

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH



Mạng máy tính - CO3094

Báo cáo

DEVELOP A SIMPLE FILE-SHARING APPLICATION

Giảng viên hướng dẫn: Nguyễn Phương Duy

Sinh viên thực hiện: 2110527 - Nguyễn Hoàng Duy Tân
2112594 - Trần Nguyễn Minh Tuệ
2110342 - Nguyễn Minh Lộc

Mục lục

| | |
|--|----|
| 1. Thành viên và phân chia công việc | 5 |
| 2. Phân tích yêu cầu | 6 |
| 2.1. Functional requirements | 6 |
| 2.1.1. Client Functions | 6 |
| 2.1.2. Server Functions | 7 |
| 2.2. Non-functional requirements | 7 |
| 2.3. Phân tích kiến trúc | 7 |
| 2.3.1. Kiến trúc Peer-to-Peer (P2P) | 7 |
| 2.3.2. Kiến trúc client-server | 7 |
| 3. Giới thiệu Protocol | 9 |
| 3.1. HyperText Transfer Protocol | 9 |
| 3.2. Transmission Control Protocol | 10 |
| 4. Socket programming with Python | 10 |
| 5. Application design | 12 |
| 5.1. Architecture design | 12 |
| 5.2. Flow protocol design | 13 |
| 5.2.1. Register | 13 |
| 5.2.2. Login | 13 |
| 5.2.3. Ping | 13 |
| 5.2.4. Discover | 13 |
| 5.2.5. Publish | 14 |
| 5.2.6. Fetch | 14 |
| 5.3. Activity diagram | 15 |
| 5.3.1. Register | 15 |
| 5.3.2. Login | 15 |
| 5.3.3. Ping | 16 |
| 5.3.4. Discover | 17 |
| 5.3.5. Publish | 18 |
| 5.3.6. Fetch | 18 |
| 6. Design UI | 19 |
| 6.1. Register and Login | 19 |
| 6.2. Publish Page | 20 |
| 6.3. Fetch Page | 21 |
| 7. Tutorial - Guides | 23 |

| | |
|-------------------------|----|
| 7.1. Server | 23 |
| 7.1.1. ping | 24 |
| 7.1.2. Discover | 26 |
| 7.2. Client | 28 |
| 7.2.1. Register | 28 |
| 7.2.2. Login | 30 |
| 7.2.3. Publish | 31 |
| 7.2.4. Fetch | 32 |
| 8. Error Handling | 35 |

Danh mục hình vẽ

| | |
|---|----|
| Hình 1: 8 giao thức mạng máy tính phổ biến (Nguồn Bytebytego) | 9 |
| Hình 2: Mô hình socket | 10 |
| Hình 3: Quy trình một kết nối socket để gửi nhận dữ liệu (Nguồn ResearchGate) | 11 |
| Hình 4: Kiến trúc tổng quan của hệ thống | 12 |
| Hình 5: Sơ đồ luồng của chức năng đăng ký | 13 |
| Hình 6: Sơ đồ luồng của chức năng đăng nhập | 13 |
| Hình 7: Sơ đồ luồng của chức năng ping | 13 |
| Hình 8: Sơ đồ luồng của chức năng discover | 13 |
| Hình 9: Sơ đồ luồng của chức năng publish | 14 |
| Hình 10: Sơ đồ luồng của chức năng fetch | 14 |
| Hình 11: Sơ đồ hoạt động của chức năng đăng ký | 15 |
| Hình 12: Sơ đồ hoạt động của chức năng đăng nhập | 15 |
| Hình 13: Sơ đồ hoạt động của chức năng ping | 16 |
| Hình 14: Sơ đồ hoạt động của chức năng discover | 17 |
| Hình 15: Sơ đồ hoạt động của chức năng publish | 18 |
| Hình 16: Sơ đồ hoạt động của chức năng fetch | 18 |
| Hình 17: UI Register and Login | 19 |
| Hình 18: UI Publish File | 20 |
| Hình 19: UI Fetch File | 21 |
| Hình 20: UI Profile | 22 |
| Hình 21: Start Server | 23 |
| Hình 22: Start Server | 24 |
| Hình 23: Ping | 25 |
| Hình 24: Ping No Online | 26 |
| Hình 25: Discover | 27 |

| | |
|--|----|
| Hình 26: Start Client | 28 |
| Hình 27: Client Register | 28 |
| Hình 28: Client Register Success | 29 |
| Hình 29: Server Register Success | 30 |
| Hình 30: Client Login | 31 |
| Hình 31: Login Success | 31 |
| Hình 32: Publish | 32 |
| Hình 33: Publish Success | 32 |
| Hình 34: Fetch | 33 |
| Hình 35: Fetch Success | 33 |
| Hình 36: Fetch Success | 34 |

Danh mục bảng biểu



1. Thành viên và phân chia công việc

| Họ và tên | Nhiệm vụ | Mức độ hoàn thành |
|----------------------|----------|-------------------|
| Nguyễn Hoàng Duy Tân | | 100% |
| Trần Nguyễn Minh Tuệ | | 100% |
| Nguyễn Minh Lộc | | 100% |

2. Phân tích yêu cầu

2.1. Functional requirements

Xây dựng một ứng dụng chia sẻ file đơn giản với giao thức được định nghĩa sẵn, sử dụng những giao thức trong TCP/IP stack.

2.1.1. Client Functions

Basic functions

Đăng ký trong kho lưu trữ

- Máy khách có thể gửi yêu cầu đăng ký file có trong kho lưu trữ cho máy chủ.
- Thông điệp đăng ký file: “publish “. File trên máy khách sẽ được thêm vào kho lưu trữ dưới tên .
- Các file sau khi đăng ký sẽ được lưu trữ trong kho lưu trữ của tài khoản được liên kết với máy chủ.

Gửi yêu cầu tải file cho server

- Máy khách có thể gửi yêu cầu tải file không có sẵn trong kho lưu trữ của mình. Lúc này máy chủ sẽ phản hồi lại danh sách các máy khách khác có file được yêu cầu.

Tải file trực tiếp từ nguồn muốn chọn

- Máy khách sau khi nhận được phản hồi từ máy chủ danh sách máy khách có sẵn file được yêu cầu có thể chọn một nguồn thích hợp và gửi yêu cầu tải file tới đó.
- Các máy khách được cung cấp một danh sách yêu cầu tải file từ các máy khách khác, máy khách có thể chọn 1 file trong danh sách và gửi yêu cầu tải file tới máy khách đó.
- Thông điệp tải file: “fetch “. Trong đó fname là 1 trong những tên file muốn chọn sau khi server đã phản hồi.

Extended functions

Đăng ký tài khoản

- Người dùng đăng ký địa chỉ của máy vào hệ thống của máy chủ.

Đăng nhập tài khoản và xác thực

- Người dùng đăng nhập tài khoản để sử dụng các chức năng của hệ thống.

Liệt kê danh sách lưu trữ

- Máy khách có thể kiểm tra danh sách các file có trong kho lưu trữ của mình.

Tìm kiếm bằng từ khóa

- Server sẽ hỗ trợ người dùng tìm kiếm file theo từ khóa.

2.1.2. Server Functions

Basic functions

Kiểm tra trạng thái máy chủ

- Máy chủ có thể kiểm tra trạng thái của máy chủ khác thông qua lệnh “ping <hostname>”

Xem danh sách file của máy khách khác

- Máy chủ có thể xem danh sách file trong kho lưu trữ của các máy khách thông qua lệnh “discover <hostname>”

Gửi thông tin cần thiết sau khi nhận yêu cầu tải file từ clients

- Sau khi nhận được yêu cầu tìm file từ người dùng, server sẽ tiến hành theo dõi và tìm kiếm để trả về các thông tin nơi đang lưu trữ các file đó cho clients: ID peer, thời gian file được cập nhật.

Extended functions

Xem file log

- Máy chủ có thể xem file log của máy khách khác thông qua.

2.2. Non-functional requirements

Giao diện người dùng – Cung cấp giao diện người dùng để sử dụng cho máy khách để nhập các lệnh và theo dõi quá trình tải tệp. **Multi-threading** – Triển khai đa luồng trong máy khách để có thể xử lý nhiều tải xuống cùng lúc. **Hiệu năng và tích hợp** – Đảm bảo rằng hệ thống hoạt động hiệu quả và có khả năng tích hợp với các mạng internet và hệ thống người dùng khác nhau.

2.3. Phân tích kiến trúc

2.3.1. Kiến trúc Peer-to-Peer (P2P)

- Kiến trúc Peer-to-Peer (P2P) là một mô hình mạng máy tính trong đó các máy tính (được gọi là nút hoặc “peers”) kết nối trực tiếp với nhau để chia sẻ tài nguyên và thông tin mà không cần sự tương tác trung tâm từ máy chủ. Trong kiến trúc P2P, mỗi máy tính có thể đồng thời hoạt động như máy khách và máy chủ, có nghĩa là chúng có khả năng yêu cầu tài nguyên từ các máy tính khác và chia sẻ tài nguyên với người khác.

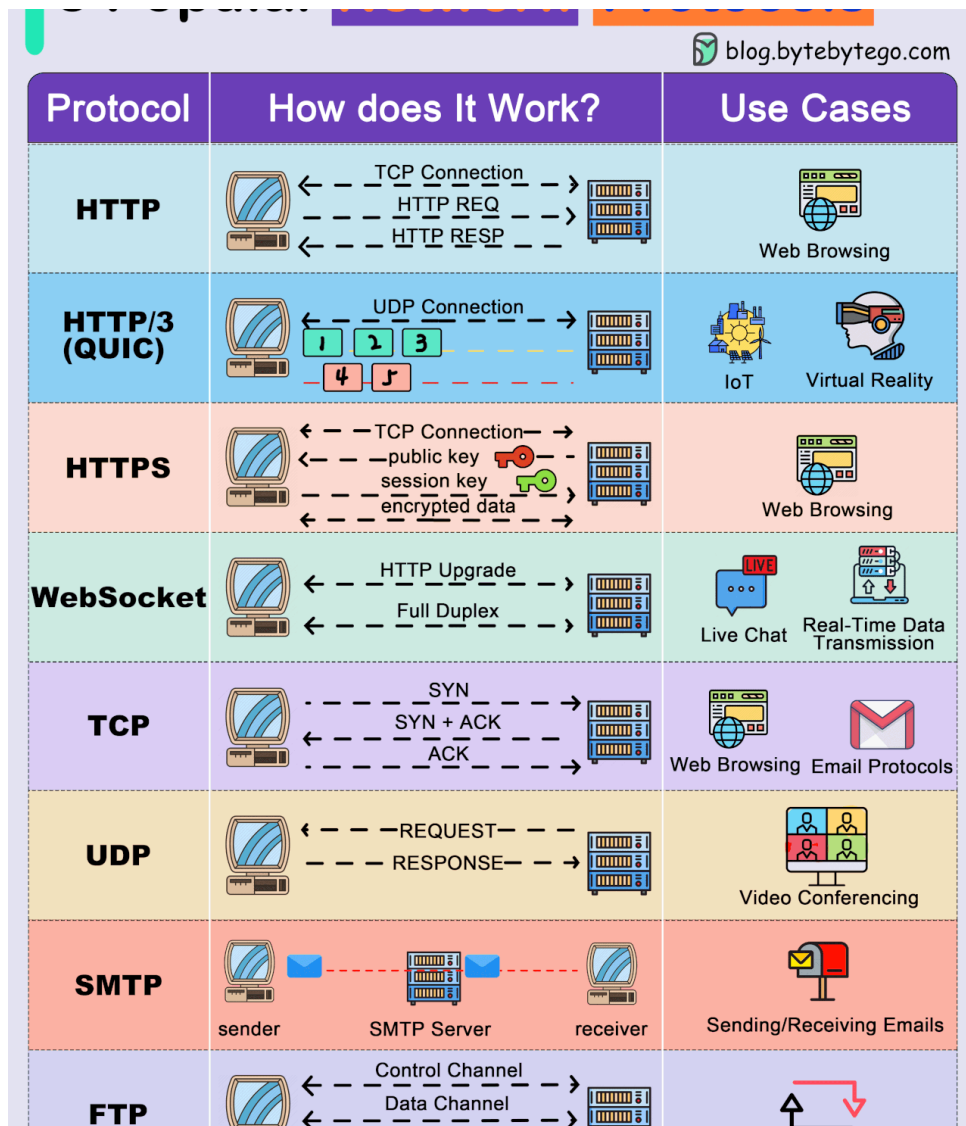
2.3.2. Kiến trúc client-server

- Kiến trúc client-server (còn được gọi là mô hình client-server) là một kiến trúc máy tính phổ biến được sử dụng trong việc tổ chức và quản lý các dịch vụ và tài nguyên trên mạng. Nó dựa trên sự phân chia các vai trò chính trong hệ thống thành hai phần: máy khách (client) và máy chủ (server). Hai phần này tương tác với nhau để cung cấp các dịch vụ, ứng dụng, và tài nguyên cho người dùng.

3. Giới thiệu Protocol

3.1. HyperText Tràner Protocol

HyperText Transfer Protocol (HTTP) là một giao thức ở tầng ứng dụng trong mô hình OSI để gửi và nhận tài liệu, hình ảnh, văn bản như HTML document. Về cơ bản, giao thức HTTP xây dựng trên cơ chế request-response trong mô hình client-server. Trong mô hình này, client có thể là một process thuộc máy tính này, server có thể là process thuộc máy tính khác, hai process thuộc hai phần cứng khác nhau khi muốn giao tiếp với nhau thì sẽ thông qua HTTP để giao tiếp. Ai là người request thì đó là client, người nhận request để response sẽ là server.



Hình 1: 8 giao thức mạng máy tính phổ biến (Nguồn Bytebytego)

3.2. Transmission Control Protocol

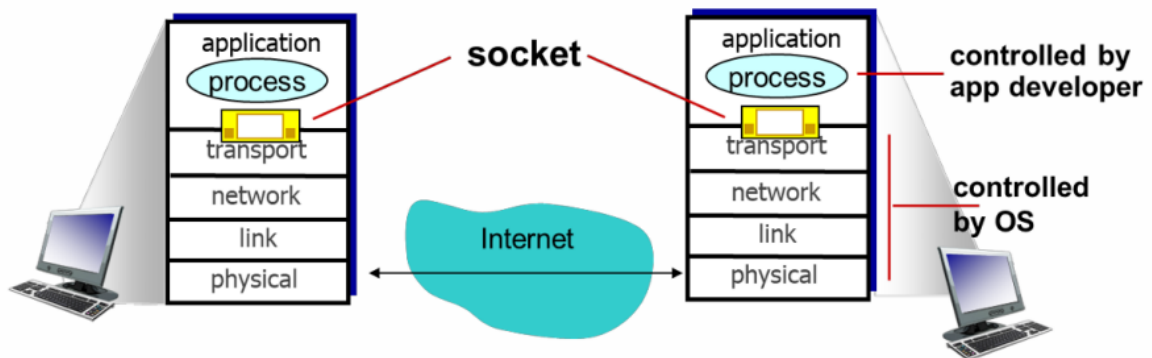
Transmission Control Protocol (TCP) là một trong các giao thức cốt lõi của bộ giao thức TCP/IP. Nhờ có TCP, các ứng dụng trên các host được nối mạng có thể tạo các kết nối với nhau, mà qua đó chúng có thể trao đổi dữ liệu hoặc các gói tin. Giao thức này đảm bảo chuyển giao dữ liệu tới nơi nhận một cách đáng tin cậy và đúng thứ tự.

Cách đặc tính cơ bản của TCP:

- Point-to-point: Trong một giao thức TCP, chỉ có một sender và một server được kết nối với nhau bằng 3-way handshaking.
- Reliable, in-order bit stream: Hỗ trợ truyền tin cậy và đúng thứ tự.
- Pipelined: Truyền song song nhằm tăng hiệu quả gửi nhận
- Flow control: Receiver kiểm soát tốc độ gửi của sender để tránh làm quá tải receiver.
- Congestion control: Tự động điều chỉnh tốc độ gửi ở mức tối đa mà không làm tắc nghẽn hệ thống.
- Full-duplex connection: hỗ trợ truyền hai chiều trong cùng một thời điểm trong một kết nối.

4. Socket programming with Python

Socket là cánh cổng ngăn cách giữa Application Layer với Transport Layer.



Hình 2: Mô hình socket

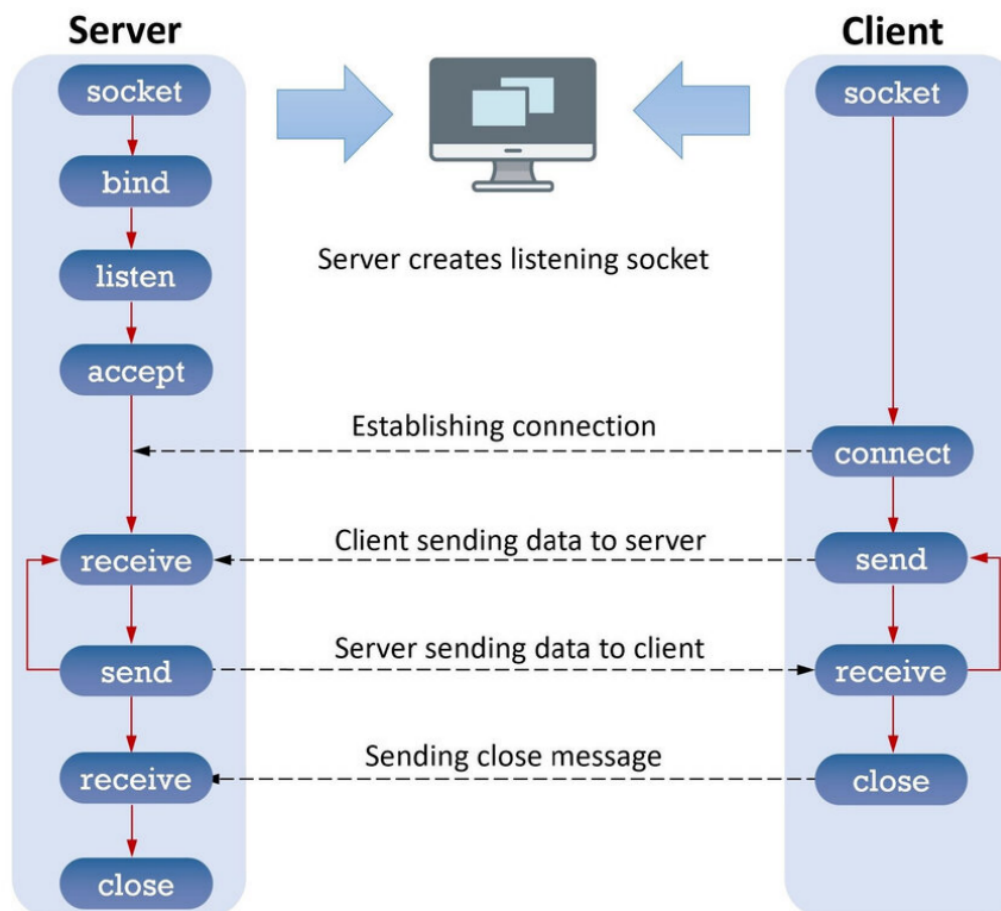
Thư viện socket của Python hỗ trợ một số API sau để thiết lập kết nối socket TCP/UDP:

- socket()
- bind()
- listen()
- accept()
- connect()

- connect_ex()
- send()
- recv()
- close()

Thư viện socket của Python hỗ trợ cả TCP socket lẫn UDP socket. Trong project này, nhóm dự định sử dụng TCP socket để hiện thực dự án.

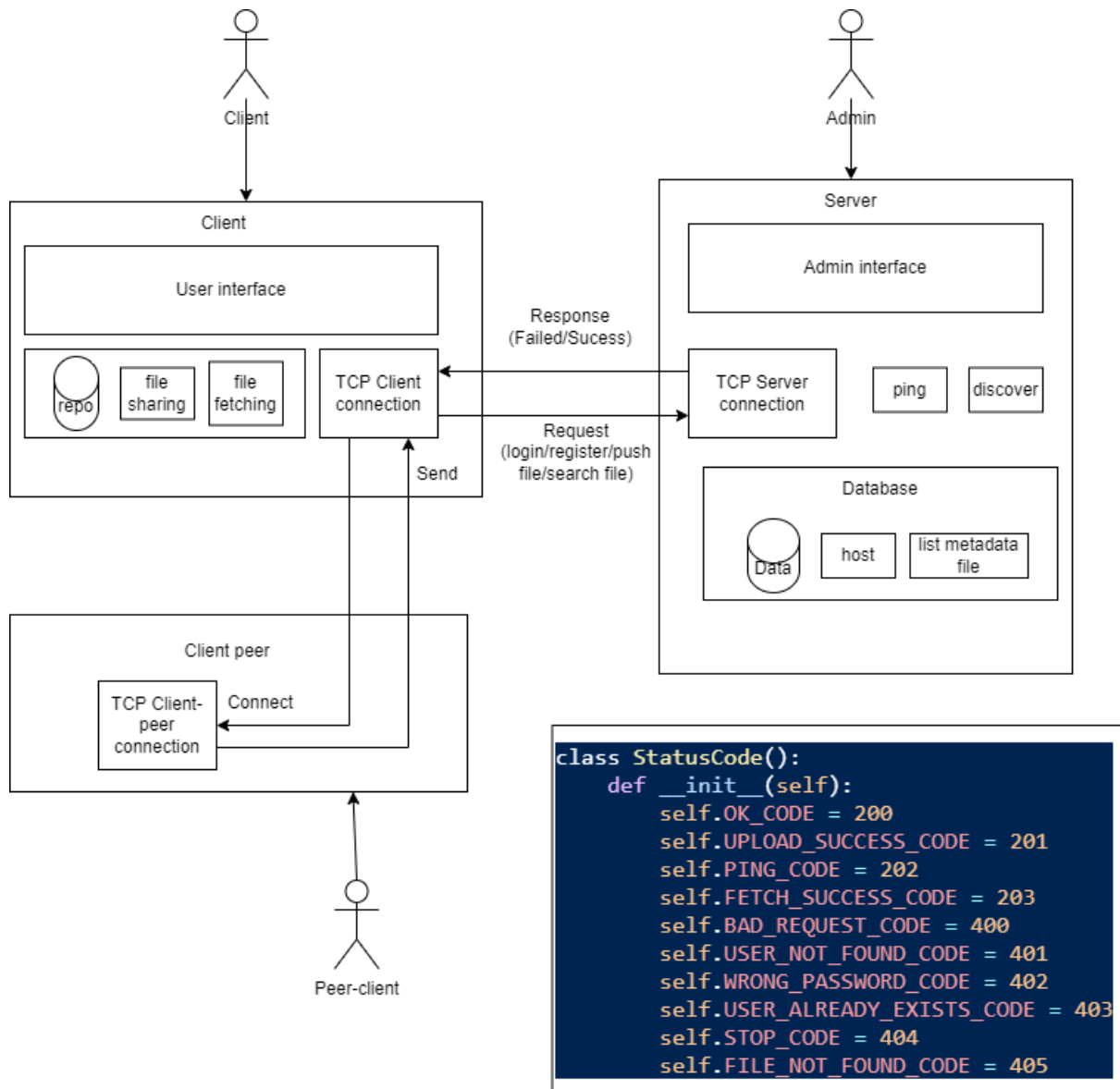
Flow của một TCP socket connection có thể được thể hiện như hình dưới đây:



Hình 3: Quy trình một kết nối socket để gửi nhận dữ liệu (Nguồn ResearchGate)

5. Application design

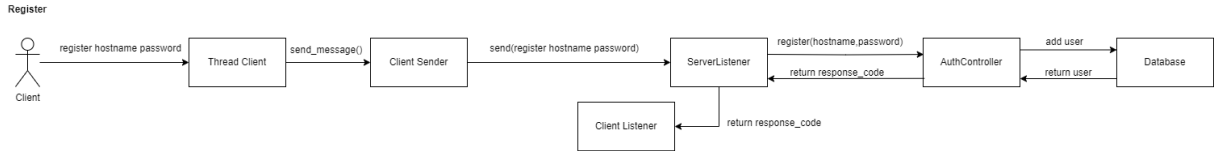
5.1. Architecture design



Hình 4: Kiến trúc tổng quan của hệ thống

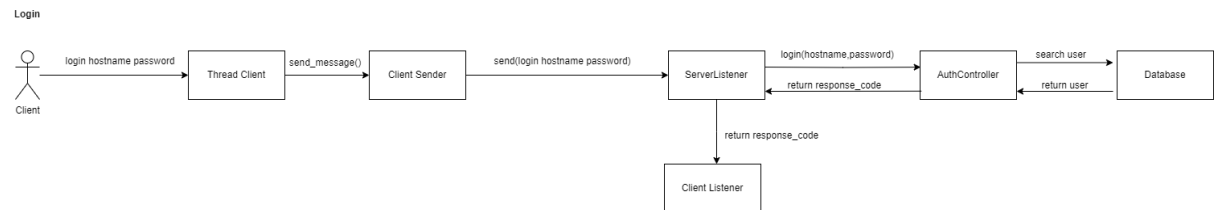
5.2. Flow protocol design

5.2.1. Register



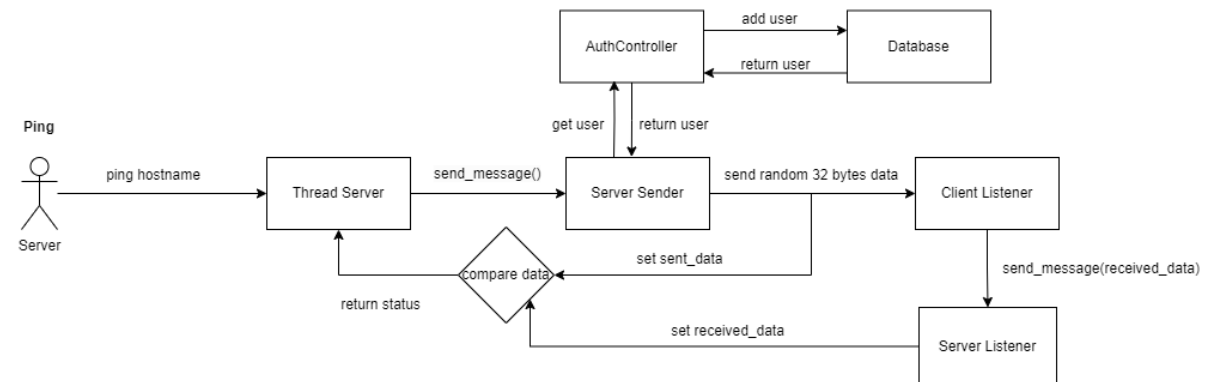
Hình 5: Sơ đồ luồng của chức năng đăng ký

5.2.2. Login



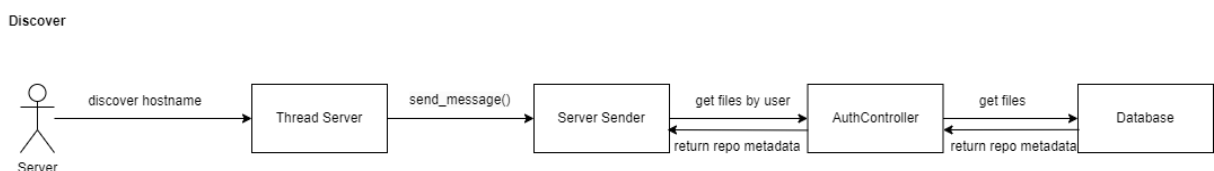
Hình 6: Sơ đồ luồng của chức năng đăng nhập

5.2.3. Ping



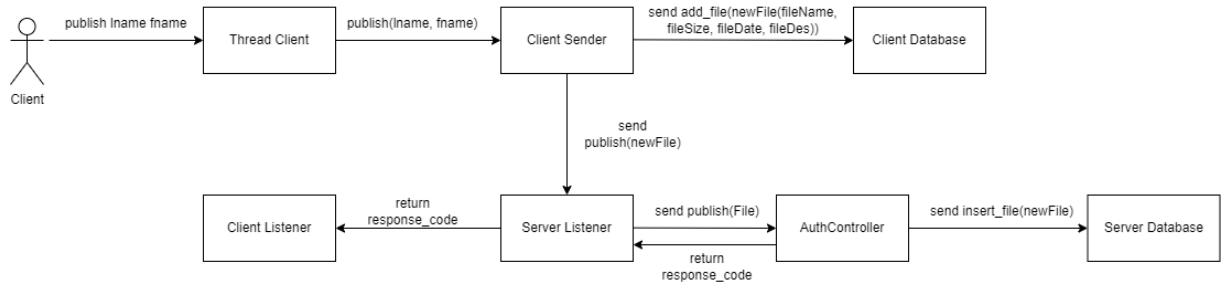
Hình 7: Sơ đồ luồng của chức năng ping

5.2.4. Discover



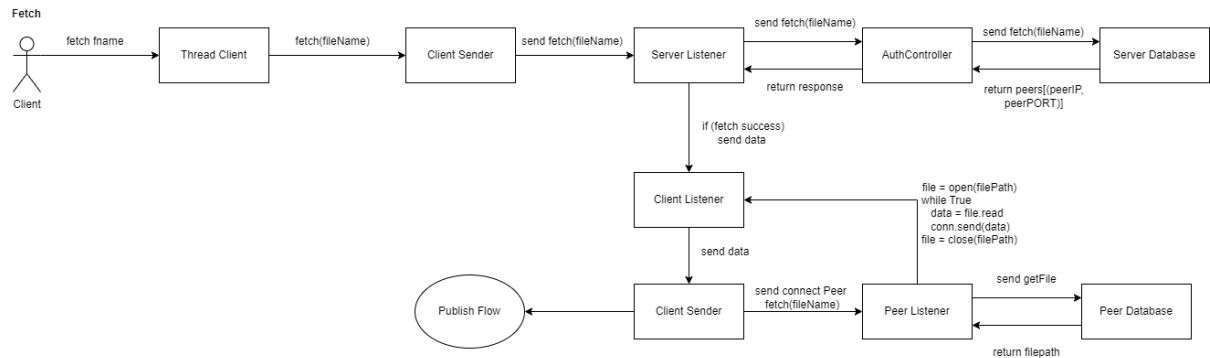
Hình 8: Sơ đồ luồng của chức năng discover

5.2.5. Publish



Hình 9: Sơ đồ luồng của chức năng publish

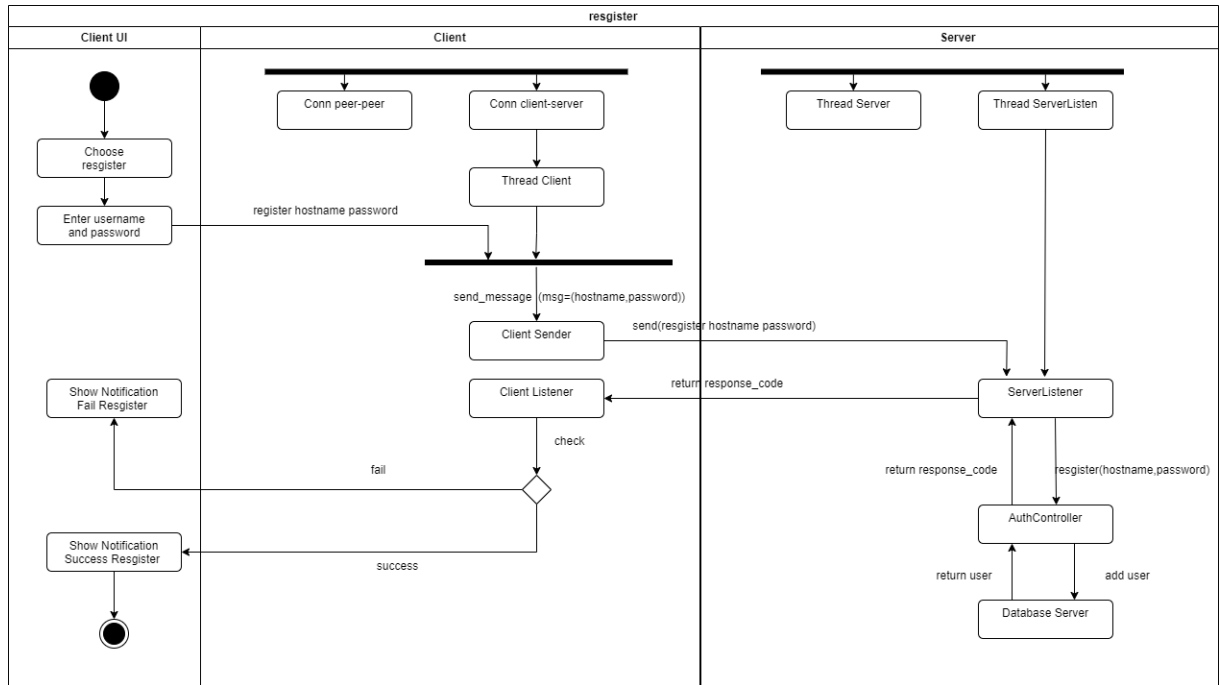
5.2.6. Fetch



Hình 10: Sơ đồ luồng của chức năng fetch

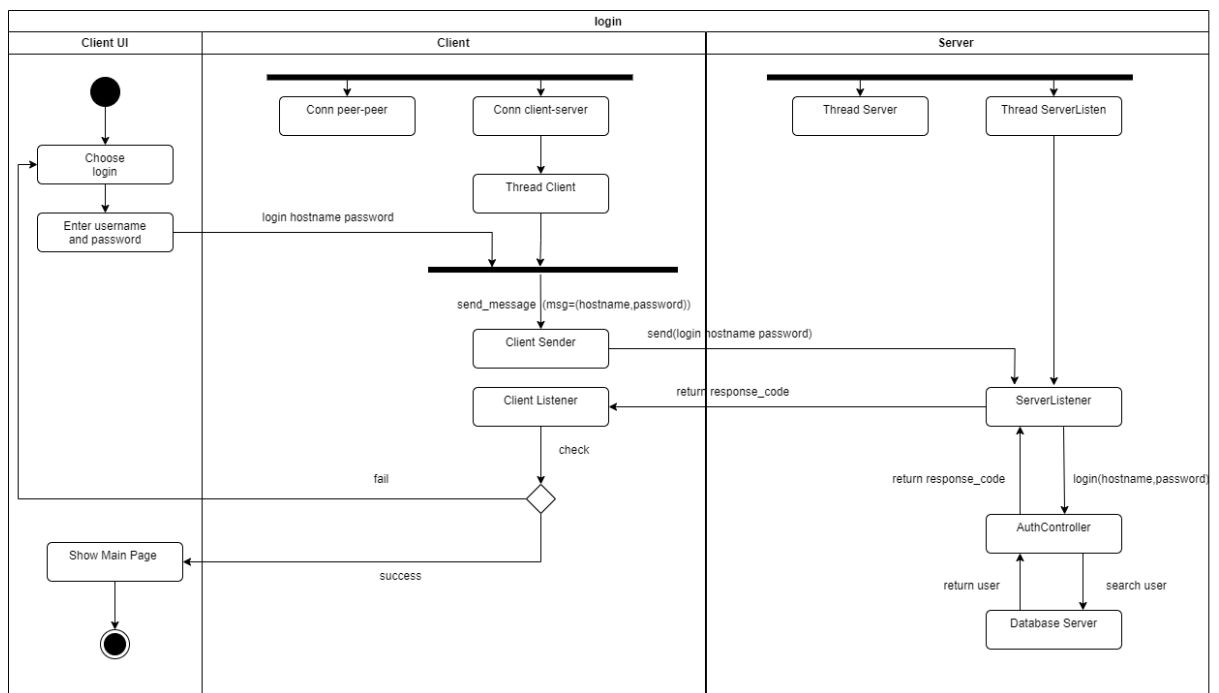
5.3. Activity diagram

5.3.1. Register



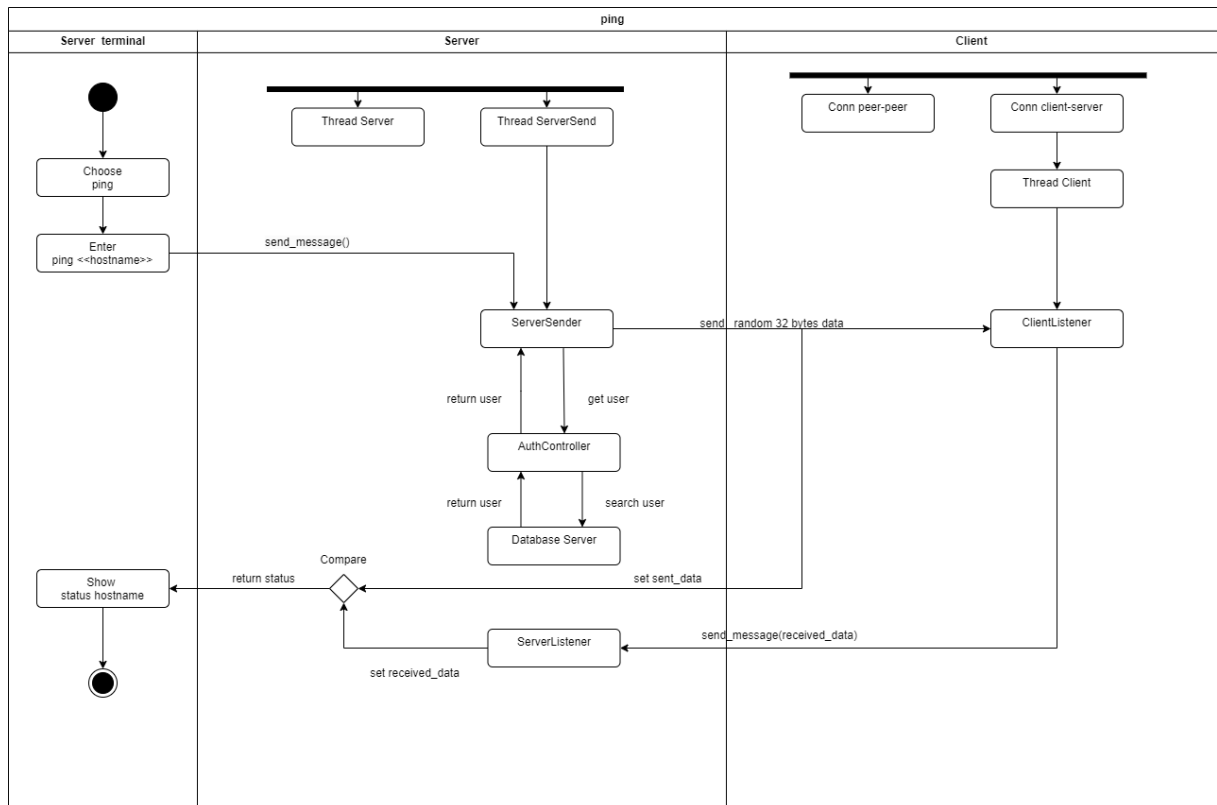
Hình 11: Sơ đồ hoạt động của chức năng đăng ký

5.3.2. Login



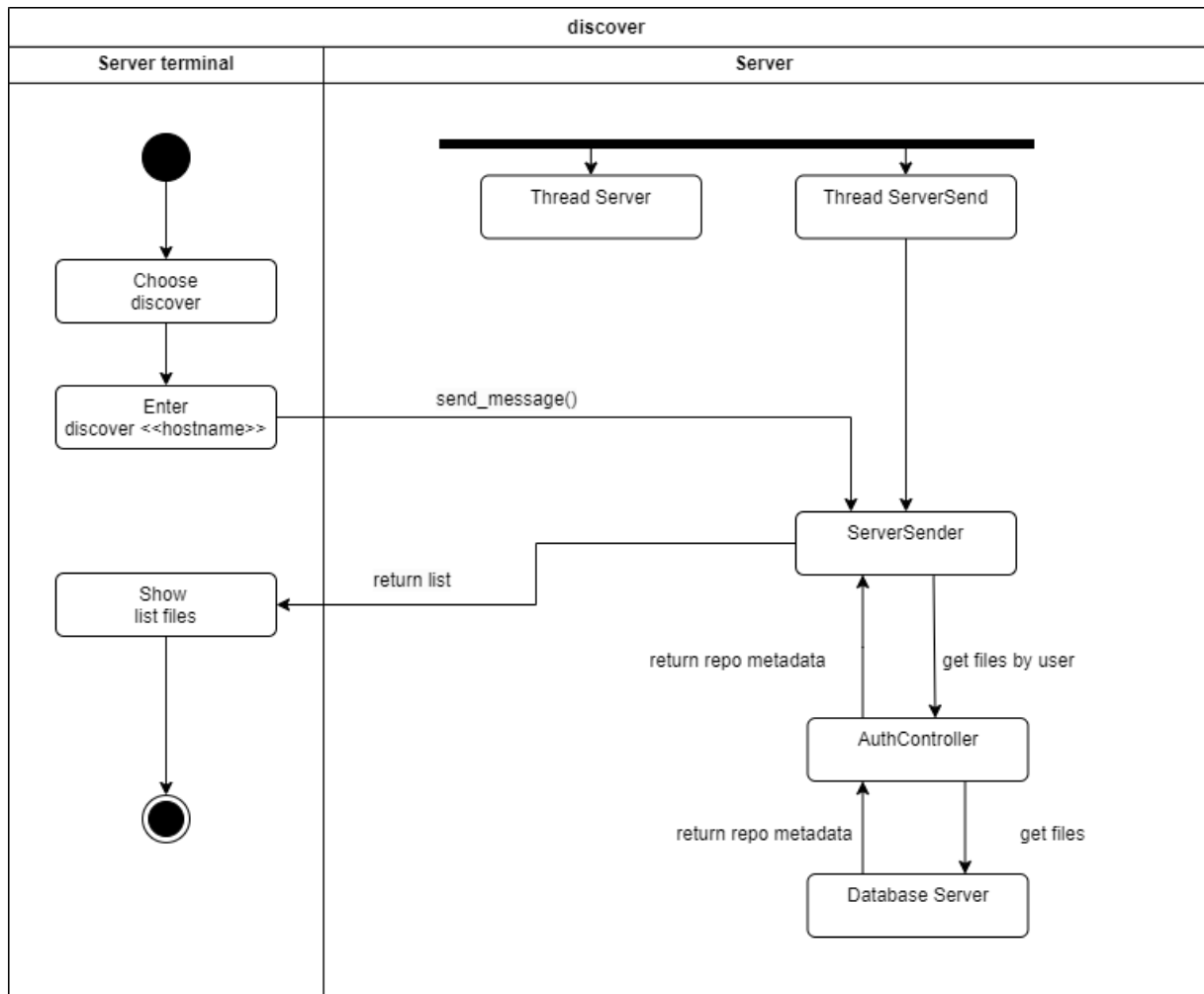
Hình 12: Sơ đồ hoạt động của chức năng đăng nhập

5.3.3. Ping



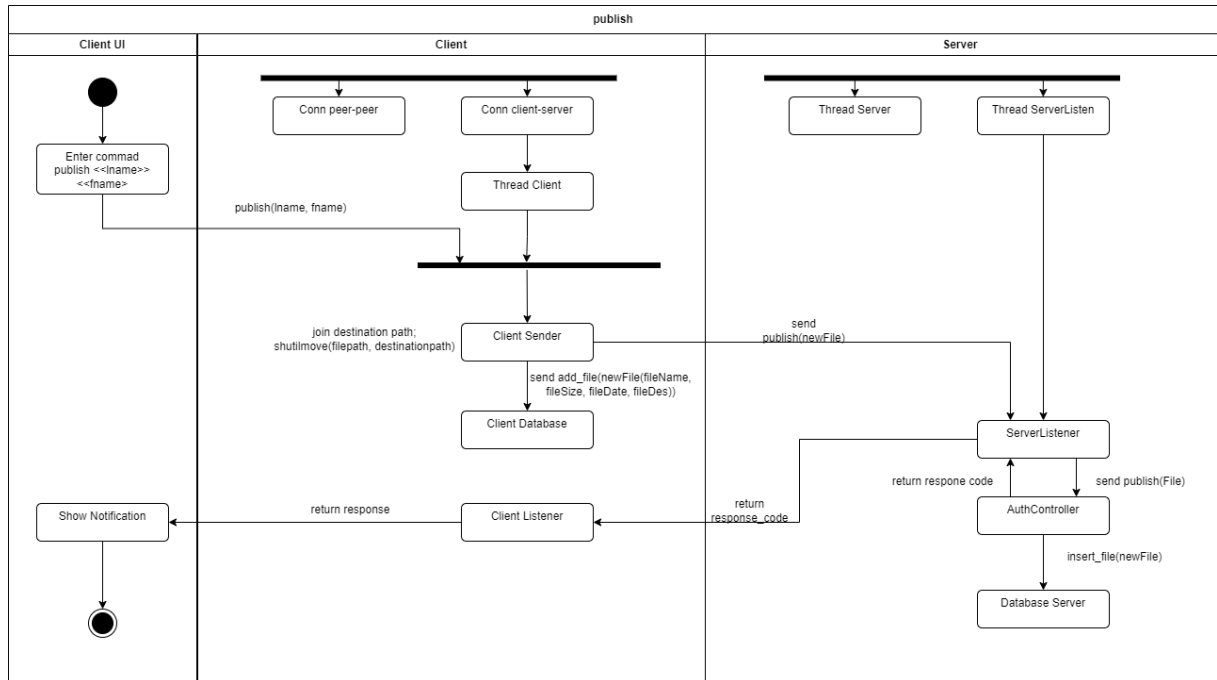
Hình 13: Sơ đồ hoạt động của chức năng ping

5.3.4. Discover



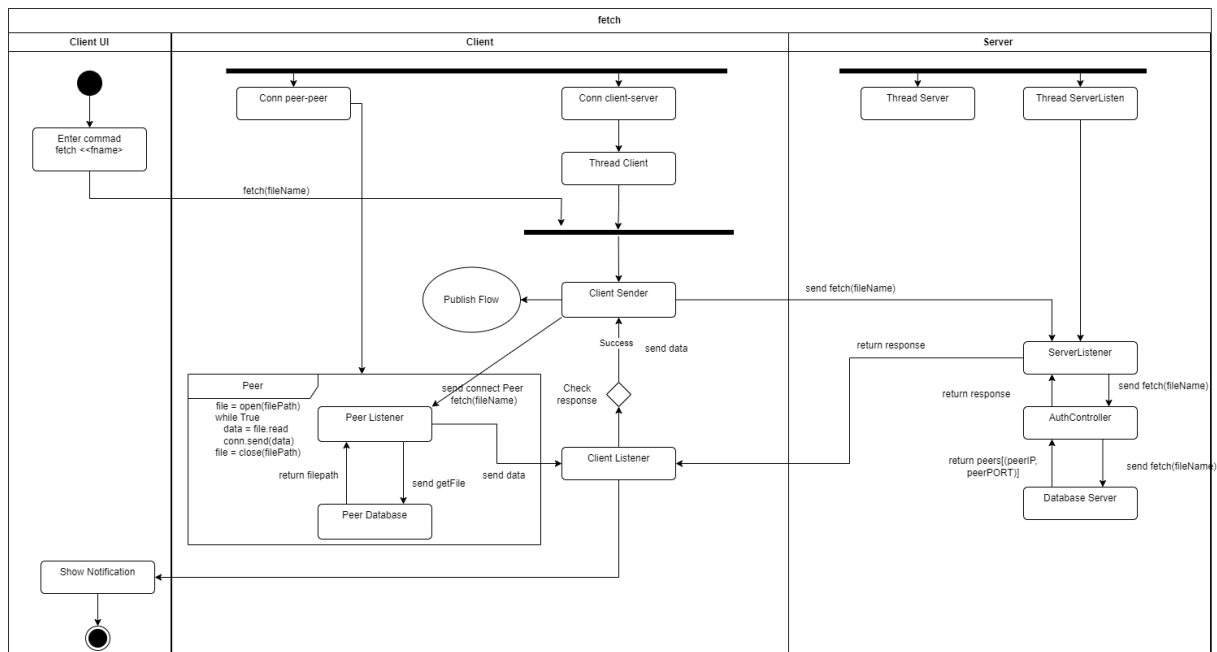
Hình 14: Sơ đồ hoạt động của chức năng discover

5.3.5. Publish



Hình 15: Sơ đồ hoạt động của chức năng publish

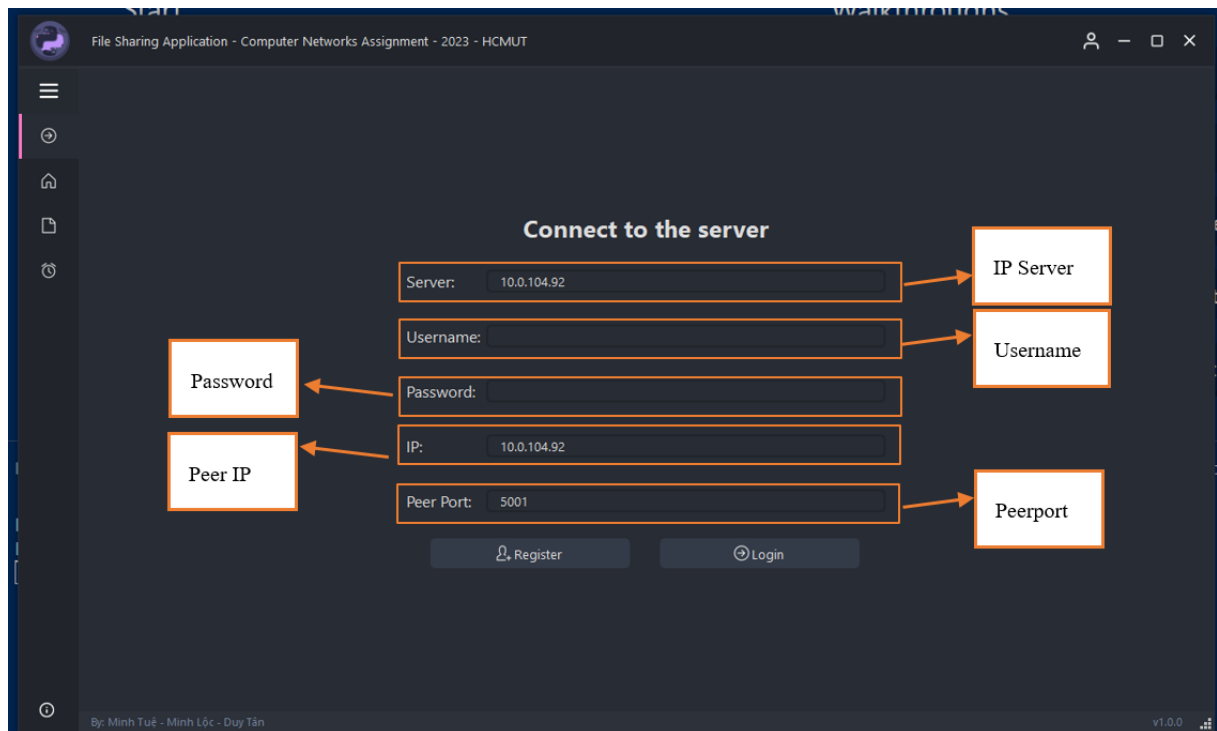
5.3.6. Fetch



Hình 16: Sơ đồ hoạt động của chức năng fetch

6. Design UI

6.1. Register and Login



The screenshot shows a web application window titled "File Sharing Application - Computer Networks Assignment - 2023 - HCMUT". The main content area is titled "Connect to the server". It contains five input fields with labels: "Server:" (value: 10.0.104.92), "Username:", "Password:", "IP:" (value: 10.0.104.92), and "Peer Port:" (value: 5001). Below these fields are two buttons: "Register" and "Login". Arrows point from labels to their respective input fields: "IP Server" points to the Server field, "Username" points to the Username field, "Password" points to the Password field, "Peer IP" points to the IP field, and "Peerport" points to the Peer Port field.

Hình 17: UI Register and Login

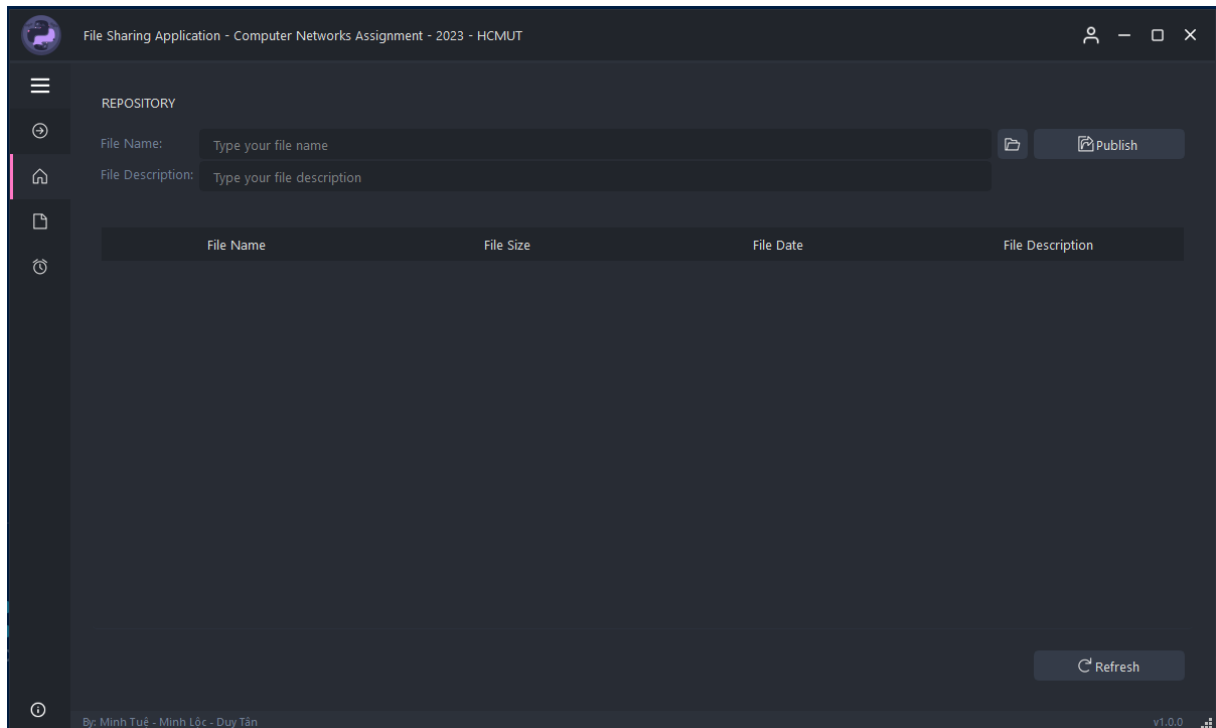
Đây là trang đăng ký, đăng nhập vào hệ thống bao gồm: địa chỉ IP Server, username, password, địa chỉ IP Peer, địa chỉ Peer Port.

Phía dưới có 2 nút: Register và Login.

Khi nhấn nút Register, hệ thống sẽ gửi thông tin đăng ký lên Server.

Khi nhấn nút Login, hệ thống sẽ gửi thông tin đăng nhập lên Server.

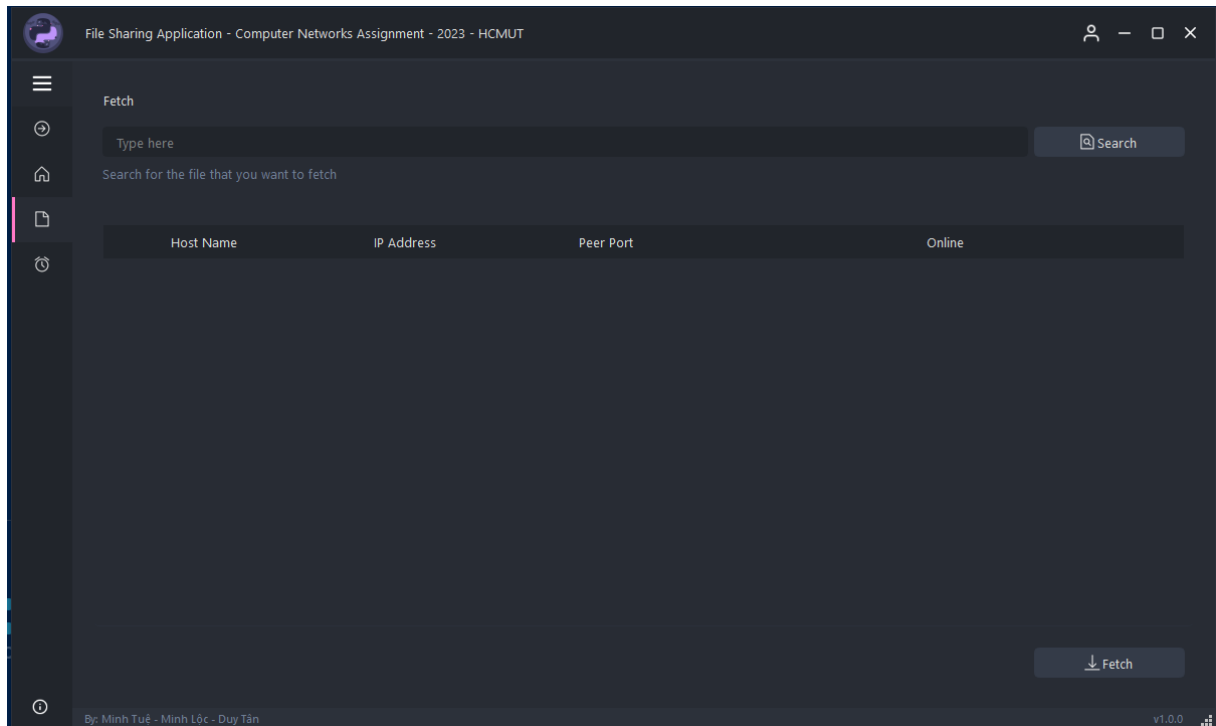
6.2. Publish Page



Hình 18: UI Publish File

Đây là trang Publish File, bao gồm: đường dẫn file, tên file fname sau khi publish, mô tả file, nút Publish. Khi nhấn nút Publish, hệ thống sẽ gửi thông tin Publish lên Server. Phía dưới là lưới hiển thị danh sách các file đã Publish.

6.3. Fetch Page



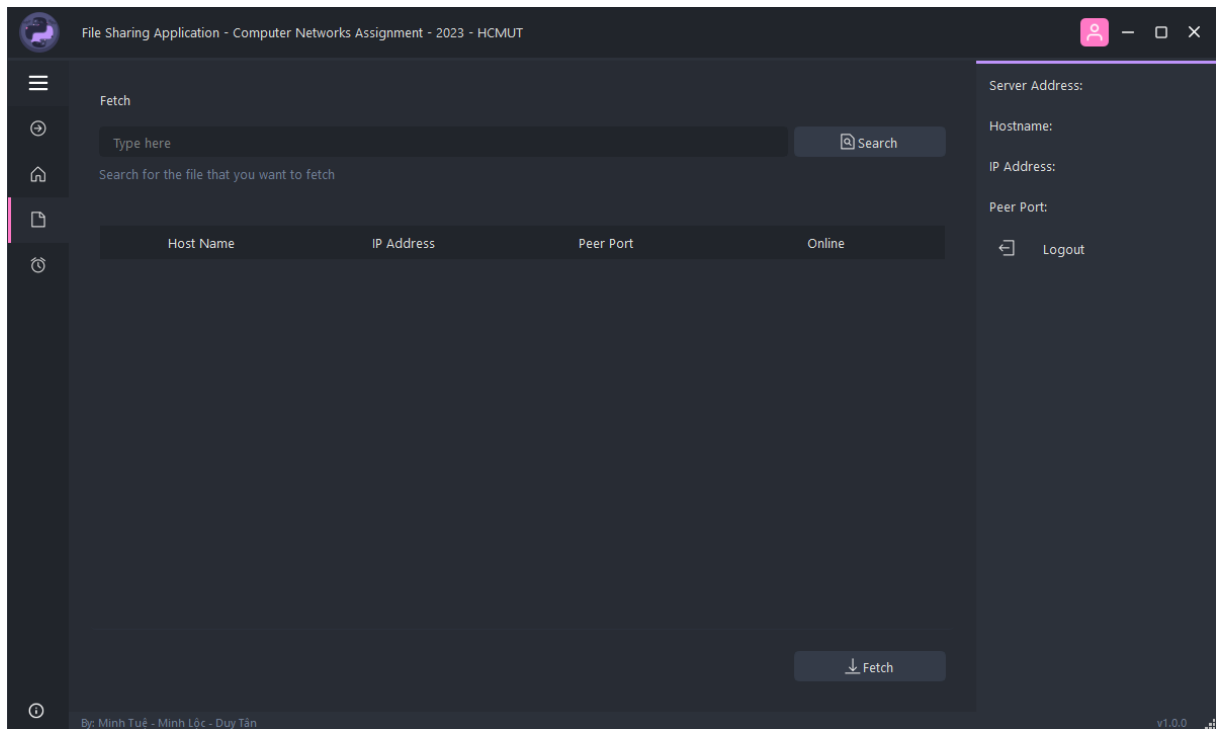
Hình 19: UI Fetch File

Đây là trang Fetch File, bao gồm: tên file fname cần Fetch, nút Search.

Sau khi Search, giao diện sẽ hiển thị danh sách các file có tên fname.

Người dùng có thể chọn 1 file trong danh sách và nhấn nút Fetch để Fetch file về.

Ngoài ra còn hiển thị thông tin người dùng: username, địa chỉ IP Peer, địa chỉ Peer Port.

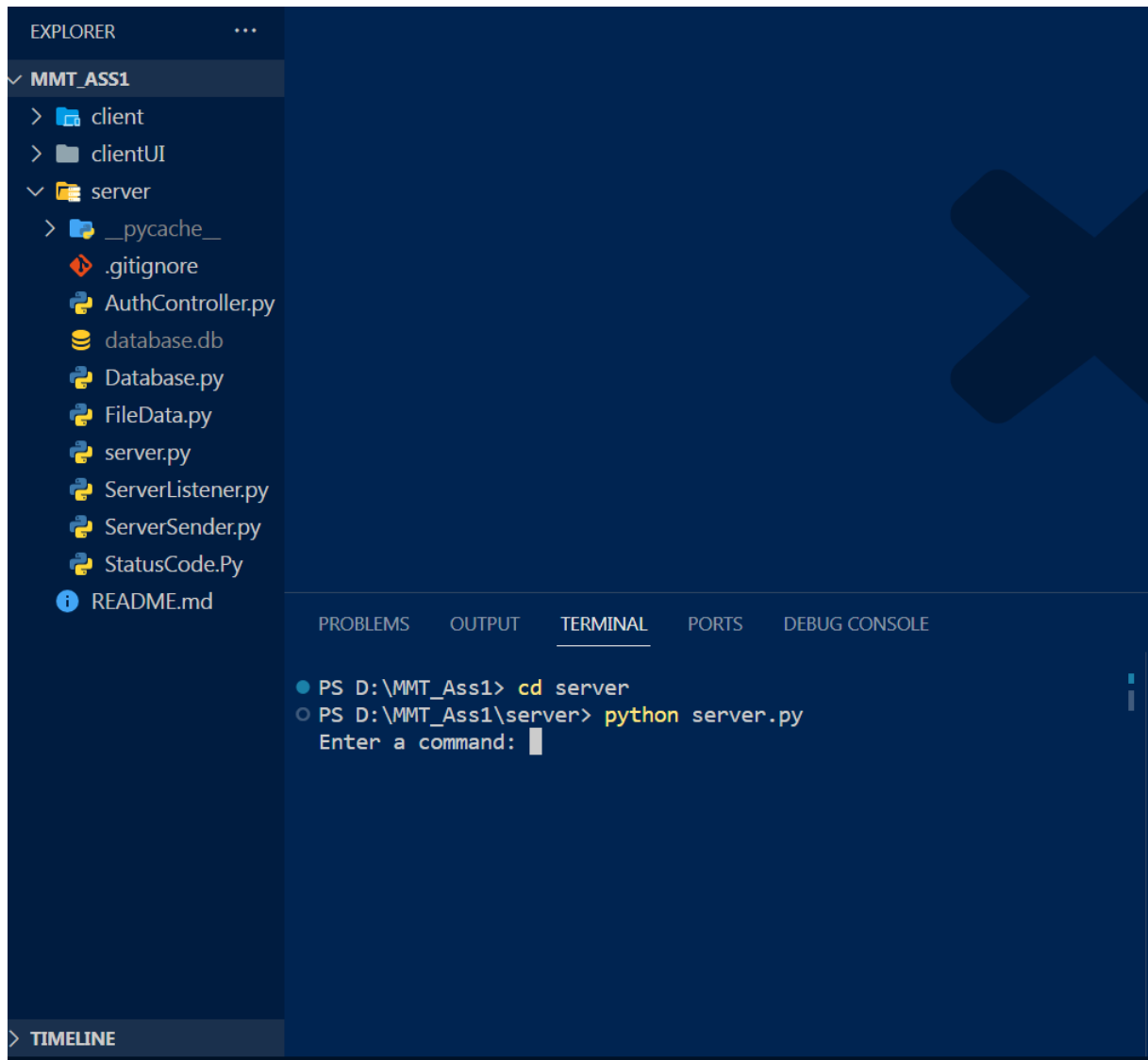


Hình 20: UI Profile

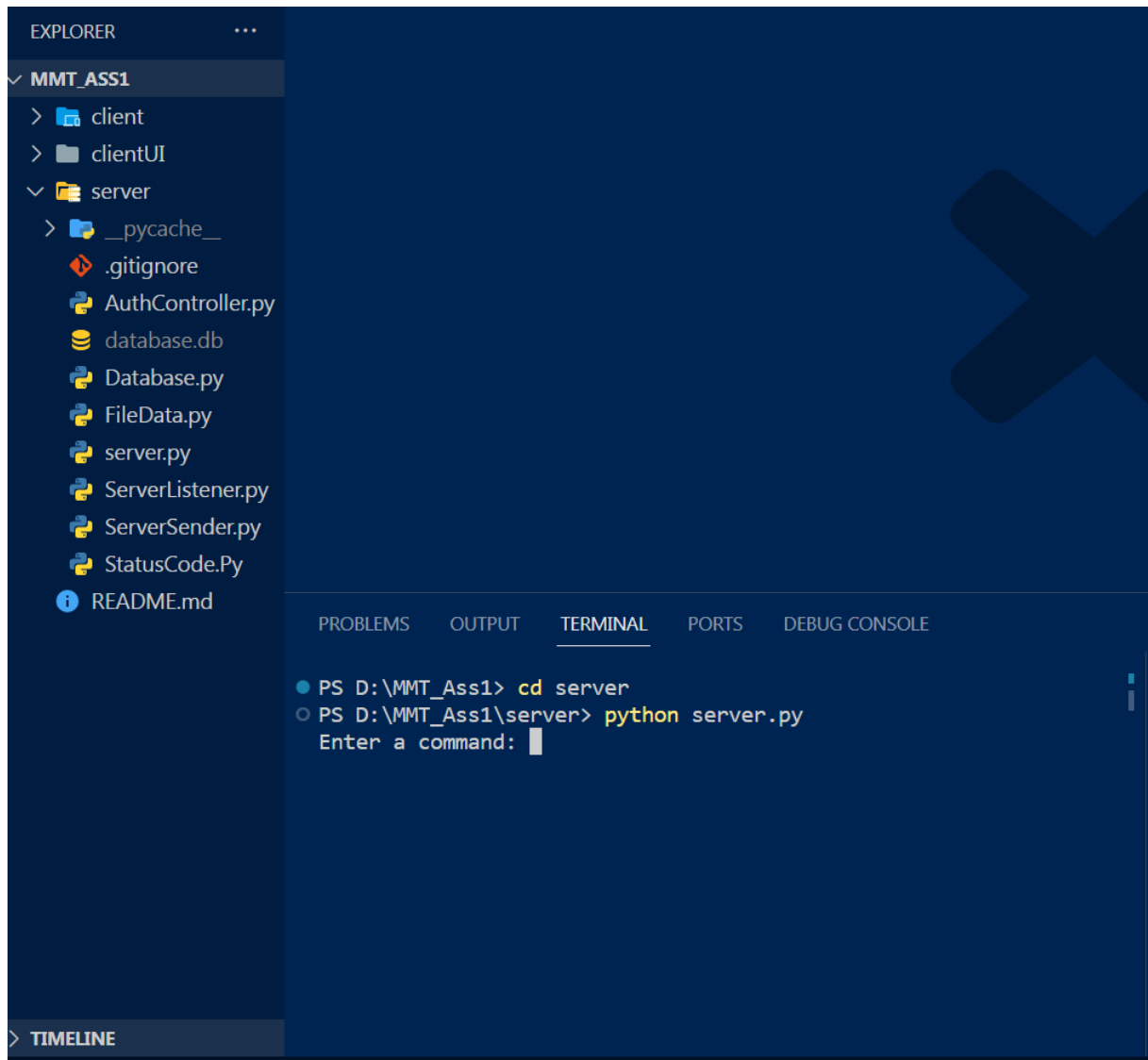
7. Tutorial - Guides

7.1. Server

Người dùng có thể start server bằng cách chạy file server.py



Hình 21: Start Server

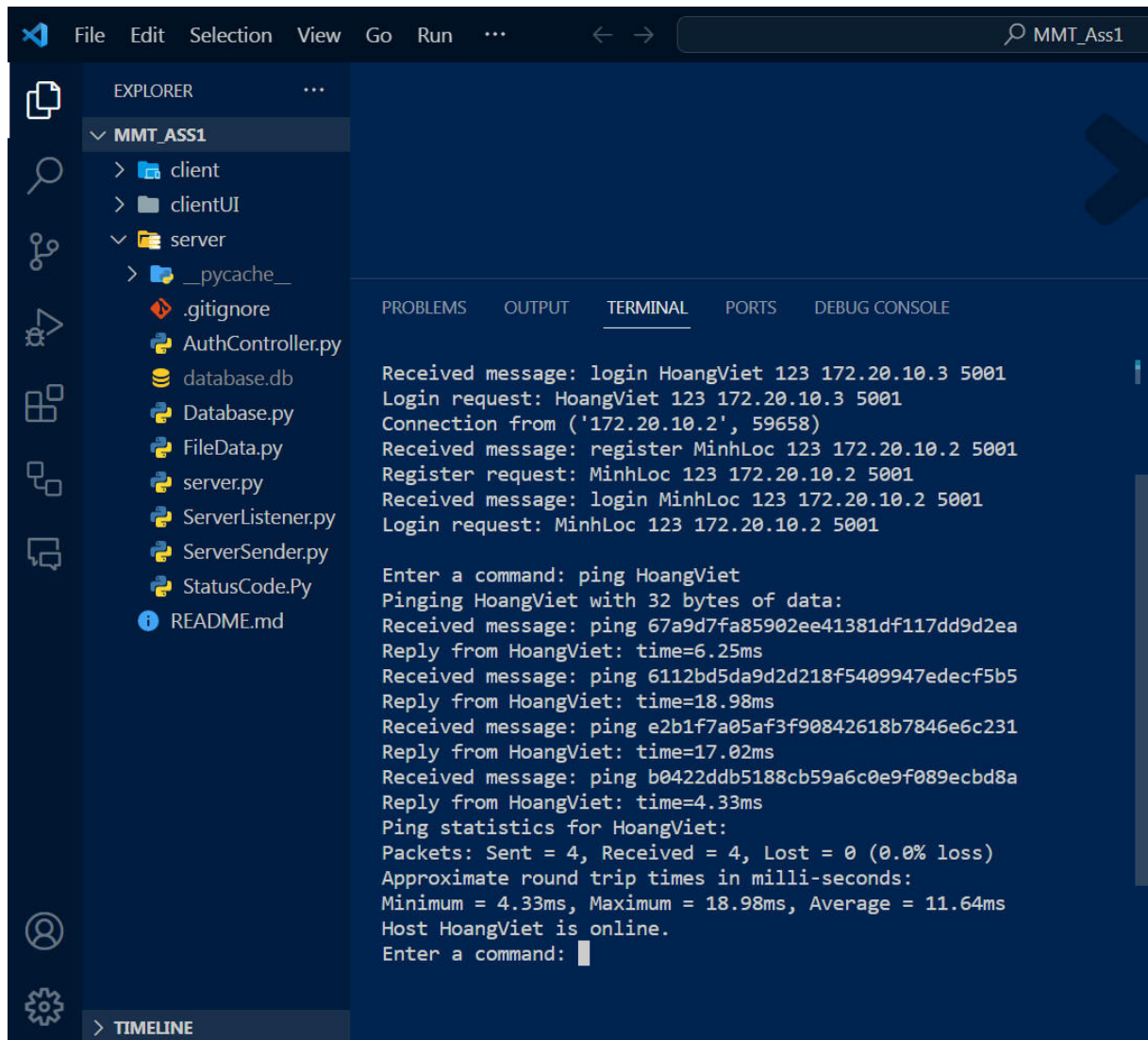


Hình 22: Start Server

7.1.1. ping

Server có thể ping tới các Peer khác bằng cách nhập ping <>

Nếu ping thành công, sẽ hiển thị thông báo client đang hoạt động



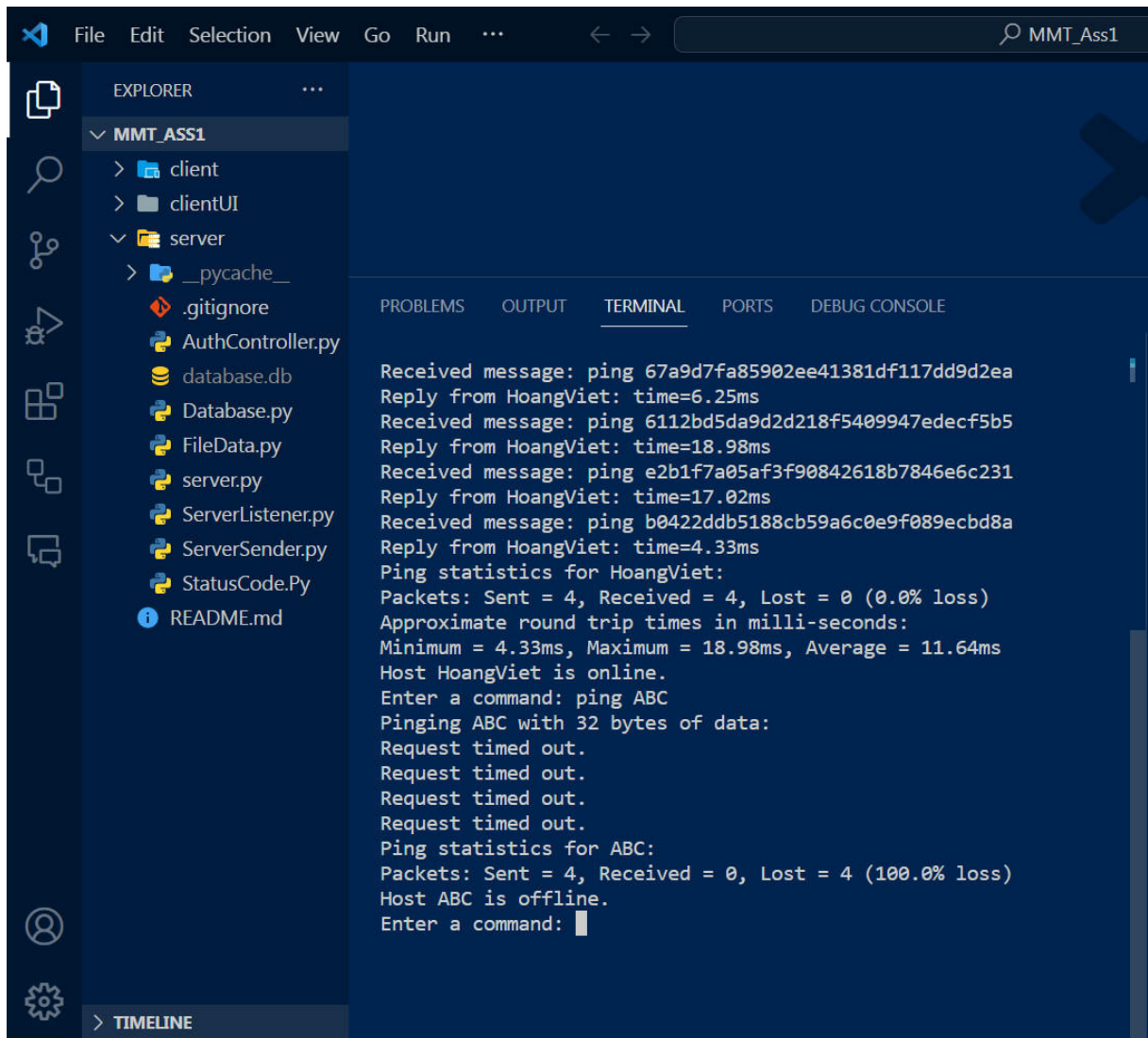
The screenshot shows the Visual Studio Code (VS Code) interface with a project named 'MMT_Ass1'. The Explorer sidebar on the left displays the project structure, including folders 'client' and 'clientUI', and a 'server' folder containing files like '__pycache__', '.gitignore', 'AuthController.py', 'database.db', 'Database.py', 'FileData.py', 'server.py', 'ServerListener.py', 'ServerSender.py', 'StatusCode.py', and 'README.md'. The bottom panel is the 'TERMINAL' tab, which shows the following output:

```
Received message: login HoangViet 123 172.20.10.3 5001
Login request: HoangViet 123 172.20.10.3 5001
Connection from ('172.20.10.2', 59658)
Received message: register MinhLoc 123 172.20.10.2 5001
Register request: MinhLoc 123 172.20.10.2 5001
Received message: login MinhLoc 123 172.20.10.2 5001
Login request: MinhLoc 123 172.20.10.2 5001

Enter a command: ping HoangViet
Pinging HoangViet with 32 bytes of data:
Received message: ping 67a9d7fa85902ee41381df117dd9d2ea
Reply from HoangViet: time=6.25ms
Received message: ping 6112bd5da9d2d218f5409947edecf5b5
Reply from HoangViet: time=18.98ms
Received message: ping e2b1f7a05af3f90842618b7846e6c231
Reply from HoangViet: time=17.02ms
Received message: ping b0422ddb5188cb59a6c0e9f089ecbd8a
Reply from HoangViet: time=4.33ms
Ping statistics for HoangViet:
Packets: Sent = 4, Received = 4, Lost = 0 (0.0% loss)
Approximate round trip times in milli-seconds:
Minimum = 4.33ms, Maximum = 18.98ms, Average = 11.64ms
Host HoangViet is online.
Enter a command: 
```

Hình 23: Ping

Nếu ping thất bại, sẽ hiển thị thông báo client không hoạt động



```
File Edit Selection View Go Run ... MMT_Ass1

EXPLORER
  MMT_ASS1
    client
    clientUI
    server
    __pycache__
    .gitignore
    AuthController.py
    database.db
    Database.py
    FileData.py
    server.py
    ServerListener.py
    ServerSender.py
    StatusCode.py
    README.md

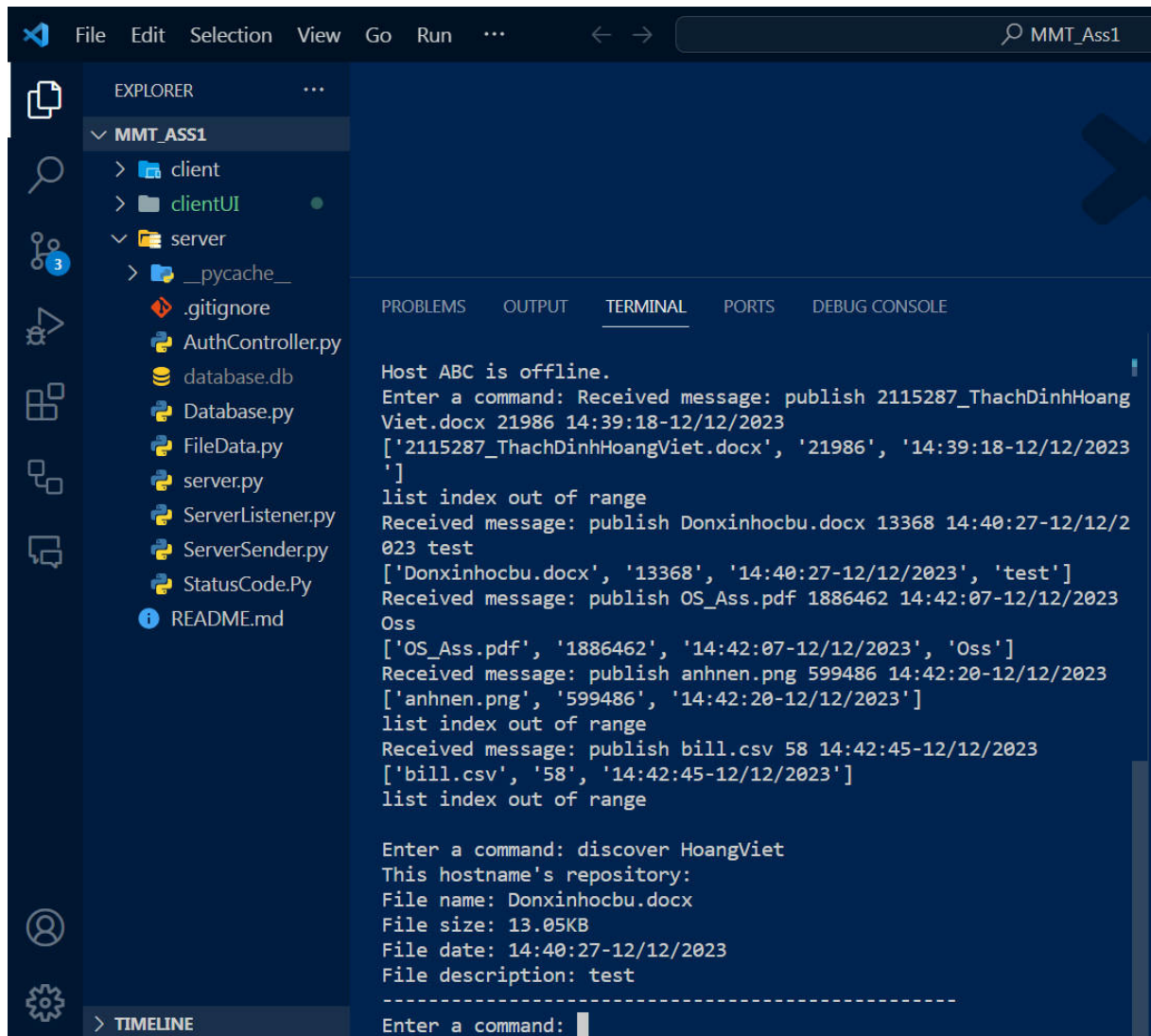
PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE

Received message: ping 67a9d7fa85902ee41381df117dd9d2ea
Reply from HoangViet: time=6.25ms
Received message: ping 6112bd5da9d2d218f5409947edecf5b5
Reply from HoangViet: time=18.98ms
Received message: ping e2b1f7a05af3f90842618b7846e6c231
Reply from HoangViet: time=17.02ms
Received message: ping b0422ddb5188cb59a6c0e9f089ecbd8a
Reply from HoangViet: time=4.33ms
Ping statistics for HoangViet:
Packets: Sent = 4, Received = 4, Lost = 0 (0.0% loss)
Approximate round trip times in milli-seconds:
Minimum = 4.33ms, Maximum = 18.98ms, Average = 11.64ms
Host HoangViet is online.
Enter a command: ping ABC
Pinging ABC with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Ping statistics for ABC:
Packets: Sent = 4, Received = 0, Lost = 4 (100.0% loss)
Host ABC is offline.
Enter a command: 
```

Hình 24: Ping No Online

7.1.2. Discover

Server có thể discover các Peer khác bằng cách nhập discover



The screenshot shows the Visual Studio Code (VS Code) interface. On the left is the Explorer sidebar showing a project named 'MMT_ASS1'. The project structure includes a 'client' folder, a 'clientUI' folder, and a 'server' folder. Inside 'server', there are files like 'AuthController.py', 'database.db', 'Database.py', 'FileData.py', 'server.py', 'ServerListener.py', 'ServerSender.py', 'StatusCode.py', and a 'README.md' file. The main editor area is split into two panes. The top pane shows the 'TERMINAL' tab with the following output:

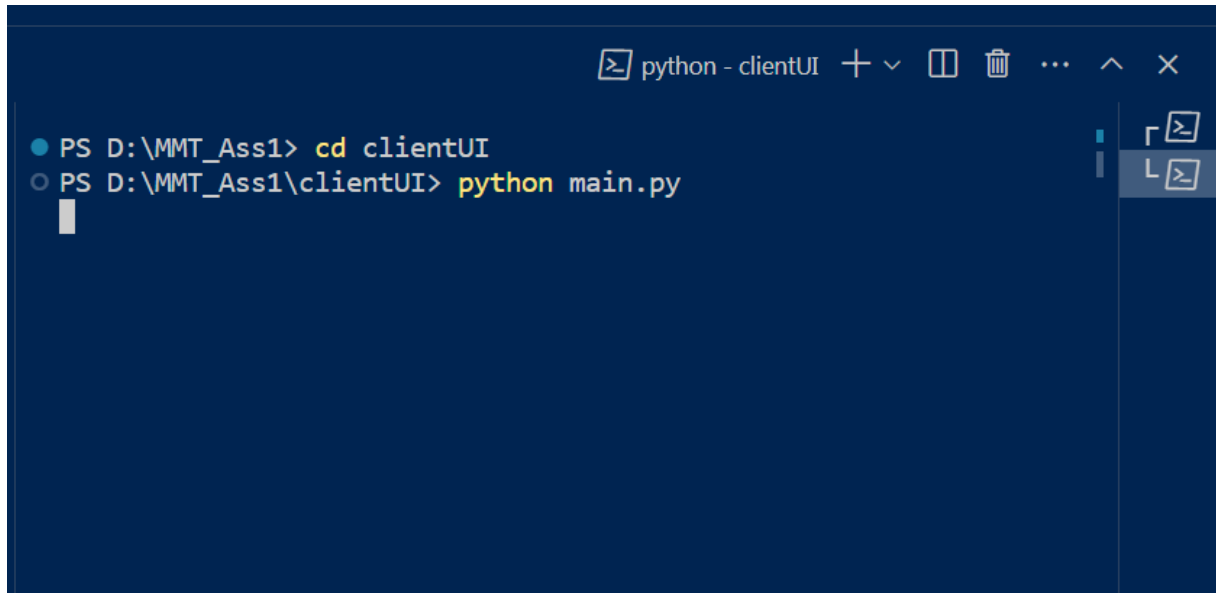
```
Host ABC is offline.  
Enter a command: Received message: publish 2115287_ThachDinhHoang  
Viet.docx 21986 14:39:18-12/12/2023  
['2115287_ThachDinhHoangViet.docx', '21986', '14:39:18-12/12/2023  
']  
list index out of range  
Received message: publish Donxinhocbu.docx 13368 14:40:27-12/12/2  
023 test  
['Donxinhocbu.docx', '13368', '14:40:27-12/12/2023', 'test']  
Received message: publish OS_Ass.pdf 1886462 14:42:07-12/12/2023  
Oss  
['OS_Ass.pdf', '1886462', '14:42:07-12/12/2023', 'Oss']  
Received message: publish anhnen.png 599486 14:42:20-12/12/2023  
['anhnen.png', '599486', '14:42:20-12/12/2023']  
list index out of range  
Received message: publish bill.csv 58 14:42:45-12/12/2023  
['bill.csv', '58', '14:42:45-12/12/2023']  
list index out of range  
  
Enter a command: discover HoangViet  
This hostname's repository:  
File name: Donxinhocbu.docx  
File size: 13.05KB  
File date: 14:40:27-12/12/2023  
File description: test  
-----  
Enter a command: 
```

The bottom pane of the editor shows the 'TIMELINE' tab, which is currently empty.

Hình 25: Discover

7.2. Client

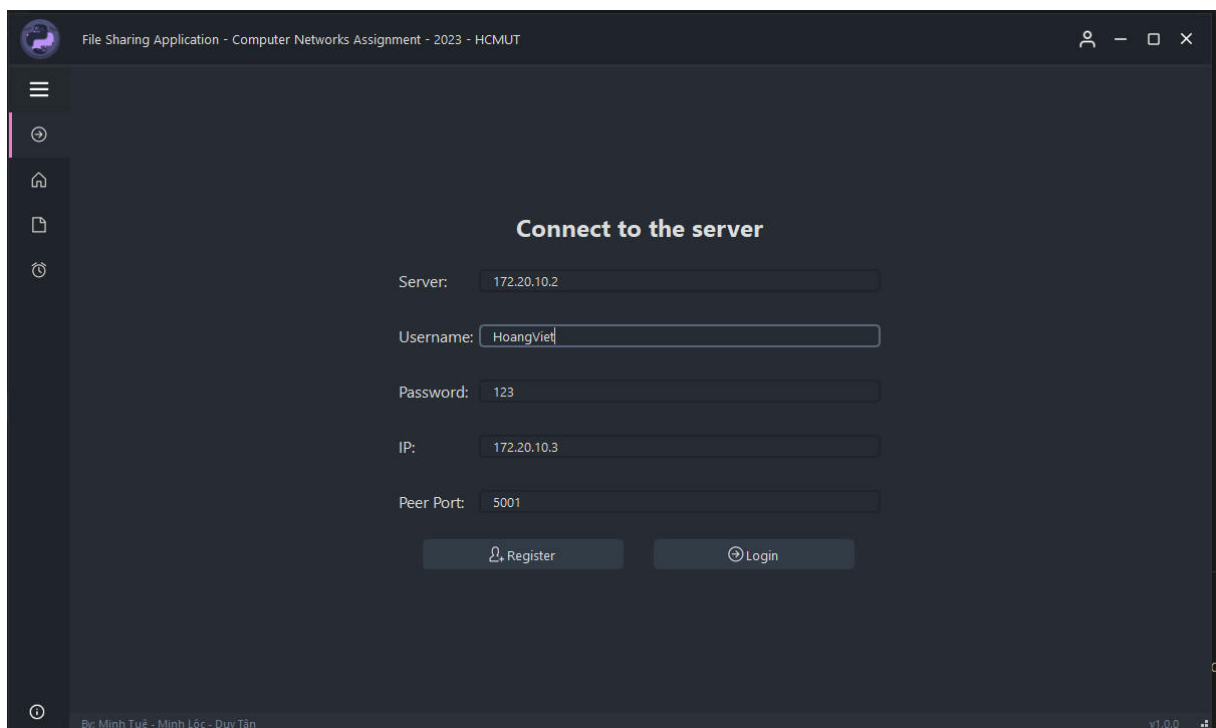
Người dùng có thể start client bằng cách chạy file main.py



Hình 26: Start Client

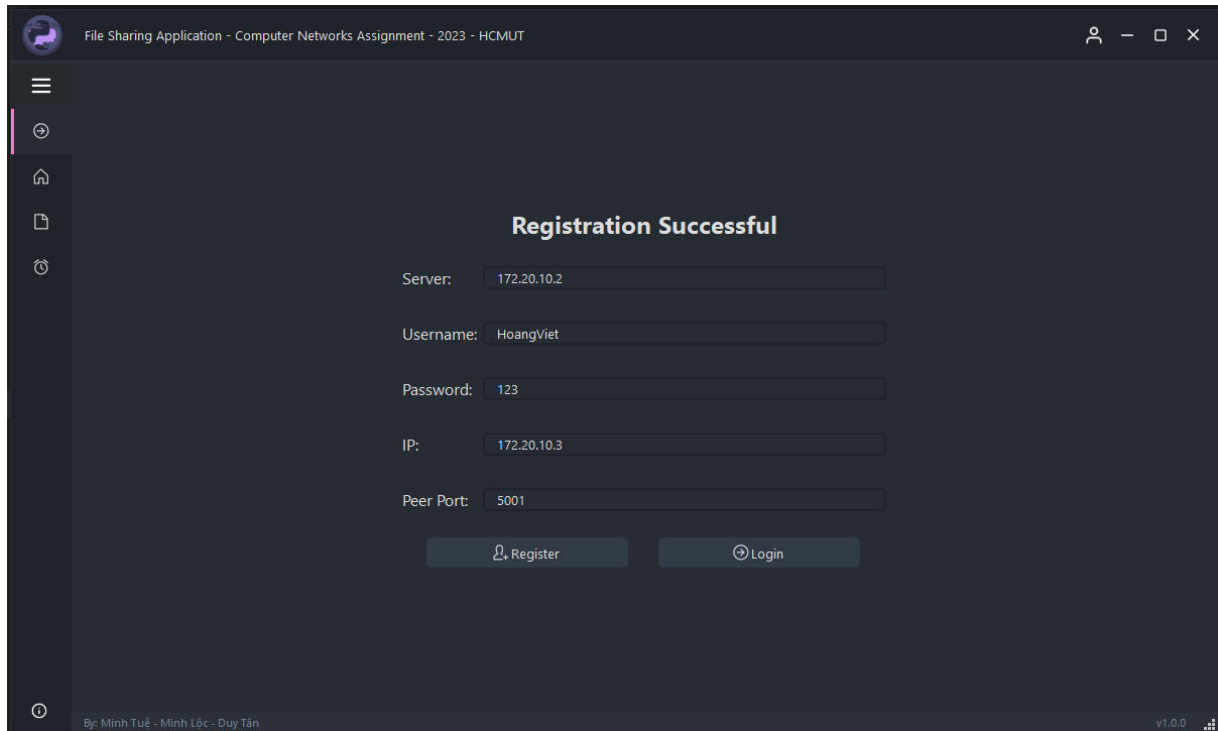
7.2.1. Register

Người dùng sẽ nhập IP Server, username, password, địa chỉ IP Peer, địa chỉ Peer Port để đăng ký tài khoản trên hệ thống.



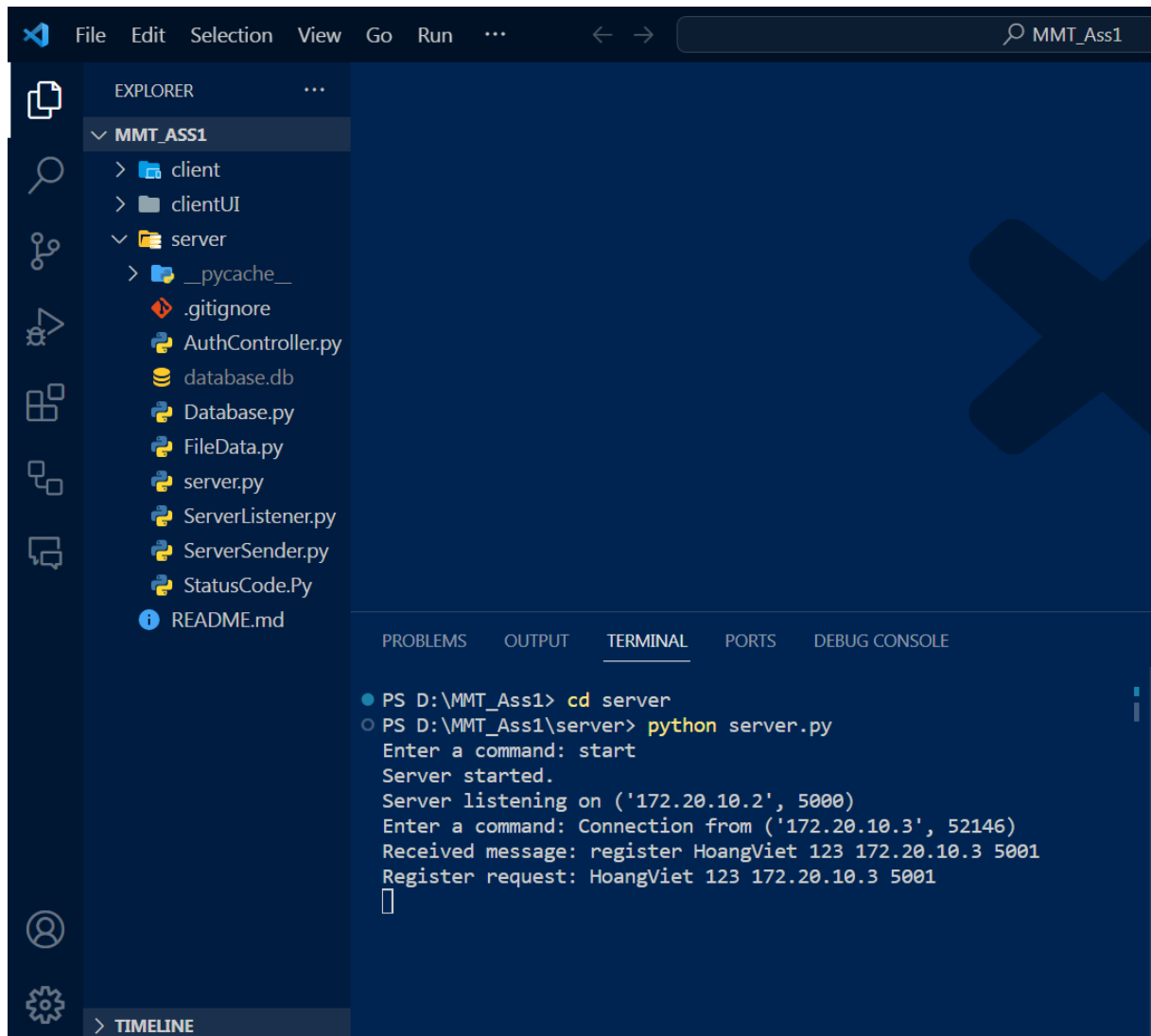
Hình 27: Client Register

Giao diện sẽ hiển thị thông báo đăng ký thành công



Hình 28: Client Register Success

Phía Server sẽ hiển thị thông báo đăng ký thành công



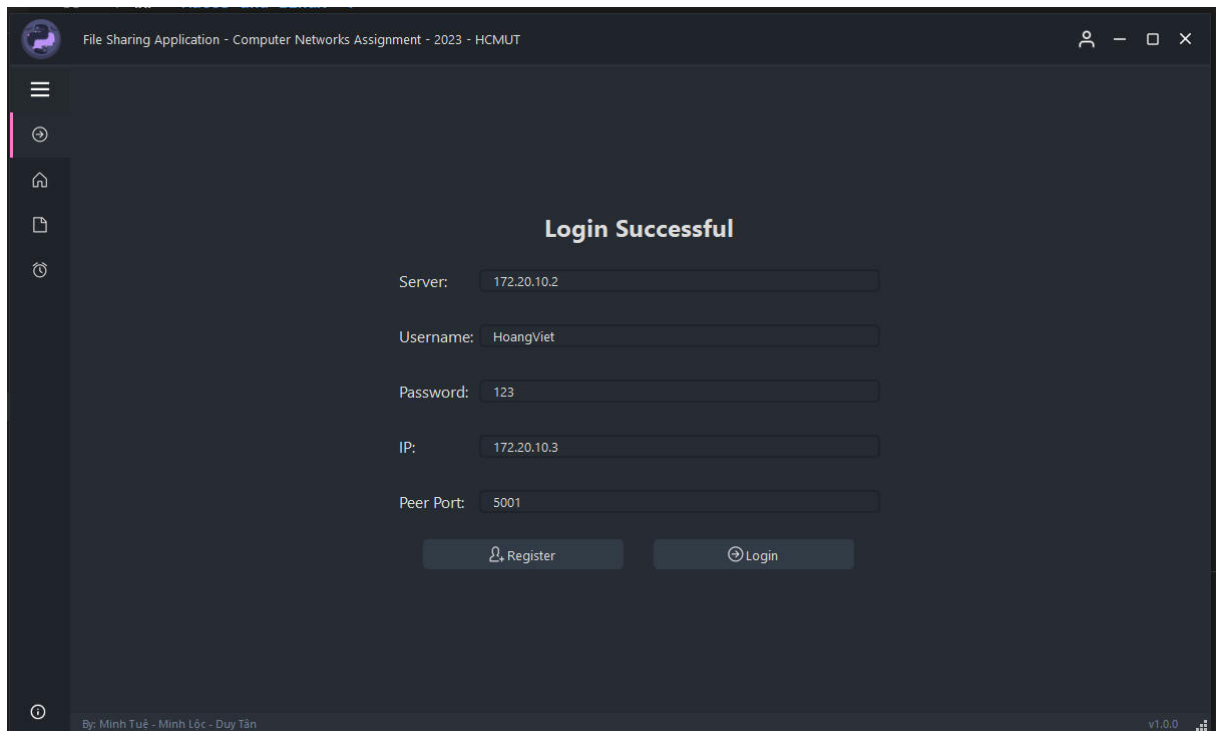
The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left displaying the project structure for 'MMT_Ass1'. The project contains folders 'client' and 'clientUI', and a 'server' folder which includes subfolders '__pycache__' and files: '.gitignore', 'AuthController.py', 'database.db', 'Database.py', 'FileData.py', 'server.py', 'ServerListener.py', 'ServerSender.py', 'StatusCode.py', and 'README.md'. The bottom panel shows the 'TERMINAL' tab with the following output:

```
PS D:\MMT_Ass1> cd server
PS D:\MMT_Ass1\server> python server.py
Enter a command: start
Server started.
Server listening on ('172.20.10.2', 5000)
Enter a command: Connection from ('172.20.10.3', 52146)
Received message: register HoangViet 123 172.20.10.3 5001
Register request: HoangViet 123 172.20.10.3 5001
█
```

Hình 29: Server Register Success

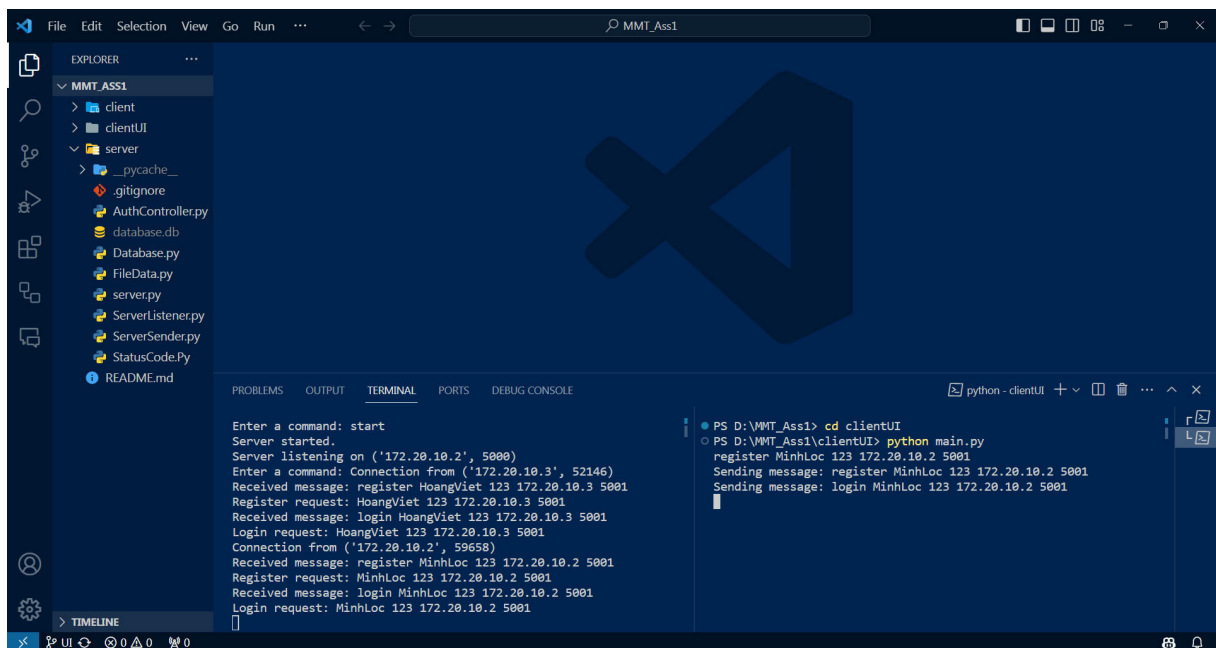
7.2.2. Login

Người dùng sẽ nhập IP Server, username, password, địa chỉ IP Peer, địa chỉ Peer Port để đăng nhập vào hệ thống.



Hình 30: Client Login

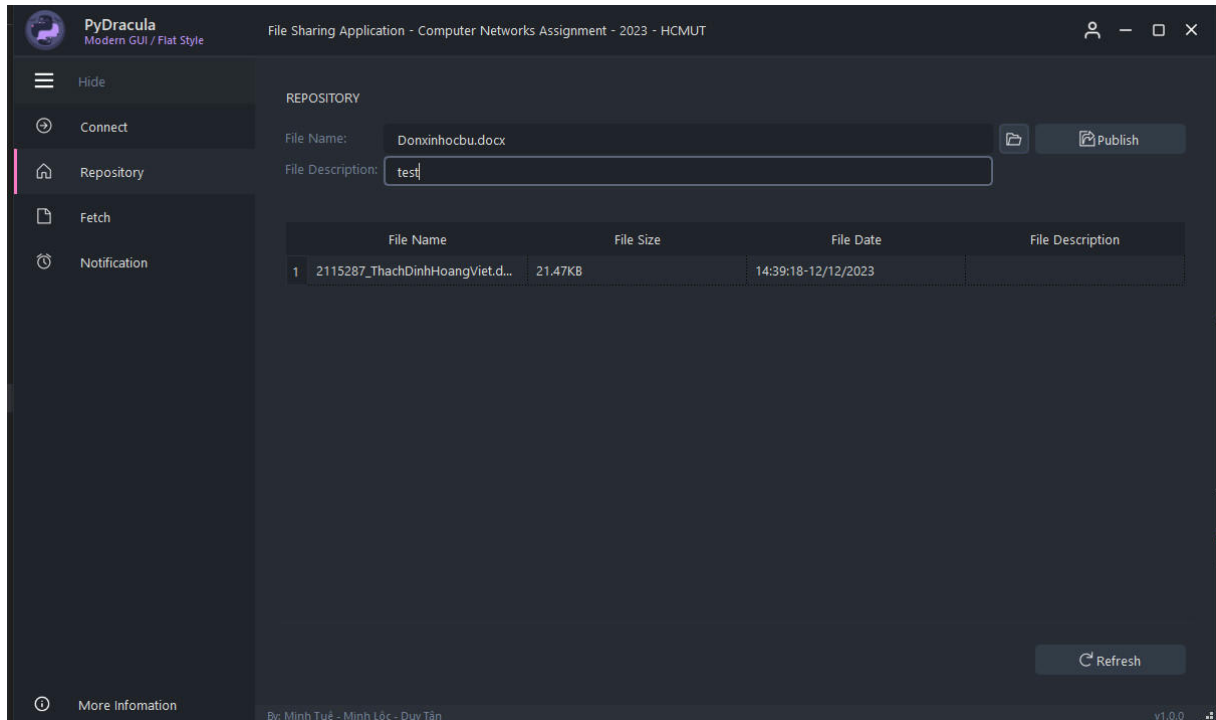
Phía Server sẽ hiển thị thông báo đăng nhập thành công



Hình 31: Login Success

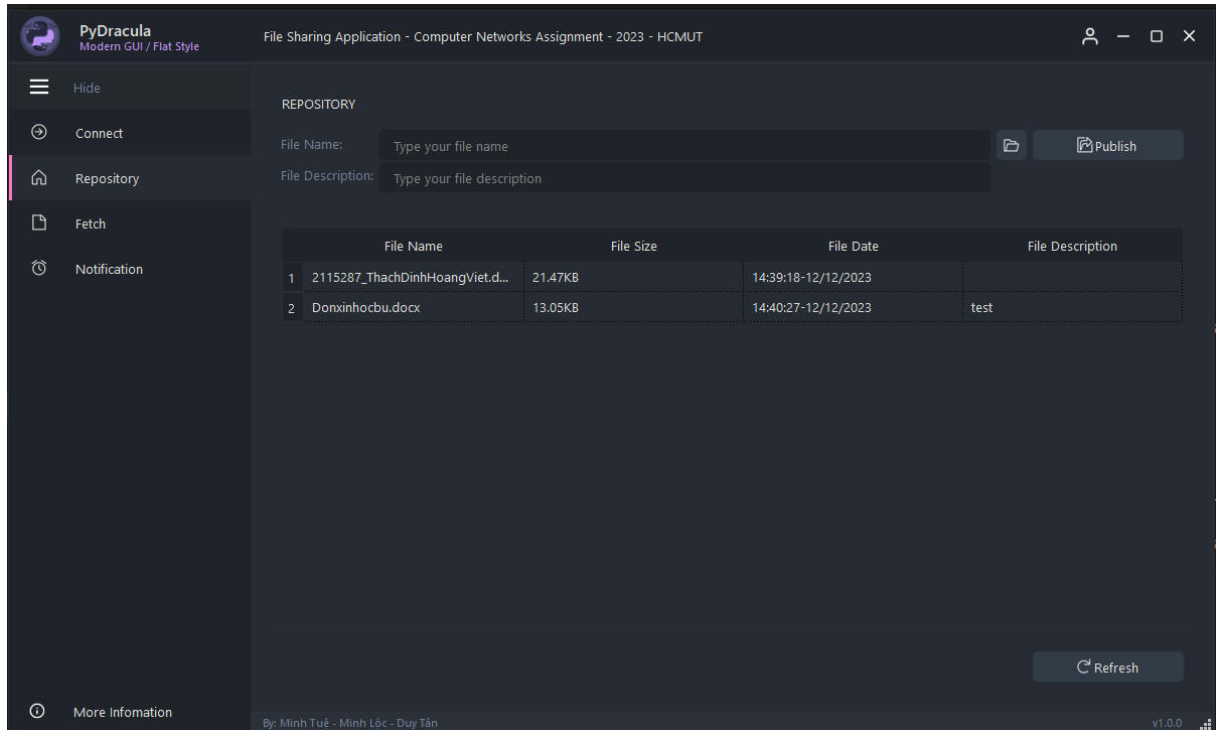
7.2.3. Publish

Người dùng sẽ nhập đường dẫn file, mô tả file và nhấn nút Publish để Publish file lên hệ thống.



Hình 32: Publish

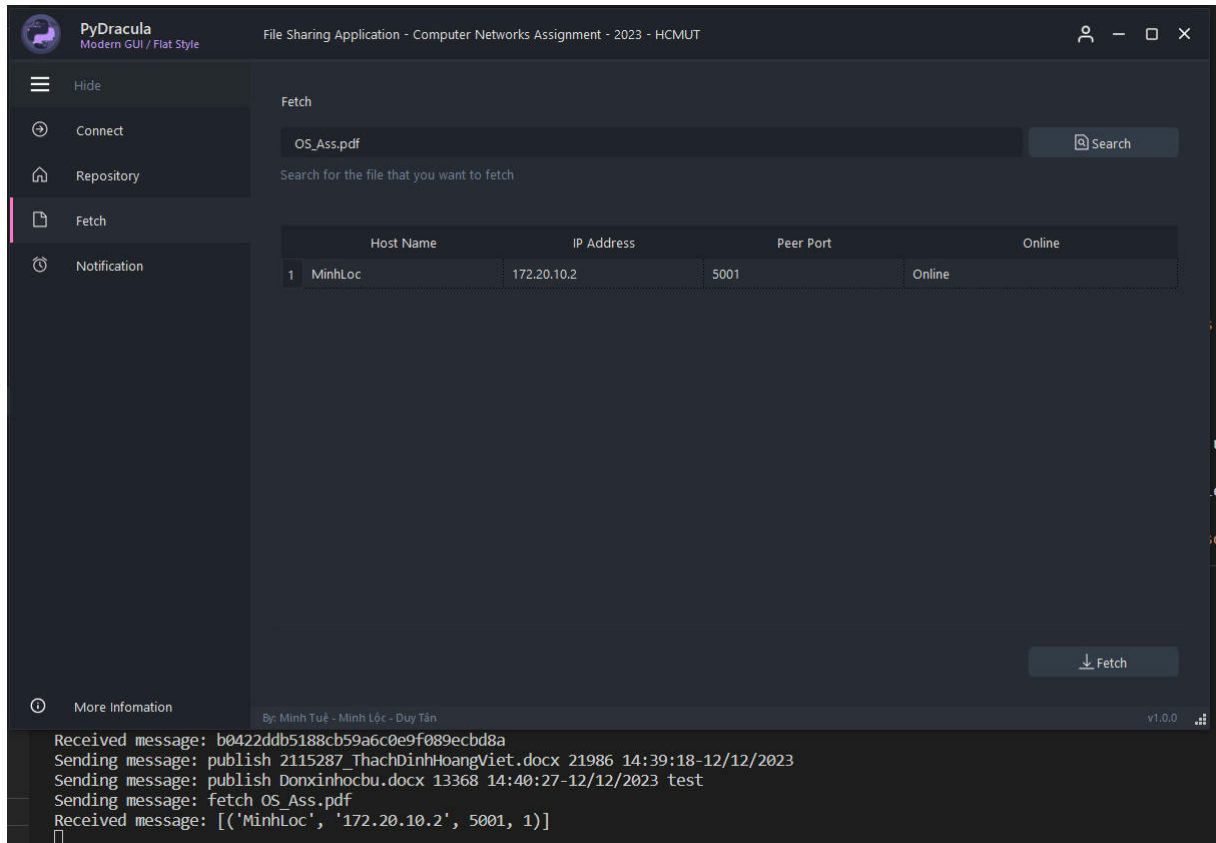
Giao diện sẽ hiển thị file đã Publish vào danh sách



Hình 33: Publish Success

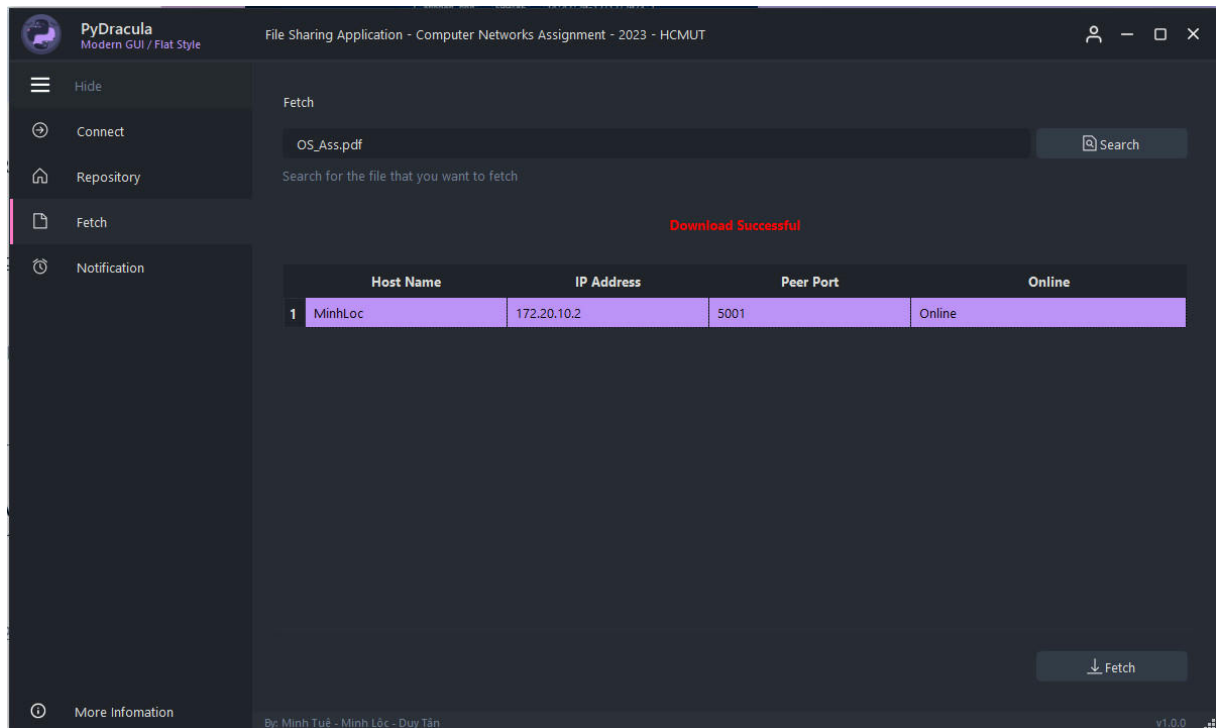
7.2.4. Fetch

Người dùng sẽ nhập tên file cần Fetch và nhấn nút Search để tìm kiếm file.



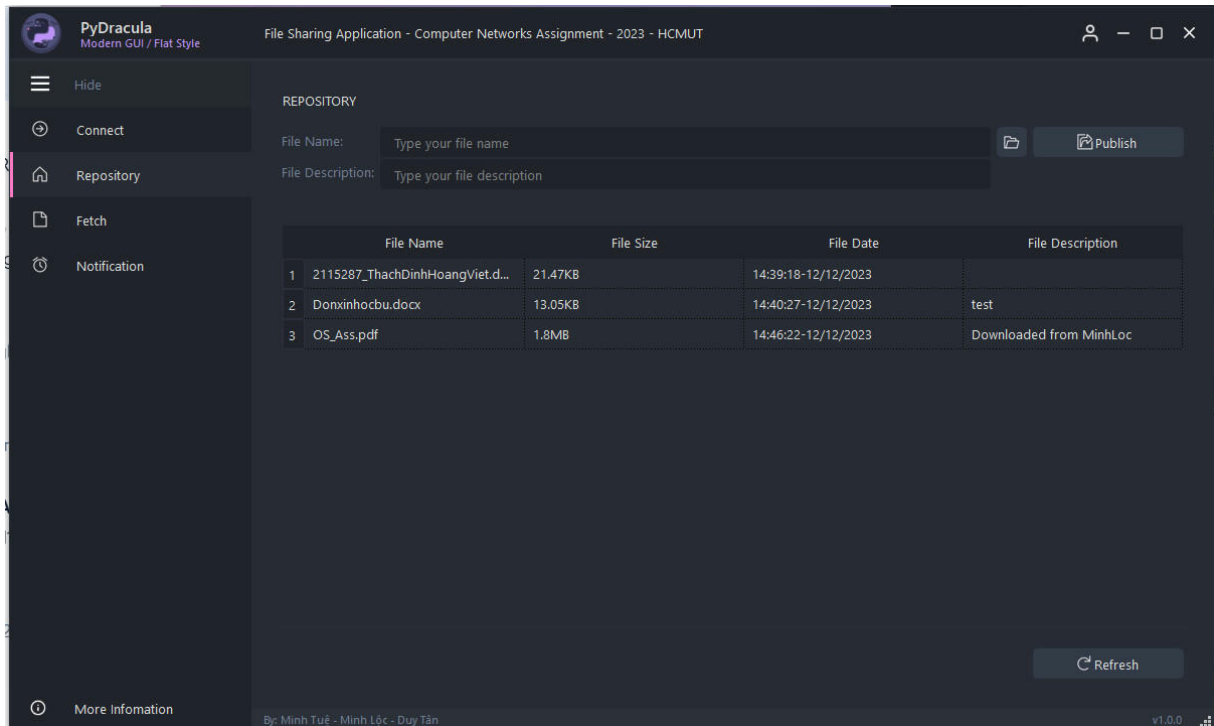
Hình 34: Fetch

Sau khi chọn file cần Fetch và nhấn nút Fetch, file sẽ được Fetch về.



Hình 35: Fetch Success

Lưới danh sách file sẽ hiển thị file đã Fetch về.



Hình 36: Fetch Success



8. Error Handling

Tài liệu tham khảo

- [1] Slide môn học Mạng máy tính CO3093.
- [2] ByteByteGo. 8 Popular Network Protocols. Truy cập từ: <https://substackcdn.com>
- [3] Md Rafiul, KabirBhagawat Baanav Yedla RaviBhagawat Baanav Yedla Ravi, Sandip Ray. A Virtual Prototyping Platform for Exploration of Vehicular Electronics. ResearchGate. Truy cập từ: https://www.researchgate.net/publication/370038185_A_Virtual_Prototyping_Platform_for_Exploration_of_Vehicular_Electronics