

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. Phân tích yêu cầu .....	6
1.1. Functional requirements .....	6
1.1.1. Client Functions .....	6
1.1.2. Server Functions .....	7
1.2. Non-functional requirements .....	7
1.3. Phân tích kiến trúc .....	7
1.3.1. Kiến trúc Peer-to-Peer (P2P) .....	7
1.3.2. Kiến trúc client-server .....	7
2. Giới thiệu Protocol .....	9
2.1. HyperText Trànner Protocol .....	9
2.2. Transmission Control Protocol .....	10
3. Socket programming with Python .....	10
4. Application design .....	12
4.1. Architecture design .....	12
4.2. Flow protocol design .....	13
4.2.1. Register .....	13
4.2.2. Login .....	13
4.2.3. Ping .....	13
4.2.4. Discover .....	13
4.2.5. Publish .....	14
4.2.6. Fetch .....	14
4.3. Activity diagram .....	15
4.3.1. Register .....	15
4.3.2. Login .....	15
4.3.3. Ping .....	16
4.3.4. Discover .....	17
4.3.5. Publish .....	18
4.3.6. Fetch .....	18
4.4. Status Code .....	19
5. Design UI .....	20
5.1. Register and Login .....	20
5.2. Publish Page .....	21
5.3. Fetch Page .....	22
6. Tutorial - Guides .....	24

6.1. Server .....	24
6.1.1. ping .....	25
6.1.2. Discover .....	27
6.2. Client .....	29
6.2.1. Register .....	29
6.2.2. Login .....	31
6.2.3. Publish .....	32
6.2.4. Fetch .....	33
7. Error Handling .....	36
7.1. Server not running .....	36
7.2. Connection error .....	36
7.3. Login & Register: Wrong input format .....	38
7.4. Register: User already exist .....	38
7.5. Login: User not found .....	39
7.6. Login: Wrong password .....	40
7.7. Login: User already online .....	40
7.8. Publish: File not found .....	42
7.9. Publish: File already published .....	42
7.10. Fetch: File not found .....	44
7.11. Fetch: File already fetched .....	44

## 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: Trao đổi status code giữa client và server .....	19
Hình 18: UI Register and Login .....	20
Hình 19: UI Publish File .....	21
Hình 20: UI Fetch File .....	22
Hình 21: UI Profile .....	23
Hình 22: Start Server .....	24
Hình 23: Start Server .....	25
Hình 24: Ping .....	26
Hình 25: Ping No Online .....	27
Hình 26: Discover .....	28
Hình 27: Start Client .....	29
Hình 28: Client Register .....	29
Hình 29: Client Register Success .....	30
Hình 30: Server Register Success .....	31
Hình 31: Client Login .....	32
Hình 32: Login Success .....	32
Hình 33: Publish .....	33
Hình 34: Publish Success .....	33
Hình 35: Fetch .....	34
Hình 36: Fetch Success .....	34
Hình 37: Fetch Success .....	35
Hình 38: Server not running .....	36
Hình 39: Connection error .....	37
Hình 40: Wrong input format .....	38
Hình 41: User already exist .....	39
Hình 42: User not found .....	39
Hình 43: Wrong password .....	40
Hình 44: User already online .....	41
Hình 45: File not found .....	42
Hình 46: File already published .....	43
Hình 47: File not found .....	44
Hình 48: File already fetched .....	45



## Danh mục bảng biểu

Bảng 1: Các status code được định nghĩa .....	19
---	----

## 1. Phân tích yêu cầu

### 1.1. 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.

#### 1.1.1. 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.

### 1.1.2. 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.

## 1.2. 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.

## 1.3. Phân tích kiến trúc

### 1.3.1. 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.

### 1.3.2. Kiến trúc client-server

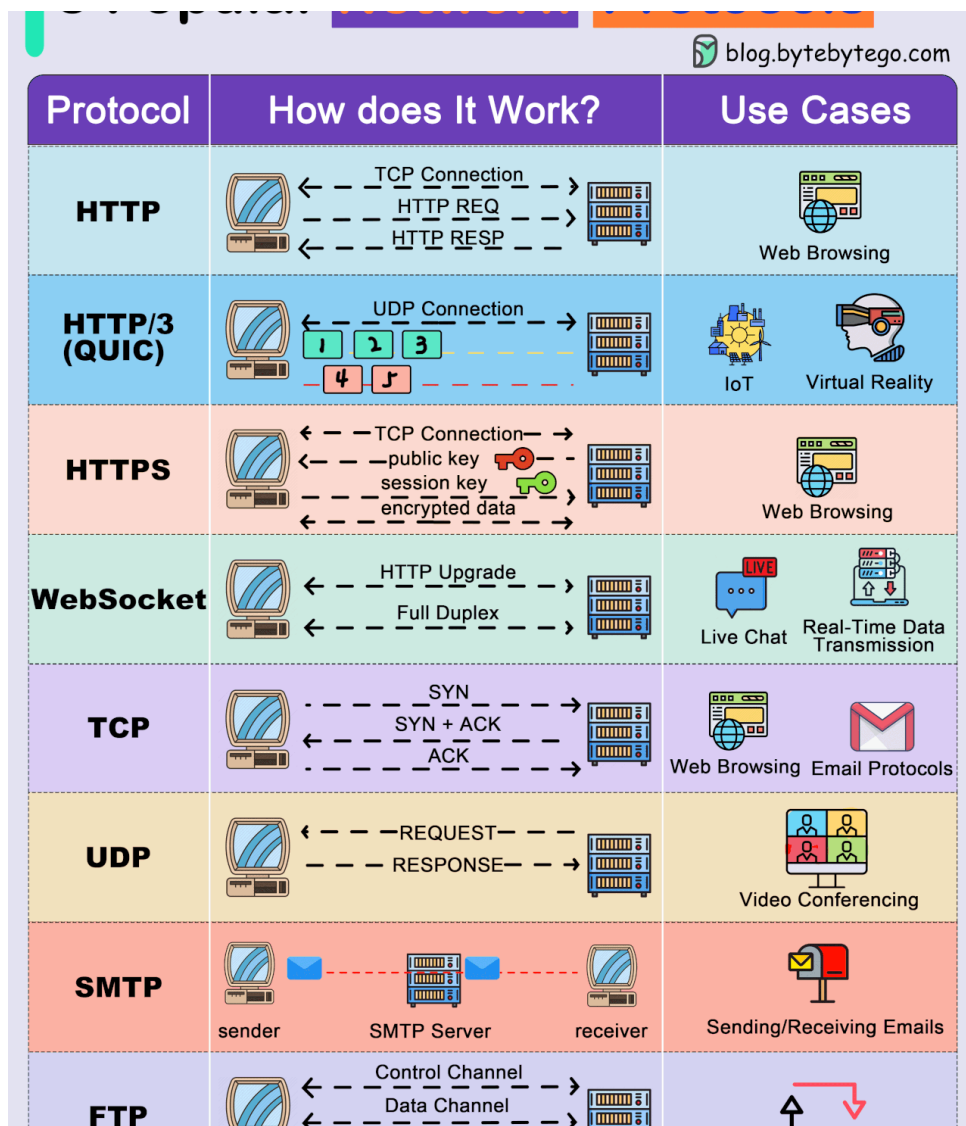
- Kiến trúc client-server (còn được gọi là mô hình client-server) là một kiến trúc máy tính phổ biến được sử dụng trong việc tổ chức và quản lý các dịch vụ và tài nguyên trên mạng. Nó dựa trên sự phân chia các vai trò chính trong hệ thống thành hai phần: máy khách (client) và máy chủ (server). Hai phần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



## 2. Giới thiệu Protocol

### 2.1. HyperText Trànner 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)

## 2.2. Transmission Control Protocol

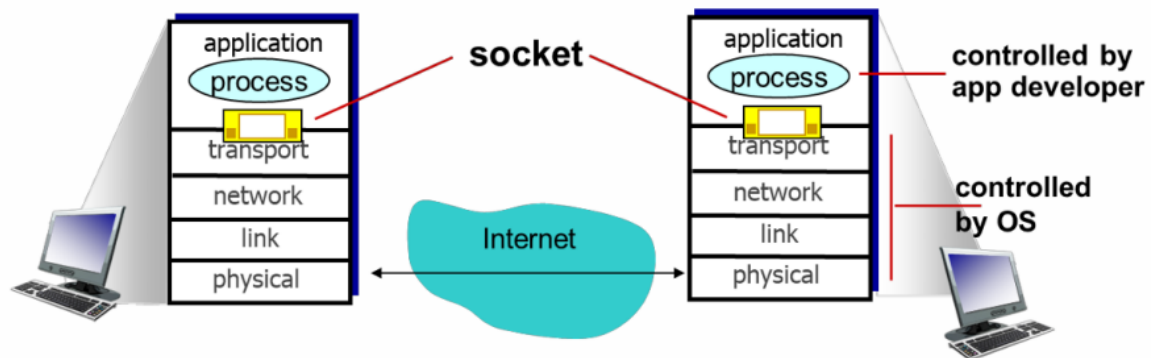
Transmission Control Protocol (TCP) là một trong các giao thức cốt lõi của bộ giao thức TCP/IP. Nhờ có TCP, các ứng dụng trên các host được nối mạng có thể tạo các kết nối với nhau, mà qua đó chúng có thể trao đổi dữ liệu hoặc các gói tin. Giao thức này đảm bảo chuyển giao dữ liệu tới nơi nhận một cách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

## 3. Socket programming with Python

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



Hình 2: Mô hình socket

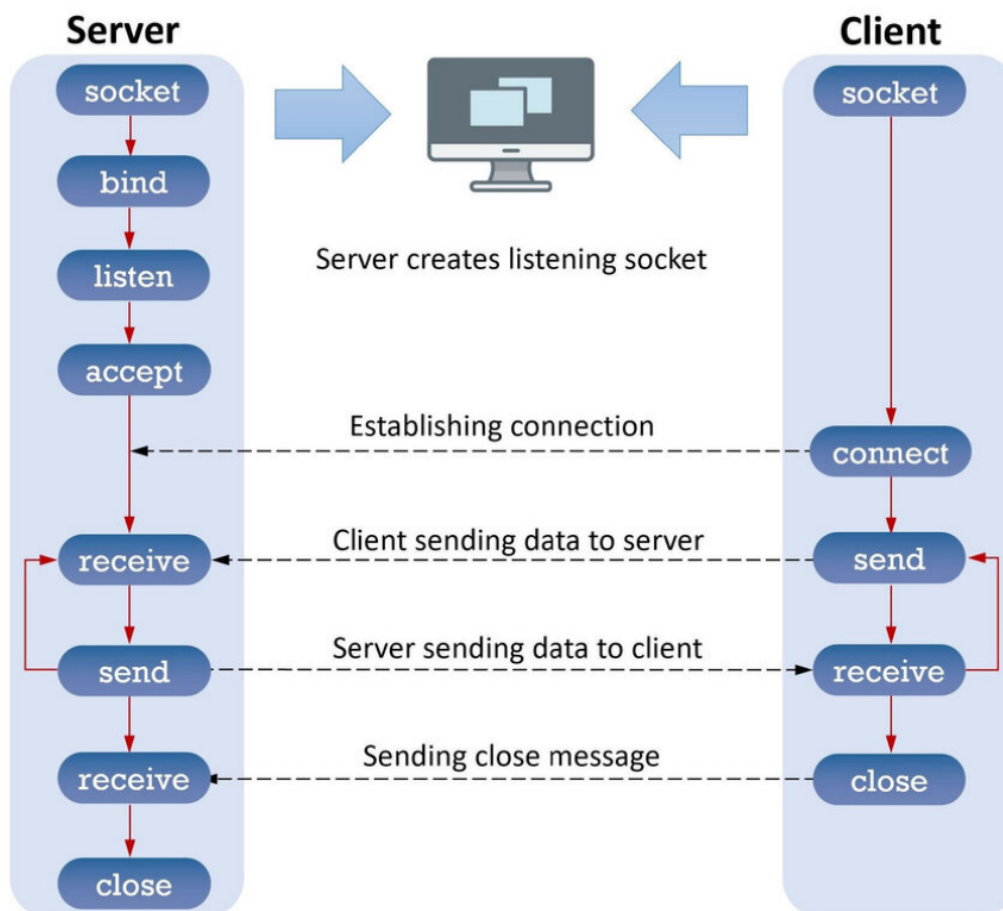
Thư viện socket của Python hỗ trợ một số API sau để thiết lập kết nối socket TCP/UDP:

- 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.

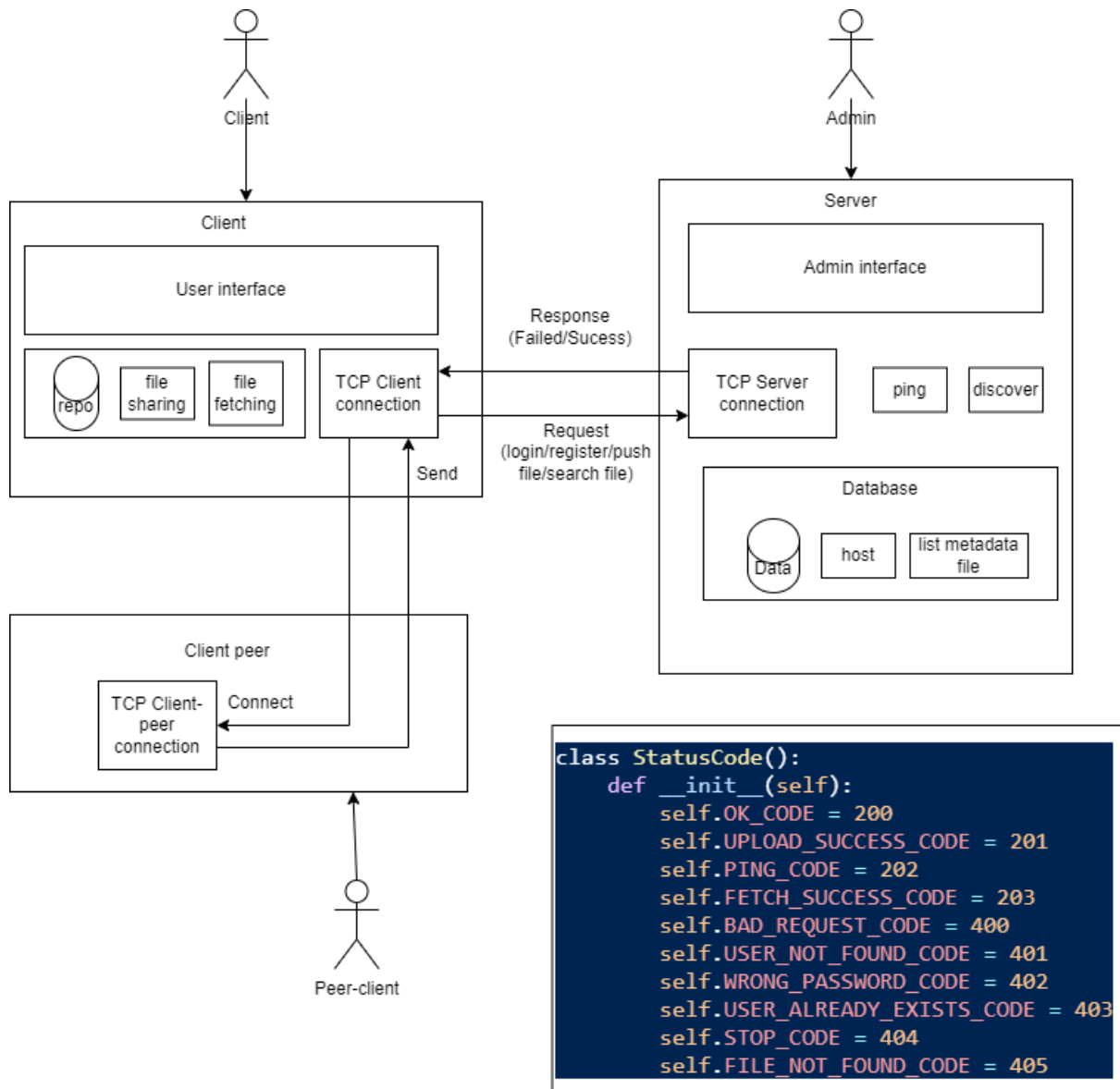
Flow của một TCP socket connection có thể được thể hiện như hình dưới đây:



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

## 4. Application design

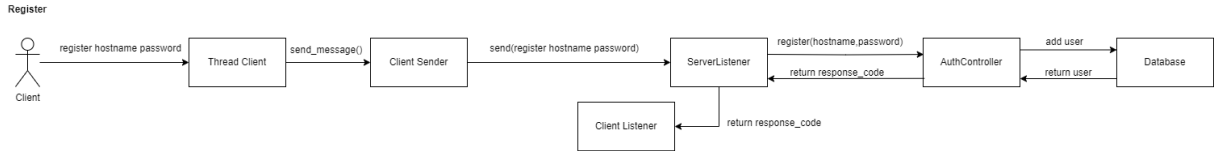
### 4.1. Architecture design



Hình 4: Kiến trúc tổng quan của hệ thống

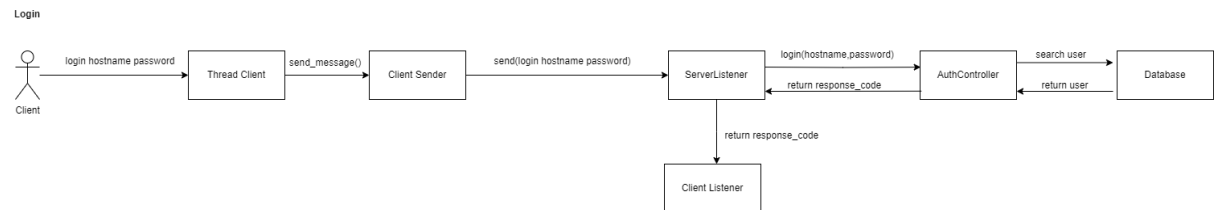
## 4.2. Flow protocol design

### 4.2.1. Register



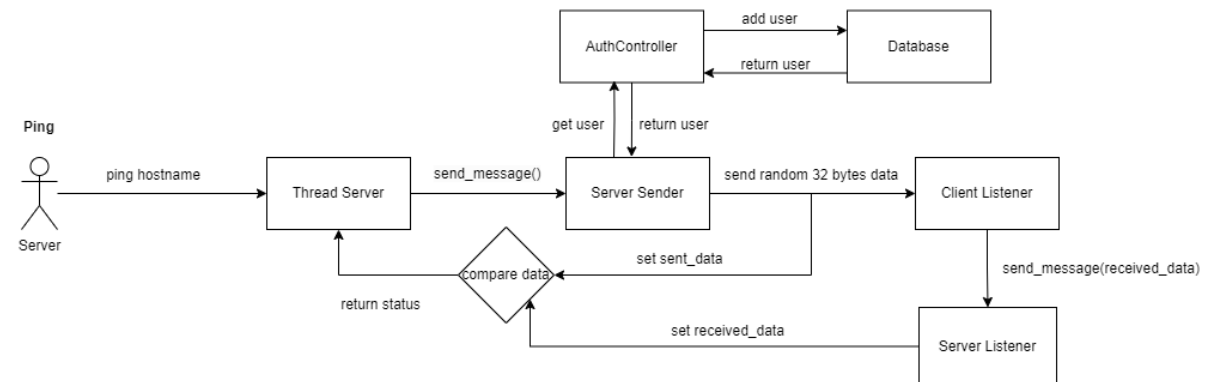
Hình 5: Sơ đồ luồng của chức năng đăng ký

### 4.2.2. Login



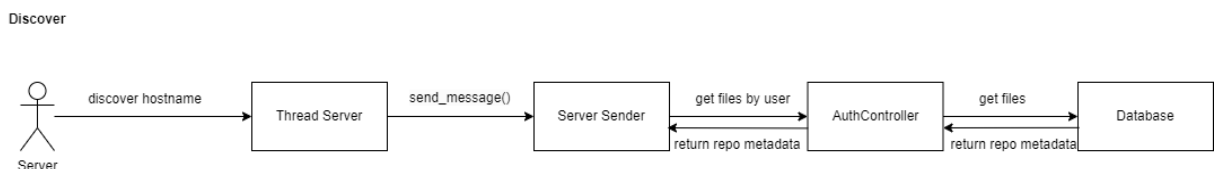
Hình 6: Sơ đồ luồng của chức năng đăng nhập

### 4.2.3. Ping



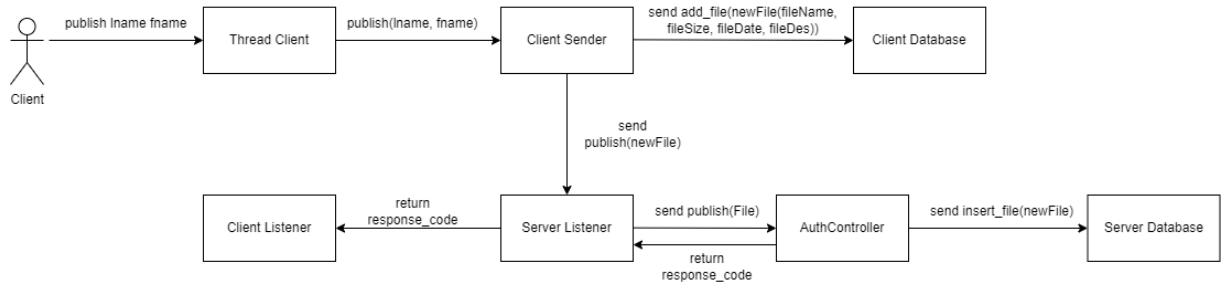
Hình 7: Sơ đồ luồng của chức năng ping

### 4.2.4. Discover



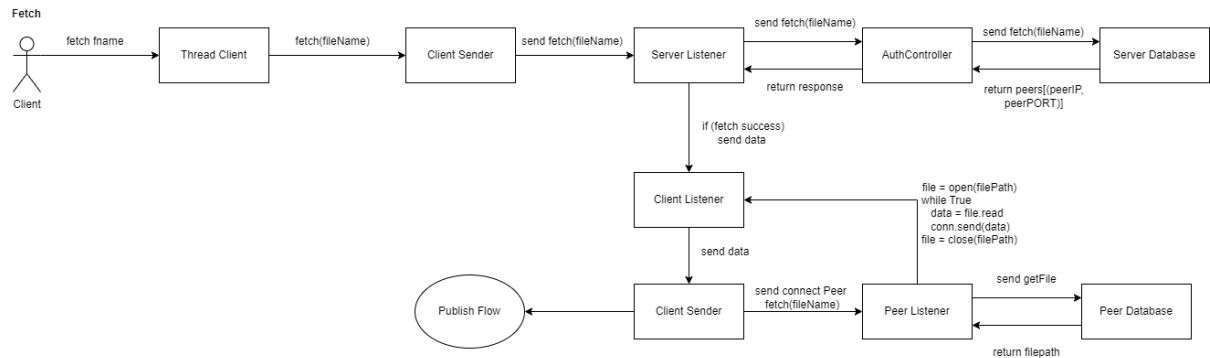
Hình 8: Sơ đồ luồng của chức năng discover

#### 4.2.5. Publish



Hình 9: Sơ đồ luồng của chức năng publish

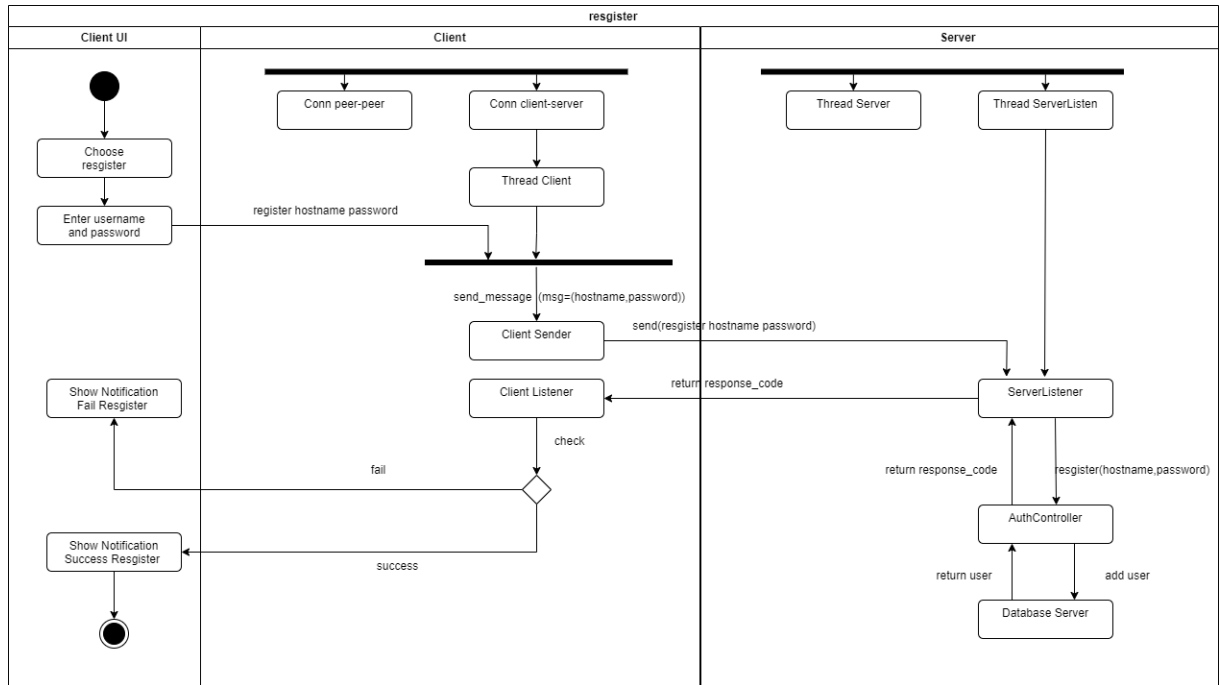
#### 4.2.6. Fetch



Hình 10: Sơ đồ luồng của chức năng fetch

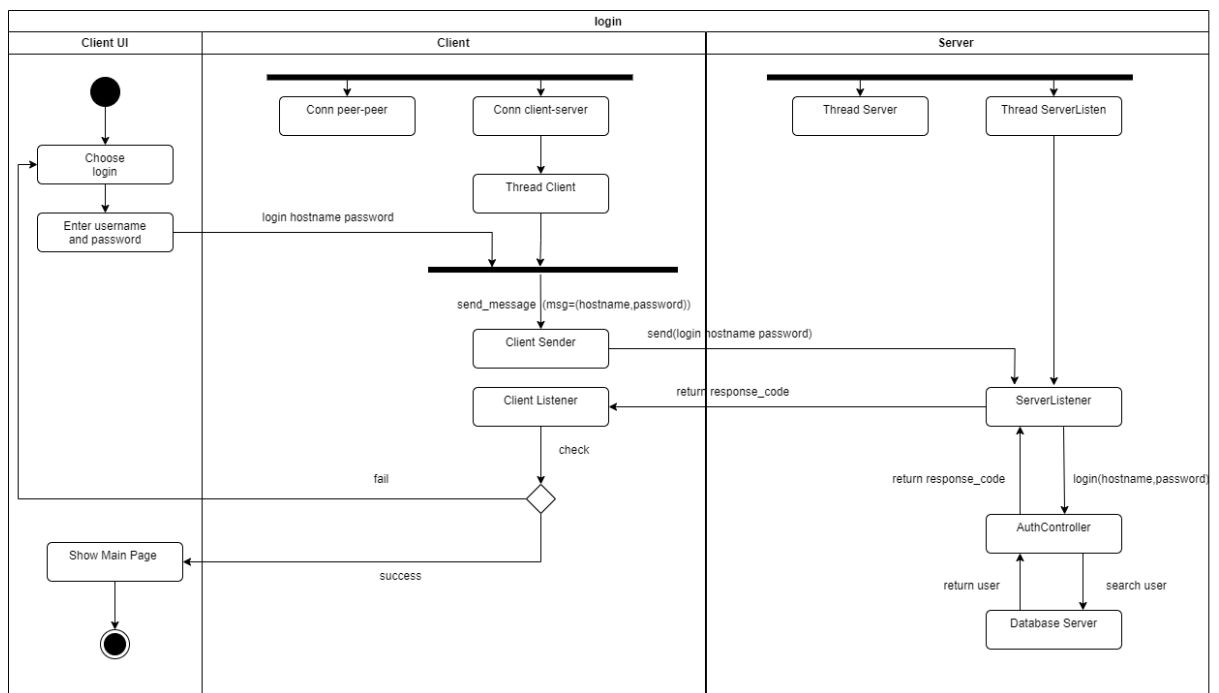
### 4.3. Activity diagram

#### 4.3.1. Register



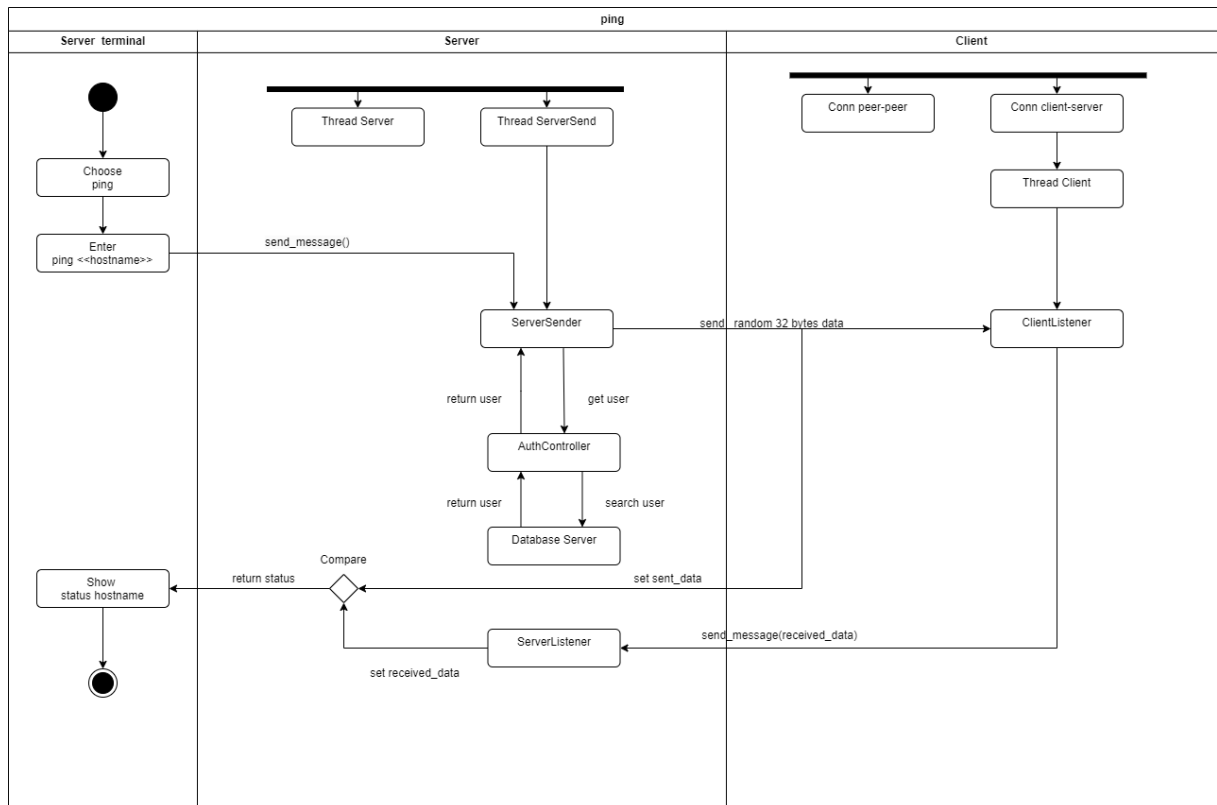
Hình 11: Sơ đồ hoạt động của chức năng đăng ký

#### 4.3.2. Login



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

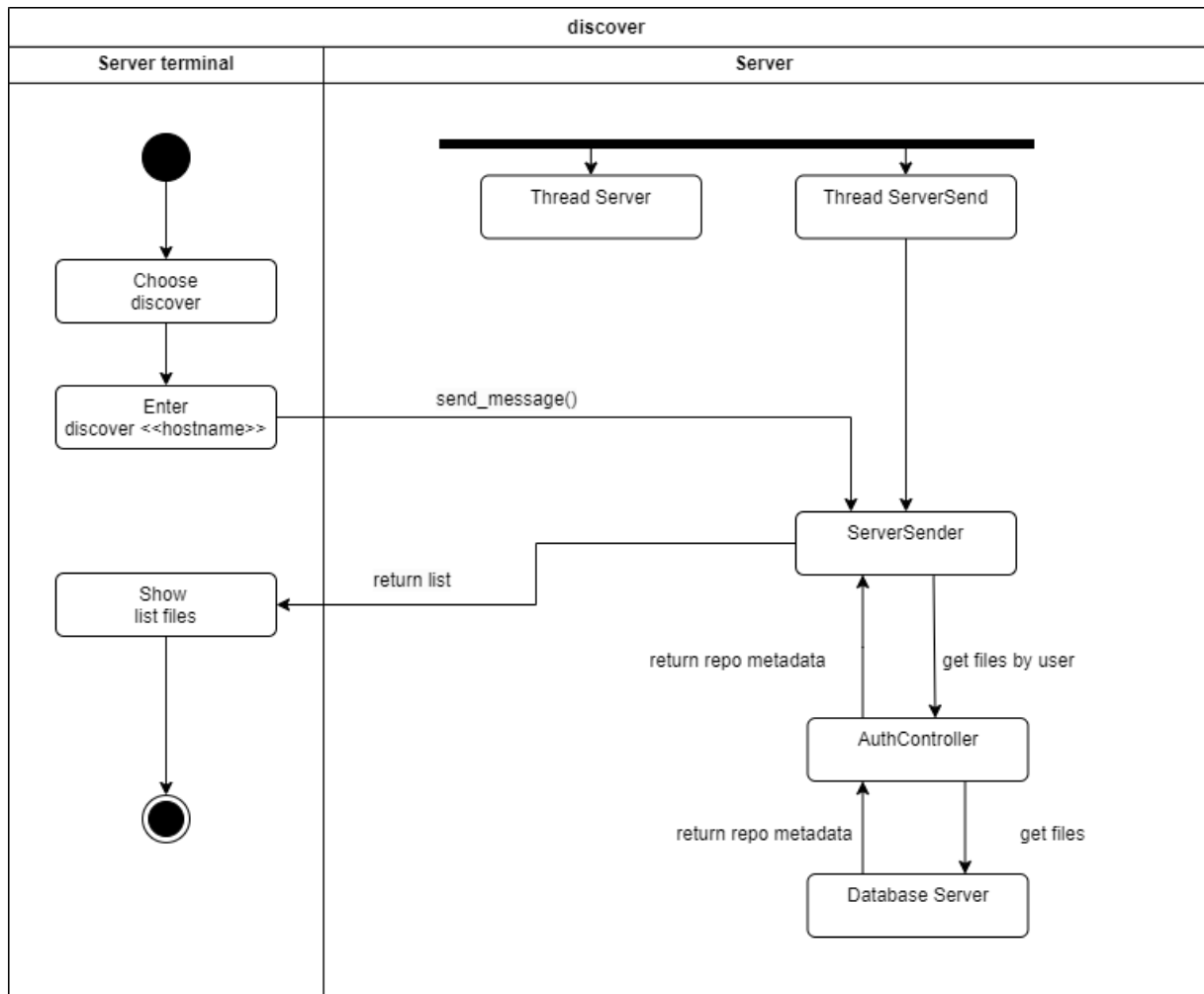
### 4.3.3. Ping



Hình 13: Sơ đồ hoạt động của chức năng ping

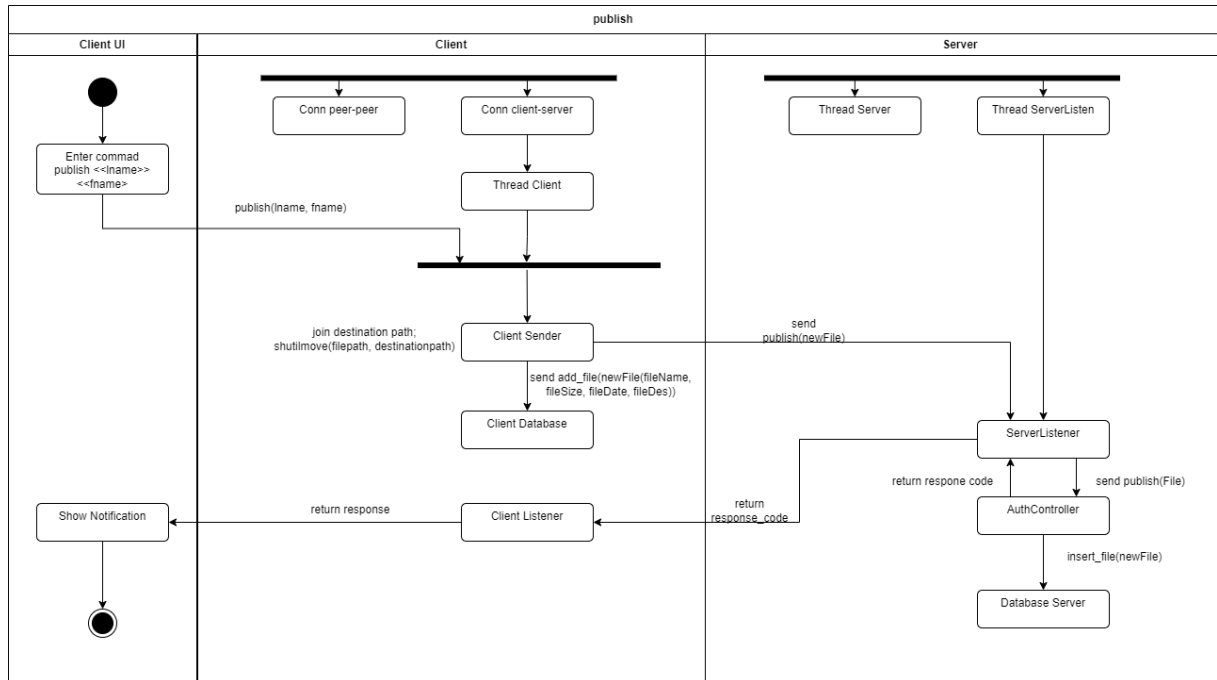


#### 4.3.4. Discover



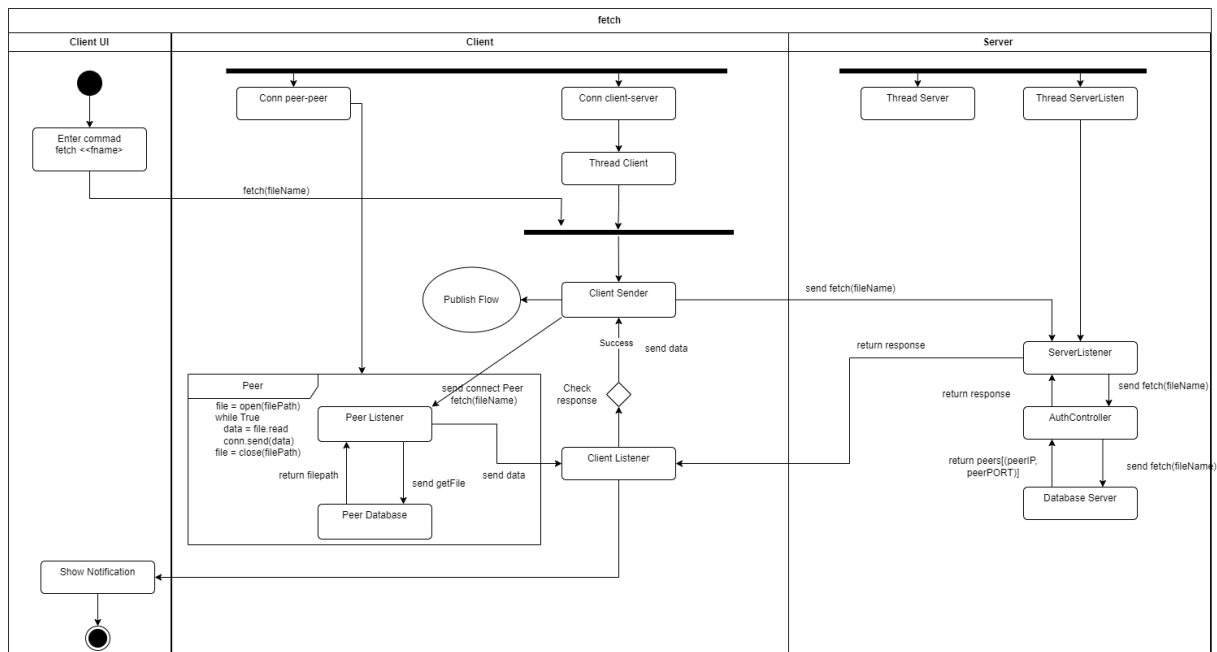
Hình 14: Sơ đồ hoạt động của chức năng discover

#### 4.3.5. Publish



Hình 15: Sơ đồ hoạt động của chức năng publish

#### 4.3.6. Fetch



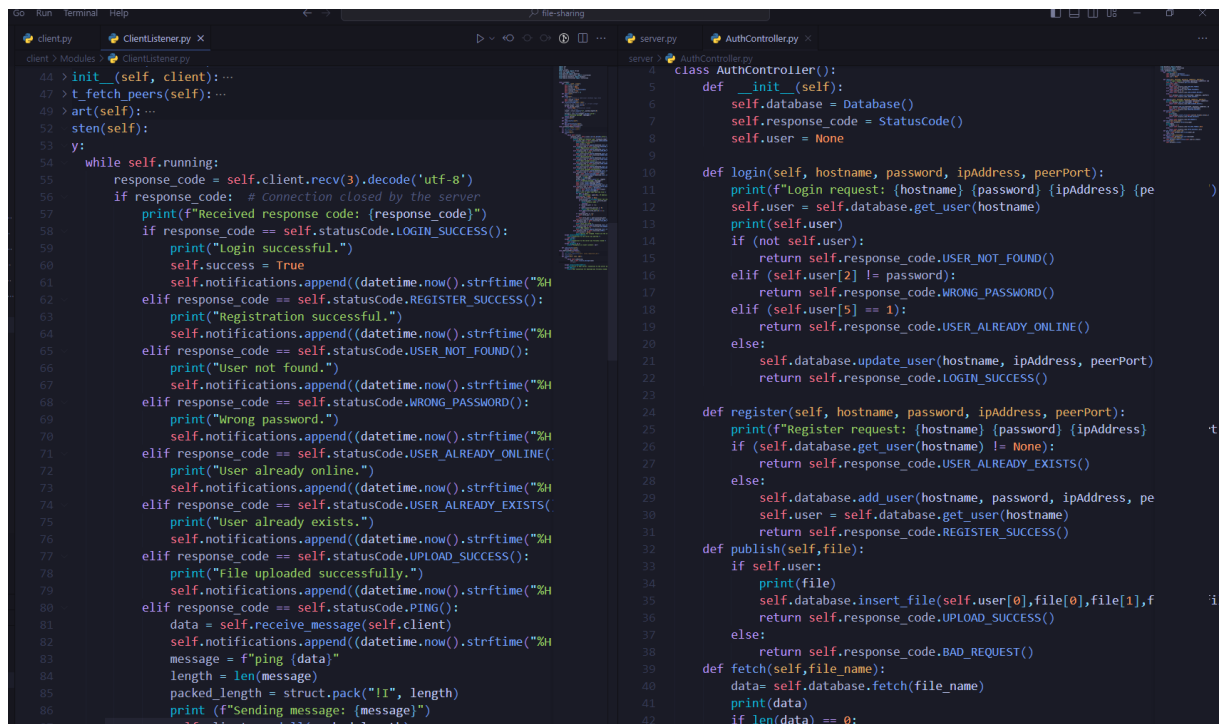
Hình 16: Sơ đồ hoạt động của chức năng fetch

#### 4.4. Status Code

Ngoài ra, nhóm còn định nghĩa thêm một số status code để phục vụ cho việc trao đổi thông tin giữa server đến client và ngược lại.

Status Code	Mô tả
200	Đăng nhập thành công
201	Đăng ký thành công
202	Publish thành công
203	Server muốn ping client
204	Fetch thành công
205	Download file thành công
400	Yêu cầu không hợp lệ
401	Không tìm thấy tài khoản
402	Sai mật khẩu
403	Tài khoản đã đăng nhập
404	Tài khoản đã tồn tại
405	Client thoát khỏi server
406	File không tồn tại

Bảng 1: Các status code được định nghĩa



```

client.py
class ClientListener:
    def __init__(self, client):
        self.client = client
        self.running = True
        self.notifications = []
        self.status_code = StatusCodes()

    def t_fetch_peers(self):
        # ...

    def art(self):
        # ...

    def sten(self):
        # ...

    while self.running:
        response_code = self.client.recv(3).decode('utf-8')
        if response_code == self.status_code.LOGIN_SUCCESS():
            print("Login successful.")
            self.success = True
            self.notifications.append((datetime.now().strftime("%H:%M:%S"), "Login successful."))
        elif response_code == self.status_code.REGISTER_SUCCESS():
            print("Registration successful.")
            self.notifications.append((datetime.now().strftime("%H:%M:%S"), "Registration successful."))
        elif response_code == self.status_code.USER_NOT_FOUND():
            print("User not found.")
            self.notifications.append((datetime.now().strftime("%H:%M:%S"), "User not found."))
        elif response_code == self.status_code.WRONG_PASSWORD():
            print("Wrong password.")
            self.notifications.append((datetime.now().strftime("%H:%M:%S"), "Wrong password."))
        elif response_code == self.status_code.USER_ALREADY_ONLINE():
            print("User already online.")
            self.notifications.append((datetime.now().strftime("%H:%M:%S"), "User already online."))
        elif response_code == self.status_code.USER_ALREADY_EXISTS():
            print("User already exists.")
            self.notifications.append((datetime.now().strftime("%H:%M:%S"), "User already exists."))
        elif response_code == self.status_code.UPLOAD_SUCCESS():
            print("File uploaded successfully.")
            self.notifications.append((datetime.now().strftime("%H:%M:%S"), "File uploaded successfully."))
        elif response_code == self.status_code.PING():
            data = self.receive_message(self.client)
            self.notifications.append((datetime.now().strftime("%H:%M:%S"), f"ping {data}"))
            length = len(message)
            packed_length = struct.pack("i", length)
            print(f"Sending message: {message}")
            self.client.send(packed_length + message)

server.py
class AuthController:
    def __init__(self):
        self.database = Database()
        self.response_code = StatusCodes()
        self.user = None

    def login(self, hostname, password, ipAddress, peerPort):
        print(f"Login request: {hostname} {password} {ipAddress} {peerPort}")
        self.user = self.database.get_user(hostname)
        print(self.user)
        if not self.user:
            return self.response_code.USER_NOT_FOUND()
        elif self.user[2] != password:
            return self.response_code.WRONG_PASSWORD()
        elif self.user[5] == 1:
            return self.response_code.USER_ALREADY_ONLINE()
        else:
            self.database.update_user(hostname, ipAddress, peerPort)
            return self.response_code.LOGIN_SUCCESS()

    def register(self, hostname, password, ipAddress, peerPort):
        print(f"Register request: {hostname} {password} {ipAddress} {peerPort}")
        if self.database.get_user(hostname) != None:
            return self.response_code.USER_ALREADY_EXISTS()
        else:
            self.database.add_user(hostname, password, ipAddress, peerPort)
            self.user = self.database.get_user(hostname)
            return self.response_code.REGISTER_SUCCESS()

    def publish(self, file):
        if self.user:
            print(file)
            self.database.insert_file(self.user[0], file[0], file[1], file[2])
            return self.response_code.UPLOAD_SUCCESS()
        else:
            return self.response_code.BAD_REQUEST()

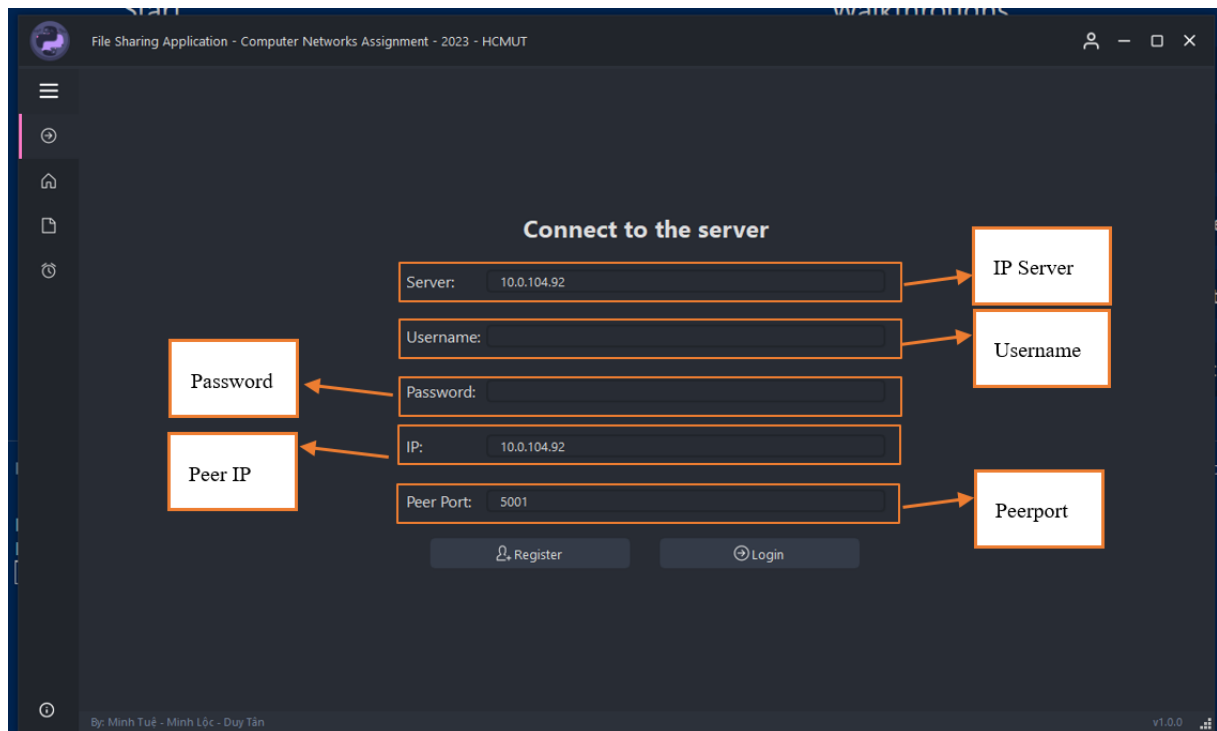
    def fetch(self, file_name):
        data = self.database.fetch(file_name)
        print(data)
        if len(data) == 0:
            return self.response_code.FILE_NOT_FOUND()
        else:
            return self.response_code.FETCH_SUCCESS()

```

Hình 17: Trao đổi status code giữa client và server

## 5. Design UI

### 5.1. Register and Login



The screenshot shows a dark-themed 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" to the Server field, "Username" to the Username field, "Password" to the Password field, "Peer IP" to the IP field, and "Peerport" to the Peer Port field. The bottom of the window shows the text "Bç: Minh Tuệ - Minh Lộc - Duy Tân" and "v1.0.0".

Hình 18: UI Register and Login

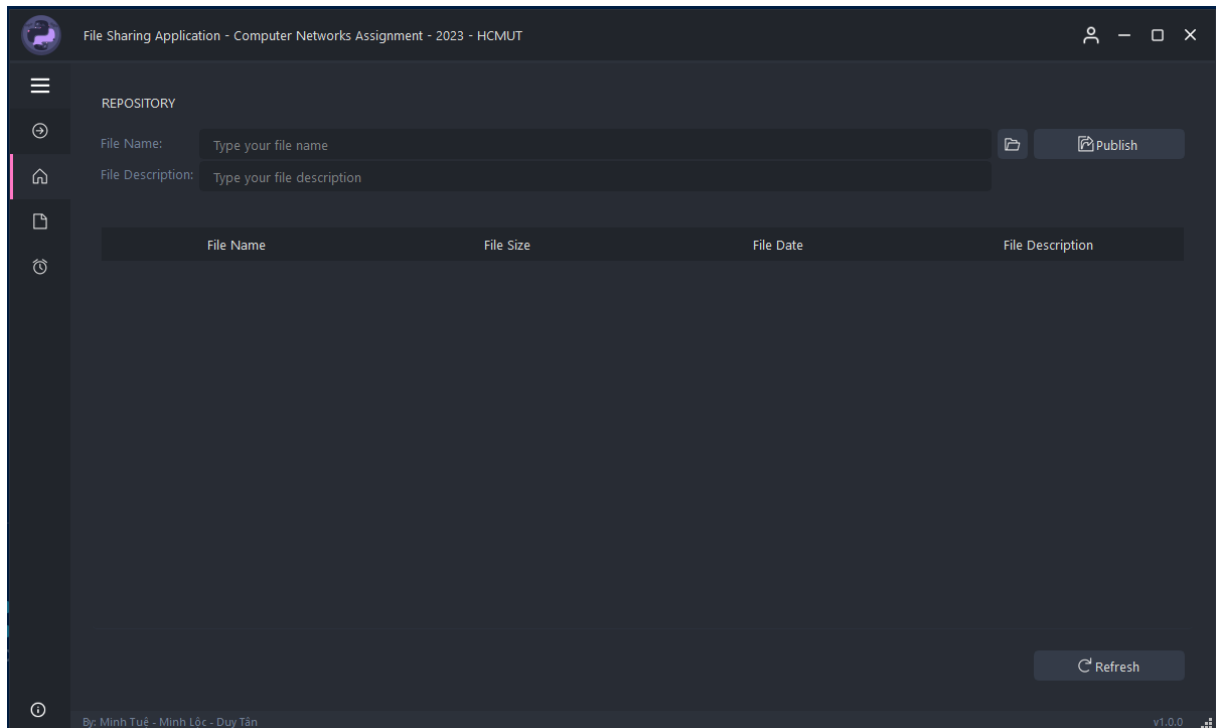
Đây là trang đăng ký, đăng nhập vào hệ thống bao gồm: địa chỉ IP Server, username, password, địa chỉ IP Peer, địa chỉ Peer Port.

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.

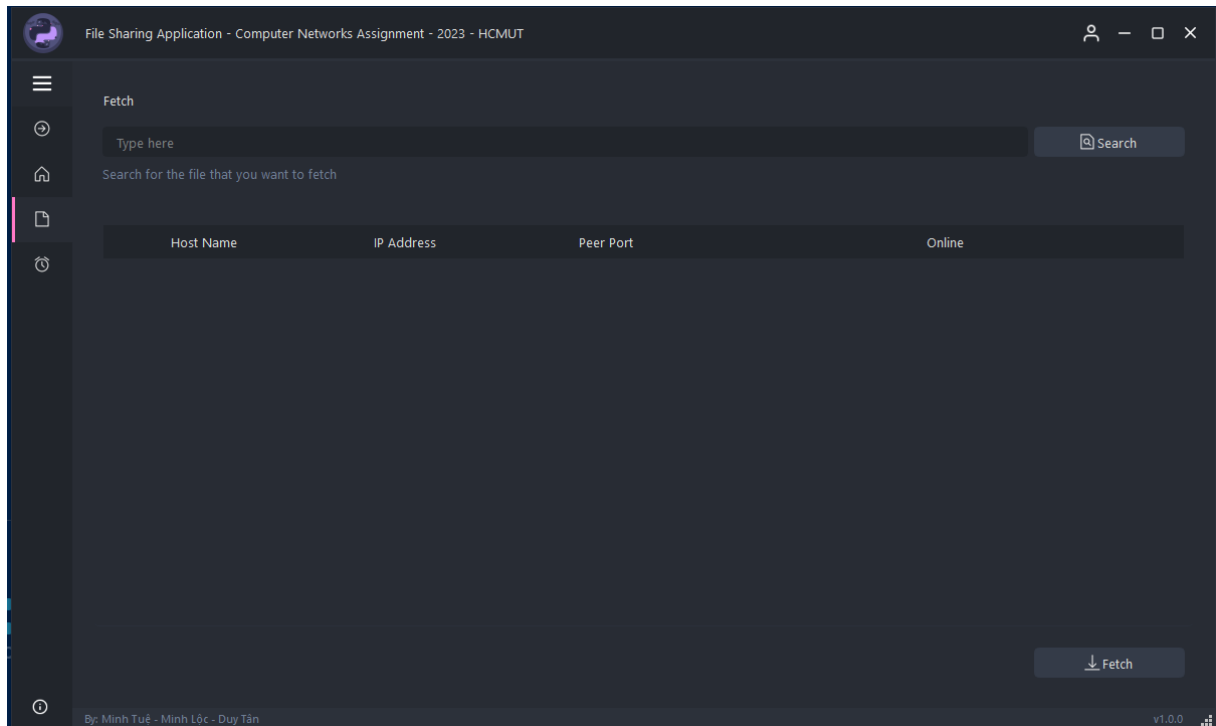
## 5.2. Publish Page



Hình 19: 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.

### 5.3. Fetch Page



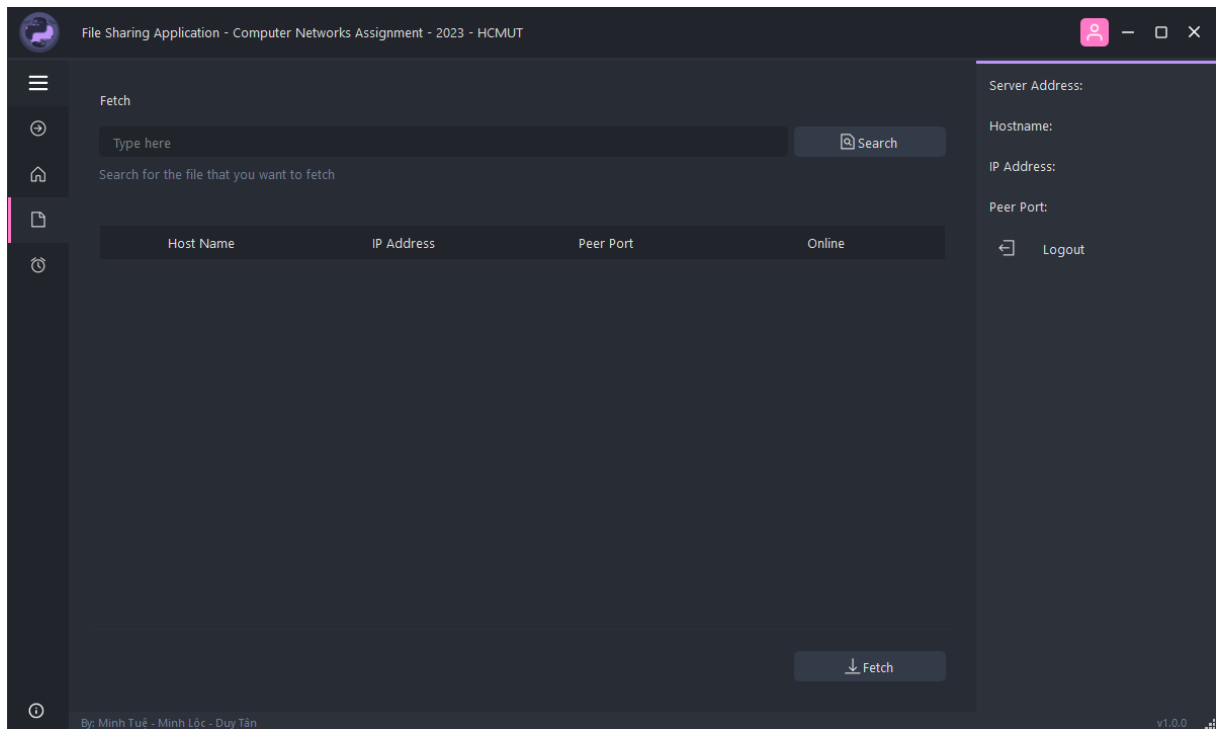
Hình 20: UI Fetch File

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.

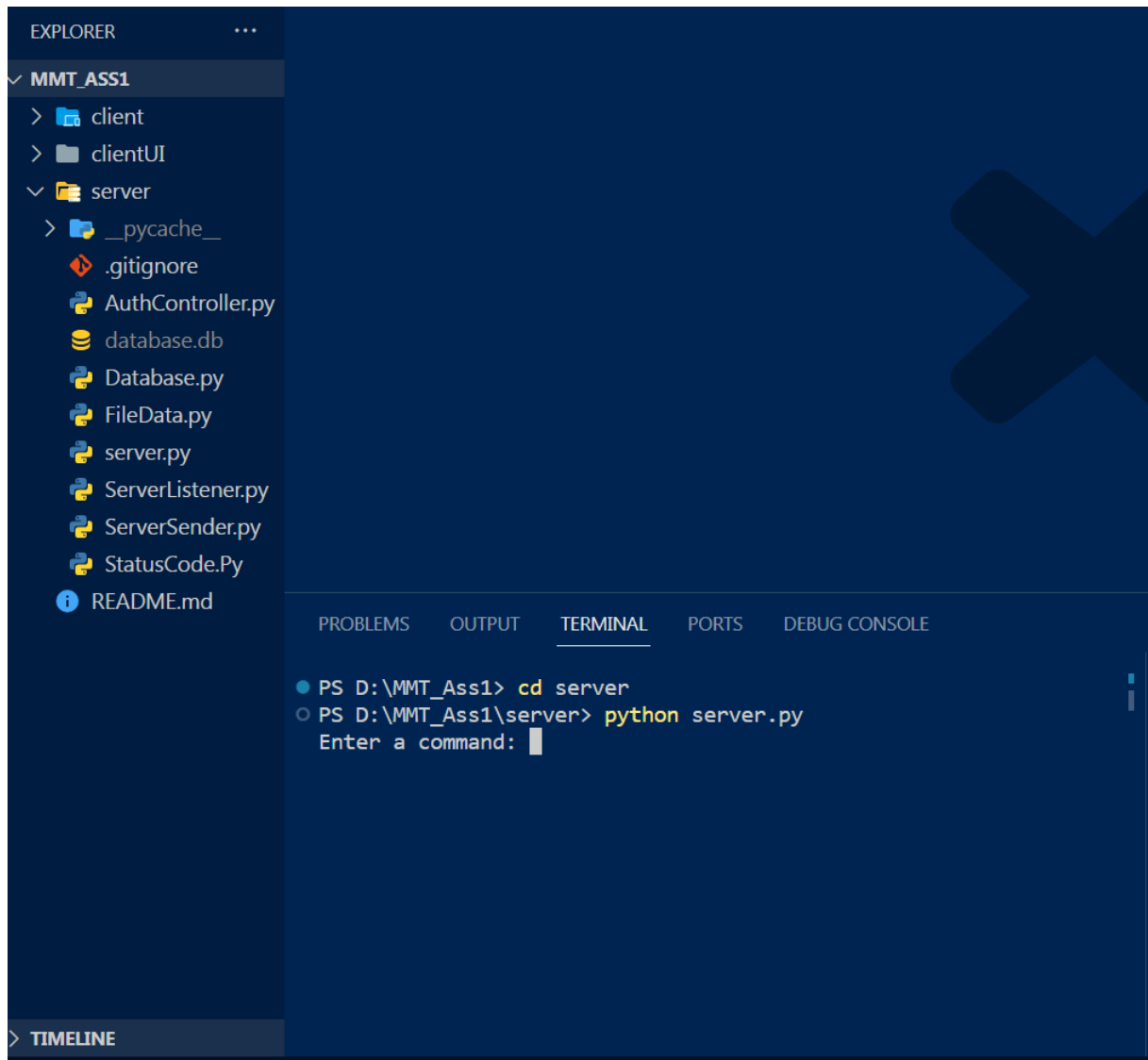


Hình 21: UI Profile

## 6. Tutorial - Guides

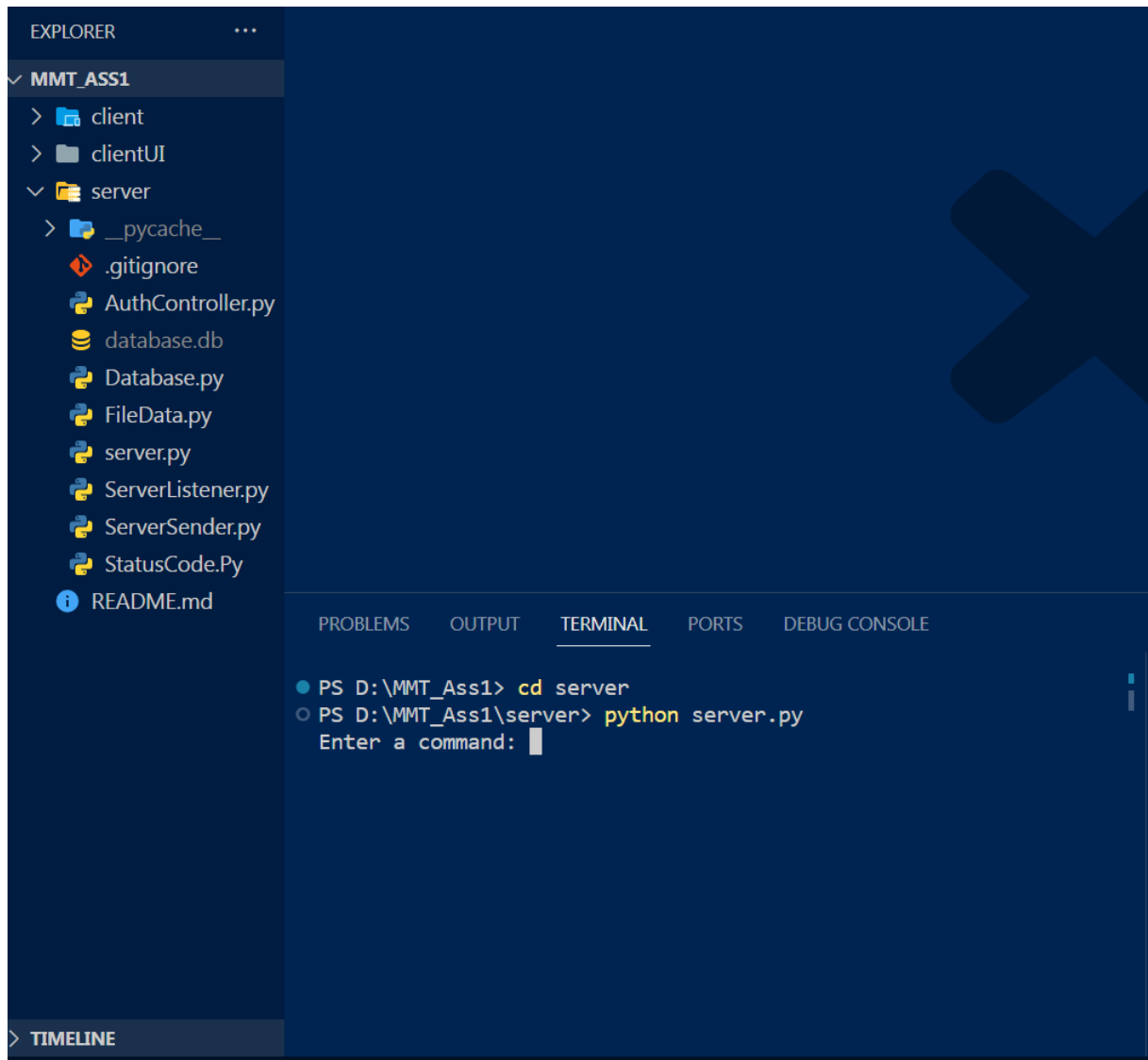
### 6.1. Server

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



Hình 22: Start Server



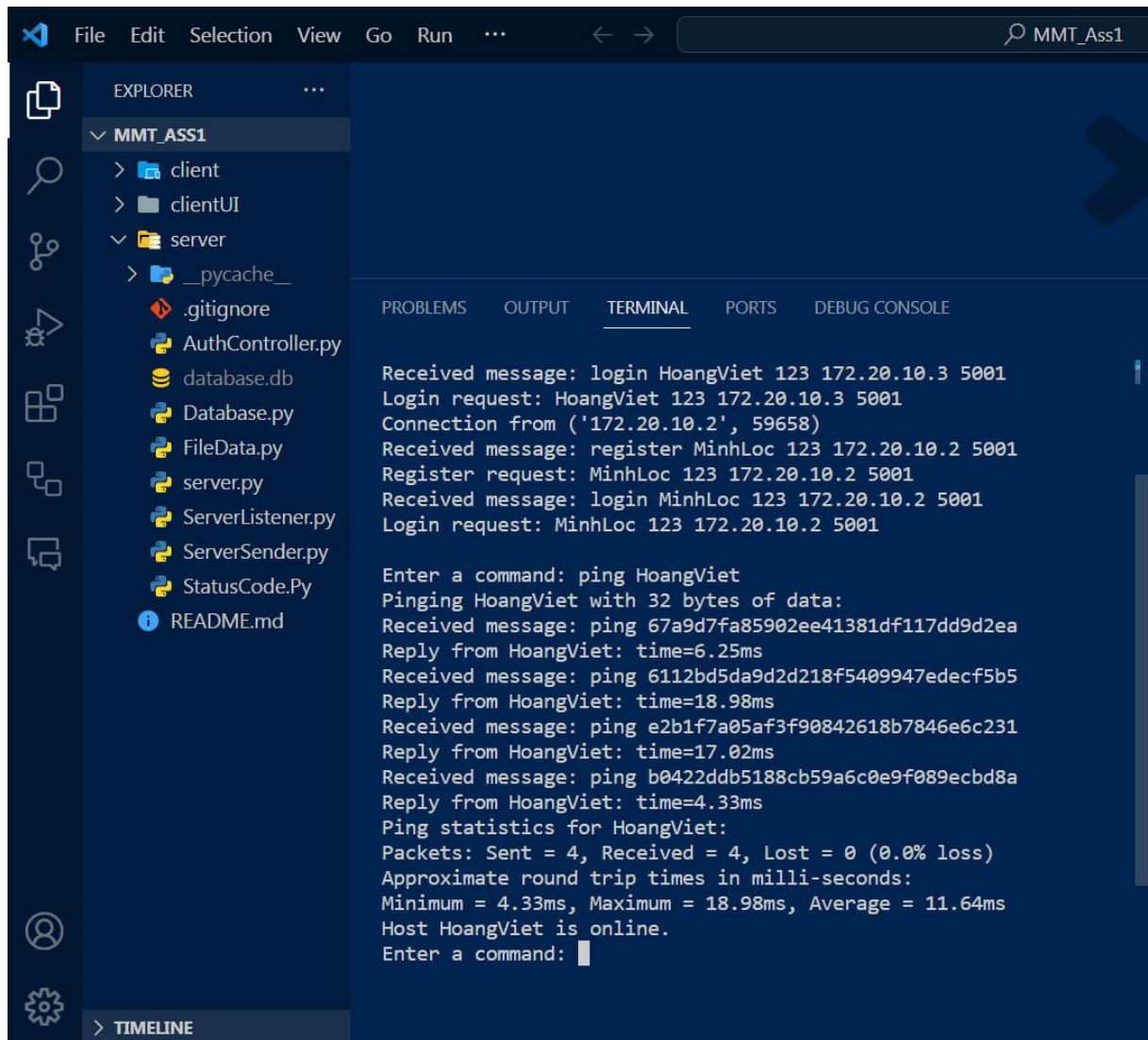


Hình 23: Start Server

### 6.1.1. 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



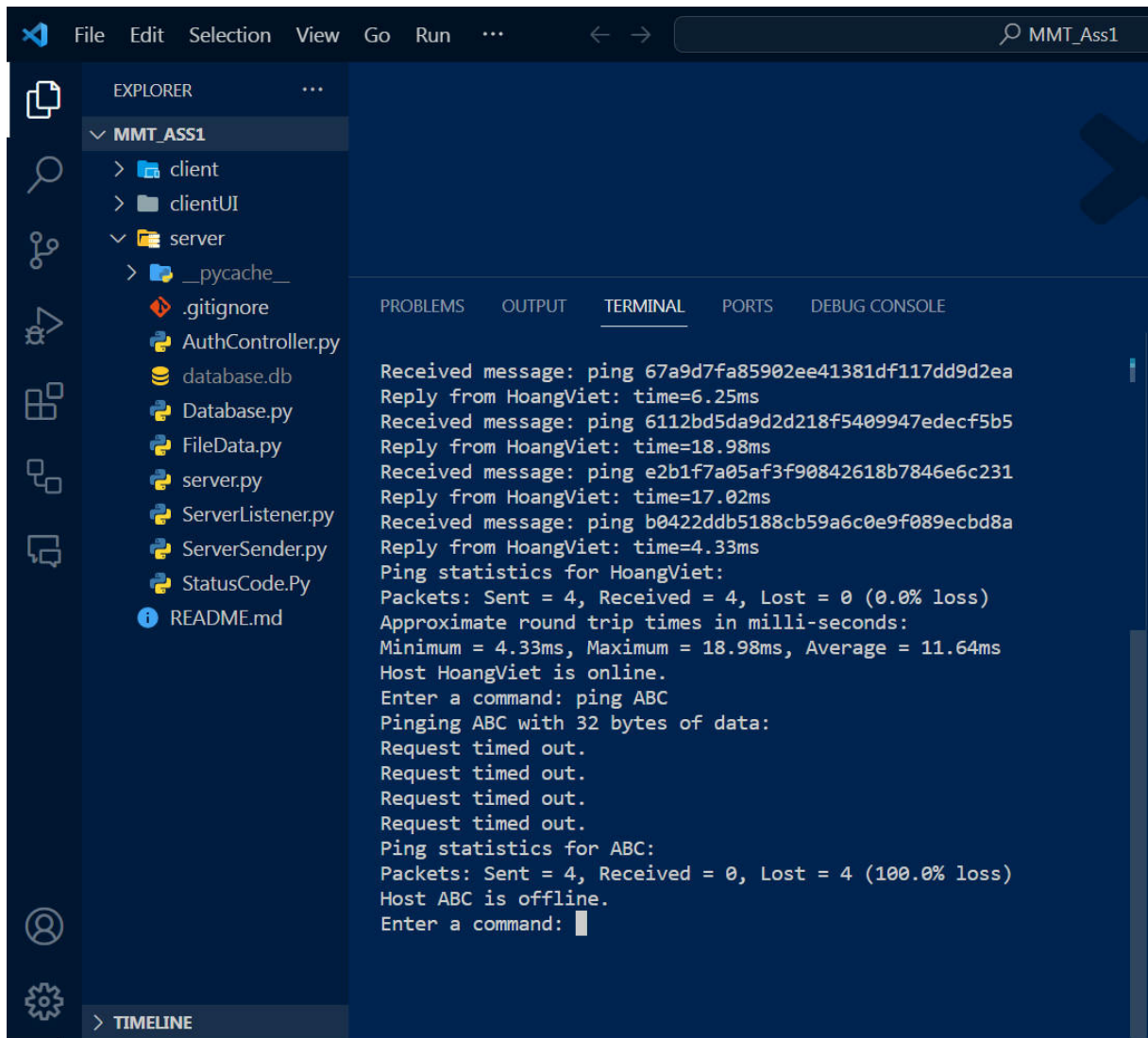
The screenshot shows the Visual Studio Code (VS Code) interface with a project named 'MMT\_Ass1'. The Explorer sidebar on the left displays the project structure, including folders 'client' and 'clientUI', and a 'server' folder containing files like '\_\_pycache\_\_', '.gitignore', 'AuthController.py', '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 24: 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

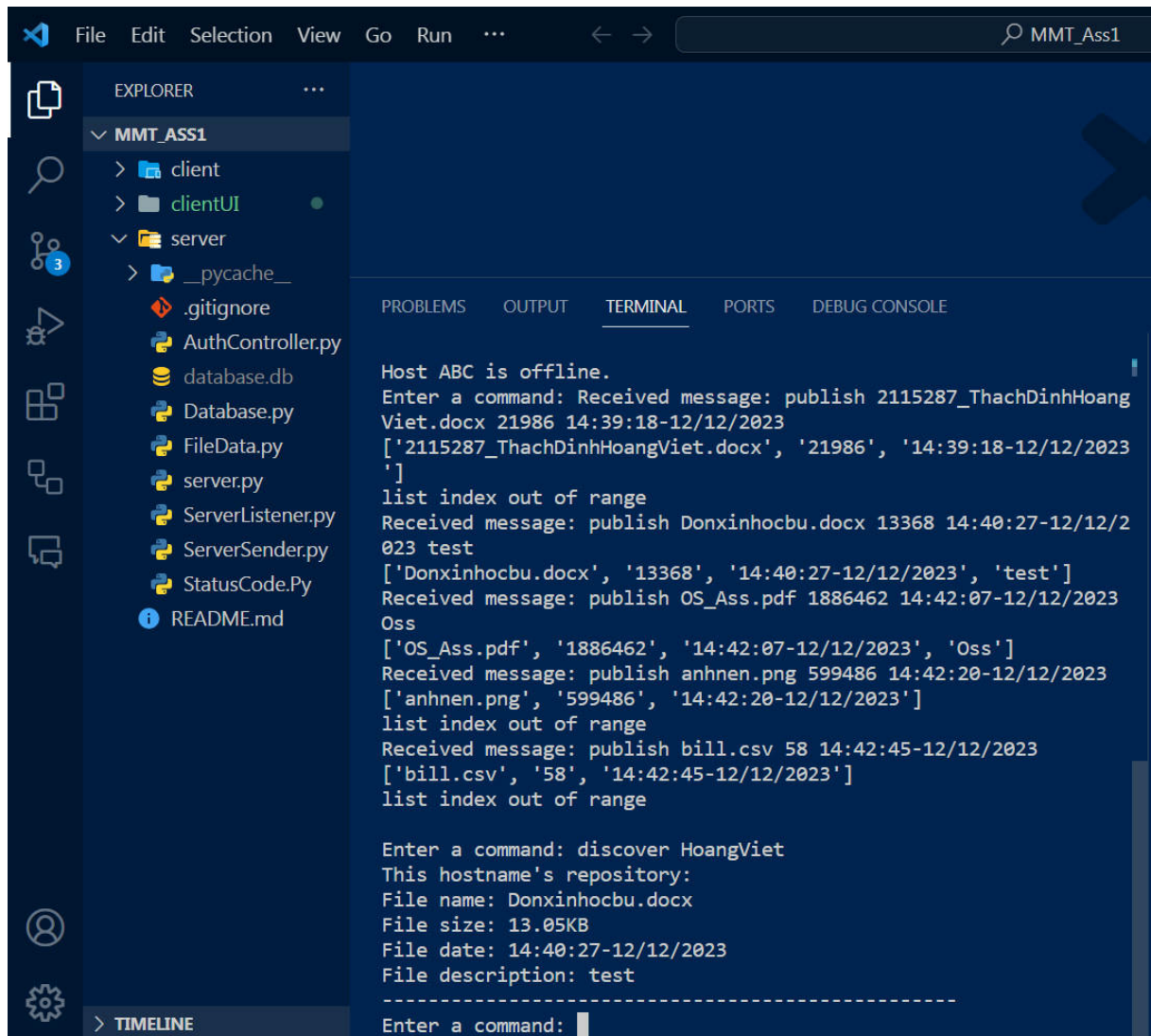
PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE

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 25: Ping No Online

### 6.1.2. Discover

Server có thể discover các Peer khác bằng cách nhập discover



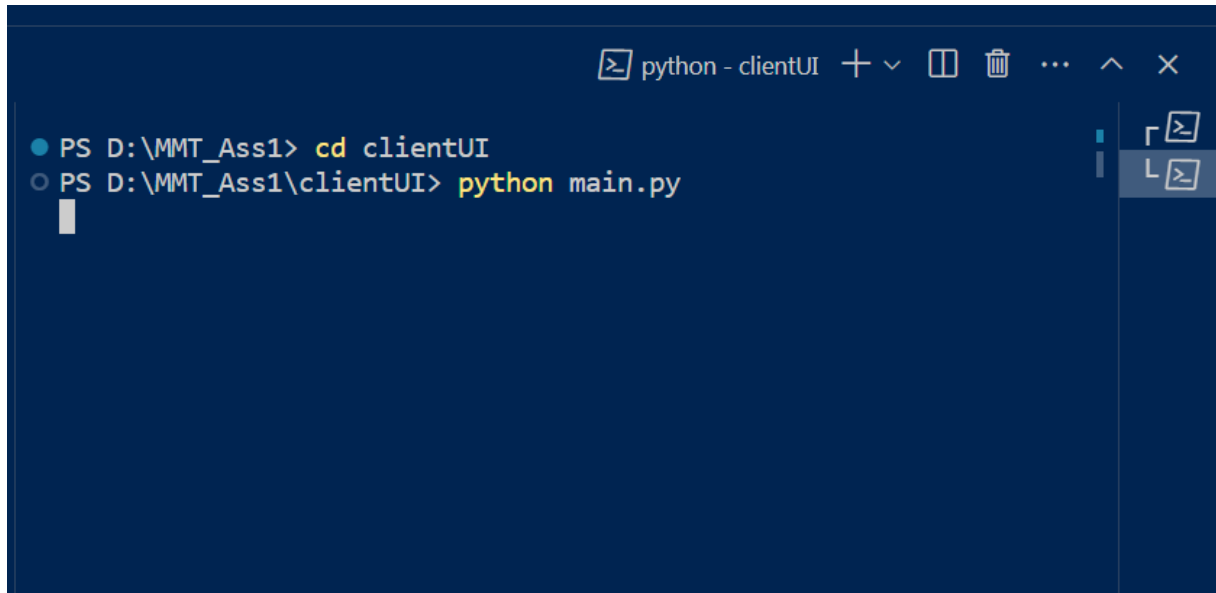
```
Host ABC is offline.
Enter a command: Received message: publish 2115287_ThachDinhHoangViet.docx 21986 14:39:18-12/12/2023
['2115287_ThachDinhHoangViet.docx', '21986', '14:39:18-12/12/2023']
list index out of range
Received message: publish Donxinhocbu.docx 13368 14:40:27-12/12/2023 test
['Donxinhocbu.docx', '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:
```

Hình 26: Discover

## 6.2. Client

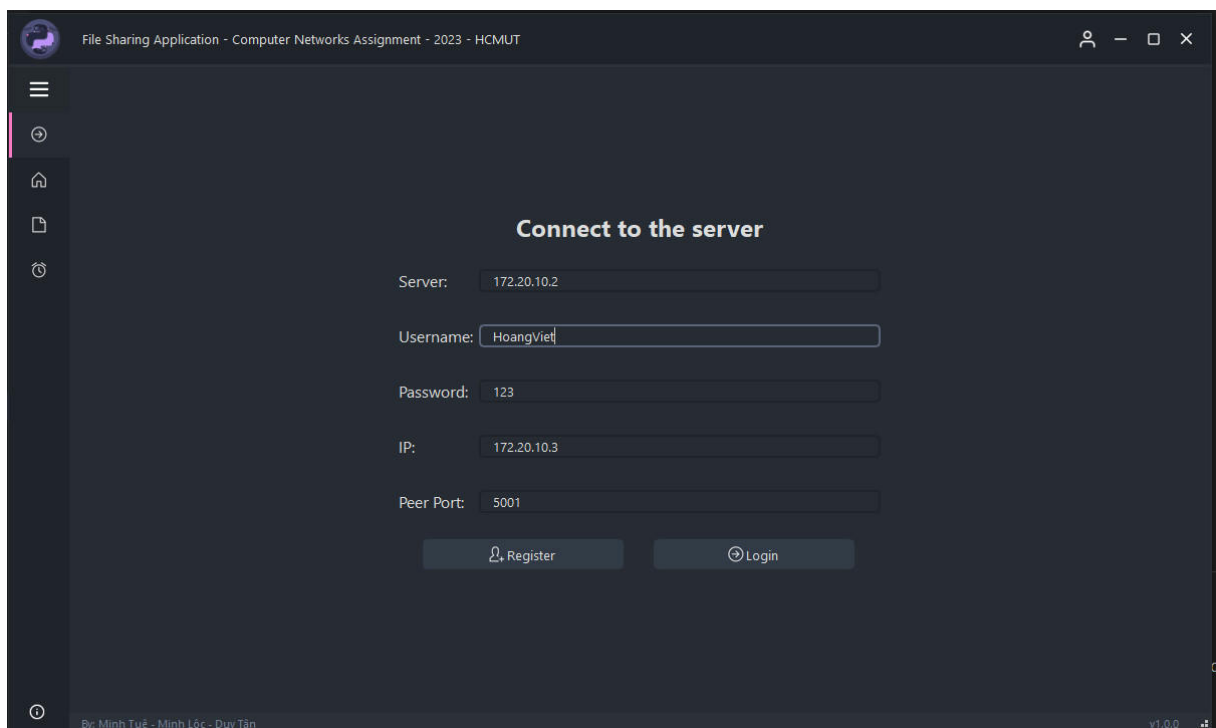
Người dùng có thể start client bằng cách chạy file main.py



Hình 27: Start Client

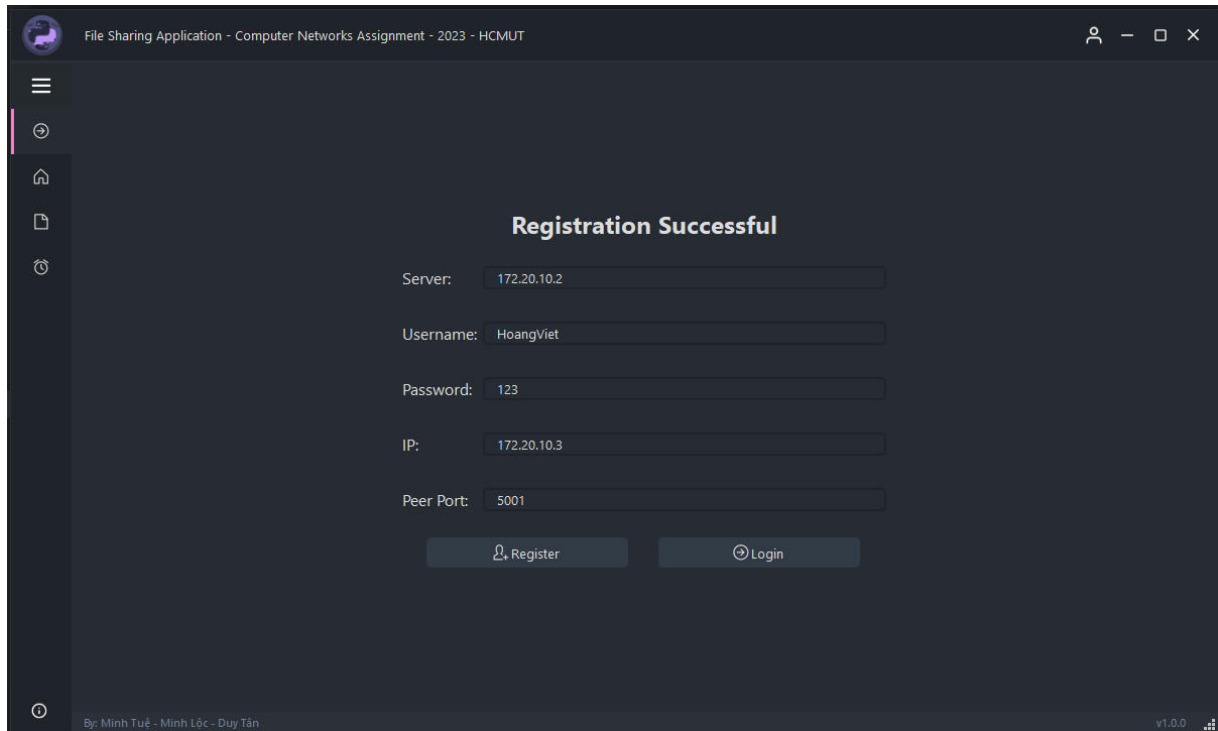
### 6.2.1. Register

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



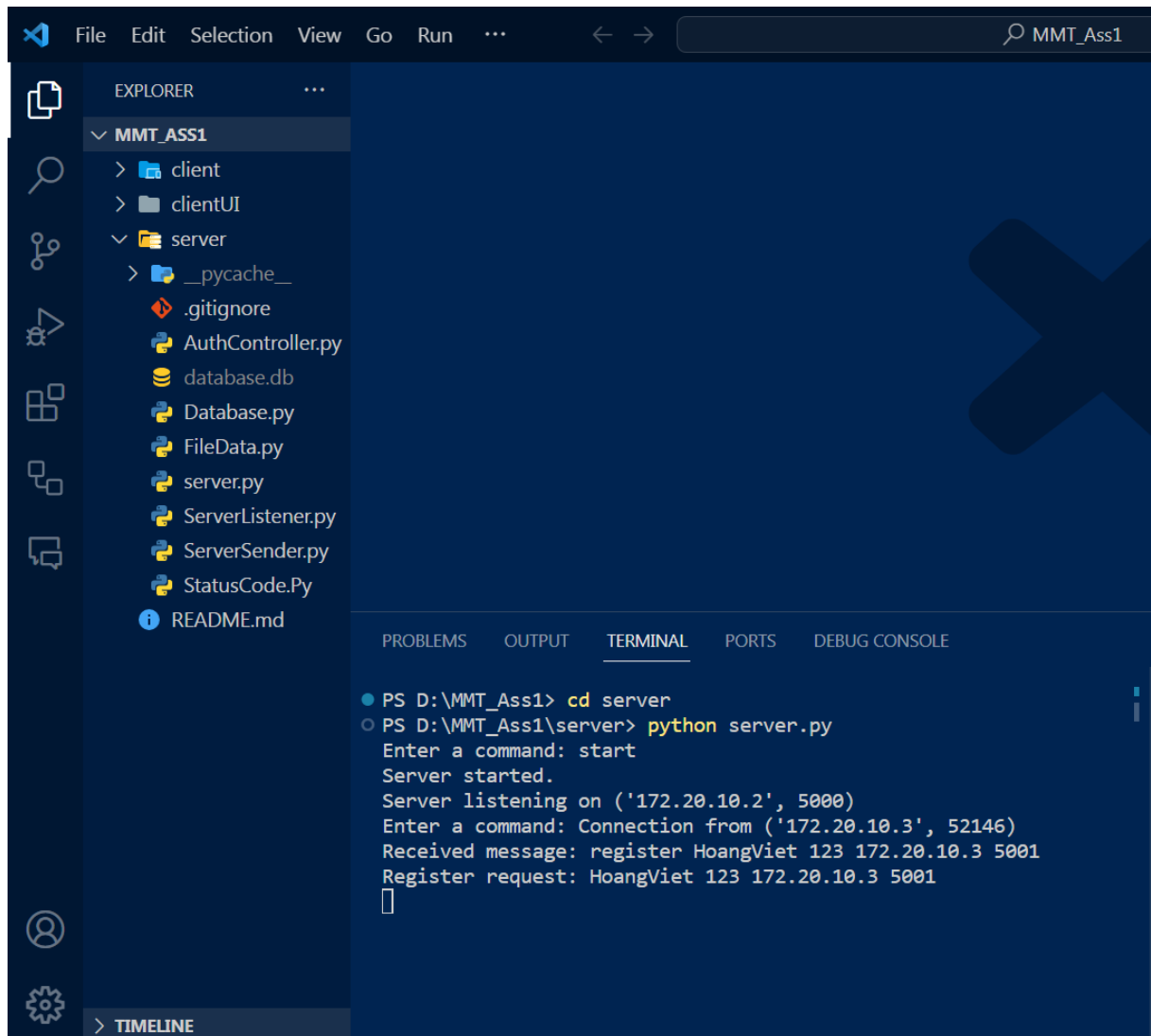
Hình 28: Client Register

Giao diện sẽ hiển thị thông báo đăng ký thành công



Hình 29: Client Register Success

Phía Server sẽ hiển thị thông báo đăng ký thành công



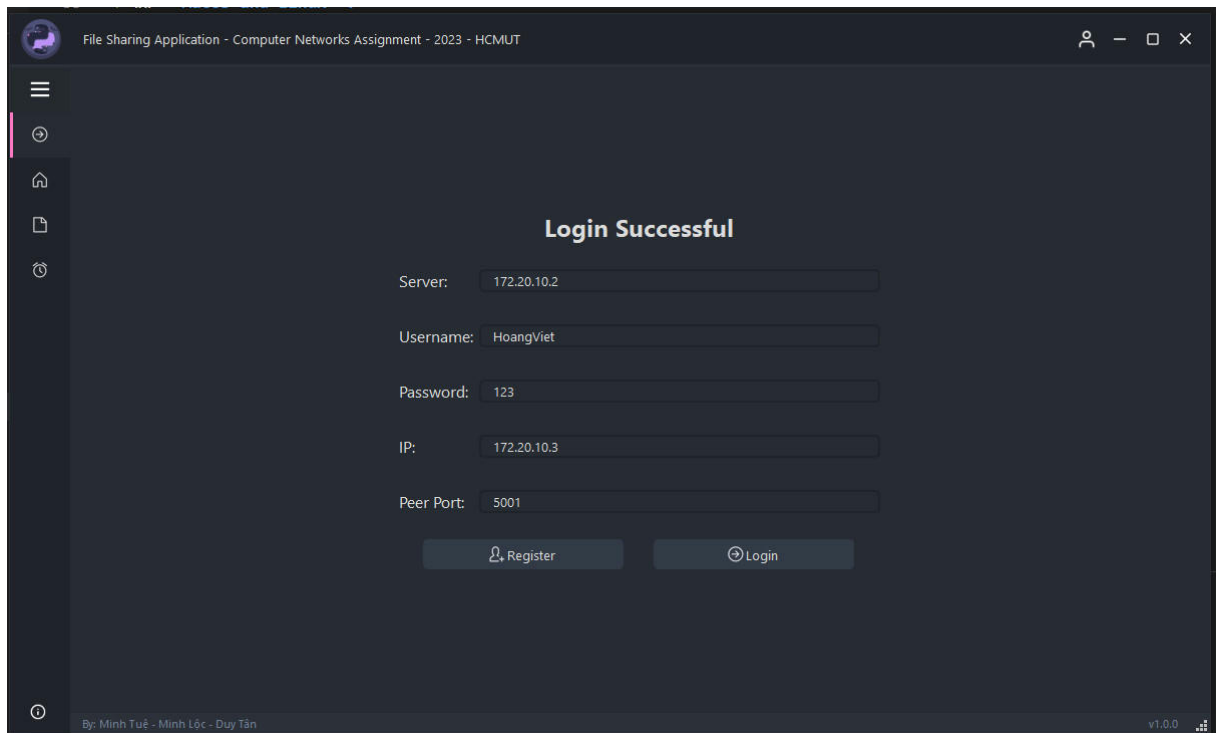
The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left displaying the project structure for 'MMT\_Ass1'. The project contains folders 'client' and 'clientUI', and a 'server' folder which includes subfolders '\_\_pycache\_\_' and files: '.gitignore', 'AuthController.py', '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 30: Server Register Success

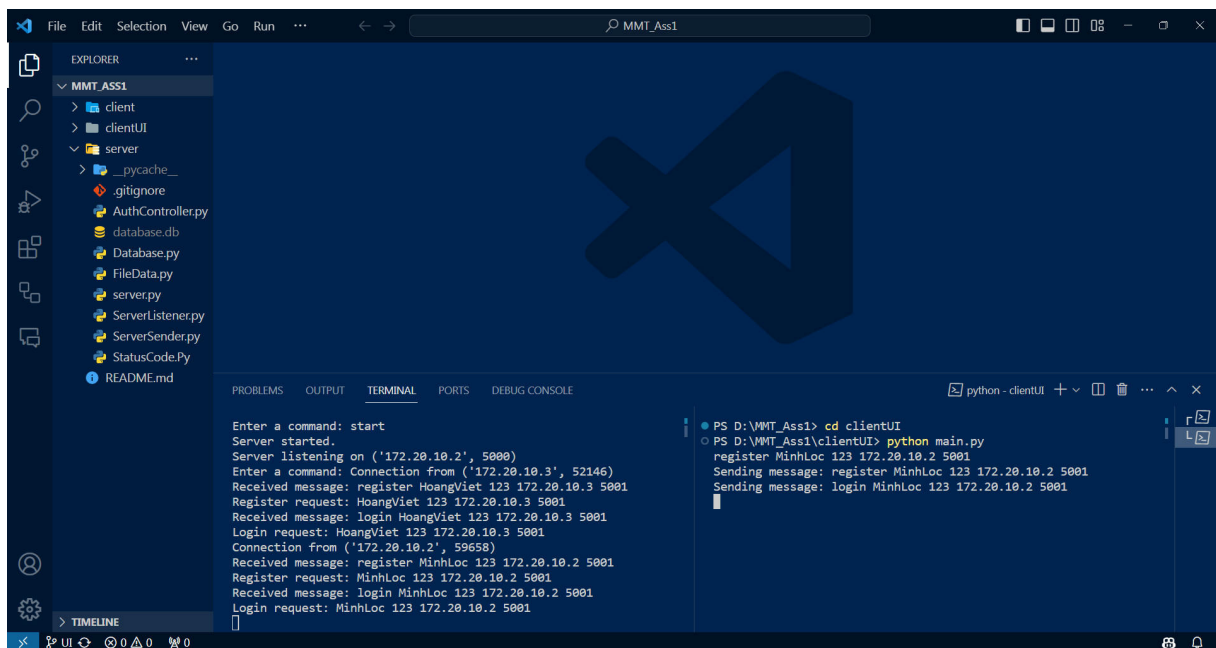
### 6.2.2. Login

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



Hình 31: Client Login

Phía Server sẽ hiển thị thông báo đăng nhập thành công

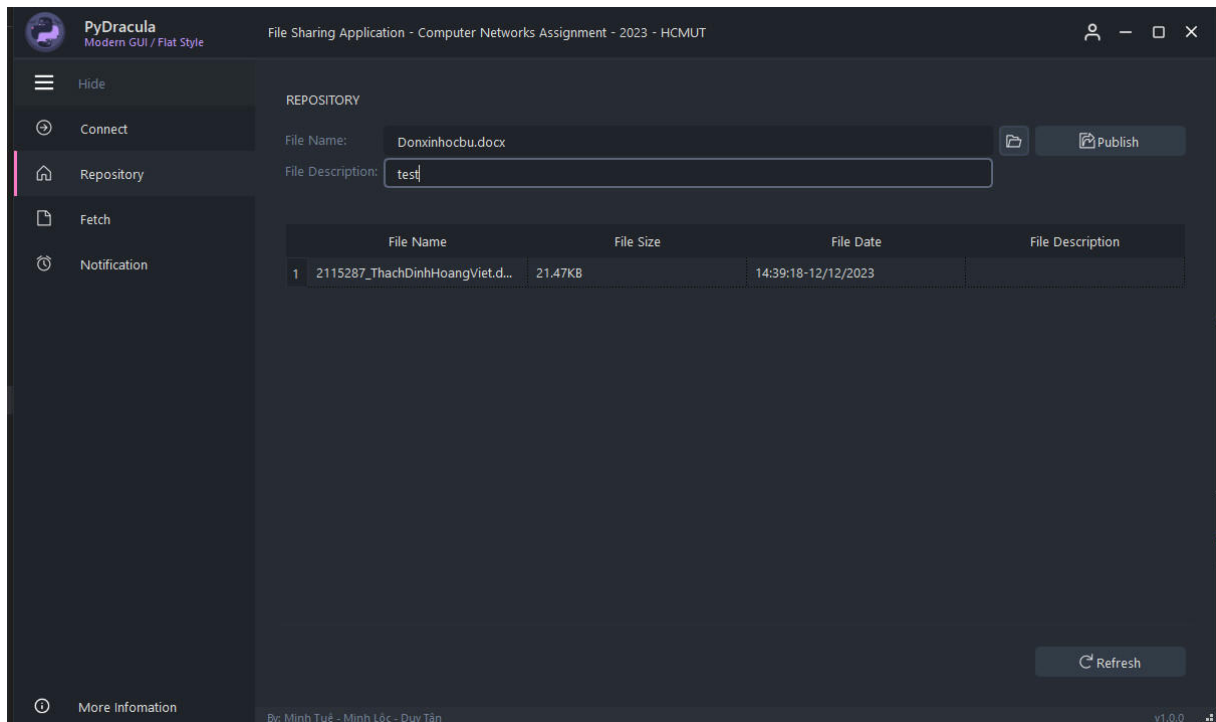


Hình 32: Login Success

### 6.2.3. Publish

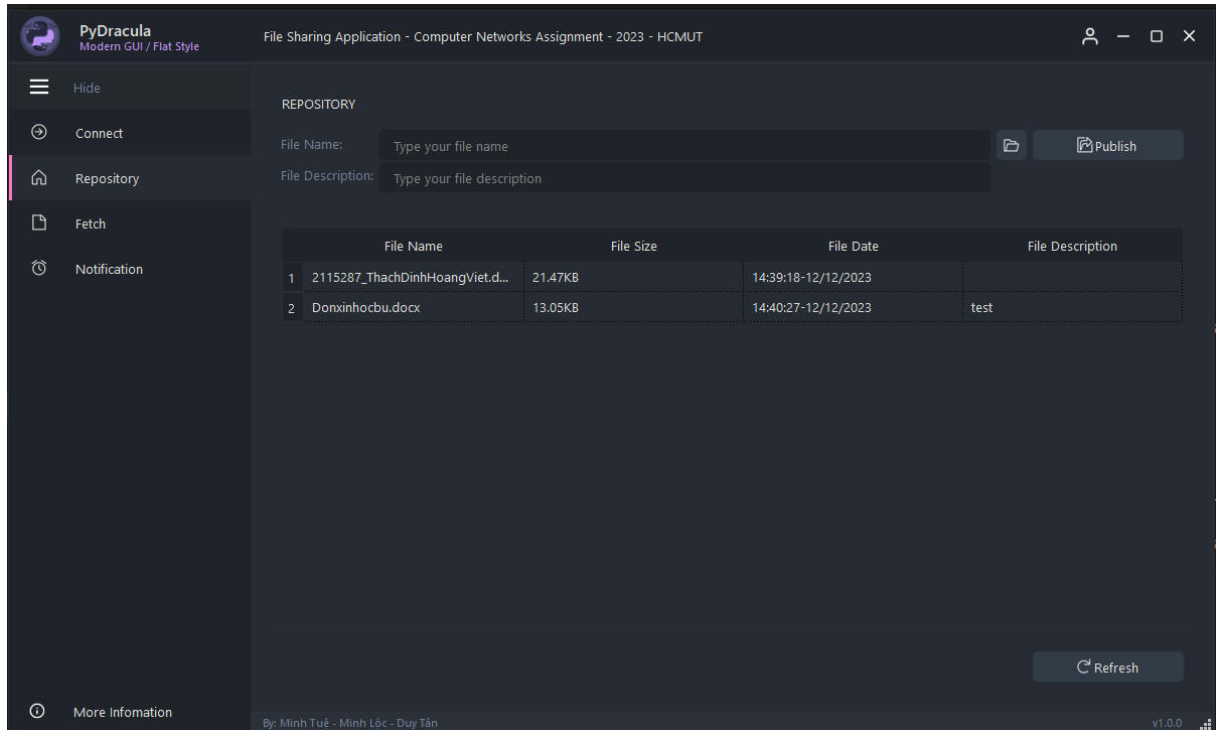
Người dùng sẽ nhập đường dẫn file, mô tả file và nhấn nút Publish để Publish file lên hệ thống.





Hình 33: Publish

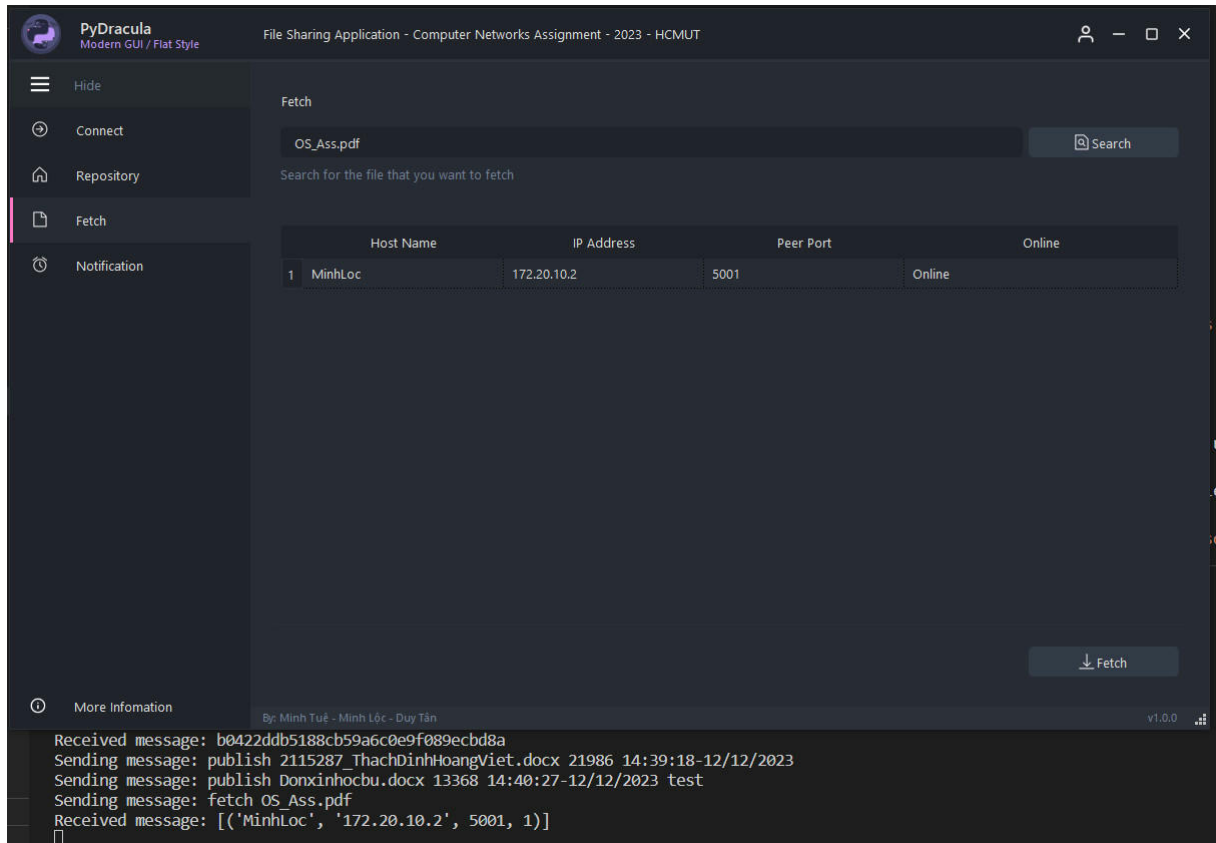
Giao diện sẽ hiển thị file đã Publish vào danh sách



Hình 34: Publish Success

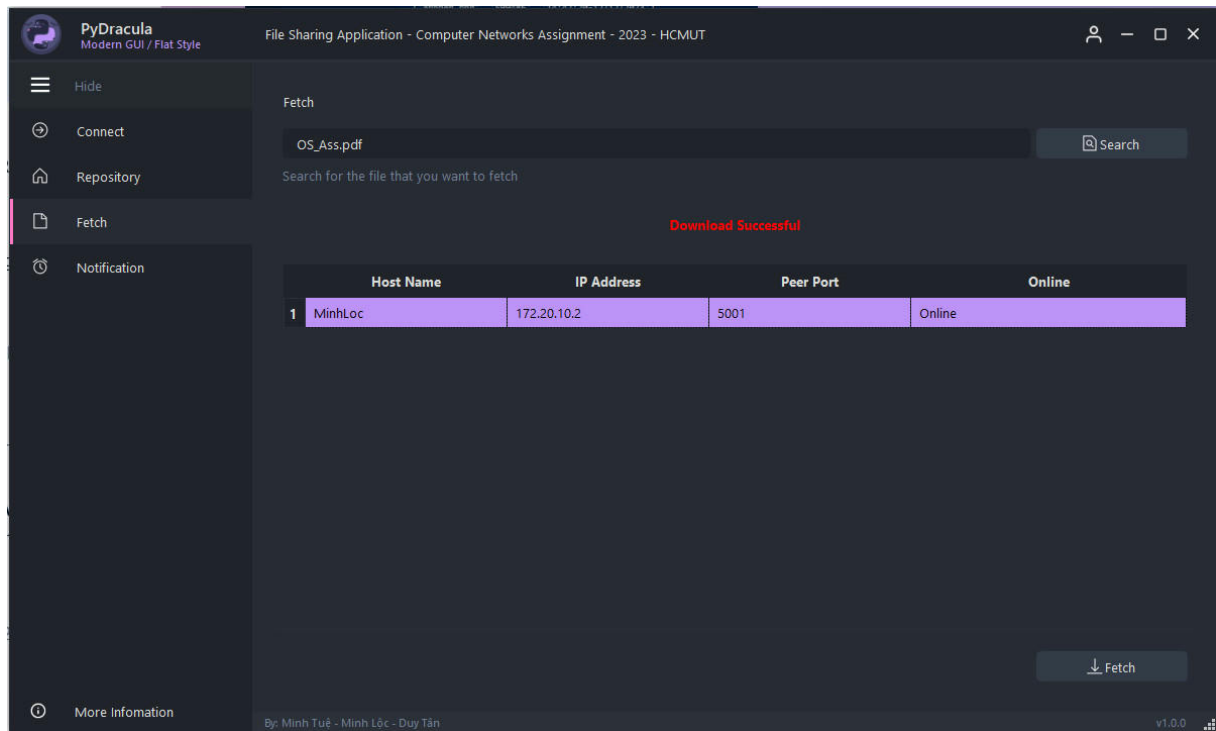
#### 6.2.4. Fetch

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



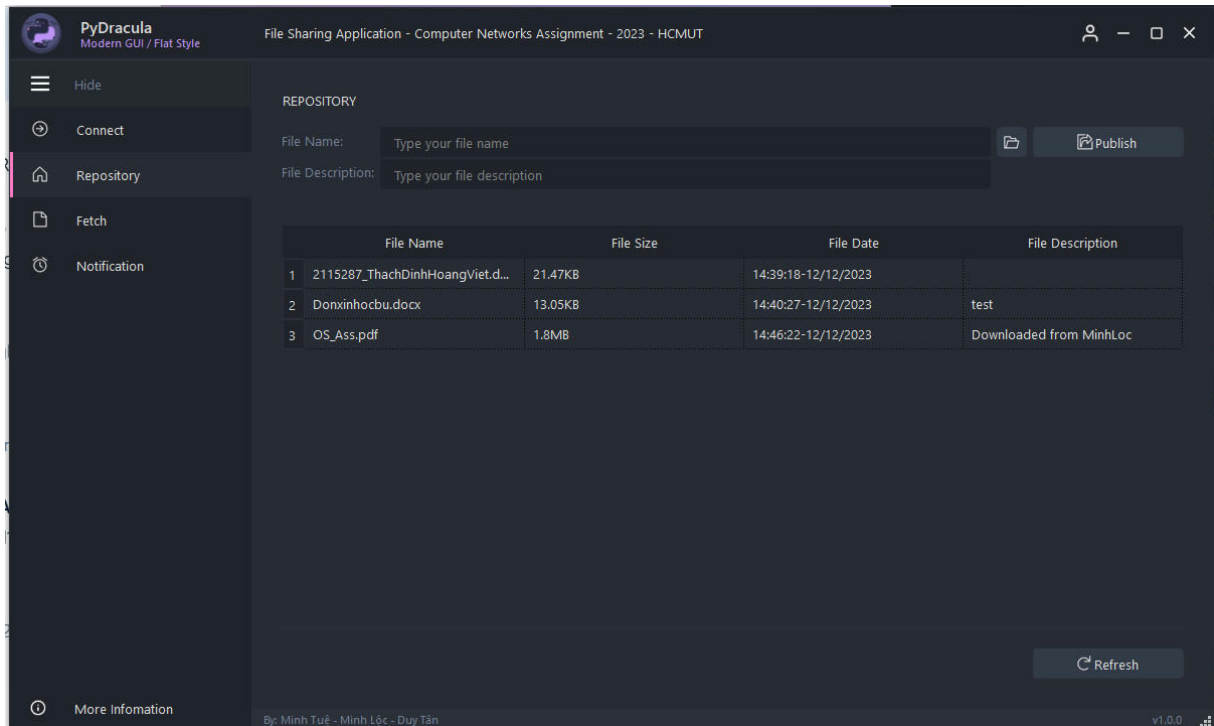
Hình 35: Fetch

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



Hình 36: Fetch Success

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

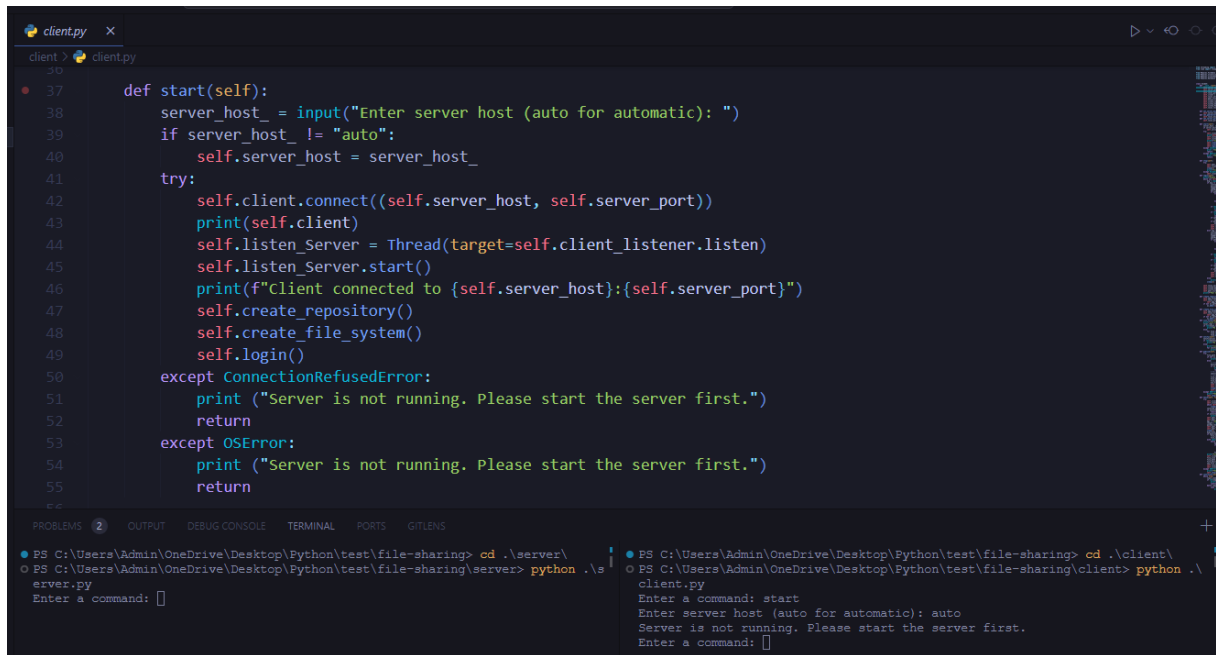


Hình 37: Fetch Success

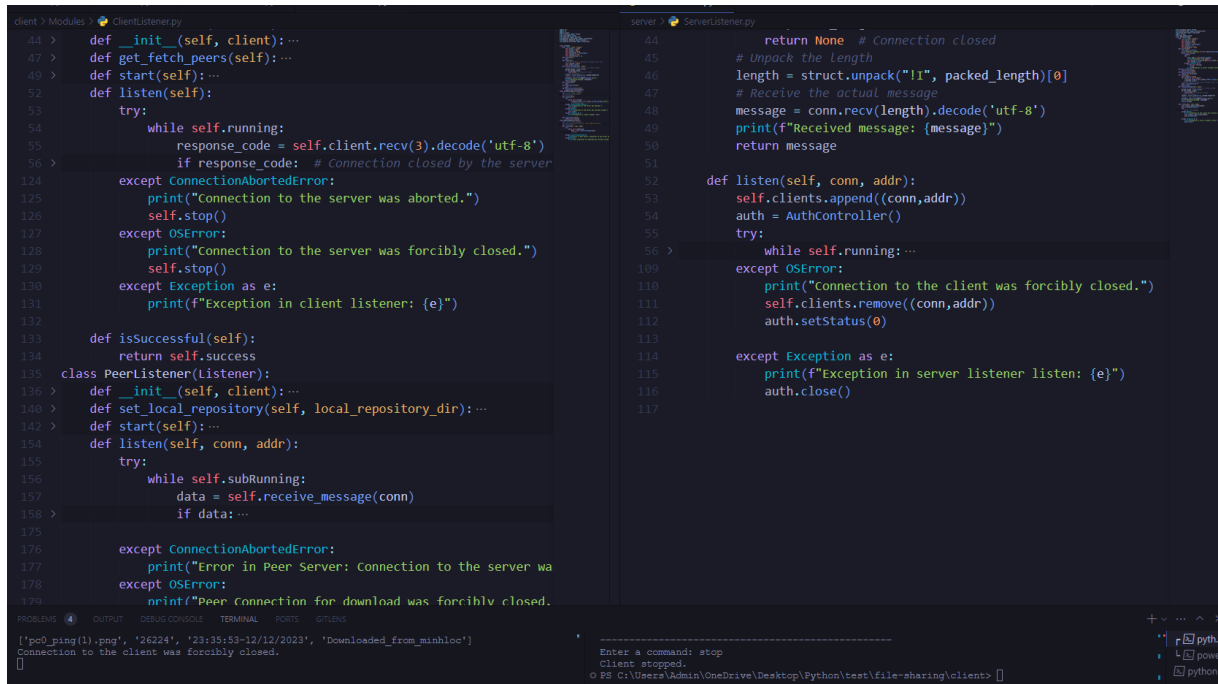
## 7. Error Handling

### 7.1. Server not running

Khi Client kết nối tới Server, nếu Server không hoạt động, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu khởi động lại Server.



```
client.py
client > client.py
37 def start(self):
38     server_host_ = input("Enter server host (auto for automatic): ")
39     if server_host_ != "auto":
40         self.server_host = server_host_
41     try:
42         self.client.connect((self.server_host, self.server_port))
43         print(self.client)
44         self.listen_Server = Thread(target=self.client_listener.listen)
45         self.listen_Server.start()
46         print(f"Client connected to {self.server_host}:{self.server_port}")
47         self.create_repository()
48         self.create_file_system()
49         self.login()
50     except ConnectionRefusedError:
51         print("Server is not running. Please start the server first.")
52         return
53     except OSError:
54         print("Server is not running. Please start the server first.")
55         return
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
261
```



```
client.py
44 > def __init__(self, client):...
47 > def get_fetch_peers(self):...
49 > def start(self):...
52 def listen(self):
53     try:
54         while self.running:
55             response_code = self.client.recv(3).decode('utf-8')
56             if response_code: # Connection closed by the server
124         except ConnectionAbortedError:
125             print("Connection to the server was aborted.")
126             self.stop()
127         except OSError:
128             print("Connection to the server was forcibly closed.")
129             self.stop()
130         except Exception as e:
131             print(f"Exception in client listener: {e}")
132
133 def isSuccessful(self):
134     return self.success
135 class PeerListener(Listener):
136 > def __init__(self, client):...
140 > def set_local_repository(self, local_repository_dir):...
142 > def start(self):...
154 def listen(self, conn, addr):
155     try:
156         while self.subRunning:
157             data = self.receive_message(conn)
158             if data: ...
175
176         except ConnectionAbortedError:
177             print("Error in Peer Server: Connection to the server wa
178         except OSError:
179             print("Peer connection for download was forcibly closed.

server.py
44         return None # Connection closed
45         # Unpack the length
46         length = struct.unpack("I", packed_length)[0]
47         # Receive the actual message
48         message = conn.recv(length).decode('utf-8')
49         print(f"Received message: {message}")
50         return message
51
52 def listen(self, conn, addr):
53     self.clients.append((conn,addr))
54     auth = AuthController()
55     try:
56 >         while self.running:...
109     except OSError:
110         print("Connection to the client was forcibly closed.")
111         self.clients.remove((conn,addr))
112         auth.setStatus(0)
113
114     except Exception as e:
115         print(f"Exception in server listener listen: {e}")
116         auth.close()
117
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

[{"peo\_ping(1).png", "26224", "23:35:53-12/12/2023", "Downloaded\_from\_minhloc"}]  
Connection to the client was forcibly closed.

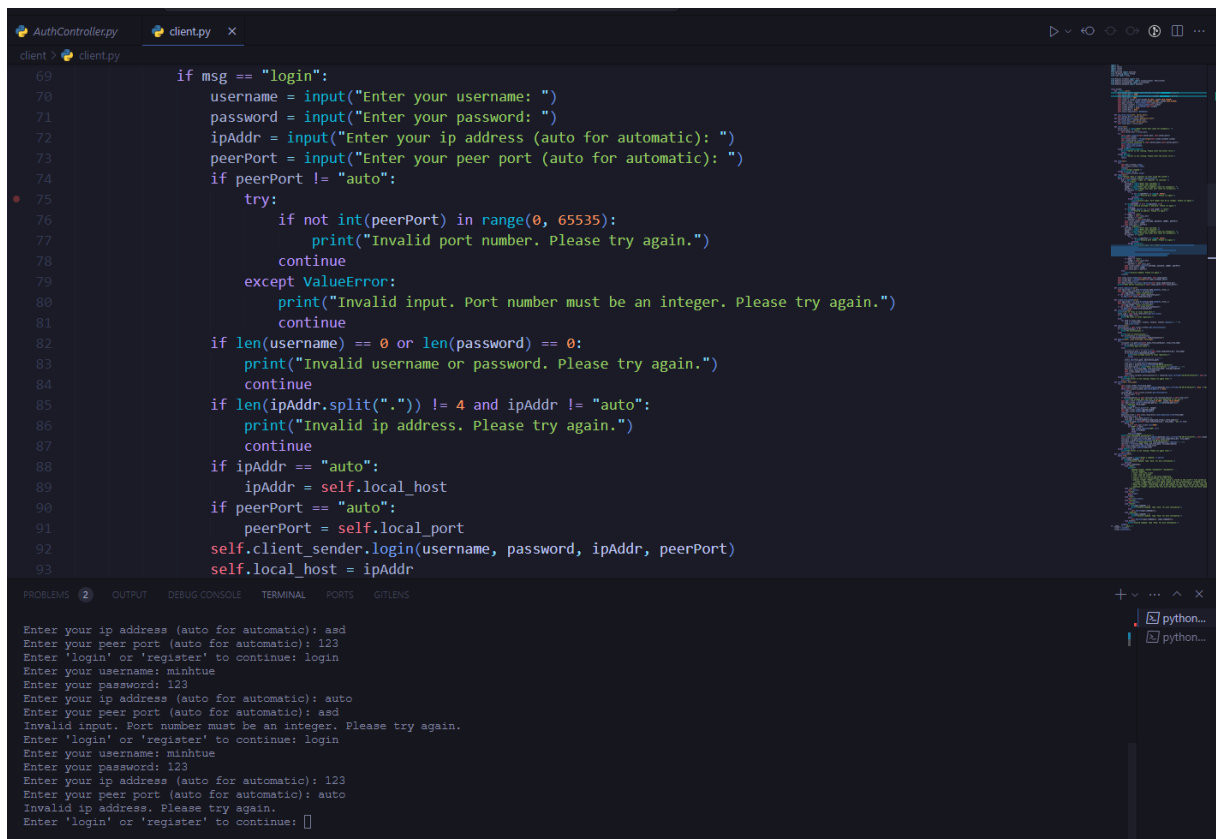
Enter a command: stop  
Client stopped.  
PS C:\Users\Admin\OneDrive\Desktop\Python\test\file-sharing\client>

Hình 39: Connection error

### 7.3. Login & Register: Wrong input format

Hệ thống sẽ kiểm tra các trường nhập vào có đúng định dạng hay không.

Nếu không đúng định dạng, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu nhập lại.



```
client.py
69         if msg == "login":
70             username = input("Enter your username: ")
71             password = input("Enter your password: ")
72             ipAddr = input("Enter your ip address (auto for automatic): ")
73             peerPort = input("Enter your peer port (auto for automatic): ")
74             if peerPort != "auto":
75                 try:
76                     if not int(peerPort) in range(0, 65535):
77                         print("Invalid port number. Please try again.")
78                         continue
79                 except ValueError:
80                     print("Invalid input. Port number must be an integer. Please try again.")
81                     continue
82             if len(username) == 0 or len(password) == 0:
83                 print("Invalid username or password. Please try again.")
84                 continue
85             if len(ipAddr.split(".")) != 4 and ipAddr != "auto":
86                 print("Invalid ip address. Please try again.")
87                 continue
88             if ipAddr == "auto":
89                 ipAddr = self.local_host
90             if peerPort == "auto":
91                 peerPort = self.local_port
92             self.client_sender.login(username, password, ipAddr, peerPort)
93             self.local_host = ipAddr

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS GIT LENS
Enter your ip address (auto for automatic): asd
Enter your peer port (auto for automatic): 123
Enter 'login' or 'register' to continue: login
Enter your username: minhtue
Enter your password: 123
Enter your ip address (auto for automatic): auto
Enter your peer port (auto for automatic): asd
Invalid input. Port number must be an integer. Please try again.
Enter 'login' or 'register' to continue: login
Enter your username: minhtue
Enter your password: 123
Enter your ip address (auto for automatic): 123
Enter your peer port (auto for automatic): auto
Invalid ip address. Please try again.
Enter 'login' or 'register' to continue: []
```

Hình 40: Wrong input format

### 7.4. Register: User already exist

Hệ thống sẽ kiểm tra xem username đã tồn tại hay chưa.

Nếu đã tồn tại, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu nhập lại.

```
9
10 def login(self, hostname, password, ipAddress, peerPort):
11     print(f"Login request: {hostname} {password} {ipAddress} {peerPort}")
12     self.user = self.database.get_user(hostname)
13     print(self.user)
14     if (not self.user):
15         return self.response_code.USER_NOT_FOUND()
16     elif (self.user[2] != password):
17         return self.response_code.WRONG_PASSWORD()
18     elif (self.user[5] == 1):
19         return self.response_code.USER_ALREADY_ONLINE()
20     else:
21         self.database.update_user(hostname, ipAddress, peerPort)
22         return self.response_code.LOGIN_SUCCESS()
23
24 def register(self, hostname, password, ipAddress, peerPort):
25     print(f"Register request: {hostname} {password} {ipAddress} {peerPort}")
26     if (self.database.get_user(hostname) != None):
27         return self.response_code.USER_ALREADY_EXISTS()
28     else:
```

Enter 'login' or 'register' to continue: register  
Enter your username: minhthue  
Enter your password: 123  
Enter your ip address (auto for automatic): auto  
Enter your peer port (auto for automatic): auto  
Sending message: register minhthue 123 192.168.1.8 5001  
Received response code: 201  
Registration successful  
Enter 'login' or 'register' to continue: register  
Enter your username: minhthue  
Enter your password: 123  
Enter your ip address (auto for automatic): auto  
Enter your peer port (auto for automatic): auto  
Sending message: register minhthue 123 192.168.1.8 5001  
Received response code: 404  
User already exists.  
Enter 'login' or 'register' to continue:

Hình 41: User already exist

## 7.5. Login: User not found

Hệ thống sẽ kiểm tra xem username đã tồn tại hay chưa.

Nếu chưa tồn tại, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu nhập lại.

```
9
10 def login(self, hostname, password, ipAddress, peerPort):
11     print(f"Login request: {hostname} {password} {ipAddress} {peerPort}")
12     self.user = self.database.get_user(hostname)
13     print(self.user)
14     if (not self.user):
15         return self.response_code.USER_NOT_FOUND()
16     elif (self.user[2] != password):
17         return self.response_code.WRONG_PASSWORD()
18     elif (self.user[5] == 1):
19         return self.response_code.USER_ALREADY_ONLINE()
20     else:
21         self.database.update_user(hostname, ipAddress, peerPort)
22         return self.response_code.LOGIN_SUCCESS()
23
```

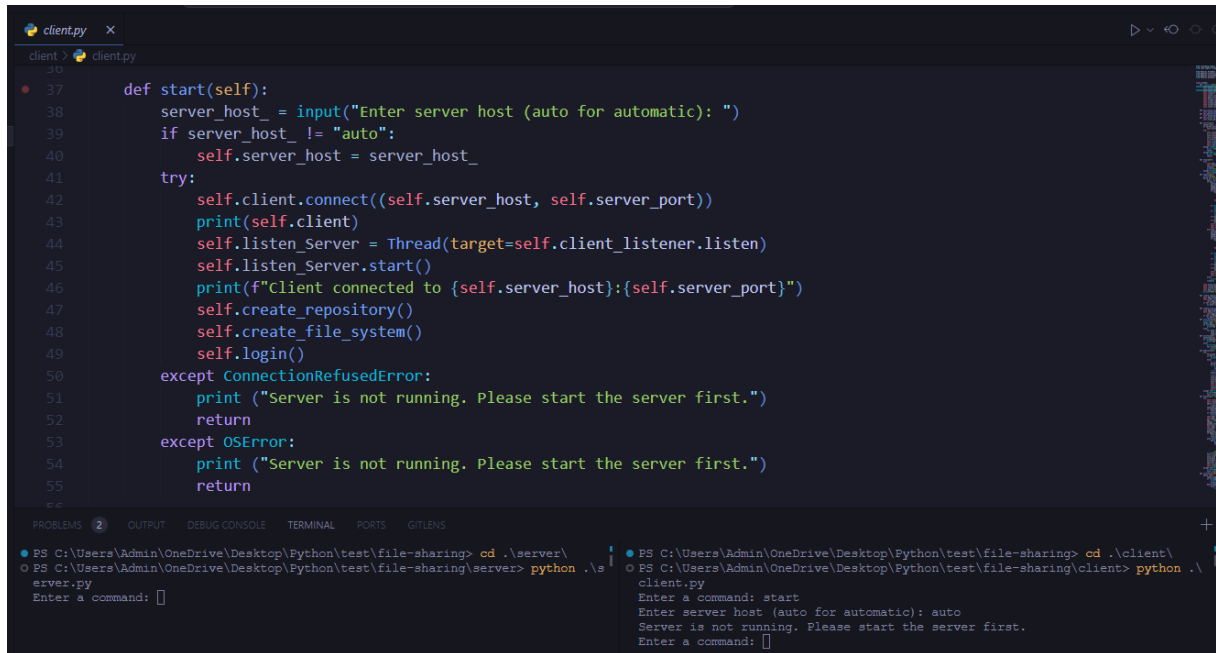
PS C:\Users\Admin\OneDrive\Desktop\Python\test\file-sharing> cd .\client\  
PS C:\Users\Admin\OneDrive\Desktop\Python\test\file-sharing\client> python .\client.py  
Enter a command: start  
Enter server host (auto for automatic): auto  
csocket.socket id=544, family=2, type=1, proto=0, laddr=('192.168.1.8', 63000), raddr=('192.168.1.8', 5000)>  
Client connected to 192.168.1.8:5000  
Please login or register to start using the system.  
Enter 'login' or 'register' to continue: login  
Enter your username: minhthue  
Enter your password: 123  
Enter your ip address (auto for automatic): auto  
Enter your peer port (auto for automatic): auto  
Sending message: login minhthue 123 192.168.1.8 5001  
Received response code: 401  
User not found.  
Enter 'login' or 'register' to continue: █

Hình 42: User not found

## 7.6. Login: Wrong password

Hệ thống sẽ kiểm tra xem password có đúng hay không.

Nếu không đúng, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu nhập lại.



```
client.py
def start(self):
    server_host_ = input("Enter server host (auto for automatic): ")
    if server_host_ != "auto":
        self.server_host = server_host_
    try:
        self.client.connect((self.server_host, self.server_port))
        print(self.client)
        self.listen_Server = Thread(target=self.client_listener.listen)
        self.listen_Server.start()
        print(f"Client connected to {self.server_host}:{self.server_port}")
        self.create_repository()
        self.create_file_system()
        self.login()
    except ConnectionRefusedError:
        print("Server is not running. Please start the server first.")
        return
    except OSError:
        print("Server is not running. Please start the server first.")
        return
```

```
PS C:\Users\Admin\OneDrive\Desktop\Python\test\file-sharing> cd .\server\
PS C:\Users\Admin\OneDrive\Desktop\Python\test\file-sharing\server> python .\server.py
Enter a command:
PS C:\Users\Admin\OneDrive\Desktop\Python\test\file-sharing> cd .\client\
PS C:\Users\Admin\OneDrive\Desktop\Python\test\file-sharing\client> python .\client.py
Enter a command: start
Enter server host (auto for automatic): auto
Server is not running. Please start the server first.
Enter a command:
```

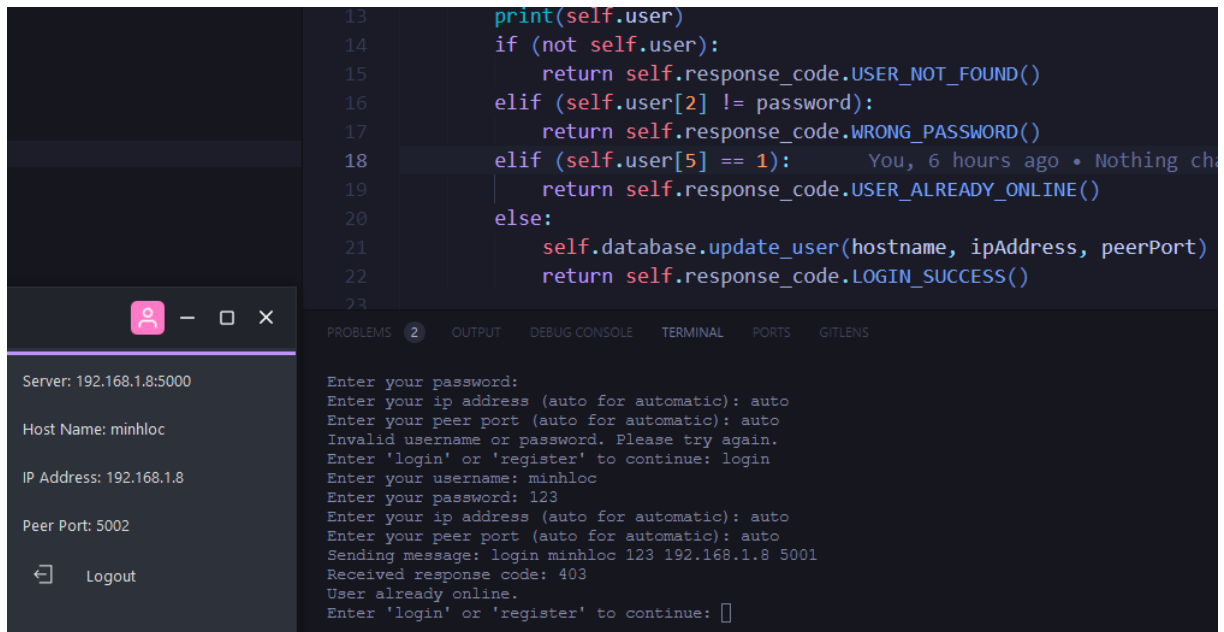
Hình 43: Wrong password

## 7.7. Login: User already online

Hệ thống sẽ kiểm tra xem username đã đăng nhập hay chưa.

Nếu đã đăng nhập, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu nhập lại.





```
13     print(self.user)
14     if (not self.user):
15         return self.response_code.USER_NOT_FOUND()
16     elif (self.user[2] != password):
17         return self.response_code.WRONG_PASSWORD()
18     elif (self.user[5] == 1):      You, 6 hours ago • Nothing ch
19         return self.response_code.USER_ALREADY_ONLINE()
20     else:
21         self.database.update_user(hostname, ipAddress, peerPort)
22         return self.response_code.LOGIN_SUCCESS()
23
```

Server: 192.168.1.8:5000

Host Name: minhloc

IP Address: 192.168.1.8

Peer Port: 5002

Logout

Enter your password:  
Enter your ip address (auto for automatic): auto  
Enter your peer port (auto for automatic): auto  
Invalid username or password. Please try again.  
Enter 'login' or 'register' to continue: login  
Enter your username: minhloc  
Enter your password: 123  
Enter your ip address (auto for automatic): auto  
Enter your peer port (auto for automatic): auto  
Sending message: login minhloc 123 192.168.1.8 5001  
Received response code: 403  
User already online.  
Enter 'login' or 'register' to continue: []

Hình 44: User already online

## 7.8. Publish: File not found

Hệ thống sẽ kiểm tra xem file có tồn tại hay không. Nếu không tồn tại, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu nhập lại.



```
159         for notification in notifications:
160             print(f"{notification[0]}: {notification[2]}")
161     def publish(self, local_file_name, file_name):
162         try:
163             file_path = os.path.join(self.local_file_system_dir, local_file_name)
164             if not os.path.exists(file_path):
165                 print("File does not exist.")
166             else:
167                 # Create the full destination path
168                 destination_path = os.path.join(self.local_respiratory_dir, file_name)
169                 if os.path.exists(destination_path):
170                     print("File already exists in local repository.")
171                 return
172             # Move the file to the destination directory
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

Received response code: 200  
Login successful.  
Peer Server listening on 192.168.1.8:5001  
Enter a command: publish momo.png momo.png  
File does not exist.  
Enter a command: █

Hình 45: File not found

## 7.9. Publish: File already published

Hệ thống sẽ kiểm tra xem file đã được Publish hay chưa. Nếu đã được Publish, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu nhập lại.

```
165         print( File does not exist. )
166     else:
167         # Create the full destination path
168         destination_path = os.path.join(self.local_respiratory_dir, file_name)
169         if os.path.exists(destination_path):
170             print("File already exists in local repository.")
171             return
172         # Move the file to the destination directory
173         shutil.move(file_path, destination_path)
174         # Update local_respiratory
175         file_size = os.path.getsize(destination_path)
176         file_date = datetime.now().strftime("%H:%M:%S-%d/%m/%Y")
177         file_description = input("Enter file description: ").replace(" ", "_")
178         new_file = File(file_name, file_size, file_date, file_description)
```

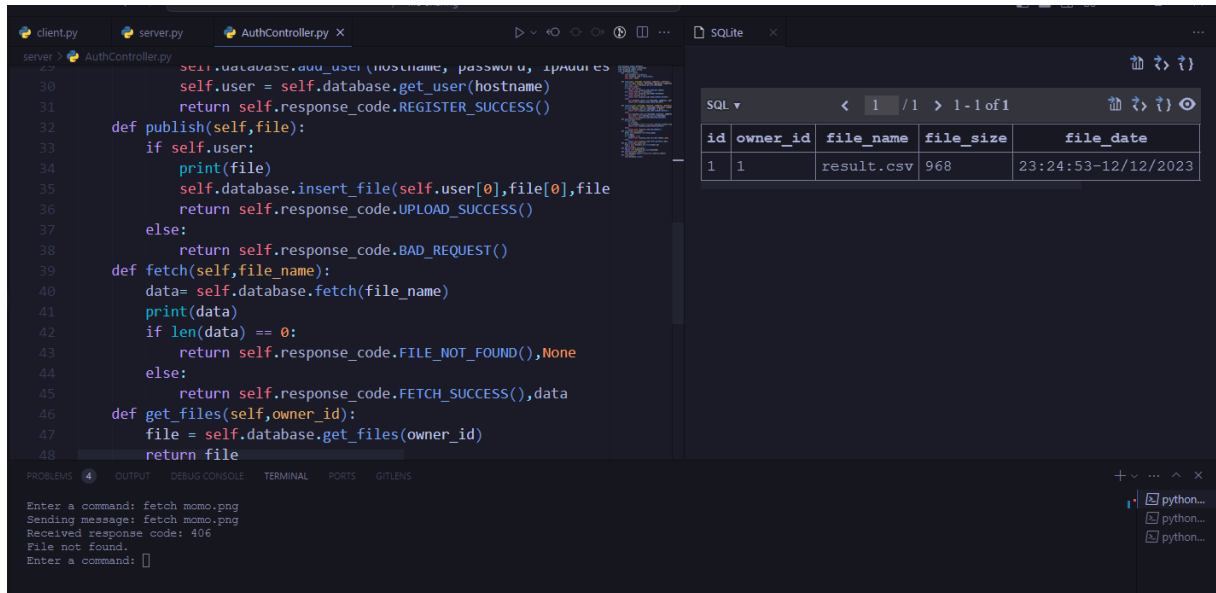
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

Received response code: 200  
Login successful.  
Peer Server listening on 192.168.1.8:5001  
Enter a command: publish momo.png momo.png  
File does not exist.  
Enter a command: publis result.csv result.csv  
Invalid command. Type 'help' for more information.  
Enter a command: publish result.csv result.csv  
Enter file description: xstk  
Sending message: publish result.csv 968 23:24:53-12/12/2023 xstk  
Received response code: 202  
File uploaded successfully.  
Enter a command: list  
List of files in local repository:  
File name: result.csv  
File size: 968B  
File date: 23:24:53-12/12/2023  
File description: xstk  
-----  
Enter a command: publish result.csvvv result.csv  
File does not exist.  
Enter a command: publish result.csv result.csv  
File does not exist.  
Enter a command: publish result.csv result.csv  
File already exists in local repository.  
Enter a command: []

Hình 46: File already published

## 7.10. Fetch: File not found

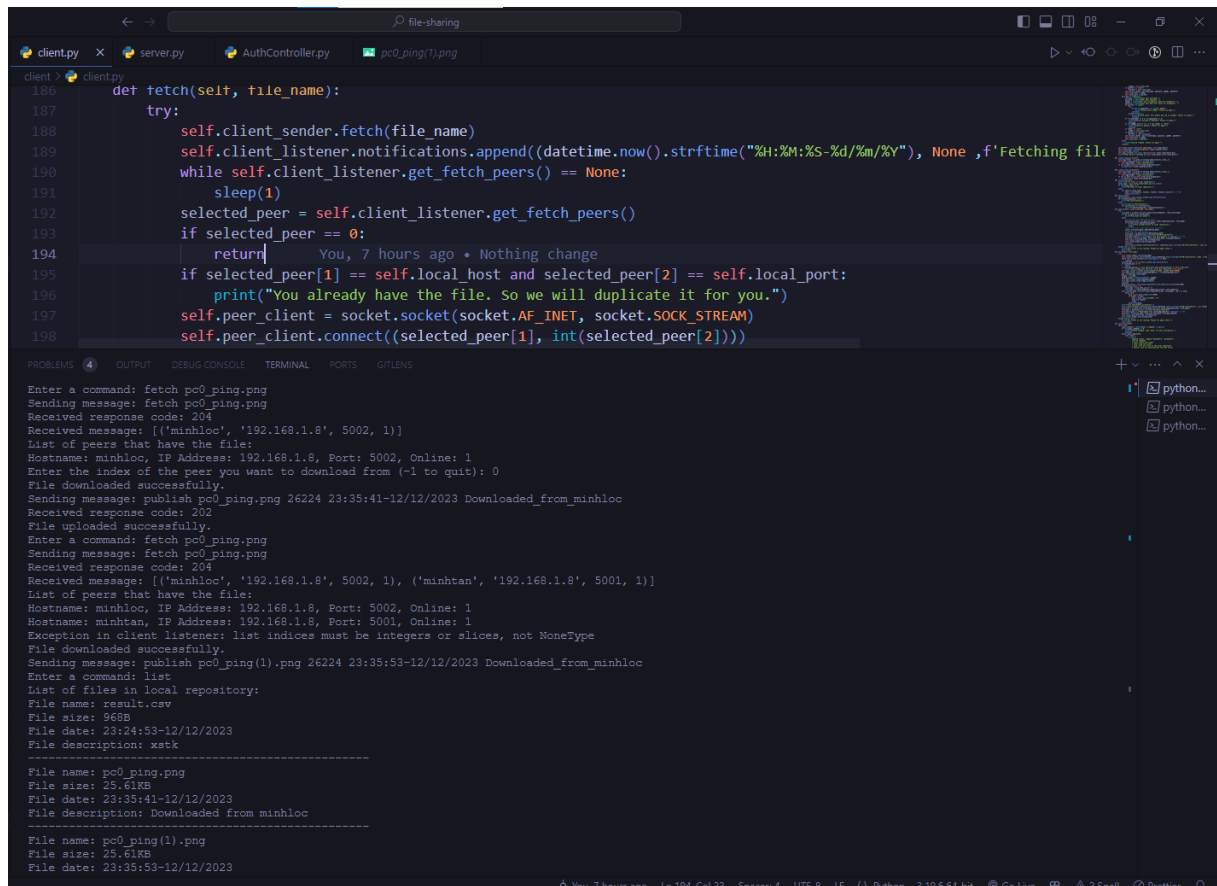
Hệ thống sẽ kiểm tra xem file có tồn tại hay không. Nếu không tồn tại, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu nhập lại.



Hình 47: File not found

## 7.11. Fetch: File already fetched

Hệ thống sẽ kiểm tra xem file đã được Fetch hay chưa. Nếu đã được fetch, hệ thống sẽ tự động duplicate file cho người dùng.



```
def fetch(self, file_name):
    try:
        self.client_sender.fetch(file_name)
        self.client_listener.notifications.append((datetime.now().strftime("%H:%M:%S-%d/%m/%Y"), None, f'Fetching file'))
        while self.client_listener.get_fetch_peers() == None:
            sleep(1)
        selected_peer = self.client_listener.get_fetch_peers()
        if selected_peer == 0:
            return You, 7 hours ago • Nothing change
        if selected_peer[1] == self.local_host and selected_peer[2] == self.local_port:
            print("You already have the file. So we will duplicate it for you.")
        self.peer_client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.peer_client.connect((selected_peer[1], int(selected_peer[2])))
```

Enter a command: fetch pc0\_ping.png  
Sending message: fetch pc0\_ping.png  
Received response code: 204  
Received message: [('minhloc', '192.168.1.8', 5002, 1)]  
List of peers that have the file:  
Hostname: minhloc, IP Address: 192.168.1.8, Port: 5002, Online: 1  
Enter the index of the peers you want to download from (~1 to quit): 0  
File downloaded successfully.  
Sending message: publish pc0\_ping.png 26224 23:35:41-12/12/2023 Downloaded\_from\_minhloc  
Received response code: 202  
File uploaded successfully.  
Enter a command: fetch pc0\_ping.png  
Sending message: fetch pc0\_ping.png  
Received response code: 204  
Received message: [('minhloc', '192.168.1.8', 5002, 1), ('minhtan', '192.168.1.8', 5001, 1)]  
List of peers that have the file:  
Hostname: minhloc, IP Address: 192.168.1.8, Port: 5002, Online: 1  
Hostname: minhtan, IP Address: 192.168.1.8, Port: 5001, Online: 1  
Exception in client listener: list indices must be integers or slices, not NoneType  
File downloaded successfully.  
Sending message: publish pc0\_ping(1).png 26224 23:35:53-12/12/2023 Downloaded\_from\_minhloc  
Enter a command: list  
List of files in local repository:  
File name: result.csv  
File size: 968B  
File date: 23:24:53-12/12/2023  
File description: xatk  
-----  
File name: pc0\_ping.png  
File size: 25.61KB  
File date: 23:35:41-12/12/2023  
File description: Downloaded from minhloc  
-----  
File name: pc0\_ping(1).png  
File size: 25.61KB  
File date: 23:35:53-12/12/2023

Hình 48: File already fetched

## 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](https://www.researchgate.net/publication/370038185_A_Virtual_Prototyping_Platform_for_Exploration_of_Vehicular_Electronics)