành phần chính của SVN bao gồm:



**Repository (Kho lưu trữ):** Nơi chứa toàn bộ mã nguồn và lịch sử thay đổi của dự án, được lưu trữ trên máy chủ trung tâm.

**Working Copy (Bản sao làm việc):** Bản sao của repository được lập trình viên tải về để chỉnh sửa và thử nghiệm.

**Checkout (Lấy bản sao):** Quá trình sao chép repository từ máy chủ về máy tính cá nhân để bắt đầu làm việc.

**Commit (Gửi thay đổi):** Ghi lại và gửi các thay đổi từ working copy lên repository, tạo ra phiên bản mới của dự án.

**Update (Cập nhật):** Quá trình đồng bộ working copy với các thay đổi mới nhất từ repository, đảm bảo rằng mọi thành viên luôn có phiên bản cập nhật.

**Merge (Hợp nhất):** Kết hợp các thay đổi từ các working copy khác nhau vào repository, giúp đồng bộ hóa công việc giữa các lập trình viên.

**Branch (Nhánh) và Tag (Đánh dấu**): Hỗ trợ tạo nhánh để phát triển tính năng mới mà không ảnh hưởng đến mã nguồn chính và đánh dấu các phiên bản quan trọng của dự án.

### 1.2.5. Một số lệnh SVN.

|  |  |
| --- | --- |
| Lệnh SVN | Chức năng |
| svn checkout <url> | Tải xuống repository từ server về máy cục bộ |
| svn update | Cập nhật các thay đổi từ repository về máy cục bộ |
| svn add <file> | Thêm file mới vào version control |
| svn commit -m "msg" | Gửi các thay đổi đã staging lên repository với nội dung mô tả |
| svn delete <file> | Xóa file khỏi version control |
| svn status | Kiểm tra trạng thái file (đã thay đổi, thêm mới, xóa...) |
| svn log | Xem lịch sử các lần commit |
| svn revert <file> | Khôi phục lại file về trạng thái trước khi chỉnh sửa |
| svn info | Hiển thị thông tin chi tiết của file hoặc thư mục |

## 1.3. Tổng quan về SSH, FTP, SFTP.

### 1.3.1. Tìm hiểu SSH.

a. Khái niệm.

SSH là một giao thức mạng dùng để kết nối đến máy tính từ xa thông qua dòng lệnh, cho phép người dùng thực hiện các thao tác quản trị hệ thống một cách an toàn nhờ vào việc **mã hóa toàn bộ quá trình truyền dữ liệu**.

b. Chức năng chính.

* Đăng nhập từ xa vào máy chủ.
* Thực thi lệnh và quản lý hệ thống từ xa.
* Chuyển file an toàn thông qua scp hoặc sftp.

c. Ưu điểm.

* Bảo mật cao, dữ liệu truyền đi được mã hóa.
* Hỗ trợ xác thực bằng mật khẩu hoặc khóa riêng (SSH key).
* Có thể chuyển file thông qua các công cụ đi kèm như scp, sftp.

d. Nhược điểm.

* Không có giao diện đồ họa (giao tiếp qua dòng lệnh).
* Yêu cầu kiến thức kỹ thuật để sử dụng hiệu quả.
* Bị giới h
* ạn truy cập nếu tường lửa chặn cổng 22.

e. Cách sử dụng.

ssh username@hostname

VD: ssh admin@192.168.1.100

### 1.3.2. Tìm hiểu về SFTP.

a. Khái niệm.

FTP là giao thức truyền tệp cổ điển giữa client và server thông qua kết nối TCP/IP. Đây là phương pháp truyền file phổ biến từ những năm đầu của Internet.

b. Chức năng chính.

* Tải lên và tải xuống file.
* Tạo, đổi tên, xóa thư mục/tệp từ xa.

c. Ưu điểm.

* Dễ sử dụng, có giao diện đồ họa thân thiện.
* Phổ biến và được hỗ trợ rộng rãi.
* Tốc độ truyền tải nhanh trong mạng nội bộ.

d. Nhược điểm.

* Không mã hóa thông tin (user, mật khẩu, dữ liệu đều truyền dưới dạng văn bản thuần).
* Dễ bị đánh cắp thông tin nếu sử dụng qua Internet.
* Cần cấu hình thêm để sử dụng qua mạng an toàn (FTPS).

e. Cách sử dụng.

Trình duyệt web: <ftp://yourserver.com>

### 1.3.3. SFTP (SSH File Transfer Protocol).

a. Khái niệm.

SFTP là giao thức truyền tệp an toàn, hoạt động trên nền tảng SSH. Mặc dù tên gọi tương tự FTP, nhưng nó hoàn toàn khác biệt và không sử dụng cổng FTP.

b. Chức năng chính.

* Truyền file an toàn từ client đến server và ngược lại.
* Tạo/xóa/sửa đổi file và thư mục.
* Kiểm soát truy cập thông qua SSH.

c. Ưu điểm.

* Bảo mật cao, sử dụng cùng nền tảng với SSH.
* Tất cả dữ liệu, kể cả mật khẩu, đều được mã hóa.
* Chỉ sử dụng một cổng duy nhất (22), dễ cấu hình trên tường lửa.

d. Nhược điểm.

* Tốc độ truyền tải có thể chậm hơn FTP do việc mã hóa dữ liệu.
* Một số phần mềm cũ không hỗ trợ SFTP.
* Yêu cầu SSH được kích hoạt và cấu hình đúng.

e. Cách sử dụng.

ssh username@hostname

VD: ssh admin@192.168.1.100

# CHƯƠNG 2. ÁP DỤNG CÁC PHẦN MỀM ĐỂ THỰC HIỆN QUẢN LÝ MÃ NGUỒN.

## 2.1. Tổng quan về project.

### 2.1.1. Giới thiệu chung về project.

Trong bối cảnh giáo dục ngày càng phát triển, việc quản lý thông tin sinh viên một cách hiệu quả, nhanh chóng và chính xác đóng vai trò then chốt trong công tác đào tạo. Phần mềm quản lý sinh viên được xây dựng nhằm đáp ứng nhu cầu này, cung cấp một hệ thống toàn diện giúp các đơn vị đào tạo theo dõi và quản lý thông tin sinh viên một cách tối ưu. Hệ thống này tích hợp các chức năng cơ bản và thiết yếu như thêm mới, chỉnh sửa, xóa, tìm kiếm thông tin sinh viên, quản lý lớp học, quản lý điểm số và nhiều chức năng hỗ trợ khác, góp phần nâng cao hiệu quả quản lý và giảm thiểu các tác vụ thủ công.

### 2.1.2. Mục tiêu dự án.

* Dự án này hướng đến việc xây dựng một hệ thống quản lý sinh viên đơn giản nhưng hiệu quả, dựa trên mô hình Client - Server phổ biến. Các mục tiêu cụ thể bao gồm:
* Xây dựng một hệ thống quản lý sinh viên có kiến trúc Client - Server rõ ràng và ổn định.
* Cung cấp các chức năng cơ bản và thiết yếu cho việc quản lý thông tin sinh viên, bao gồm: thêm mới, chỉnh sửa, xóa và tìm kiếm sinh viên .
* Đảm bảo hệ thống có khả năng triển khai dễ dàng trên môi trường mạng nội bộ (LAN) hoặc trên môi trường Internet, tạo sự linh hoạt trong việc sử dụng.

### 2.2.3. Công cụ và công nghệ sử dụng.

Để đạt được các mục tiêu đề ra, dự án đã sử dụng các công cụ và công nghệ sau

a. Giao diện người dùng (User Interface - UI).

Ngôn ngữ: HTML, CSS, JavaScript - Các ngôn ngữ nền tảng cho việc xây dựng giao diện web, đảm bảo tính tương tác và hiển thị thông tin.

b. Web Server.

* Hệ điều hành máy chủ: Ubuntu

Ubuntu 22.04 LTS được sử dụng làm hệ điều hành máy chủ vì tính ổn định, mã nguồn mở, dễ sử dụng và cộng đồng hỗ trợ mạnh mẽ.

Máy chủ được cấu hình trực tiếp bằng Terminal, thuận tiện cho việc cài đặt và quản lý dịch vụ.

- Apache Web Server:

Apache HTTP Server (thường gọi là Apache) là một phần mềm máy chủ web mã nguồn mở phổ biến nhất trên thế giới, được phát triển bởi Apache Software Foundation. Nó có nhiệm vụ xử lý và phản hồi các yêu cầu HTTP từ trình duyệt người dùng, và trả về nội dung web tương ứng (HTML, PHP, CSS, JS…).

c. Cơ sở dữ liệu.

MySQL (MariaDB): Một hệ quản trị cơ sở dữ liệu mã nguồn mở mạnh mẽ và được sử dụng rộng rãi. MySQL (hoặc MariaDB, một nhánh phát triển của MySQL) được lựa chọn để lưu trữ toàn bộ thông tin của hệ thống, bao gồm thông tin sinh viên, thông tin lớp học, điểm số và các dữ liệu liên quan khác một cách có cấu trúc và an toàn.

d. Backend – Xử lý logic.

Ngôn ngữ lập trình PHP đóng vai trò trung tâm trong việc xử lý các logic nghiệp vụ của hệ thống. PHP là một ngôn ngữ kịch bản mã nguồn mở chạy phía máy chủ (server-side), đặc biệt phù hợp cho việc xây dựng các ứng dụng web động. Với cú pháp đơn giản, dễ học và dễ triển khai, PHP cho phép các lập trình viên dễ dàng kết nối và tương tác với cơ sở dữ liệu MySQL để thực hiện các thao tác như thêm mới, chỉnh sửa, xóa, và truy vấn thông tin sinh viên. PHP tiếp nhận dữ liệu từ giao diện người dùng qua các biểu mẫu HTML, sau đó xử lý và thực hiện truy vấn đến cơ sở dữ liệu, đồng thời trả về kết quả phù hợp cho người dùng. Ngoài ra, PHP còn hỗ trợ thiết lập các cơ chế bảo mật cơ bản như kiểm tra đầu vào, ngăn chặn SQL Injection và XSS, góp phần bảo vệ hệ thống.

e. Môi trường phát triển.

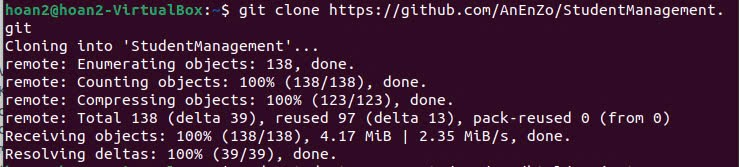
Hệ điều hành: Ubuntu 22.04 LTS - Một Linux ổn định và phổ biến, cung cấp một môi trường phát triển mạnh mẽ và thân thiện với các công cụ và công nghệ được sử dụng trong dự án.

Công cụ:

* VS Code (Visual Studio Code): Một trình soạn thảo mã nguồn mạnh mẽ, miễn phí và được tích hợp nhiều tính năng hỗ trợ lập trình hiệu quả.
* phpMyAdmin: Một công cụ web-based miễn phí, được viết bằng PHP, dùng để quản trị cơ sở dữ liệu MySQL một cách trực quan thông qua giao diện đồ họa.
* Terminal: Giao diện dòng lệnh trên Ubuntu, được sử dụng để thực hiện các thao tác hệ thống, quản lý server và chạy các lệnh cần thiết trong quá trình phát triển.

## 2.2. Build Project trên VM remote access.

Bước 1: Tải bản mã nguồn về máy ảo.



Bước 2: Chuyển vào thư mục để cấu hình.

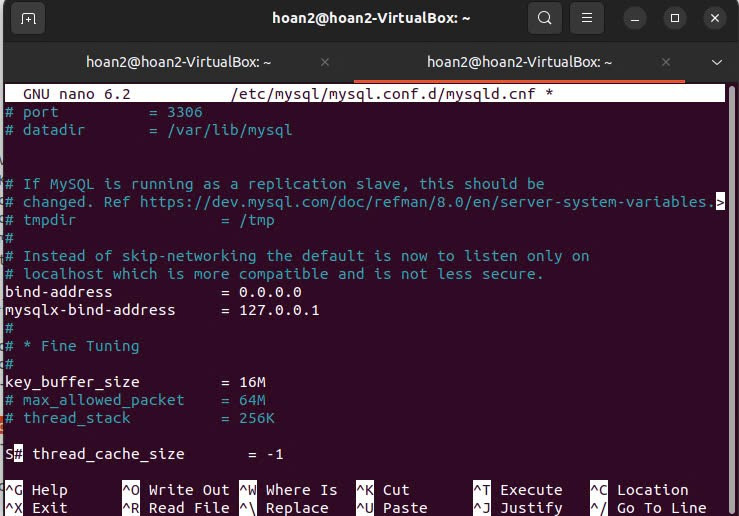


Bước 3: Cài mysql-server trên máy ảo.

sudo apt install mysql-server

Bước 4: Cấu hình cho mysql cho phép mysql-serve truy cập ngoài localhost bằng cách sửa bind-address = 127.0.0.1 thành bind-address = 0.0.0.0

sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf



Bước 5: Tạo user mysql truy cập từ xa.

CREATE USER 'hoanls12'@'%' IDENTIFIED BY '123456';

GRANT ALL PRIVILEGES ON \*.\* TO 'tenuser'@'%' WITH GRANT OPTION;

FLUSH PRIVILEGES;

EXIT;

Bước 6: - Cài phpmyadmin.

sudo apt install phpmyadmin

- Kích hoạt phpMyAdmin trong Apache:

sudo ln -s /etc/phpmyadmin/apache.conf /etc/apache2/conf-available/phpmyadmin.conf

sudo a2enconf phpmyadmin

sudo a2enconf phpmyadmin

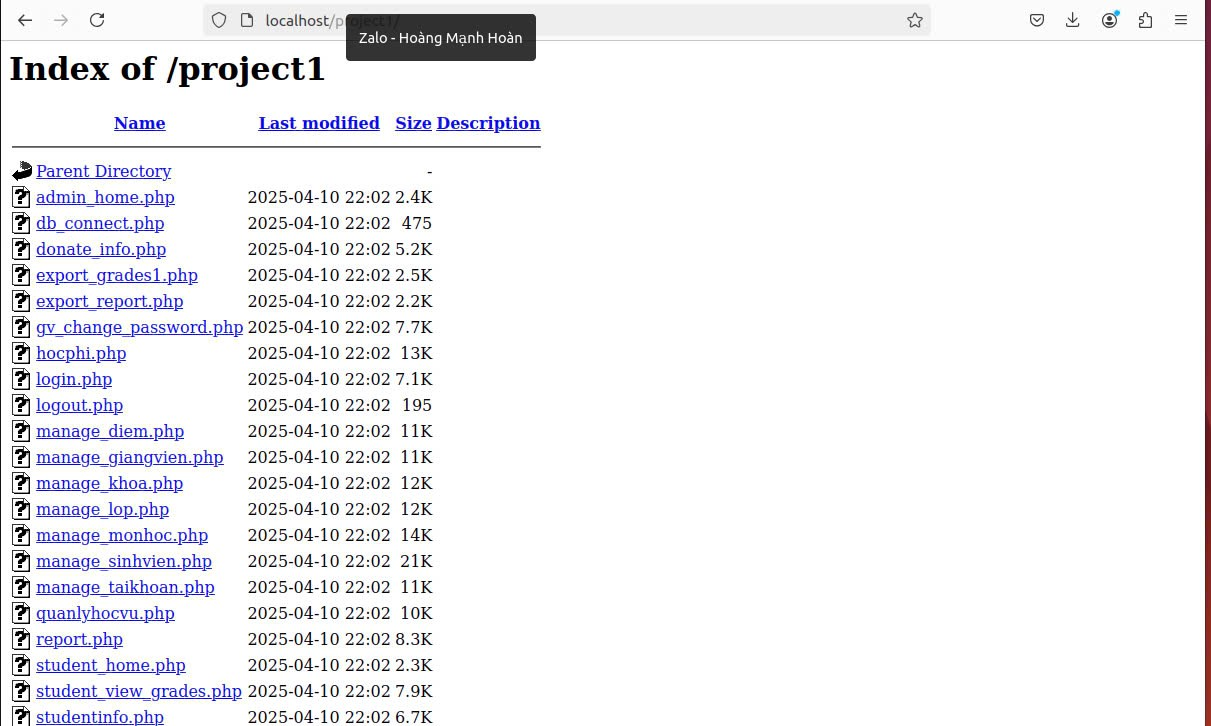
Bước 7: Tạo cơ sở dữ liệu.

Bước 8: Cấp quyền thư mục chạy.

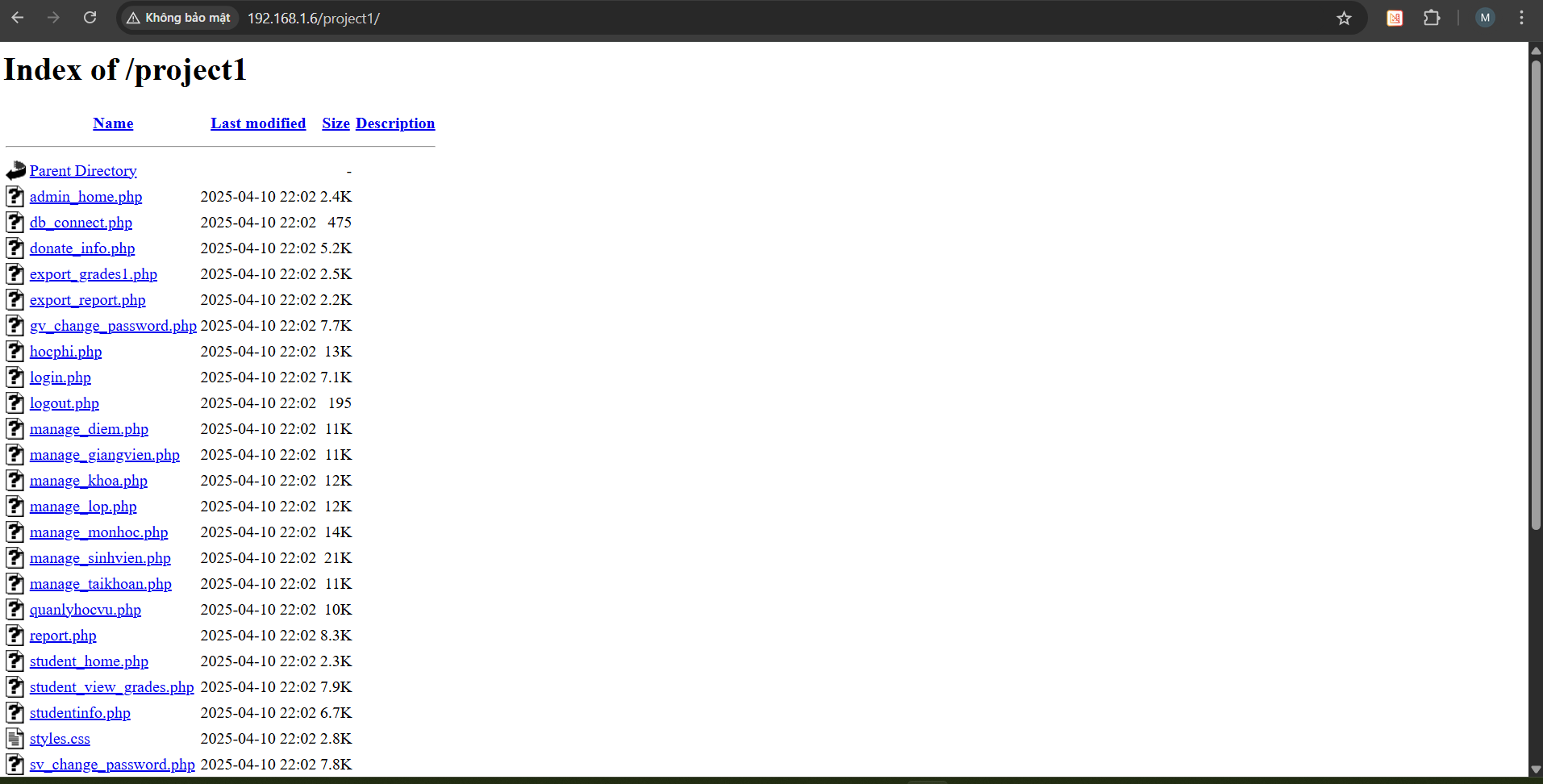
sudo chown -R www-data:www-data /var/www/html/project1

sudo chmod -R 755 /var/www/html/project1

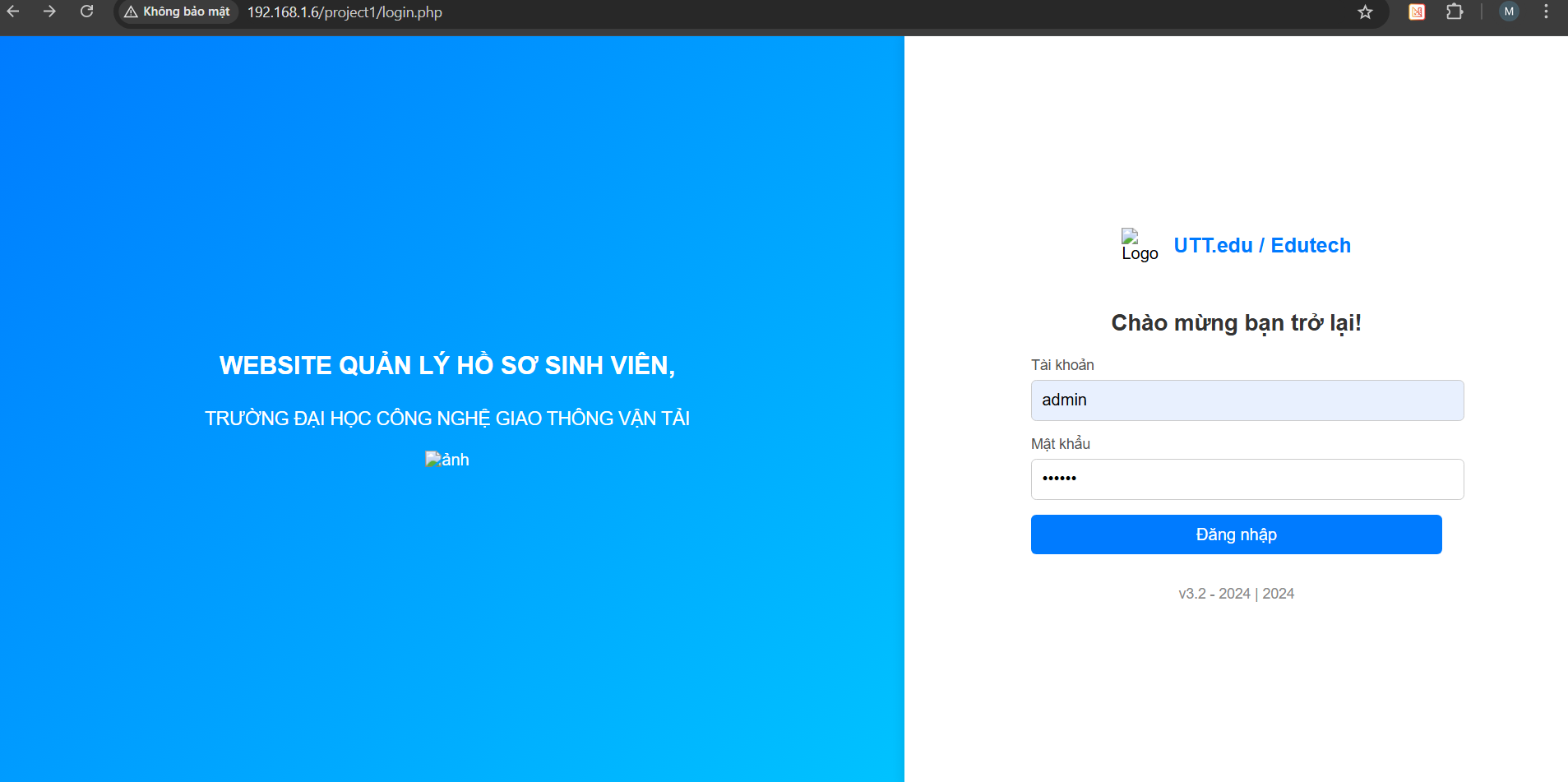
Bước 9: Truy cập trang web trên máy ảo bằng địa chỉ <http://localhost/project1>



Hoặc truy cập bằng IP máy ảo http://192.168.1.6/project1/



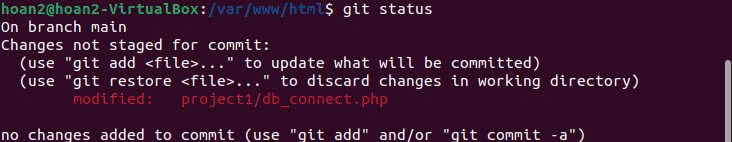
Như vậy ta đã triển khai 1 project lên máy ảo mà có thể truy cập bằng máy vật lý.



## 2.3 Sử dụng công cụ Git để quản lý mã nguồn.

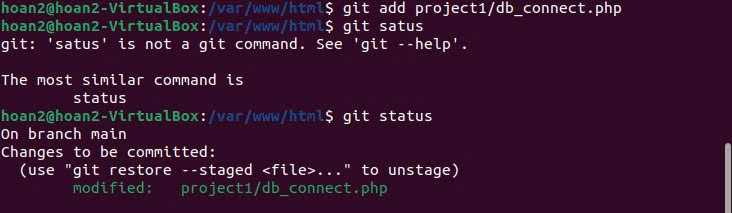
- Tạo thư mục Git: sudo git init

-Kiểm tra trạng của thư mục: sudo git status



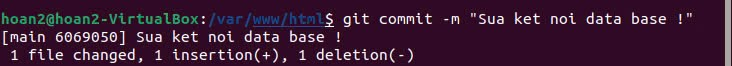
- Chon file trước khi commit vào bộ nhớ tạm:

sudo git add project1/db\_connect.php



-Lưu các thay đôi và vùng tạm:

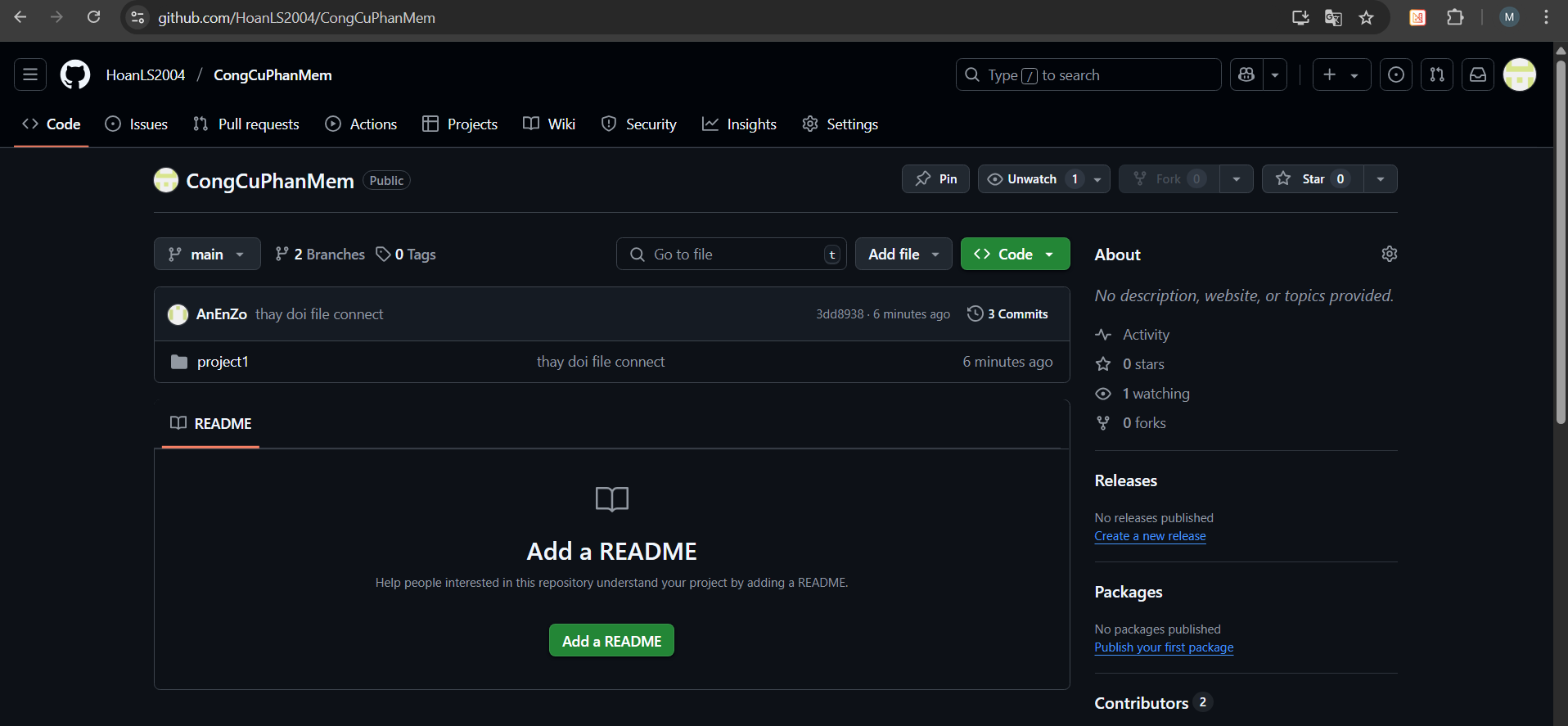
git commit -m "Sua file ket noi data base"



-Kết nối link github:

git remote add origin https://github.com/HoanLS2004/CongCuPhanMem.git

- Đẩy file lên github: git push origin main



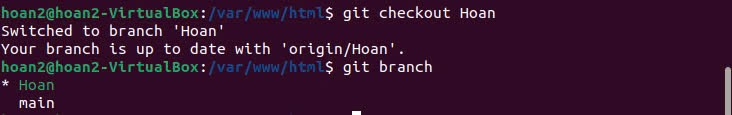
- Tạo nhánh mới:

git branch Hoan

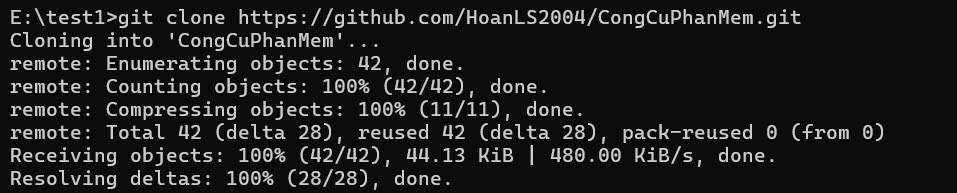
git push -u origin main



-Chuyển nhánh: git checkout Hoan

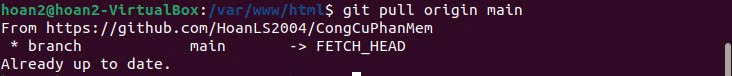


**-** Dowload file về: git clone https://github.com/HoanLS2004/CongCuPhanMem.git

****

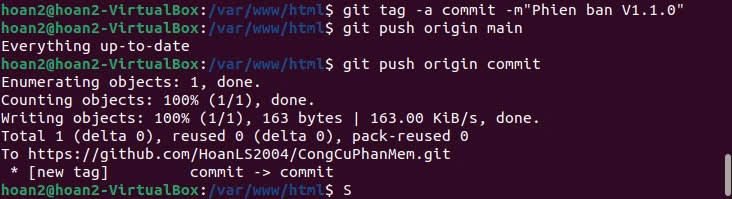
- Fetch dữ liệu mới nhất từ GitHub: git fetch origin

- Cập nhật code mới nhất từ repositore: git pull origin main



- Tạo tag: git tag -a commit -m “Phien ban V1.1.0”

- Đẩy tag lên github: git push origin commit



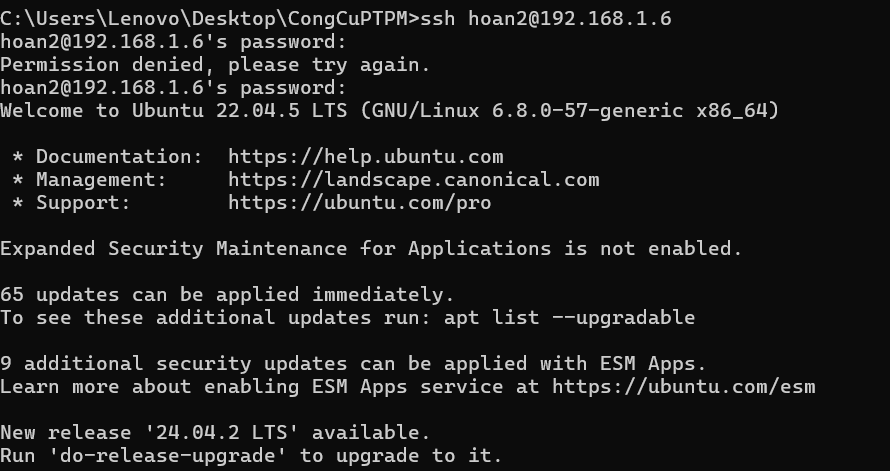
## 2.4. Cấu hình truy cập ssh, ftp, sftp.

### 2.4.1. Cấu hình ssh.

- Cài ssh: sudo apt install openssh-server -y

- Khởi động ssh: sudo systemctl enable ssh.

- Truy cập máy chủ qua ssh: ssh hoan2@192.168.1.6



- Khởi đông lại ssh: sudo systemctl restart ssh

### 2.4.2 Cấu hình ftp.

- Cài ftp: sudo apt install vsftpd

- Cấu hình: sudo apt install vsftpd

Chỉnh sửa file cấu hình:

listen=YES

listen\_ipv6=NO

anonymous\_enable=NO

local\_enable=YES

write\_enable=YES

chroot\_local\_user=YES

allow\_writeable\_chroot=YES

user\_sub\_token=$USER

local\_root=/home/$USER/ftp

pasv\_enable=Yes

- Tạo thư mục cho người dùng FTP:

sudo mkdir -p /home/hoanftp/ftp/upload

sudo chown nobody:nogroup /home/hoanftp/ftp

sudo chmod a-w /home/hoanftp/ftp

sudo chown hoanftp:hoanftp /home/hoanftp/ftp/upload

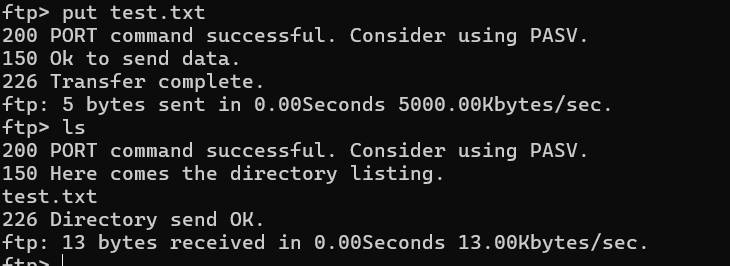
Như vậy ta đã có thể sử dụng giao thức ftp để truyền file giữa các máy với nhau.

- Đẩy file lên máy ảo:

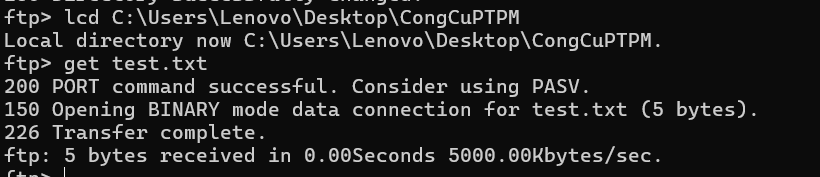
ftp> cd upload

ftp> lcd C:\Users\Lenovo\Desktop\CongCuPTPM

ftp> put test.txt



- Lấy file từ máy ảo về



- Khởi động lại ftp: sudo systemctl restart vstpd

### 2.4.2. Cấu hình sftp.

- Tạo user để truy cập mỗi sftp: sudo adduser hoandsftp

- Tạo thư mục:

sudo mkdir -p /home/hoansftp/upload

sudo chown root:root /home/hoansftp

sudo chmod 755 /home/hoansftp

sudo chown hoansftp:sftpusers /home/hoansftp/upload

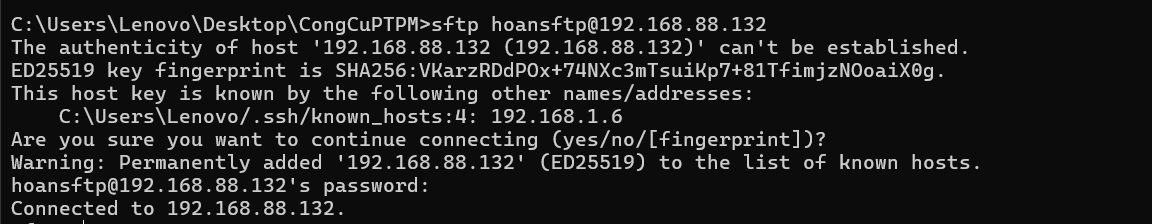
- Cấu hình để cấp quyền trong file: sudo nano /etc/ssh/sshd\_config

Chỉnh sửa nội dung file để có thể truy cập:



- Khởi động lại sftp: sudo systemctl restart ssh

Như vậy ta có thể truyền file qua giao thức sftp.



- Đẩy file lên máy ảo:

sftp> cd uploads

sftp> lcd C:\Users\Lenovo\Desktop\CongCuPTPM

sftp> put test2.txt

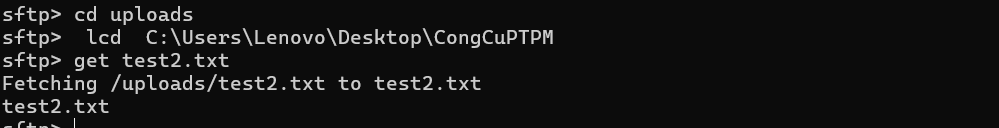


- Tải file từ trên máy ảo về:

sftp> cd uploads

sftp> lcd C:\Users\Lenovo\Desktop\CongCuPTPM

sftp> get test2.txt



# KẾT LUẬN

Qua quá trình tìm hiểu và triển khai project trên môi trường máy ảo, kết hợp với việc sử dụng các công cụ phát triển phần mềm như Git hoặc SVN, em đã có cái nhìn tổng quan và thực tiễn hơn về quy trình xây dựng và quản lý một hệ thống phần mềm chuyên nghiệp.

Việc cài đặt môi trường phát triển trên máy ảo, cấu hình cho phép truy cập từ máy khác qua địa chỉ IP nội bộ hoặc tên miền ảo không chỉ giúp mô phỏng môi trường triển khai thực tế mà còn nâng cao khả năng làm việc nhóm và kiểm thử trên nhiều thiết bị. Đồng thời, việc áp dụng công cụ quản lý mã nguồn như Git đã hỗ trợ hiệu quả trong việc lưu trữ, theo dõi và quản lý sự thay đổi của mã nguồn trong suốt quá trình phát triển.

Thông qua bài thực hành này, em không chỉ nâng cao kỹ năng về cấu hình hệ thống, quản lý phiên bản và triển khai ứng dụng, mà còn rèn luyện tư duy làm việc theo quy trình chuyên nghiệp – một yếu tố cần thiết đối với các lập trình viên trong thời đại công nghệ hiện nay.

Tuy vẫn còn một số hạn chế do thời gian và phạm vi kiến thức, nhưng bài làm đã phần nào đạt được mục tiêu đề ra. Em hy vọng đây sẽ là nền tảng để tiếp tục phát triển những kỹ năng quan trọng hơn trong các dự án thực tế trong tương lai.