

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



**BÁO CÁO BÀI TẬP LỚN 1**  
**MẠNG MÁY TÍNH (CO3093)**  
**HỌC KỲ 241 - NĂM HỌC 2024-2025**

---

**NETWORK APPLICATION PROGRAMMING**  
**A SIMPLE FILE-SHARING APPLICATION**

---

GV hướng dẫn: Lê Bảo Khánh  
Sinh viên: Hồ Minh Hưng - 2211361  
Lê Thị Hoàng Anh - 2210070  
Nguyễn Minh Khánh - 2111493  
Nguyễn Tuấn Duy - 2110934

TP. Hồ Chí Minh, Tháng 11/2024



## Mục lục

<b>1 Lời mở đầu</b>	<b>2</b>
1.1 Tổng quan bài tập lớn . . . . .	2
1.2 Mục tiêu và kết quả cần đạt được . . . . .	2
<b>2 Cơ sở lý thuyết – Giải pháp đề xuất</b>	<b>2</b>
2.1 Mô hình Peer-to-Peer (P2P) tập trung . . . . .	2
2.1.1 Tổng quan . . . . .	2
2.1.2 Giao thức truyền dữ liệu . . . . .	3
2.2 Phát triển ứng dụng . . . . .	4
2.2.1 Ngôn ngữ sử dụng . . . . .	4
2.2.2 Giao diện người dùng . . . . .	4
2.2.3 Backend . . . . .	4
2.2.4 Hệ quản trị cơ sở dữ liệu . . . . .	4
2.3 Tổng quan công nghệ sử dụng . . . . .	5
<b>3 Yêu cầu sản phẩm</b>	<b>5</b>
3.1 Yêu cầu chức năng . . . . .	5
3.2 Yêu cầu phi chức năng . . . . .	5
3.3 Yêu cầu về quản lý dữ liệu . . . . .	6
<b>4 Thiết kế hệ thống</b>	<b>6</b>
4.1 Tổng quan hệ thống . . . . .	6
4.2 Chi tiết Server và một số hàm quan trọng . . . . .	8
4.3 Chi tiết Client . . . . .	8
<b>5 Kiểm thử</b>	<b>9</b>
5.1 Kiểm thử cài đặt hệ thống . . . . .	9
5.1.1 Kiểm tra kết nối với database . . . . .	9
5.1.2 Kiểm thử khả năng khởi động và ghi chép của server . . . . .	10
5.2 Kiểm thử chức năng . . . . .	11
5.2.1 Kiểm tra khả năng publish một file . . . . .	11
5.2.2 Kiểm tra khả năng fetch một file (từ nhiều client) . . . . .	12
5.2.3 Kiểm thử khả năng loại bỏ các pieces trùng lặp đã tồn tại trong database . . . . .	14
<b>6 Hướng dẫn người dùng</b>	<b>15</b>
6.1 Hướng dẫn cài đặt . . . . .	15
6.1.1 Đối với server . . . . .	15
6.1.2 Đối với client . . . . .	16
6.2 Thực hiện hệ thống . . . . .	17
6.2.1 Đối với quản trị viên . . . . .	17
6.2.2 Đối với người dùng khách . . . . .	17
<b>7 Tài liệu tham khảo</b>	<b>19</b>



## 1 Lời mở đầu

### 1.1 Tổng quan bài tập lớn

Bài tập lớn yêu cầu xây dựng một ứng dụng tương tự như Torrent dựa trên giao thức TCP/IP và hỗ trợ chuyển dữ liệu đa hướng (MDDT). Ứng dụng bao gồm hai loại host: tracker và node. Tracker đóng vai trò trung tâm, theo dõi các node đang kết nối đến và lưu trữ thông tin các phần dữ liệu có trên node. Node sẽ thông báo cho tracker về các tệp tin có trong khu lưu trữ cục bộ trên máy của node, nhưng không truyền dữ liệu tệp tin lên tracker. Khi một node cần một tệp nhưng không có trong kho lưu trữ, node có thể gửi request đến tracker. Từ đó, ứng dụng trả về các node khác nhau đang giữ các phần file (pieces of file), node request sẽ có khả năng tải xuống đồng thời nhiều tệp hoặc các phần của tệp từ nhiều node khác nhau. Vì vậy, ứng dụng yêu cầu việc triển khai node code phải đa luồng.

### 1.2 Mục tiêu và kết quả cần đạt được

Mục tiêu của bài tập lớn là xây dựng một ứng dụng đơn giản tương tự như Torrent dựa trên TCP/IP protocol với khả năng truyền dữ liệu đa chiều (MDDT). Kết quả cần đạt được bao gồm:

- Xây dựng hai loại host: Tracker và Node.
- Tracker quản lý được các node đã kết nối và thông tin các phần của file mà node đang giữ.
- Tracker có thể phản hồi các yêu cầu về thông tin của các node đang giữ các phần của file, hoặc về thông tin của các file đó.
- Node có thể thông báo cho Tracker về các files đang giữ trên local repository.
- Node có thể gửi yêu cầu cho tracker để lấy thông tin của các node đang giữ các phần của file hoặc thông tin của các file đó.
- Kết nối nhiều node với nhau để trao đổi nhiều files và các phần của file một cách đồng thời bằng đa luồng.
- Các node trao đổi file thành công.
- Ngoài ra, đảm bảo các tính năng người dùng khác.

## 2 Cơ sở lý thuyết – Giải pháp đề xuất

### 2.1 Mô hình Peer-to-Peer (P2P) tập trung

#### 2.1.1 Tổng quan

Mạng Peer-to-Peer (P2P) là một mô hình mạng trong đó các máy tính (gọi là peer) kết nối và chia sẻ tài nguyên trực tiếp với nhau mà không cần thông qua một máy chủ trung gian. Trong mạng này, mỗi peer có thể vừa là máy khách (client) vừa là máy chủ (server), giúp phân tán tài nguyên và tăng khả năng chia sẻ dữ liệu một cách hiệu quả. Các peer có thể trao đổi dữ liệu, tệp tin, và tài nguyên khác một cách trực tiếp mà không cần phụ thuộc vào hệ thống tập trung, giúp giảm thiểu tắc nghẽn mạng và tăng khả năng mở rộng. Mạng P2P có thể được phân loại dựa trên cách tổ chức và điều phối các peer: Hai loại chính là mạng P2P tập trung (centralized P2P) và mạng P2P phi tập trung (decentralized P2P).

Trong mạng P2P tập trung, một máy chủ trung tâm (tracker) được sử dụng để quản lý danh sách các peer và điều phối kết nối giữa các peer trong mạng. Khi một peer muốn tìm kiếm dữ liệu hoặc tài nguyên, nó sẽ liên hệ với tracker để biết các peer khác đang có dữ liệu mong muốn. Tracker đóng vai trò hỗ trợ việc kết nối ban đầu giữa các peer, sau đó các peer sẽ thực hiện việc truyền dữ liệu trực tiếp với nhau. Mô hình này dễ dàng triển khai và quản lý hơn so với mạng phi tập trung nhưng phụ thuộc vào máy chủ trung tâm, dẫn đến khả năng bị gián đoạn nếu máy chủ bị lỗi.



## 2.1.2 Giao thức truyền dữ liệu

### 2.1.2.1 Giao thức TCP/IP

TCP/IP (Transmission Control Protocol/Internet Protocol) là một bộ giao thức được sử dụng phổ biến cho truyền dữ liệu trên Internet và các mạng khác. TCP/IP bao gồm hai phần chính:

- Transmission Control Protocol (TCP): Là một giao thức truyền tải đáng tin cậy được sử dụng để chia nhỏ và lắp ráp các gói dữ liệu trước khi chúng được gửi qua mạng. TCP đảm bảo rằng các gói dữ liệu được gửi đi sẽ đến đích một cách chính xác và đúng thứ tự. Nó xác định các kết nối, đồng bộ hóa truyền tải và quản lý lưu lượng dữ liệu.
- Internet Protocol (IP): Là một giao thức định tuyến và chuyển tiếp dữ liệu trong mạng. IP định danh và địa chỉ hóa các thiết bị trong mạng, đồng thời quản lý việc định tuyến (routing) các gói dữ liệu qua mạng. Giao thức này đảm bảo gói dữ liệu được gửi đến đúng đích và định vị các thiết bị trong mạng.

Một số điểm nổi bật của giao thức TCP/IP được sử dụng trong ứng dụng:

- Kết nối: Sử dụng `socket.socket(socket.AF_INET, socket.SOCK_STREAM)` để tạo một socket TCP. Sau đó, gọi `sock.connect()` để kết nối đến một máy chủ xác định.
- Truyền dữ liệu: Gửi dữ liệu bằng `sock.sendall()` và nhận dữ liệu bằng `sock.recv()`.

### 2.1.2.2 Giao thức tự định nghĩa

Giao thức được thiết kế để phù hợp với yêu cầu của ứng dụng chia sẻ file. Giao thức này sử dụng định dạng JSON để định nghĩa các hành động và thông điệp giữa client và server.

Chức năng chính của giao thức bao gồm:

- **Định nghĩa các hành động:** Các hành động như xuất bản (publish), lấy file (fetch), và gửi file (send\_file) được xác định thông qua các thông điệp JSON.
- **Truyền thông điệp:** JSON cho phép truyền tải dữ liệu phức tạp, bao gồm các thông tin như tên file, kích thước file, và các mã băm (hash) của từng phần dữ liệu.

Cách thức hoạt động của giao thức có thể được mô tả như sau:

- Client gửi yêu cầu với thông điệp JSON chứa các thông tin cần thiết đến server. Ví dụ, khi client muốn xuất bản một file, nó sẽ gửi một thông điệp với các thông tin như tên file, kích thước file và mã băm của các phần.

```
{  
    "action": "publish",  
    "peers_port": peers_port,  
    "peers_hostname": peers_hostname,  
    "file_name": file_name,  
    "file_size": file_size,  
    "piece_hash": piece_hash,  
    "piece_size": piece_size,  
    "num_order_in_file": num_order_in_file  
}
```

- Server phân tích thông điệp và thực hiện hành động tương ứng, chẳng hạn như lưu trữ thông tin file hoặc gửi file về cho client.

Giao thức này cung cấp một lớp giao tiếp bổ sung cho giao thức TCP/IP, cho phép hệ thống hoạt động một cách hiệu quả và linh hoạt trong việc chia sẻ và quản lý file.



## 2.2 Phát triển ứng dụng

### 2.2.1 Ngôn ngữ sử dụng

Python là ngôn ngữ lập trình chính được sử dụng để phát triển ứng dụng. Với cú pháp đơn giản và dễ đọc, Python cho phép lập trình viên dễ dàng phát triển các tính năng phức tạp cũng như duy trì mã nguồn hiệu quả.

### 2.2.2 Giao diện người dùng

Giao diện người dùng của ứng dụng được xây dựng dưới dạng *Command Line Interface* (CLI). Điều này cho phép người dùng tương tác với hệ thống thông qua các lệnh nhập vào từ bàn phím. Việc sử dụng CLI giúp tối ưu hóa tài nguyên và cho phép người dùng thực hiện các tác vụ một cách nhanh chóng và hiệu quả.

Máy khách có trình thông dịch shell lệnh đơn giản được sử dụng để chấp nhận hai loại lệnh.

- **publish file\_name:** Một tập hợp các tệp cục bộ trên máy khách được chia thành các phần (kích thước 512kb). Người dùng có thể chọn các phần nội dung được chia sẻ để gửi thông tin đến máy chủ.
- **fetch file\_name:** Máy chủ gửi thông tin về các phần tệp được chia sẻ bởi các đối tác mà máy khách chưa có. Máy khách có thể chọn bất kỳ phần nào để tải xuống. Sau khi tất cả các phần được tải xuống, chúng sẽ được hợp nhất thành một tệp gốc.

### 2.2.3 Backend

Ứng dụng được phát triển với một số thư viện Python chính như sau:

- **socket:** Thư viện này được sử dụng để thiết lập kết nối mạng giữa máy chủ và máy khách, cho phép truyền tải dữ liệu qua giao thức TCP/IP. Nó cung cấp các phương thức để gửi và nhận dữ liệu, đảm bảo khả năng giao tiếp giữa các nút trong mạng.
- **threading:** Thư viện này cho phép xử lý đồng thời nhiều kết nối từ các máy khách khác nhau. Việc sử dụng threading giúp ứng dụng có thể phục vụ nhiều người dùng một cách hiệu quả mà không làm gián đoạn hoạt động của máy chủ.
- **logging:** Thư viện logging được sử dụng để ghi lại các sự kiện và lỗi trong quá trình hoạt động của ứng dụng. Điều này giúp theo dõi hiệu suất và phát hiện lỗi một cách nhanh chóng.
- **JSON:** Thư viện này hỗ trợ mã hóa và giải mã dữ liệu trong định dạng JSON, giúp việc truyền tải thông tin giữa máy chủ và máy khách trở nên dễ dàng và hiệu quả.
- **psycopg2:** Thư viện này cho phép kết nối và tương tác với cơ sở dữ liệu PostgreSQL, thực hiện các truy vấn SQL để lưu trữ và truy xuất dữ liệu liên quan đến người dùng và các tệp tin được chia sẻ.
- **tkinter:** Đây là một thư viện tạo giao diện người dùng (GUI), giúp người dùng dễ dàng tương tác với ứng dụng một cách trực quan.
- **hashlib:** Thư viện này gồm các thuật toán băm để tạo thành mã băm. Trong bài tập lần này, nhóm lựa chọn thuật toán SHA-1 trong thư viện hashlib.

### 2.2.4 Hệ quản trị cơ sở dữ liệu

(BONUS)

Hệ thống cơ sở dữ liệu được sử dụng trong ứng dụng là PostgreSQL, một hệ quản trị cơ sở dữ liệu quan hệ mạnh mẽ và linh hoạt. PostgreSQL cho phép lưu trữ dữ liệu một cách có tổ chức, đồng thời hỗ trợ thực hiện các truy vấn phức tạp để quản lý thông tin về các máy khách và tệp tin mà họ chia sẻ.



Mục tiêu	Giải pháp
Mô hình kết nối	Mô hình P2P tập trung
Giao thức truyền dữ liệu	TCP/IP
Version Control and Source Management	Git, GitHub
Ngôn ngữ lập trình	Python
Hệ quản trị cơ sở dữ liệu	PostgreSQL

### 2.3 Tổng quan công nghệ sử dụng

## 3 Yêu cầu sản phẩm

### 3.1 Yêu cầu chức năng

#### Yêu cầu đối với server:

- Quản lý dữ liệu của các client: Server trở thành quản lý của hệ thống truyền file, nhằm theo dõi client đang kết nối, file mà client đang giữ.
- Lưu trữ danh sách các client đang kết nối đến server và danh sách file của các client đó: Tính năng nhằm quản lý các file nằm ở máy hostname. Server tạo danh sách các client đã kết nối và các file của client đã kết nối. Khi client kết nối với server, server có thể theo dõi thông tin như địa chỉ IP, Port, trạng thái của client. Thông tin này giúp server xác định và quản lý các client một cách hiệu quả nhằm tránh, ngăn chặn các cuộc tấn công mạng hoặc giả mạo IP Address. Ngoài ra server còn tạo một danh sách để quản lý và tìm kiếm các file mà client đã đưa lên thông qua lệnh publish. Danh sách này có thể bao gồm thông tin của file như tên file, kích thước file, thông tin các piece như kích thước, mã hash của từng phần trong file, tên máy client chứa file hostname, địa chỉ IP và port đang chứa file. Bằng cách theo dõi thông tin này, server có thể dễ dàng biết và trả lời được thông tin về máy đang giữ file mà các máy khác yêu cầu đang cần.
- Phản hồi yêu cầu của client: Server và các client được kiến trúc theo mô hình Server-Client vì thế nên khi client gửi yêu cầu thì server sẽ nhận và xử lý nó. Sau đó, server lại gửi phản hồi mà client cần.

#### Yêu cầu đối với client:

- Gửi thông báo cho server các file mình có thể chia sẻ: Tính năng này sẽ gửi thông tin những file đang có ở máy Client lên server, sẽ có 2 trường hợp xảy ra: Nếu kết nối được đến Server, Client sẽ gửi danh sách tên file lên Server. Nếu không, Client sẽ báo lỗi.
- Gửi thông tin để tìm file trên cơ sở dữ liệu: Tính năng này sẽ gửi thông tin (tên file) lên Server, và sau đó Server sẽ thực hiện tìm kiếm file này ở các máy Client đang kết nối với Server và ở cơ sở dữ liệu. Sẽ có 2 trường hợp xảy ra:
  - Nếu tìm ra, Server sẽ gửi về Client đang thực hiện yêu cầu các thông tin về file (tên của file ở local, file đang ở client nào). Sau đó, nếu người dùng ở máy Client đang thực hiện yêu cầu muốn tải về file đấy, file sẽ được chuyển thẳng giữa 2 máy Client, không thông qua Server.
  - Nếu không tìm ra, hệ thống sẽ báo lỗi và kết thúc quá trình tìm kiếm. Khi này, nếu người dùng muốn, có thể nhập lại tên file và nhấn nút để tiếp tục thực hiện tìm kiếm mới.
- Hỗ trợ upload và download file chia sẻ với hai lệnh “publish” và “fetch”: Client sẽ hỗ trợ hai lệnh chính là “publish” và “fetch”.
  - publish file\_name:** Client có file tên file\_name sẽ được thêm vào cơ sở dữ liệu của server với nhiều thông tin liên quan đến client và file đó.
  - fetch file\_name:** Gửi yêu cầu tìm kiếm tệp file\_name lên server. Nếu tìm thấy, client sẽ nhận được thông tin kết nối của client đang chứa tập tin đó và tiến hành truyền tải dữ liệu.

### 3.2 Yêu cầu phi chức năng

Hiệu suất: Hệ thống sẽ hỗ trợ khả năng tạo nhiều kết nối đồng thời bằng cách tạo mỗi thread cho một kết nối, giúp tăng hiệu suất cũng như khả năng truyền tải.

Thiết kế dễ sử dụng: Với 3 câu lệnh “publish”, “fetch” và “exit”, người dùng đã có thể sử dụng được sản phẩm và kết nối cũng như chia sẻ nội dung với người dùng khác.

Có giao diện đồ họa người dùng: Ngoài việc sử dụng CLI, ứng dụng còn có GUI đối với những người chưa biết cách thao tác với CLI, giúp ứng dụng dễ tiếp cận và trực quan hơn.

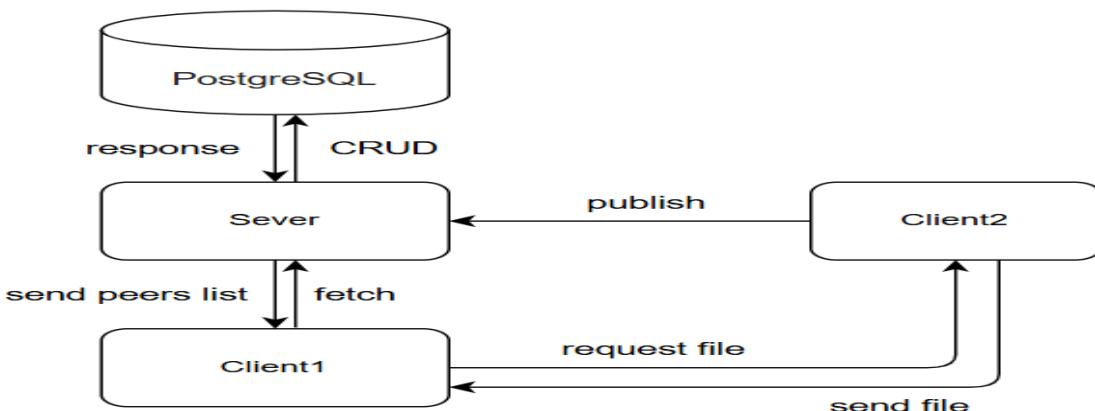
### 3.3 Yêu cầu về quản lý dữ liệu

**Server:** Thông tin của các tập tin gửi về server phải được lưu trữ trên hệ thống quản lý cơ sở dữ liệu. Ngoài ra, khi kết nối với client, server phải biết các thông tin cơ bản như tên, port của client đang kết nối với server cũng như thời điểm của hành động. Đồng thời phải loại bỏ đi các pieces bị trùng lặp của một file trên hệ thống.

**Client:** Tất cả thông tin chi tiết về các file khi gửi về server được lưu trên hệ quản trị cơ sở dữ liệu đều phải được lưu trữ và đảm bảo có thể truy cập dễ dàng và chính xác.

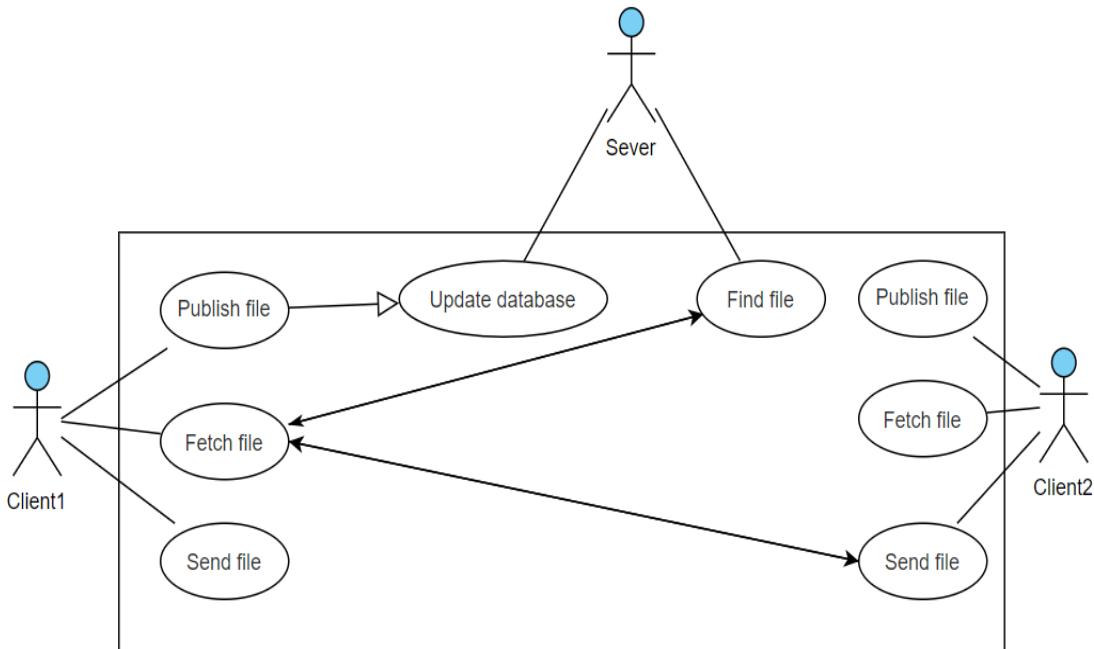
## 4 Thiết kế hệ thống

### 4.1 Tổng quan hệ thống

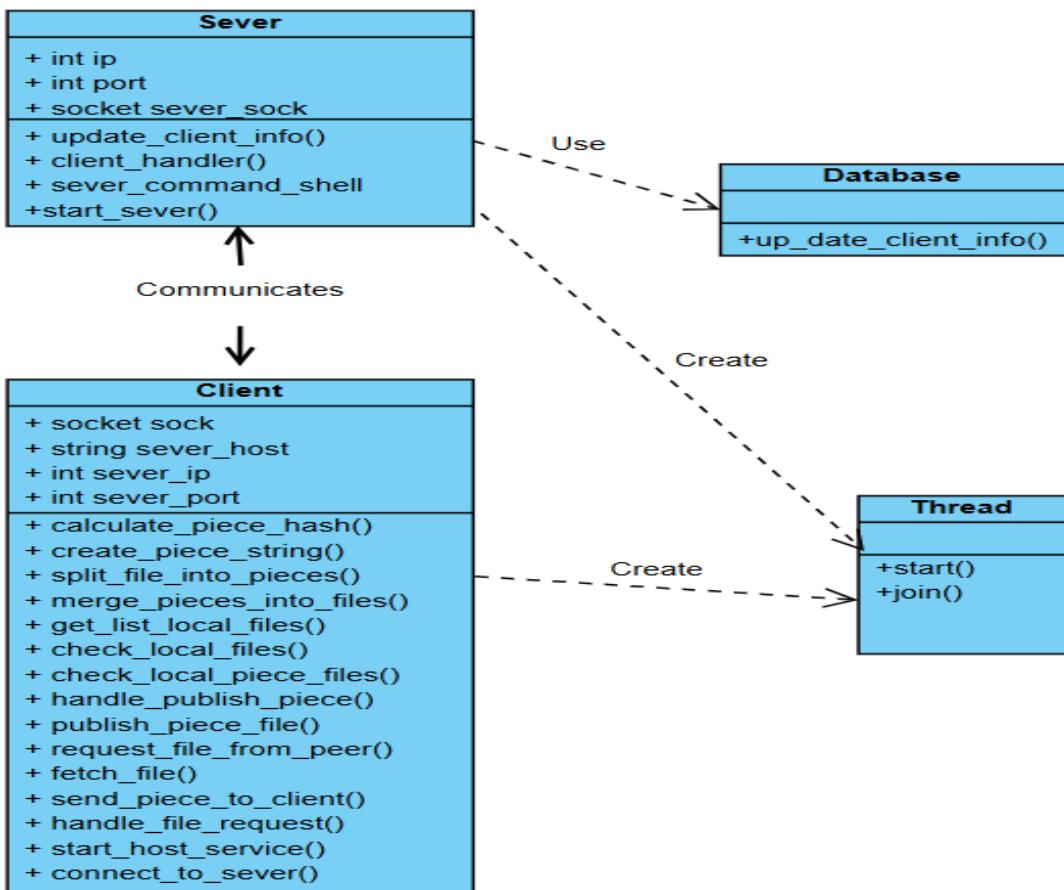


- Các client publish file để lưu thông tin file vào cơ sở dữ liệu.
- Khi muốn tải một file, client 1 gửi yêu cầu fetch tới sever để lấy các thông tin về các peer có các mảnh file. Sever sẽ gửi danh sách các peer có các mảnh file client 1 đang cần.
- Client 1 kết nối tới các peer đó và yêu cầu tải các mảnh file. Các peer gửi các mảnh file cho client 1.
- Các client đều có thể publish, fetch cũng như chia sẻ những gì mình đã tải cho các client khác.
- Danh sách các peer, lịch sử hoạt động, thông tin các file được chia sẻ đều được lưu trong Database và có thể thêm/xoá/đọc/sửa bởi sever.

### Use-case diagram



### Class diagram





## 4.2 Chi tiết Server và một số hàm quan trọng

- **Cấu hình Logging và kết nối cơ sở dữ liệu:** Server kết nối tới cơ sở dữ liệu PostgreSQL, tạo một con trỏ (cur) để thực hiện các thao tác với dữ liệu client và file.
- **Hàm log\_event:** Ghi lại các sự kiện của server để tiện trong việc theo dõi hoạt động.
- **Hàm update\_client\_info:** Lưu thông tin của client và các file được client chia sẻ vào cơ sở dữ liệu. Thông tin được lưu trữ bao gồm IP, port, hostname của client và các thuộc tính của file như tên, kích thước, mã hash của từng piece trong file. Khi có một piece cùng mã hash được gửi lên database, server sẽ xóa đi thông tin của piece cũ đó và cập nhật thông tin của piece mới lên để đảm bảo một piece chỉ có một mã hash.
- **Hàm client\_handler:** Xử lý kết nối của từng client trong một luồng riêng biệt. Các lệnh chính bao gồm:
  - introduce: Ghi nhận hostname của peer vào active\_connections, cho biết kết nối đang hoạt động.
  - publish: Peer thông báo rằng nó có một file để chia sẻ. Hàm update\_client\_info được gọi để cập nhật cơ sở dữ liệu với thông tin peer và file. Sau khi cập nhật thành công, gửi lại thông báo xác nhận.
  - fetch: Thực hiện truy vấn trong cơ sở dữ liệu để tìm các peer có cùng tên file nhưng khác hash và thứ tự của mảnh file, nhằm đảm bảo tìm được các mảnh file khác mà peer chưa có. Nếu tìm thấy các peer phù hợp, danh sách thông tin của các peer này được gửi về dưới dạng JSON. Nếu không có peer nào có file này, gửi lại thông báo lỗi.
  - file\_list: In ra danh sách các tệp được cung cấp bởi peer.
- **Hàm request\_file\_list\_from\_client:** Yêu cầu danh sách file từ client, dùng kết nối với một cổng cụ thể của client.
- **Hàm discover\_files:** Gọi hàm request\_file\_list\_from\_client để lấy danh sách file mà client sở hữu.
- **Hàm ping\_host:** Kiểm tra tình trạng kết nối với client bằng cách gửi lệnh ping.
- **Hàm server\_command\_shell:** Đọc các lệnh từ người dùng và thực thi chúng.
  - discover: Khám phá file từ một client cụ thể.
  - ping: Kiểm tra trạng thái kết nối của client.
  - exit: Thoát chương trình.
- **Hàm start\_server:** Tạo socket cho server, lắng nghe kết nối và xử lý từng client trong luồng riêng biệt.

**Khởi động server:** Tạo một luồng cho server để lắng nghe kết nối, sau đó mở server\_command\_shell để thực hiện các lệnh điều khiển.

**Tóm lại:** Server sử dụng mô hình Server-Client để quản lý danh sách các file được chia sẻ từ client, cung cấp tính năng yêu cầu và cập nhật file, hỗ trợ tìm kiếm và kiểm tra trạng thái client trong hệ thống.

## 4.3 Chi tiết Client

- **Hàm calculate\_piece\_hash :** Tính hash SHA-1 cho một mảnh file. Hash được sử dụng để xác thực tính toàn vẹn của mỗi mảnh file.
- **Hàm create\_pieces\_string :** Tạo ra danh sách hash từ các mảnh dữ liệu.
- **Hàm split\_file\_into\_pieces:** Chia file gốc thành các mảnh nhỏ có kích thước cố định (piece\_length). Kết quả là một danh sách chứa đường dẫn của các mảnh file vừa tạo.
- **Hàm merge\_pieces\_into\_file:** Ghép các mảnh file lại thành file hoàn chỉnh.
- **Hàm get\_list\_local\_files :** Lấy danh sách các file trong thư mục, danh sách này liệt kê các file có thể chia sẻ hoặc tải về từ peer khác.
- **Hàm check\_local\_files :** Kiểm tra xem file có tồn tại trong thư mục hiện tại hay không.
- **Hàm check\_local\_piece\_files :** Kiểm tra xem các piece\_file có tồn tại trong thư mục hiện tại hay không.



- **Hàm handle\_publish\_piece :** Xử lý yêu cầu publish các mảnh của file đã chọn, gồm hash và số thứ tự mảnh.
- **Hàm publish\_piece\_file:** Gửi thông tin publish (file name, file size, piece hash, và piece size) đến server qua socket.
- **Hàm request\_file\_from\_peer:** Tải về một mảnh dữ liệu từ peer đã chọn, yêu cầu tải file từ peer đó bằng cách kết nối qua peers\_ip và peer\_port.
- **Lớp PieceDownloader:** Lớp này chịu trách nhiệm tải xuống một piece cụ thể bằng một thread riêng biệt, giúp hiện thực hóa tải đồng thời các piece khác nhau.
- **Hàm fetch\_file:** Gửi yêu cầu fetch để lấy các thông tin về peer có mảnh file. Nếu tất cả các mảnh có sẵn từ các peer được xác định, các mảnh sẽ được ghép lại để tạo file hoàn chỉnh.
- **Hàm send\_piece\_to\_client:** Gửi một mảnh file đến peer yêu cầu.
- **Hàm handle\_file\_request:** Xử lý yêu cầu nhận mảnh file từ các peer khác.
- **Hàm start\_host\_service:** Tạo luồng lắng nghe các yêu cầu tải mảnh file từ peer khác trên port được chọn.
- **Hàm connect\_to\_server:** Thiết lập kết nối đến server và gửi thông tin giới thiệu peer (hostname và port).
- **Hàm main** Khởi tạo luồng dịch vụ host, thiết lập kết nối với server, và lắng nghe các lệnh từ các client (publish, fetch, hoặc exit).

## 5 Kiểm thử

### 5.1 Kiểm thử cài đặt hệ thống

#### 5.1.1 Kiểm tra kết nối với database

Các bước thực hiện như sau:

1. Khởi động server và thực hiện **publish** một file bất kỳ.

```
PS F:\HK241\MMT\BTL1> cd server
PS F:\HK241\MMT\BTL1\server> py server.py
Server command: 2024-10-31 00:17:16,254 - INFO - Server started and is listening for connections.
2024-10-31 00:17:38,118 - INFO - Active connections: 2
2024-10-31 00:17:38,118 - INFO - Connection established with VioletEvergarden/192.168.0.3:65433
2024-10-31 00:17:51,486 - INFO - Updating client info in database for hostname: VioletEvergarden/192.168.0.3:65433
2024-10-31 00:17:51,503 - INFO - Database update complete for hostname: VioletEvergarden/192.168.0.3:65433
```

2. Kiểm tra trên hệ QTCSQL đã có thông tin chưa. Nếu có tức là hệ thống đã kết nối client và server tới database thành công.

### 5.1.2 Kiểm thử khả năng khởi động và ghi chép của server

- ## 1. Khởi động server.

```
PS F:\HK241\MMT\BTL1> cd server  
PS F:\HK241\MMT\BTL1\server> py server.py  
Server command: 2024-10-31 00:17:16,254 - INFO - Server started and is listening for connections.  
2024-10-31 00:17:38,118 - INFO - Active connections: 2  
2024-10-31 00:17:38,118 - INFO - Connection established with VioletEvergarden/192.168.0.3:65433)  
2024-10-31 00:17:51,486 - INFO - Updating client info in database for hostname: VioletEvergarden/192.168.0.3:65433  
2024-10-31 00:17:51,503 - INFO - Database update complete for hostname: VioletEvergarden/192.168.0.3:65433
```

- Khởi động các client khác và kiểm tra server có đang lắng nghe không. Nếu có tức là server đã khởi động thành công.

```
PS F:\HK241\MMT\BTL1> cd client1  
PS F:\HK241\MMT\BTL1\client1> py client.py
```

```
PS F:\HK241\MMT\BT1> cd client3  
PS F:\HK241\MMT\BT1\client3> py client.py
```

```
PS F:\HK241\MMT\BTL1> cd server
PS F:\HK241\MMT\BTL1\server> py server.py
Server command: 2024-10-31 00:17:16,254 - INFO - Server started and is listening for connections.
2024-10-31 00:17:38,118 - INFO - Active connections: 2
2024-10-31 00:17:38,118 - INFO - Connection established with VioletEvergarden/192.168.0.3:65433)
2024-10-31 00:17:51,486 - INFO - Updating client info in database for hostname: VioletEvergarden/192.168.0.3:65433
2024-10-31 00:17:51,503 - INFO - Database update complete for hostname: VioletEvergarden/192.168.0.3:65433
2024-10-31 00:18:04,055 - INFO - Active connections: 3
2024-10-31 00:18:04,056 - INFO - Connection established with VioletEvergarden/192.168.0.3:65435)
2024-10-31 00:18:17,228 - INFO - Updating client info in database for hostname: VioletEvergarden/192.168.0.3:65435
```

## 5.2 Kiểm thử chức năng

### 5.2.1 Kiểm tra khả năng publish một file

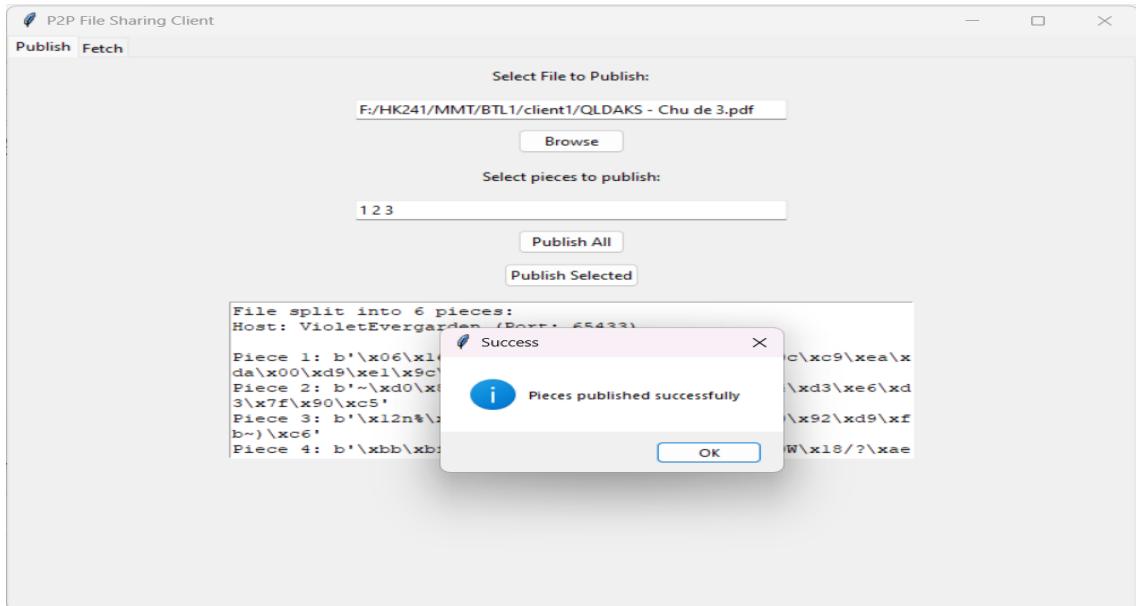
Phía server:

```
PS F:\HK241\MMT\BTL1> cd server  
PS F:\HK241\MMT\BTL1\server> py server.py  
Server command: 2024-10-31 00:17:16,254 - INFO - Server started and is listening for connections.  
2024-10-31 00:17:38,118 - INFO - Active connections: 2  
2024-10-31 00:17:38,118 - INFO - Connection established with VioletEvergarden/192.168.0.3:65433  
2024-10-31 00:17:51,486 - INFO - Updating client info in database for hostname: VioletEvergarden/192.168.0.3:65433  
2024-10-31 00:17:51,503 - INFO - Database update complete for hostname: VioletEvergarden/192.168.0.3:65433
```

Phia client:

```
PS F:\HK241\MMT\BTL1> cd client1
PS F:\HK241\MMT\BTL1> py client.py
Enter command (publish file_name/ fetch file_name/ exit): publish data.pdf
File data.pdf have ['data.pdf_piece1', 'data.pdf_piece2', 'data.pdf_piece3', 'data.pdf_piece4', 'data.pdf_piece5', 'data.pdf_piece6', 'data.pdf_piece7', 'data.pdf_piece8', 'data.pdf_piece9', 'data.pdf_piece10', 'data.pdf_piece11', 'data.pdf_piece12', 'data.pdf_piece13']
piece: ['b'z\xcd\x02\xe4?\x9e\xee\xbc~\g\x9f\x88\xfe\xc3\x06N\xd5\x0b\xa6'', "b'\x86\x8b_\xec\xd3\x86<\xd4\xcb\xbe\xf4\x00\xdc\xfd\xedv=\xdf\xf5\xad'", "b'\x97\x85tB~\x9b\xef.\x]\xb5:NT\xf03?1\x01'', "b'q\x01\xc9\x84\xab0\xd0\xf1\xfe\x8e\xc2\xe5\xeb\xf0jm\xa1\x7d\xc5'', "b'\x87\x9c\xr\xb3\xcb\xc5\xcb\xaa:\x7ewjfc\xfb\xd1N5'', "b'\xe4j?\xbd\xc7\x10s\xb1\x40A\x97\xad\x92\xaf\x11\x92\xf9'', "b'b\x2a\x25G,\x3c\r\x17Y\xf7\xe9\x86\xd5\x1ar\xb1X'', "b'\xdb\xe0\xca\t\xc8\xea\x00\xb2c\x\x9c\xb1&\#x4f}\xf6*\x7a'', "b'\xe4R\xde+\r7\xd1j\x12\x88\xde\xb4\xcd\x9b:\x81\xb1\xc5\x13'', "b'\xc1\xe\xfe\xb0C\x15\xf0\x9e\xf3\x19FD\xa6\x8a\xb9\xf9\x8a\xf0\xec'', "b'a\xea#\x08M\xcbe!\x8F\xdf\x1f\xde\x7c\x83q\xaf'', "b'\x1\xdfHz\xff&N\x7f\xae\x1f\x92hR\xeed2K\xe5Ay'', "b'\x8d\x1e\x88v\x\xeb\xcb*\x1l\x8b\x9c\xc9\x00\xm''.]
Please select num piece in file to publish:1 2 3 4 5 6 7 8 9
You was selected:
Number 1 : b'z\xcd\x02\xe4?\x9e\xee\xbc~\g\x9f\x88\xfe\xc3\x06N\xd5\x0b\xa6'
Number 2 : b'\x86\x8b_\xec\xd3\x86<\xd4\xcb\xbe\xf4\x00\xdc\xfd\xedv=\xdf\xf5\xad'
Number 3 : b'\x97\x85tB~\x9b\xef.\x]\xb5:NT\xf03?1\x01'
Number 4 : b'q\x01\xc9\x84\xab0\xd0\xf1\xfe\x8e\xc2\xe5\xeb\xf0jm\xa1\x7d\xc5'
Number 5 : b'\x87\x9c\xr\xb3\xcb\xc5\xcb\xaa:\x7ewjfc\xfb\xd1N5'
Number 6 : b'\xe4j?\xbd\xc7\x10s\xb1\x40A\x97\xad\x92\xaf\x11\x92\xf9'
Number 7 : b'b\x2a\x25G,\x3c\r\x17Y\xf7\xe9\x86\xd5\x1ar\xb1X'
Number 8 : b'\xdb\xe0\xca\t\xc8\xea\x00\xb2c\x\x9c\xb1&\#x4f}\xf6*\x7a'
Number 9 : b'\xe4R\xde+\r7\xd1j\x12\x88\xde\xb4\xcd\x9b:\x81\xb1\xc5\x13'
File list updated successfully.
Enter command (publish file name/ fetch file name/ exit):
```

Hoặc đổi với GUI của client:



Kết quả ở database với thao tác của GUI:

	peers_ip	peers_port	peers_hostname	file_name	file_size	piece_hash
1	192.168.1.17	65433	VioletEvergarden	QLDAKS - Chu de 3.pdf	3104577	b'\x06\x16\xcf\xbf\x1af\xb4SJ\xf8\x00\x0c\xc9\xea\xda\x00\xd9\xe1\x9c'
2	192.168.1.17	65433	VioletEvergarden	QLDAKS - Chu de 3.pdf	3104577	b'~\xd0\xbb\x0f\xba\xe2&\xa2\xd2\x18\x04\xd3\xe6\xd3\x7f\x90\xc5'
3	192.168.1.17	65433	VioletEvergarden	QLDAKS - Chu de 3.pdf	3104577	b'\x12n%\x99\x13\xcc\xe1\xa1'V[k\xde\xa0\x92\xd9\xfb~]\xc6'

### 5.2.2 Kiểm tra khả năng fetch một file (từ nhiều client)

Phía server:

```
PS F:\HK241\MMT\BTL1> cd server
PS F:\HK241\MMT\BTL1\server> py server.py
Server command: 2024-10-31 00:17:16,254 - INFO - Server started and is listening for connections.
2024-10-31 00:17:38,118 - INFO - Active connections: 2
2024-10-31 00:17:38,118 - INFO - Connection established with VioletEvergarden/192.168.0.3:65433
2024-10-31 00:17:51,486 - INFO - Updating client info in database for hostname: VioletEvergarden/192.168.0.3:65433
2024-10-31 00:17:51,503 - INFO - Database update complete for hostname: VioletEvergarden/192.168.0.3:65433
2024-10-31 00:18:04,055 - INFO - Active connections: 3
2024-10-31 00:18:04,056 - INFO - Connection established with VioletEvergarden/192.168.0.3:65435)
2024-10-31 00:18:17,228 - INFO - Updating client info in database for hostname: VioletEvergarden/192.168.0.3:65435
2024-10-31 00:18:17,233 - INFO - Database update complete for hostname: VioletEvergarden/192.168.0.3:65435
2024-10-31 00:19:08,062 - INFO - Active connections: 4
2024-10-31 00:19:08,063 - INFO - Connection established with VioletEvergarden/192.168.0.3:65434)
2024-10-31 00:25:32,948 - INFO - Connection with ('192.168.0.3', 60742) has been closed.
2024-10-31 00:25:48,100 - INFO - Active connections: 4
2024-10-31 00:25:48,100 - INFO - Connection established with VioletEvergarden/192.168.0.3:65434)
```

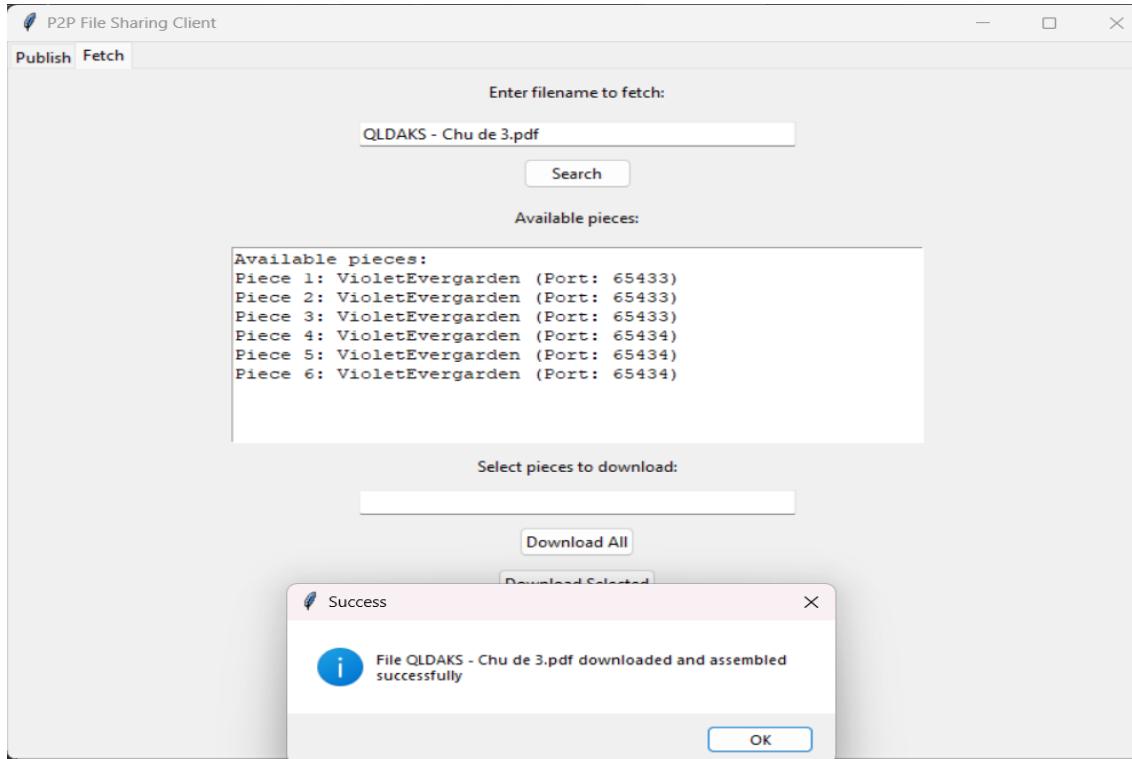
Phía client:



```
PS F:\HK241\MMT\BT11\client2> py client.py
Enter command (publish file name/ fetch file_name/ exit): fetch data.pdf
Hosts with the file data.pdf:
Number: 1 VioletEvergarden/192.168.0.3:65433 piece_hash: b'z\xcd\x02\xe4?\x9e\xeeN\xbc~g\x9f\x88\xfe\xc3\x06N\xd5\x0b\x
xa6' file_size: 6293030 piece_size: 524288 num_order_in_file: 1
Number: 2 VioletEvergarden/192.168.0.3:65433 piece_hash: b'\x86\x8b_\xecd\xd3\x86<\xd4\xcb\xbe\xf4\xao\xdc\xfd\xedv=\\xd
f\xf5\xad' file_size: 6293030 piece_size: 524288 num_order_in_file: 2
Number: 3 VioletEvergarden/192.168.0.3:65433 piece_hash: b'\x97\x85t\x9b\xe1\xfem.]\'bs:NT\xf03?1\x01' file_size: 62
93030 piece_size: 524288 num_order_in_file: 3
Number: 4 VioletEvergarden/192.168.0.3:65433 piece_hash: b'q\x01\xc9\x84\xb0\xd0\xf1\xfe\x8e\xc2\xe5\xeb\xf0jm\xa1\x
7d\xc5' file_size: 6293030 piece_size: 524288 num_order_in_file: 4
Number: 5 VioletEvergarden/192.168.0.3:65433 piece_hash: b'\x87\x9cXr\xb3\xcb\xc5i\xcb\xaa:\x87ljc\xfb\xd1Nz5' file_si
ze: 6293030 piece_size: 524288 num_order_in_file: 5
Number: 6 VioletEvergarden/192.168.0.3:65433 piece_hash: b'\xe4j?\xb0\xc7\x05\xb1\x4oA\x97\xad\x92\xaf\\\'x92\xf
9' file_size: 6293030 piece_size: 524288 num_order_in_file: 6
Number: 7 VioletEvergarden/192.168.0.3:65435 piece_hash: b'b\x2sG,\xc3\r\x17]Y\xf7\xe9\x86\xd5\x1ar\xb1}X' file_size
: 6293030 piece_size: 524288 num_order_in_file: 7
Number: 8 VioletEvergarden/192.168.0.3:65435 piece_hash: b'\xd0\xe0"\xca\t\xc8\xea\x00\xb2c\xd9\x9c\xb1&\#xf4}\xf6*\x
a7' file_size: 6293030 piece_size: 524288 num_order_in_file: 8
Number: 9 VioletEvergarden/192.168.0.3:65435 piece_hash: b'\xe4R\xde+\r7\xd1j\x12\x88\xde\xb4\xcd\x9b:\x81\xb1\xc5\x
3;' file_size: 6293030 piece_size: 524288 num_order_in_file: 9
Number: 10 VioletEvergarden/192.168.0.3:65435 piece_hash: b'\xc1\x8e\xfe\xb0\x15\xf0\x9e\xf3\x19FD\xa6\x8\xb9\xf9\x
a8\xf0\xec' file_size: 6293030 piece_size: 524288 num_order_in_file: 10
Number: 11 VioletEvergarden/192.168.0.3:65435 piece_hash: b'a\xea#\x88M\xbe!\xb8-F\xdf1\xdeT\x8c3q\xaf' file_size: 6
293030 piece_size: 524288 num_order_in_file: 11
Number: 12 VioletEvergarden/192.168.0.3:65435 piece_hash: b'1\xdfHz\xff&N\x7F\xae\x1f\x92hR\xeed2K\xe5Ay' file_size: 6
293030 piece_size: 524288 num_order_in_file: 12
Number: 13 VioletEvergarden/192.168.0.3:65435 piece_hash: b'\x8d\x1e~B\x88v&\x0b"\xcb*\n1\x8bJ\xc9\xc9o$' file_size:
6293030 piece_size: 524288 num_order_in_file: 13
Enter the number of the host to download from (or 'all' to download all pieces): all
Piece of file: data.pdf_piece1 has been fetched from peer.
Piece of file: data.pdf_piece2 has been fetched from peer.
Piece of file: data.pdf_piece3 has been fetched from peer.
Piece of file: data.pdf_piece4 has been fetched from peer.
Piece of file: data.pdf_pieces has been fetched from peer.
Piece of file: data.pdf_piece6 has been fetched from peer.
Piece of file: data.pdf_piece7 has been fetched from peer.
Piece of file: data.pdf_piece8 has been fetched from peer.
Piece of file: data.pdf_piece9 has been fetched from peer.
Piece of file: data.pdf_piece10 has been fetched from peer.
```

Piece of file: data.pdf\_piece11 has been fetched from peer.  
Piece of file: data.pdf\_piece12 has been fetched from peer.  
Piece of file: data.pdf\_piece13 has been fetched from peer.  
Got all the parts and created the file data.pdf  
Enter command (publish file name/ fetch file name/ exit):

Hoặc đối với GUI:



### 5.2.3 Kiểm thử khả năng loại bỏ các pieces trùng lắp đã tồn tại trong database

Ta sẽ cho client1 publish các pieces từ 1 đến 9 và client2 là từ 7 đến 13. Sau đó xem trong database liệu có sự trùng lặp nào không.

```
PS F:\HK241\MMT\BT1> cd client1
PS F:\HK241\MMT\BT1\client1> py client.py
Enter command (publish file_name/ fetch file_name/ exit): publish data.pdf
File data.pdf have ['data.pdf_piece1', 'data.pdf_piece2', 'data.pdf_piece3', 'data.pdf_piece4', 'data.pdf_piece5', 'data.pdf_piece6', 'data.pdf_piece7', 'data.pdf_piece8', 'data.pdf_piece9', 'data.pdf_piece10', 'data.pdf_piece11', 'data.pdf_piece12', 'data.pdf_piece13']
piece: ["b'z\xcd\x02\xe4?\x9e\xeeN\xbcg\x9f\x88\xfe\xc3\x06N\xd5\x0b\xa6'", "b'\\x86\x8b_\xec\xd3\x86\xd4\xcb\xbe\xfa\x0d\xcd\xfd\xedv=\xfdf\xf5\xad'", "b'\\x97\x85tB-\x9b\xe1\xfem.\x9b\xe1\xf0jm\xa1\xad7d\xc5'", "b'\\x87\x9cR\xb3\xcb\xc5i\xxa\xaa:\x9ewj\xfb\xd1Nz5'", "b'\\xe4j?\x9bd\xc7\x10s\xb1\x40a\x97\xad\x92\x91\x92\xf9'", "b'b\x2a\x25G,\x9c3\r\x17Y\x9f\x9e\x86\xd5\xar\xb1}X'", "b'\\xdb\xe0\xca\t\xc8\xea\x00\xb2c\xd9\x9c\xb1&\#x4f4}\x9f6*\x9a7'", "b'\\xe4R\xde+\r7\xd1j\x12\x88\xde\xb4\xcd\x9b:\x81\xb1\xc5\x13;'", "b'\\x1\x8e\xfe\xb0\x15\xf00\x9e\xf3\x19FD\xa6\xaa\xb9\xf9\xaa\xf0\xec'", "b'a\xea#\x08'M\xcbe!\x8b\xf1\xdf1\xde\x8c3\xaa\xf1'", "b'1\xdfHz\x9f8m\x7f\xae\x1f\x92hR\xeed2k\x5Ay'", "b'\\x8d\xte~\b\x88&\x0b\xcb*\x1l\x8b\x9c\x90$ml'"].
Please select num piece in file to publish:1 2 3 4 5 6 7 8 9
You was selected:
Number 1 : b'z\xcd\x02\xe4?\x9e\xeeN\xbcg\x9f\x88\xfe\xc3\x06N\xd5\x0b\xa6'
Number 2 : b'\x86\x8b_\xec\xd3\x86\xd4\xcb\xbe\xfa\x0d\xcd\xfd\xedv=\xfdf\xf5\xad'
Number 3 : b'\x97\x85tB-\x9b\xe1\xfem.\x9b\xe1\xf0jm\xa1\xad7d\xc5'
Number 4 : b'q\x01\xc9\x84\xb0\xd0\xf1\xfe\x8e\xc2\xe5\xeb\xf0jm\xa1\xad7d\xc5'
Number 5 : b'\x87\x9cR\xb3\xcb\xc5i\xcb\xaa:\x9ewj\xfb\xd1Nz5'
Number 6 : b'\\xe4j?\x9bd\xc7\x10s\xb1\x40a\x97\xad\x92\x91\x92\xf9'
Number 7 : b'b\x2a\x25G,\x9c3\r\x17Y\x9f\x9e\x86\xd5\xar\xb1}X'
Number 8 : b'\xdb\xea\xca\t\xc8\xea\x00\xb2c\xd0\x9c\xb1&\#x4f4}\x9f6*\xaa\xf1'
Number 9 : b'\\xe4R\xde+\r7\xd1j\x12\x88\xde\xb4\xcd\x9b:\x81\xb1\xc5\x13;
File list updated successfully.
```



```
PS F:\HK241\MMT\BTL1> cd client3
PS F:\HK241\MMT\BTL1\client3> py client.py
Enter command (publish file_name/ fetch file_name/ exit): publish data.pdf
File data.pdf have ['data.pdf_piece1', 'data.pdf_piece2', 'data.pdf_piece3', 'data.pdf_piece4', 'data.pdf_piece5', 'data.pdf_piece6', 'data.pdf_piece7', 'data.pdf_piece8', 'data.pdf_piece9', 'data.pdf_piece10', 'data.pdf_piece11', 'data.pdf_piece12', 'data.pdf_piece13']
piece: ["b'z\\xcd\\x02\\xe4?\\x9e\\xeeN\\xbc~g\\x9f\\x88\\xfe\\xc3\\x06N\\xd5\\x0b\\xa6'", "b'\\x86\\x8b_\\xec\\xd3\\x86<\\xd4\\xcb\\xbe\\xf4\\xae\\xdc\\xfd\\xedv=\\xdf\\xf5\\xad'", "b'\\x97\\x85b~\\xb0\\xe1\\xfem.]\\xb5:NT\\xf03?1\\x01'", "b'q\\x01\\xc9\\x84\\xb0g\\xd0\\xf1\\xfe\\xc2\\xe5\\xeb\\xf0jm\\xa1\\xa7d\\xc5'", "b'\\x87\\x9c\\r\\xb3\\xcb\\xc5i\\xcb\\xaa:\\xe7Wjc\\xfb\\xd1NZ'", "b'\\xe4j?\\xbd\\xc7\\x10s\\xb1\\xa4oA\\x97\\xad\\x92\\x91\\xf9'", "b'b\\xa25g,\\xc3\\r\\x17Y\\xf7\\x9e\\x86\\xd5\\x1ar\\xb1}X'", "b'\\xe4R\\xde+\\r7\\xd1j\\x12\\x88\\xde\\xb4\\xcd\\xb9:\\x81\\xb1\\xc5\\x13;', "b'\\x9c\\x84\\xf6*\\xa7\\'', "b'\\xe4R\\xde+\\r7\\xd1j\\x12\\x88\\xde\\xb4\\xcd\\xb9:\\x81\\xb1\\xc5\\x13;'", "b'\\xc1\\x8e\\xfe\\xb0C\\x15\\xf00\\x9e\\xf3\\x19FD\\xa6\\xa8\\xb9\\xf9\\xa8\\xf0\\xec'", "b'a\\xea#\\x08'M\\xcb!\\xb8-F\\xdfHz\\xff&N\\x7f\\xae\\x1f\\x92hR\\xeed2K\\xe5Ay'", "b'\\x8d\\x1e-B\\x88v&\\xb0\"\\xcb*\\n\\x8bJ\\xc9\\xc9o$\\m'"]
Please select num piece in file to publish:7 8 9 10 11 12 13
You was selected:
Number 7 : b'b\\xa25g,\\xc3\\r\\x17Y\\xf7\\x9e\\x86\\xd5\\x1ar\\xb1}X'
Number 8 : b'\\xd\\xe0\\xca\\t\\xc8\\xae\\x00\\xb2c\\xd9\\x9c\\xb1&#\\xf4\\xf6*\\xa7'
Number 9 : b'\\xe4R\\xde+\\r7\\xd1j\\x12\\x88\\xde\\xb4\\xcd\\xb9:\\x81\\xb1\\xc5\\x13;'
Number 10 : b'\\xc1\\x8e\\xfe\\xb0C\\x15\\xf00\\x9e\\xf3\\x19FD\\xa6\\xa8\\xb9\\xf9\\xa8\\xf0\\xec'
Number 11 : b'a\\xea#\\x08'M\\xcb!\\xb8-F\\xdfHz\\xff&N\\x7f\\xae\\x1f\\x92hR\\xeed2K\\xe5Ay'
Number 12 : b'1\\xdftHz\\xff&N\\x7f\\xae\\x1f\\x92hR\\xeed2K\\xe5Ay'
Number 13 : b'\\x8d\\x1e-B\\x88v&\\xb0\"\\xcb*\\n\\x8bJ\\xc9\\xc9o$\\m'
File list updated successfully.
```

peers_ip	peers_port	peers_hostname	file_name	file_size	piece_hash
character varying (255)					
1 192.168.0.3	65433	VioletEvergarden	data.pdf	6293030	b'z\\xcd\\x02\\xe4?\\x9e\\xeeN\\xbc~g\\x9f\\x88\\xfe\\xc3\\x06N\\xd5\\x0b\\x
2 192.168.0.3	65433	VioletEvergarden	data.pdf	6293030	b'\\x86\\x8b_\\xec\\xd3\\x86<\\xd4\\xcb\\xbe\\xf4\\xae\\x00\\xd\\xedv=\\xdf\\xf5\\xad'
3 192.168.0.3	65433	VioletEvergarden	data.pdf	6293030	b'\\x97\\x85b~\\xb0\\xe1\\xfem.]\\xb5:NT\\xf03?1\\x01'
4 192.168.0.3	65433	VioletEvergarden	data.pdf	6293030	b'q\\x01\\xc9\\x84\\xb0g\\xd0\\xf1\\xfe\\xc2\\xe5\\xeb\\xf0jm\\xa1\\xa7d\\xc5'
5 192.168.0.3	65433	VioletEvergarden	data.pdf	6293030	b'\\x87\\x9c\\r\\xb3\\xcb\\xc5i\\xcb\\xaa:\\xe7Wjc\\xfb\\xd1NZ'
6 192.168.0.3	65433	VioletEvergarden	data.pdf	6293030	b'\\xe4j?\\xbd\\xc7\\x10s\\xb1\\xa4oA\\x97\\xad\\x92\\x91\\xf9'
7 192.168.0.3	65435	VioletEvergarden	data.pdf	6293030	b'\\xa25g,\\xc3\\r\\x17Y\\xf7\\x9e\\x86\\xd5\\x1ar\\xb1}X'
8 192.168.0.3	65435	VioletEvergarden	data.pdf	6293030	b'\\xd\\xe0\\xca\\t\\xc8\\xae\\x00\\xb2c\\xd9\\x9c\\xb1&#\\xf4\\xf6*\\xa7'
9 192.168.0.3	65435	VioletEvergarden	data.pdf	6293030	b'\\xe4R\\xde+\\r7\\xd1j\\x12\\x88\\xde\\xb4\\xcd\\xb9:\\x81\\xb1\\xc5\\x13;'
10 192.168.0.3	65435	VioletEvergarden	data.pdf	6293030	b'\\xc1\\x8e\\xfe\\xb0C\\x15\\xf00\\x9e\\xf3\\x19FD\\xa6\\xa8\\xb9\\xf9\\xa8\\xf0\\xec'
11 192.168.0.3	65435	VioletEvergarden	data.pdf	6293030	b'a\\xea#\\x08'M\\xcb!\\xb8-F\\xdfHz\\xff&N\\x7f\\xae\\x1f\\x92hR\\xeed2K\\xe5Ay'
12 192.168.0.3	65435	VioletEvergarden	data.pdf	6293030	b'1\\xdftHz\\xff&N\\x7f\\xae\\x1f\\x92hR\\xeed2K\\xe5Ay'
13 192.168.0.3	65435	VioletEvergarden	data.pdf	6293030	b'\\x8d\\x1e-B\\x88v&\\xb0\"\\xcb*\\n\\x8bJ\\xc9\\xc9o\$\\m'

## 6 Hướng dẫn người dùng

### 6.1 Hướng dẫn cài đặt

#### 6.1.1 Đôi với server

Đảm bảo các thư viện logging, socket, threading, json, psycopg2, sys của Python đã được tải về.

Tải và cài đặt PostgreSQL, pgAdmin4. Sau đó thiết lập thông tin cơ bản cho database bằng SQL Shell (sử dụng các thiết lập mặc định).

Tạo bảng dữ liệu cho database bằng lệnh:

```
CREATE TABLE peers (
    peers_ip VARCHAR(255),
    peers_port VARCHAR(255),
    peers_hostname VARCHAR(255),
    file_name VARCHAR(255),
    file_size VARCHAR(255),
    piece_hash VARCHAR(255),
    piece_size VARCHAR(255),
```



```
    num_order_in_file VARCHAR(255)
);
```

Kiểm tra địa chỉ IP của máy tính bằng lệnh “ipconfig /all” trong Command Prompt.

```
Wireless LAN adapter Wi-Fi:

Connection-specific DNS Suffix . :
Description . . . . . : Killer(R) Wi-Fi 6 AX1650i 160MHz Wireless Network Adapter (201NGW)
Physical Address. . . . . : 4C-03-4F-E2-51-35
DHCP Enabled. . . . . : Yes
Autoconfiguration Enabled . . . . . : Yes
Link-local IPv6 Address . . . . . : fe80::552:fb1e:652e:6a5b%8(PREFERRED)
IPv4 Address. . . . . : 192.168.0.3(PREFERRED)
Subnet Mask . . . . . : 255.255.255.0
```

Với địa chỉ IP của máy tính là 192.168.0.3, ta sẽ thiết lập địa chỉ IP này cho server.py

```
if __name__ == "__main__":
    SERVER_HOST = '192.168.0.3'
    SERVER_PORT = 65432
    #SERVER_HOST='0.0.0.0'
    # Start server in a separate thread
    server_thread = threading.Thread(target=start_server)
    server_thread.start()

    # Start the server command shell in the main thread
    server_command_shell()

    # Signal the server to shutdown
    print("Server shutdown requested.")

    sys.exit(0)
```

### 6.1.2 Đổi với client

Đảm bảo các thư viện socket, json, os, threading, shlex, hashlib, math, tkinter của Python đã được cài đặt. Kiểm tra địa chỉ IP của máy tính bằng lệnh “ipconfig /all” trong Command Prompt.

```
Wireless LAN adapter Wi-Fi:

Connection-specific DNS Suffix . :
Description . . . . . : Killer(R) Wi-Fi 6 AX1650i 160MHz Wireless Network Adapter (201NGW)
Physical Address. . . . . : 4C-03-4F-E2-51-35
DHCP Enabled. . . . . : Yes
Autoconfiguration Enabled . . . . . : Yes
Link-local IPv6 Address . . . . . : fe80::552:fb1e:652e:6a5b%8(PREFERRED)
IPv4 Address. . . . . : 192.168.0.3(PREFERRED)
Subnet Mask . . . . . : 255.255.255.0
```



Thiết lập địa chỉ IP cho client.py, SERVER\_PORT là cổng kết nối của server, đồng thời đảm bảo rằng CLIENT\_PORT khác với SERVER\_PORT.

```
if __name__ == "__main__":
    # Replace with your server's IP address and port number
    SERVER_HOST = '192.168.0.3'
    #SERVER_HOST = '0.0.0.0'
    SERVER_PORT = 65432
    CLIENT_PORT = 65434
    main(SERVER_HOST, SERVER_PORT,CLIENT_PORT)
```

## 6.2 Thực hiện hệ thống

### 6.2.1 Đổi với quản trị viên

Chạy câu lệnh “py server.py” để khởi động server.

### 6.2.2 Đổi với người dùng khách

Chạy câu lệnh “py client.py” để khởi động chương trình client. Lúc này, người dùng sẽ gửi thông tin đã tham gia vào hệ thống cho phía server.

```
PS F:\HK241\MMT\BTL1> cd client1
PS F:\HK241\MMT\BTL1\client1> py client.py
```

```
PS F:\HK241\MMT\BTL1> cd server
PS F:\HK241\MMT\BTL1\server> py server.py
Server command: 2024-10-31 00:17:16,254 - INFO - Server started and is listening for connections.
2024-10-31 00:17:38,118 - INFO - Active connections: 2
2024-10-31 00:17:38,118 - INFO - Connection established with VioletEvergarden/192.168.0.3:65433)
2024-10-31 00:17:51,486 - INFO - Updating client info in database for hostname: VioletEvergarden/192.168.0.3:65433
2024-10-31 00:17:51,503 - INFO - Database update complete for hostname: VioletEvergarden/192.168.0.3:65433
```

Tiếp theo, người dùng sẽ chọn các lệnh như “publish”, “fetch” và “exit”. Giả sử chọn lệnh “fetch”, thực hiện theo đúng cú pháp hiển thị, hệ thống sẽ đưa ra các piece có thể tải về



```
Enter command (publish file_name/ fetch file_name/ exit): fetch data.pdf
Hosts with the file data.pdf:
Number: 1 VioletEvergarden/192.168.0.3:65433 piece_hash: b'z\xcd\x02\xe4?\x9e\xeeN\xbc~g\x9f\x88\xfe\xc3\x06N\xd5\x0b\xa6' file_size: 6293030 piece_size: 524288 num_order_in_file: 1
Number: 2 VioletEvergarden/192.168.0.3:65433 piece_hash: b'\x86\x8b_\xec\xd3\x86<\xd4\xcb\xbe\xf4\xa0\xdc\xfd\xedv=\xdf\xf5\xad' file_size: 6293030 piece_size: 524288 num_order_in_file: 2
Number: 3 VioletEvergarden/192.168.0.3:65433 piece_hash: b'\x97\x85tB~\x9b\xe1\xfem.\]\xb5:NT\xf03?1\x01' file_size: 6293030 piece_size: 524288 num_order_in_file: 3
Number: 4 VioletEvergarden/192.168.0.3:65433 piece_hash: b'q\x01\xc9\x84\xb0\xd0\xf1\xfe\x8e\xc2\xe5\xeb\xf0jm\xa1\xa7d\xc5' file_size: 6293030 piece_size: 524288 num_order_in_file: 4
Number: 5 VioletEvergarden/192.168.0.3:65433 piece_hash: b'\x87\x9c\xr\xb3\xcb\xc5\xcb\xaa;\xel7j\xfb\xd1NZ5' file_size: 6293030 piece_size: 524288 num_order_in_file: 5
Number: 6 VioletEvergarden/192.168.0.3:65433 piece_hash: b'\xe4?\xb0\xc7\x10s\xb1\x40A\x97\xad\x92\x912\xaf\\x92\xf9' file_size: 6293030 piece_size: 524288 num_order_in_file: 6
Number: 7 VioletEvergarden/192.168.0.3:65435 piece_hash: b'b\xa2sG,\xc3\r\x17]Y\xf7\xe9\x86\xd5\x1ar\xb1)X' file_size: 6293030 piece_size: 524288 num_order_in_file: 7
Number: 8 VioletEvergarden/192.168.0.3:65435 piece_hash: b'\xdb\xe0"\xca\t\xc8\xea\x00\xb2c\xd9\x9c\xb1#\xf4\xf6*\xa7' file_size: 6293030 piece_size: 524288 num_order_in_file: 8
Number: 9 VioletEvergarden/192.168.0.3:65435 piece_hash: b'\xe4R\xde+\r\xd1\x12\x88\xde\xb4\xcd\x9b:\x81\xb1\xc5\x13;' file_size: 6293030 piece_size: 524288 num_order_in_file: 9
Number: 10 VioletEvergarden/192.168.0.3:65435 piece_hash: b'\xc1\x8e\xfe\xb0\x15\xf0\x9e\xf3\x19FD\xab\x8\xb9\xf9\x8\xf0\xec' file_size: 6293030 piece_size: 524288 num_order_in_file: 10
Number: 11 VioletEvergarden/192.168.0.3:65435 piece_hash: b'a\xea#\x08'M\xcb\xb8-F\xdf1\xdeT\x8c3q\xaf' file_size: 6293030 piece_size: 524288 num_order_in_file: 11
Number: 12 VioletEvergarden/192.168.0.3:65435 piece_hash: b'1\xdfHz\xff&\nx7f\xae\x1f\x92hR\xeed2K\xe5Ay' file_size: 6293030 piece_size: 524288 num_order_in_file: 12
Number: 13 VioletEvergarden/192.168.0.3:65435 piece_hash: b'\x8d\x1e~8\x88&\x0b"\xcb*\n\x8bJ\xc9\xc9o$' file_size: 6293030 piece_size: 524288 num_order_in_file: 13
```

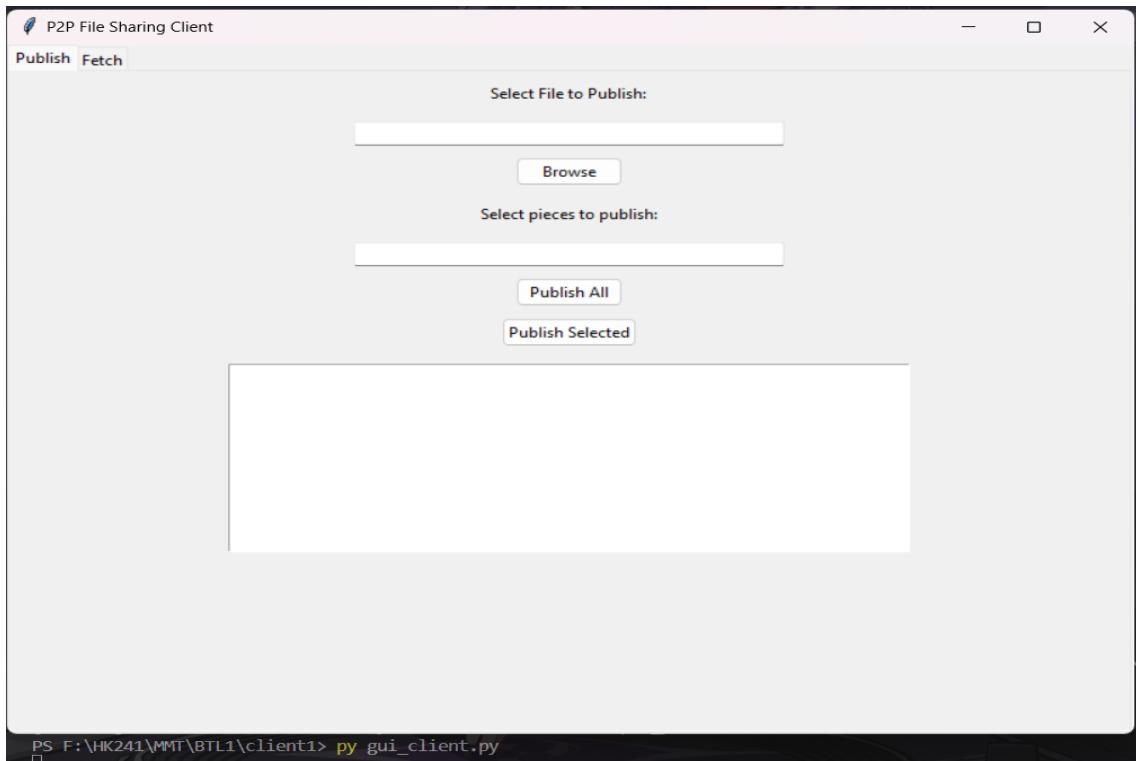
Người dùng nhập số thứ tự các piece để chọn hoặc ghi “all” để tải tất cả

```
Enter the number of the host to download from (or 'all' to download all pieces)all
Piece of file: data.pdf_piece1 has been fetched from peer.
Piece of file: data.pdf_piece2 has been fetched from peer.
Piece of file: data.pdf_piece3 has been fetched from peer.
Piece of file: data.pdf_piece4 has been fetched from peer.
Piece of file: data.pdf_piece5 has been fetched from peer.
Piece of file: data.pdf_piece6 has been fetched from peer.
Piece of file: data.pdf_piece7 has been fetched from peer.
Piece of file: data.pdf_piece8 has been fetched from peer.
Piece of file: data.pdf_piece9 has been fetched from peer.
Piece of file: data.pdf_piece10 has been fetched from peer.
Piece of file: data.pdf_piece11 has been fetched from peer.
Piece of file: data.pdf_piece12 has been fetched from peer.
Piece of file: data.pdf_piece13 has been fetched from peer.
Got all the parts and created the file data.pdf:
Enter command (publish file name/ fetch file name/ exit):
```

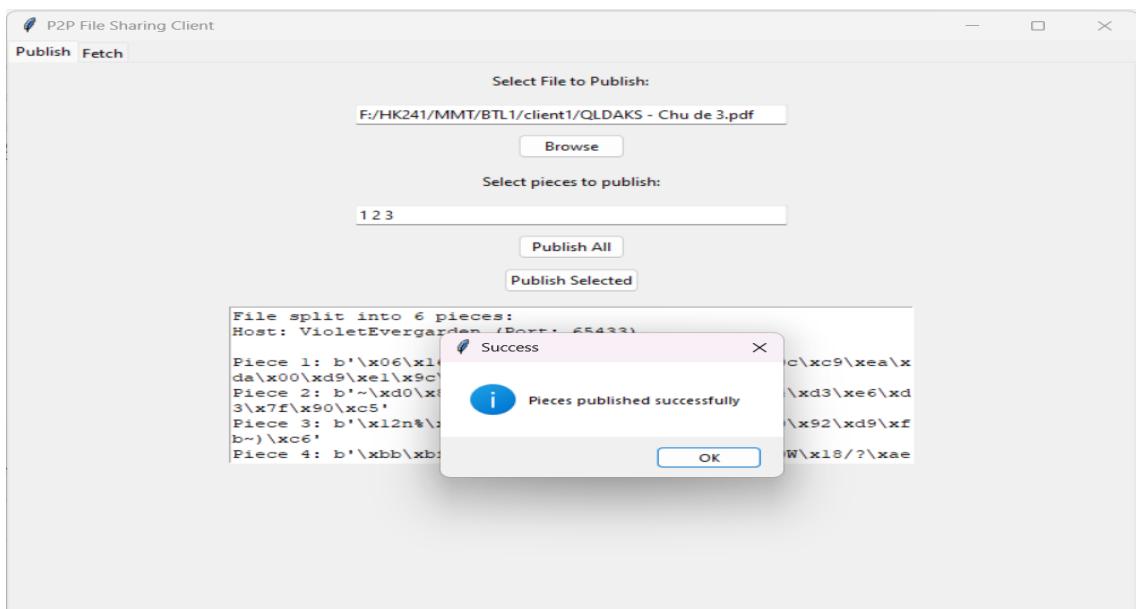
Sau đó, người dùng có thể dùng lệnh “exit” để ngắt kết nối với server.

Tuy nhiên, người dùng cũng có thể sử dụng GUI để thực hiện các thao tác publish hay fetch file. Dưới đây là ví dụ về thao tác publish bằng GUI:

Vào View, chọn Command Palette, tại đây nhập >Python: Create Environment... để tạo môi trường ảo. Tạo Terminal mới, chạy câu lệnh "py gui\_client.py" để khởi động GUI. Một cửa sổ sẽ xuất hiện. Ở lại trang "Publish" để tiếp tục thao tác (hoặc chuyển sang Fetch để thực hiện fetch file):



Nhấn "Browse" để chọn file cần chia sẻ. Ở ô trống phía dưới sẽ xuất hiện danh sách các piece chúng ta có thể publish. Bây giờ có hai ô lựa chọn là "Publish All" và "Publish Selected". Nhấn Publish All để publish tất cả các piece hoặc nhập chuỗi các piece tùy ý (ví dụ: 1 2 3) và nhấn vào lựa chọn thứ hai. Sau khi thành công, sẽ có thông báo thao tác thành công.



## 7 Tài liệu tham khảo

- W3Schools. "HTML, CSS, JavaScript and Web Development Resources." Available at: <https://www.w3schools.com>.
- Python Documentation. "Python Official Documentation." Available at: <https://docs.python.org/3/>.



- Python Package Index (PyPI). Available at: <https://pypi.org/>.
- Real Python. "Python Articles and Tutorials." Available at: <https://realpython.com/> .