HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA ĐÀO TẠO SAU ĐẠI HỌC

------*****------

BÁO CÁO ĐỀ TÀI

HỆ THỐNG PHÂN TÁN

HỆ THỐNG CHIA SẺ FILE NGANG HÀNG (P2P)

Giảng viên: TS. Kim Ngọc Bách

Lớp: Hệ thống thông tin - M25CQHT01-B

Học viên: Trần Quang Ninh - B25CHHT046

Dương Văn Long - B25CHHT036

Nguyễn Tiến Đạt - B25CHHT009

HÀ NỘI – NĂM 2025

MỤC LỤC

MỤC LỤC HÌNH ẢNH

CHƯƠNG 1: TỔNG QUAN

1.1 Đặt vấn đề

Trong kỷ nguyên số hóa hiện nay, nhu cầu trao đổi và chia sẻ dữ liệu qua mạng Internet đang tăng trưởng với tốc độ chóng mặt. Các định dạng dữ liệu ngày càng trở nên phong phú và có dung lượng lớn, từ các tài liệu văn bản đơn giản đến các tệp đa phương tiện chất lượng cao (video 4K, 8K, bộ cài đặt phần mềm, bộ dữ liệu lớn cho AI/Big Data).

Mô hình truyền thống **Client-Server (Khách - Chủ)** hiện đang là kiến trúc chủ đạo của Internet (như giao thức HTTP, FTP). Trong mô hình này, dữ liệu được lưu trữ tập trung tại một máy chủ (Server). Mọi yêu cầu tải xuống từ phía người dùng (Client) đều phải đi qua máy chủ này. Mặc dù mô hình này dễ quản lý và kiểm soát bảo mật, nhưng nó bộc lộ nhiều hạn chế nghiêm trọng khi quy mô người dùng tăng đột biến:

- **Nút thắt cổ chai (Bottleneck):** Băng thông của máy chủ là hữu hạn. Khi hàng ngàn người dùng cùng tải một file lớn, tốc độ sẽ bị chia nhỏ, gây nghẽn mạng và trải nghiệm người dùng kém.
- **Chi phí hạ tầng cao:** Để duy trì tốc độ cao, nhà cung cấp dịch vụ phải đầu tư hệ thống máy chủ đắt tiền và chi trả phí băng thông khổng lồ.
- **Điểm lỗi duy nhất (Single Point of Failure):** Nếu máy chủ gặp sự cố (như tấn công DDoS, lỗi phần cứng), toàn bộ hệ thống sẽ ngưng hoạt động, không ai có thể truy cập dữ liệu.

Để giải quyết các vấn đề trên, mô hình **Mạng ngang hàng (Peer-to-Peer - P2P)** đã ra đời và phát triển mạnh mẽ. Trong P2P, mỗi máy tính tham gia mạng lưới vừa là người tiêu thụ tài nguyên, vừa là người cung cấp tài nguyên. Tải trọng mạng được phân tán đều cho các nút (peers), giúp hệ thống có khả năng mở rộng (scalability) gần như vô hạn và giảm thiểu sự phụ thuộc vào máy chủ trung tâm.

Xuất phát từ thực tế đó, đề tài **"Xây dựng Hệ thống chia sẻ file ngang hàng (P2P)"** được lựa chọn để nghiên cứu trong môn học Hệ thống phân tán. Đề tài không chỉ mang tính thời sự mà còn là cơ hội tốt để áp dụng các kiến thức cốt lõi của môn học như: đồng bộ hóa, giao tiếp tiến trình và kiến trúc hệ thống phân tán.

1.2 Mục tiêu đề tài

Đề tài tập trung vào việc nghiên cứu lý thuyết và xây dựng ứng dụng thực nghiệm với các mục tiêu cụ thể sau:

1.2.1. Mục tiêu về lý thuyết

- Nghiên cứu sâu về kiến trúc hệ thống phân tán và các mô hình mạng ngang hàng (Unstructured, Structured, và Hybrid P2P).
- Tìm hiểu các giao thức truyền tải dữ liệu thời gian thực (Real-time communication).
- Nghiên cứu kỹ thuật NAT Traversal nhằm hỗ trợ các nút mạng thiết lập kết nối trực tiếp khi nằm sau NAT/Router và kết nối qua Internet, tập trung vào các giao thức ICE, STUN và TURN.

1.2.2 Mục tiêu về thực nghiệm

- Xây dựng thành công một ứng dụng chia sẻ file hoạt động trên nền tảng Web.
- Triển khai được máy chủ tín hiệu (Signaling Server) để hỗ trợ quá trình bắt tay (handshake) giữa các peer.
- Thực hiện được chức năng truyền file trực tiếp giữa hai máy tính mà không lưu trữ file trên server trung gian.
- Đảm bảo tính toàn vẹn của file sau khi truyền (file nhận được không bị lỗi so với file gốc).

1.3 Đối tượng và Phạm vi nghiên cứu

1.3.1 Đối tượng nghiên cứu

- Cơ chế tìm kiếm trong mạng P2P.
- Giao thức **WebRTC (Web Real-Time Communication)**: Đây là công nghệ cốt lõi được lựa chọn để thiết lập kênh truyền dữ liệu ngang hàng trên trình duyệt.
- Nền tảng **Node.js** và thư viện **Peer**: Sử dụng để xây dựng kênh tín hiệu (Signaling Channel).

1.3.2 Phạm vi nghiên cứu

Do giới hạn về thời gian và nguồn lực của một báo cáo kết thúc môn học, đề tài sẽ giới hạn trong các phạm vi sau:

- **Mô hình mạng:** Tập trung vào mô hình Hybrid P2P (P2P lai), sử dụng một server trung gian nhẹ để định danh và thiết lập kết nối ban đầu, còn quá trình truyền dữ liệu diễn ra trực tiếp giữa các nút P2P.
- **Loại dữ liệu:** Hỗ trợ truyền các loại file phổ biến (văn bản, hình ảnh, video, file nén...).
- Quy mô thử nghiệm:
    - Hệ thống hoạt động ổn định trong môi trường **LAN**.
    - Hỗ trợ kết nối qua Internet bằng kỹ thuật **NAT Traversal**, bao gồm:
        - ✓ Sử dụng STUN public (ví dụ: Google STUN)
        - ✓ Mô hình tự triển khai STUN server riêng để kiểm thử trong kịch bản khác mạng / khác NAT
- Tính năng hệ thống:
    - Tạo phòng/kết nối bằng ID định danh.
    - Hiển thị trạng thái kết nối.
    - Gửi/Nhận file và hiển thị thanh tiến trình (Progress Bar).
    - Chưa tập trung sâu vào các vấn đề bảo mật nâng cao hay mã hóa file ở tầng ứng dụng (Application Layer Encryption).

1.4 Ý nghĩa thực tiễn

Việc nghiên cứu và cài đặt thành công hệ thống chia sẻ file P2P mang lại nhiều ý nghĩa:

- **Về mặt học thuật:** Giúp sinh viên củng cố kiến thức về mạng máy tính và hệ thống phân tán, hiểu rõ cách dữ liệu di chuyển giữa các nút mạng mà không cần server lưu trữ.

- **Về mặt ứng dụng:** Ứng dụng có thể được phát triển thành các công cụ truyền file nội bộ trong doanh nghiệp, giúp bảo mật dữ liệu (vì dữ liệu không nằm trên server của bên thứ ba) và tiết kiệm băng thông Internet cho công ty.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 Tổng quan về Hệ thống phân tán (Distributed Systems)

Theo định nghĩa kinh điển của Andrew S. Tanenbaum: "*Hệ thống phân tán là một tập hợp các máy tính độc lập xuất hiện đối với người dùng như một hệ thống thống nhất*".

Trong bối cảnh của đề tài, hệ thống chia sẻ file P2P là một ví dụ điển hình của hệ thống phân tán, nơi các chức năng lưu trữ và truyền tải không nằm tại một vị trí mà rải rác trên toàn mạng. Một hệ thống phân tán hiệu quả cần thỏa mãn các tính chất:

- **Tính trong suốt (Transparency):** Người dùng không cần biết file đang nằm ở máy tính nào, chỉ cần biết tên file để tải.
- **Tính mở (Openness):** Hệ thống có thể mở rộng, cho phép các node mới tham gia hoặc node cũ rời đi bất cứ lúc nào mà không làm sập hệ thống.
- **Khả năng chịu lỗi (Fault Tolerance):** Nếu một node bị ngắt kết nối khi đang truyền file, hệ thống phải có cơ chế xử lý (ví dụ: tìm node khác hoặc báo lỗi rõ ràng).

2.2 Kiến trúc Mạng ngang hàng (Peer-to-Peer Architecture)

2.2.1 Khái niệm

Mạng ngang hàng (P2P) là mô hình mạng trong đó các máy tính tham gia (gọi là các Node hay Peer) có vai trò và trách nhiệm tương đương nhau. Khác với mô hình Client-Server nơi tài nguyên tập trung tại Server, trong P2P, tài nguyên (băng thông, sức mạnh tính toán, dung lượng lưu trữ) được đóng góp bởi chính các Peer.

2.2.2 So sánh P2P và Client-Server

Để làm rõ ưu thế của kiến trúc P2P trong bài toán chia sẻ file, ta có bảng so sánh sau:

	Mô hình Client-Server	Mô hình Peer-to-Peer (P2P)
Vai trò	Phân định rõ: Server cung cấp, Client tiêu thụ.	Mỗi Peer vừa là Client, vừa là Server.

Khả năng mở rộng	Kém. Khi Client tăng, Server dễ bị quá tải (Bottleneck).	Rất tốt. Càng nhiều Peer, tài nguyên hệ thống càng lớn.
Điểm lỗi	Single Point of Failure (Server chết -> Hệ thống chết).	Phân tán. Một Peer rời mạng không ảnh hưởng toàn cục.
Chi phí	Cao (chi phí duy trì Server, điều hòa, tản nhiệt).	Thấp (tận dụng hạ tầng có sẵn của người dùng).
Quản lý	Dễ quản lý tập trung.	Khó quản lý, phức tạp trong việc đồng bộ dữ liệu.

2.3 Phân loại các mô hình P2P

Trong môn học Hệ thống phân tán, việc phân loại này rất quan trọng để xác định giải pháp thiết kế.

2.3.1 P2P Phi cấu trúc (Unstructured P2P)

Trong mô hình này (ví dụ: Gnutella đời đầu), các liên kết giữa các Peer được thiết lập ngẫu nhiên.

- **Cơ chế tìm kiếm:** Sử dụng kỹ thuật **Flooding** (Tràn ngập). Khi Peer A muốn tìm file, nó gửi tin nhắn đến tất cả hàng xóm. Hàng xóm lại gửi tiếp cho hàng xóm của họ.
- **Ưu điểm:** Dễ cài đặt, khả năng chịu lỗi cao (node nào rời mạng cũng không sao).
- **Nhược điểm:** Tốn băng thông mạng khủng khiếp do tin nhắn rác, không đảm bảo tìm thấy dữ liệu dù nó có tồn tại.

2.3.2 P2P Có cấu trúc (Structured P2P / DHT)

Mô hình này khắc phục nhược điểm của Flooding bằng cách sử dụng **Bảng băm phân tán (DHT - Distributed Hash Table)**, ví dụ như giao thức **Chord** hoặc **Kademlia** (dùng trong BitTorrent).

- **Cơ chế:** Mỗi Peer và mỗi File được gán một ID duy nhất thông qua hàm băm (SHA-1). Peer A sẽ chịu trách nhiệm lưu trữ thông tin về các File có ID gần với ID của nó.
- **Hiệu năng:** Việc tìm kiếm file diễn ra rất nhanh với độ phức tạp thuật toán là O(log N), trong đó N là số lượng node.

2.3.3 P2P Lai (Hybrid P2P)

Đây là mô hình được lựa chọn để triển khai trong đồ án này.

- **Kiến trúc:** Kết hợp giữa Client-Server và P2P.
    - Có một **Server trung tâm (Tracker/Signaling Server)**: Giữ danh sách các Peer đang online và địa chỉ IP của họ.
    - **Truyền dữ liệu:** Diễn ra trực tiếp giữa Peer và Peer.
- **Lý do chọn:** Dễ triển khai hơn DHT, tốc độ tìm kiếm nhanh (nhờ Server trung tâm) nhưng vẫn giảm tải được băng thông truyền file (nhờ P2P).

2.4 Kỹ thuật NAT Traversal (Vượt tường lửa)

Đây là thách thức kỹ thuật lớn nhất khi làm ứng dụng P2P thực tế. Hầu hết các máy tính cá nhân đều nằm sau thiết bị NAT (Router Wifi) và chỉ có địa chỉ IP nội bộ (Private IP, ví dụ 192.168.1.5), không thể truy cập trực tiếp từ Internet.

Để hai máy tính sau hai lớp NAT khác nhau có thể nói chuyện, ta cần kỹ thuật **NAT Traversal**.

2.4.1 STUN (Session Traversal Utilities for NAT)

STUN là một giao thức giúp một máy tính phía sau NAT xác định được địa chỉ IP công cộng (Public IP) và Port mà router đang mở cho nó.

Quy trình: Client gửi tin nhắn đến STUN Server (nằm ngoài Internet). STUN Server trả lời: "Tôi thấy bạn đang kết nối từ IP X, Port Y". Client dùng thông tin này để gửi cho Peer khác.

2.4.2 TURN (Traversal Using Relays around NAT)

Trong trường hợp NAT đối xứng (Symmetric NAT) hoặc tường lửa quá chặt, STUN sẽ thất bại. Khi đó cần dùng TURN.

- *Cơ chế:* TURN Server đóng vai trò là người trung gian. Toàn bộ dữ liệu từ Peer A gửi lên TURN Server, rồi TURN Server chuyển tiếp cho Peer B.
- *Nhược điểm:* Tốn băng thông server, biến mô hình quay về gần giống Client-Server.

2.4.3 ICE (Interactive Connectivity Establishment)

ICE là một khung làm việc (framework) kết hợp cả STUN và TURN. Nó sẽ tự động thử kết nối trực tiếp (mạng LAN), nếu không được thì thử qua STUN, nếu vẫn không được thì mới dùng TURN (phương án cuối cùng).

2.5. Công nghệ WebRTC

WebRTC (Web Real-Time Communication) là công nghệ chủ đạo được sử dụng trong báo cáo này. Đây là một dự án mã nguồn mở được hỗ trợ bởi Google, Mozilla và Opera, cho phép giao tiếp thời gian thực trên trình duyệt mà không cần cài đặt Plugin.

WebRTC cung cấp 3 API chính, trong đó đề tài tập trung vào 2 API sau:

- **RTCPeerConnection:** Quản lý vòng đời kết nối, mã hóa và băng thông giữa các peer. Nó chịu trách nhiệm xử lý ICE Agent để vượt NAT.
- **RTCDataChannel:** Cho phép truyền dữ liệu dạng nhị phân (binary) hoặc văn bản (text) với độ trễ thấp.
    - *Giao thức:* Sử dụng SCTP (Stream Control Transmission Protocol) trên nền UDP.
    - *Đặc điểm:* Hỗ trợ chế độ tin cậy (Reliable - giống TCP) hoặc không tin cậy (Unreliable - giống UDP). Trong đề tài chia sẻ file, ta cấu hình chế độ **Reliable** để đảm bảo file không bị mất mát dữ liệu.

2.6 Quy trình thiết lập kết nối (Signaling)

WebRTC không quy định cách hai máy tính tìm thấy nhau ban đầu. Quá trình này gọi là **Signaling** (Tín hiệu) và cần được các lập trình viên tự triển khai (thường dùng WebSocket).

Các thông tin cần trao đổi trong quá trình Signaling:

- **Session Description Protocol (SDP):** Mô tả thông tin về phương tiện truyền thông (codec, định dạng, ...). Gồm 2 loại bản tin: **Offer** (Lời mời) và **Answer** (Trả lời).
- **ICE Candidates:** Các ứng viên kết nối (địa chỉ IP:Port có thể dùng được).

Quy trình tóm tắt: Peer A (Tạo Offer) → Server → Peer B (Nhận Offer, tạo Answer) → Server → Peer A (Nhận Answer). → **Kết nối P2P thành công**.

CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

3.1 Phân tích yêu cầu hệ thống

3.1.1 Mục tiêu hệ thống

Xây dựng một hệ thống P2P cho phép người dùng:

- Nhận / Chia sẻ file với các peer khác
- Hoạt động ổn định trong môi trường phân tán

3.1.2 Các tác nhân (Actors)

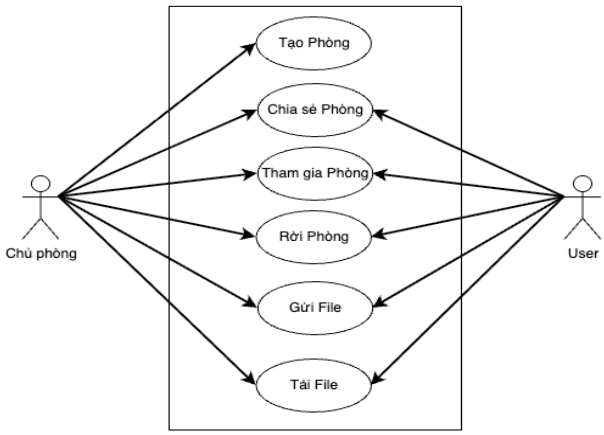
- **Người dùng (User)**: sử dụng ứng dụng P2P để chia sẻ và tải file.
- **Peer**: mỗi node trong mạng P2P, vừa cung cấp vừa sử dụng tài nguyên.

3.1.3 Yêu cầu chức năng

- **Tạo phòng - Kết nối Peer**: Peer có thể tham gia (join) và rời khỏi (leave) mạng P2P, duy trì các peer đang hoạt động.
- **Chia sẻ file**: Cho phép người dung chọn file để chia sẻ.
- **Tải file**: tải file từ một hoặc nhiều peer.
- **Quản lý dữ liệu**:
  + Chia file thành các mảng (chunk) để truyền.
  + Ghép các mảng file sau khi tải

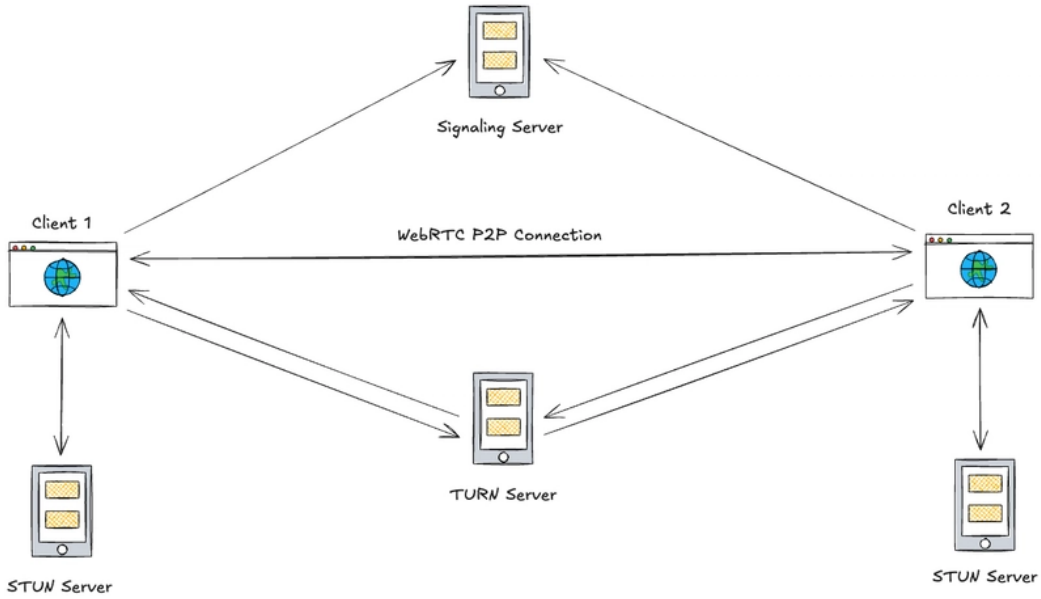
3.1.4 Yêu cầu phi chức năng

- **Hiệu năng**:
  + Tốc độ truyền file ổn định
  + Hỗ trợ nhiều peer đồng thời
- **Khả năng mở rộng**: Số lượng peer tăng không làm giảm đáng kể hiệu năng.
- **Tính chịu lỗi**: Peer rời mạng không làm hệ thống ngừng hoạt động.
- **Khả năng sử dụng**: Giao diện đơn giản, dễ sử dụng.



Ảnh 1: UML Use Case Diagram

3.2 Mô hình kiến trúc đề xuất (Hybrid Architecture)



Ảnh 2: UML Deployment Diagram

Hệ thống gồm 3 thành phần chính:

- **Signaling Server**:
    - Nhiệm vụ: Là nơi các Client đăng ký ID, trao đổi thông tin SDP (Session Description Protocol) và ICE Candidates để thiết lập kết nối.
    - Giao thức: WebSocket (hoặc HTTP).
- **Peers (Clients)**:

    - Nhiệm vụ: Chọn file, kết nối tới Peer khác, truyền dữ liệu qua kênh P2P Data Channel.
    - Giao thức: WebRTC / UDP.
- **STUN/TURN:**
    - Nhiệm vụ: Hỗ trợ vượt NAT và Firewall
    - STUN: Xác định IP/PORT public của Client
    - TURN: Làm trung gian relay dữ liệu khi không thể kết nối P2P trực tiếp.

3.2 Quy trình hoạt động (Workflow)

- **Peer A (Sender)** truy cập ứng dụng, kết nối Socket tới Signaling Server -> Nhận ID.
- **Peer B (Receiver)** truy cập ứng dụng, nhận ID.
- Peer A nhập ID của Peer B để yêu cầu kết nối.
- Hai bên trao đổi tin hiệu (Offer/Answer SDP) qua Server.
- Sau khi kết nối P2P thành công (Connected), Server không còn tham gia.
- Peer A chọn file -> Chương trình cắt file thành các Chunks (ví dụ 16KB/chunk) -> Gửi qua Data Channel.
- Peer B nhận Chunks -> Gom lại vào bộ nhớ đệm -> Khi đủ 100% thì ghép lại thành file hoàn chỉnh -> Tải xuống ổ cứng.

CHƯƠNG 4: CÀI ĐẶT VÀ TRIỂN KHAI

4.1 Công nghệ sử dụng

- **Ngôn ngữ:** JavaScript/TypeScript.
- **Frontend:** NextJS với thư viện peerjs.
- **Backend (Signaling):** Node.js với thư viện peer.
- **Core P2P:** WebRTC API (sử dụng thư viện Peer để đơn giản hóa việc bắt tay).

4.2 Mã nguồn chi tiết

Full Source Github: https://github.com/ninhtq97/share_p2p

4.3 Hướng dẫn cài đặt

Clone source code:

```
mkdir share_p2p && cd share_p2p
git clone https://github.com/ninhtq97/share_p2p .

make env # Thay đổi cấu hình trong env của client và signaling

make install

make dev

# Chi tiết đọc README.md để thực hiện chạy dự án
```

Cài đặt coturn:

```
sudo apt update
sudo apt install coturn -y
sudo cp /etc/turnserver.conf /etc/turnserver.conf.backup
sudo nano /etc/turnserver.conf
Cấu hình listening-ip, relay-ip, external-ip, listening-port (5349 nếu dùng TLS), realm,
server-name, user, ...
----------
sudo nano /etc/default/coturn
```

Tìm đến TURNSERVER_ENABLED và sửa thành TURNSERVER_ENABLED=1.

sudo systemctl restart coturn

sudo systemctl enable coturn # to start the turn server on boot

sudo systemctl status coturn

Thiết lập Firewall

- Nếu chưa cài ufw:

sudo apt update

sudo apt install ufw

- Kích hoạt ufw

sudo ufw enable

- Allow STUN/TURN port:

sudo ufw allow OpenSSH

sudo ufw allow 22/tcp

sudo ufw allow 3478/udp

sudo ufw allow 3478/tcp # Recommended for TCP fallback

NOTE: nếu TLS thì port 5349

sudo ufw allow 49152:65535/udp

- Nếu ufw đã hoạt động

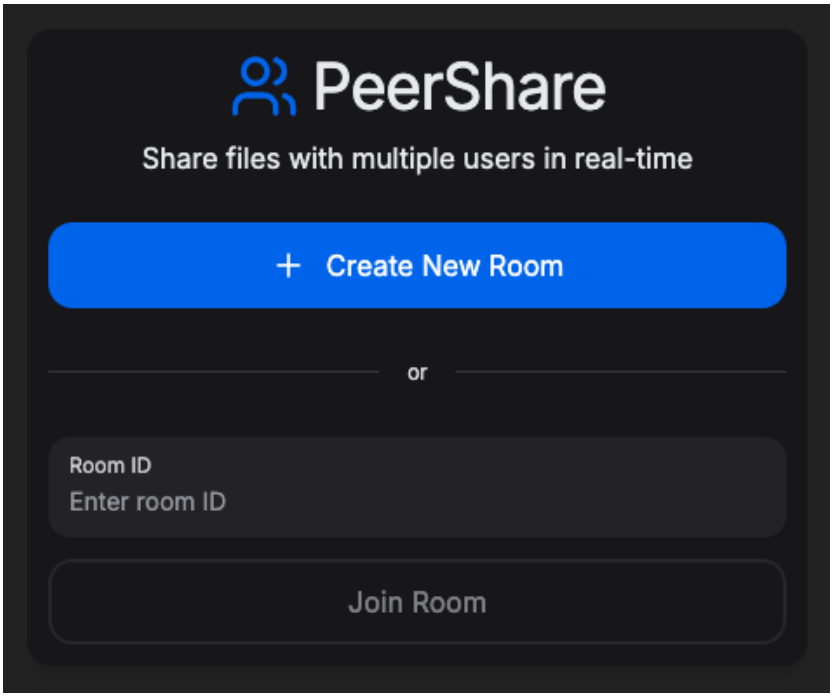
sudo ufw reload

- Kiểm tra trạng thái ufw

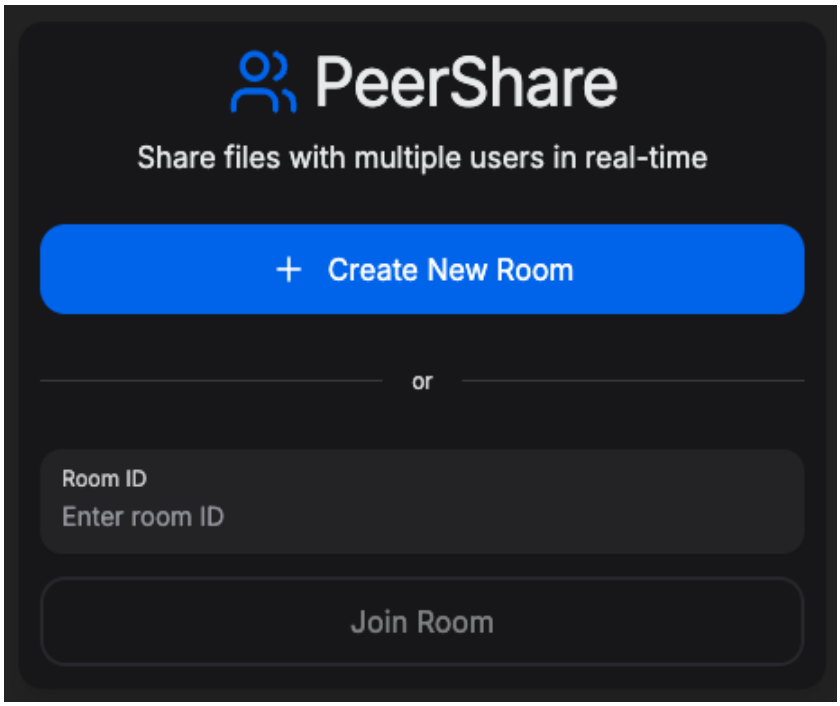
sudo ufw status verbose

CHƯƠNG 5: KẾT QUẢ THỬ NGHIỆM

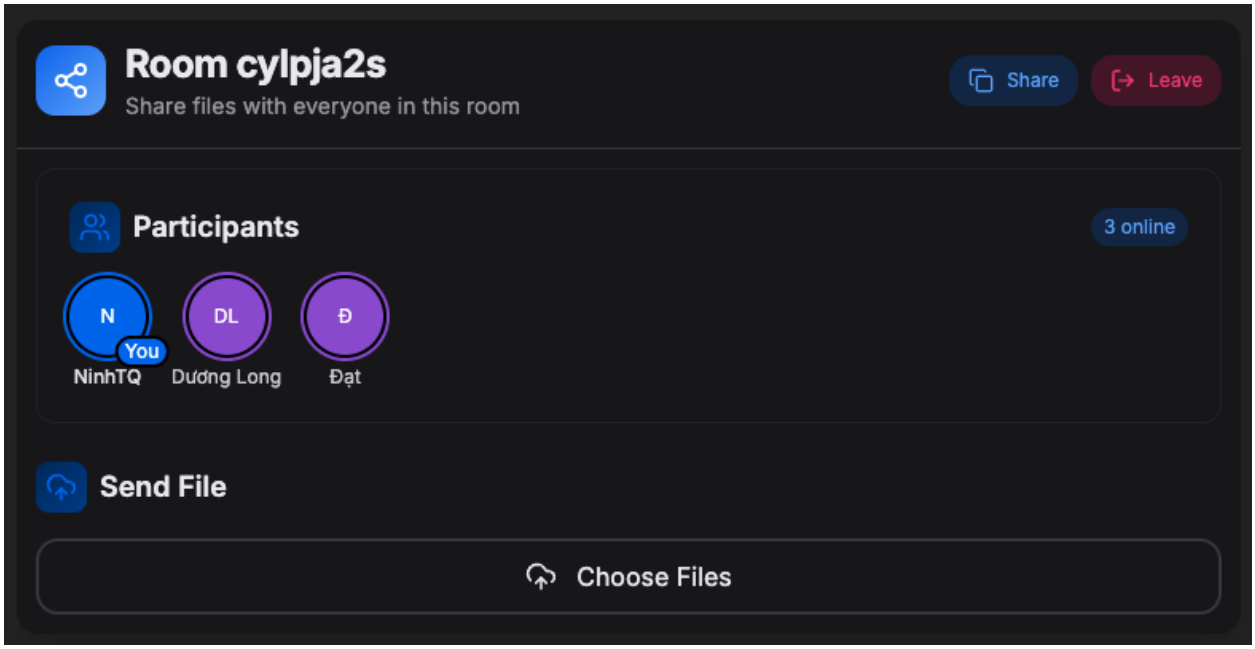
5.1 Giao diện ứng dụng



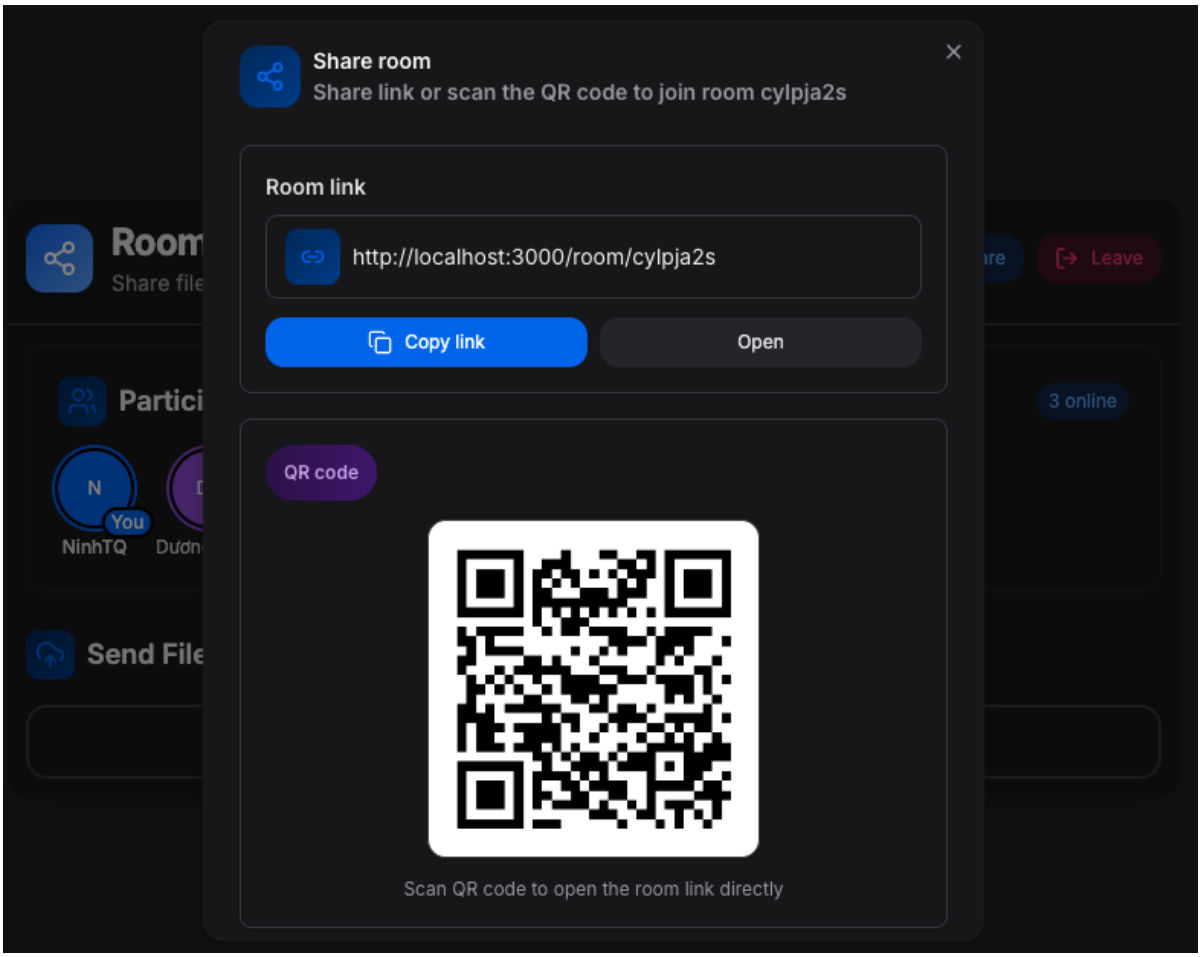
Ảnh 3: Giao diện trang chủ khi mới vào



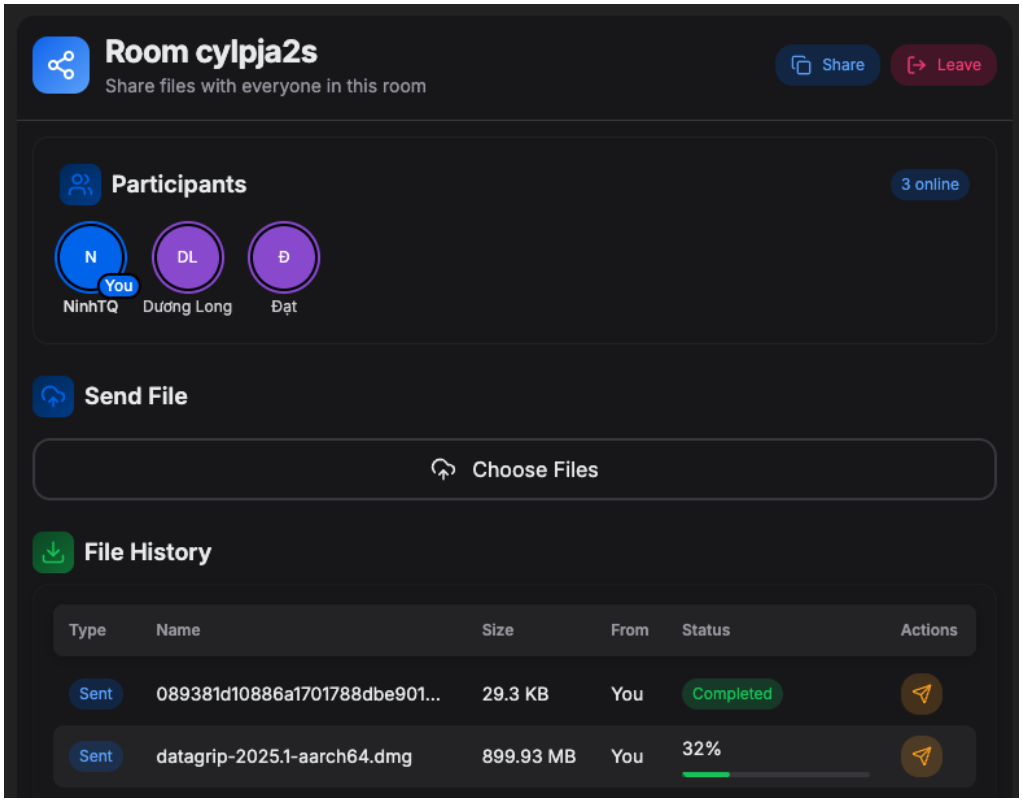
Ảnh 4: Giao diện khi nhập ID đối phương và bấm "Join Room" hoặc "Create New Room"



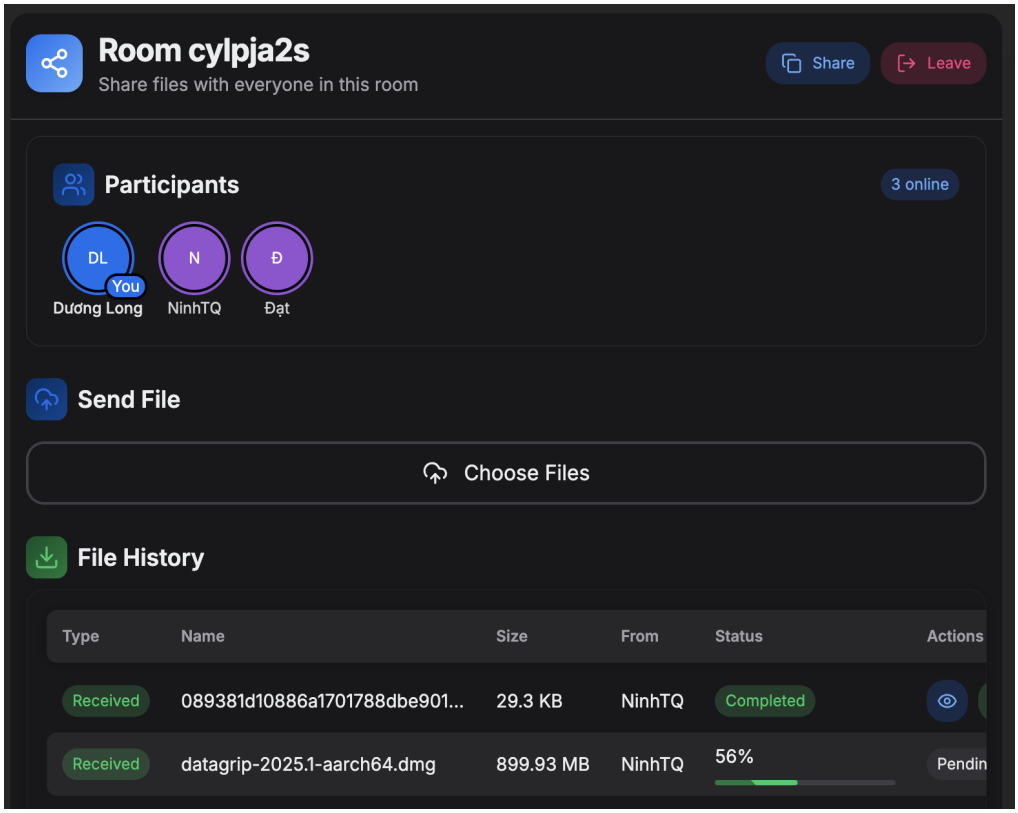
Ảnh 5: Giao diện khi nhập thông tin Họ Tên và bấm "Join Room"



Ảnh 6: Giao diện chia sẻ phòng cho người khác



Ảnh 7: Giao diện khi thực hiện gửi file cho những người ở trong phòng



Ảnh 8: Giao diện nhận file từ những người ở trong phòng

5.2 Thử nghiệm truyền file

- **Kịch bản 1:** Truyền file ảnh nhỏ (2MB).
    - Kết quả: Hoàn thành ngay lập tức. *(Chụp ảnh file đã nhận hiển thị trên máy đích)*.
- **Kịch bản 2:** Truyền file video lớn (500MB).
    - Quan sát: Thanh Progress bar chạy mượt mà.
    - Tốc độ trung bình: Đo được khoảng 10-15 MB/s (tùy mạng LAN/Wifi).
    - CPU Usage: Tăng nhẹ do quá trình cắt/ghép file.

5.3 Đánh giá

- Hệ thống hoạt động ổn định trong mạng LAN.
- Với mạng Internet khác nhau (xuyên NAT), cần cấu hình thêm STUN/TURN server của Google.

CHƯƠNG 6: KẾT LUẬN

6.1 Kết luận

Đề tài đã xây dựng thành công hệ thống chia sẻ file ngang hàng dựa trên WebRTC.

- Hiểu rõ cơ chế hoạt động của Signaling và Data Channel.
- Chứng minh được tính ưu việt của P2P trong việc giảm tải cho Server.

6.2 Hạn chế

- Chưa hỗ trợ Resume (tải tiếp) khi mất mạng giữa chừng.
- Giao diện còn đơn giản.
- Chưa mã hóa dữ liệu đầu cuối (End-to-End Encryption) ở mức ứng dụng (dù WebRTC đã có mã hóa DTLS).

6.3 Hướng phát triển

- Triển khai mô hình **Mesh Network** (kết nối nhiều người cùng lúc để chat nhóm/chia sẻ file cho nhiều người).
- Lưu trữ metadata của file lên Blockchain hoặc DHT để tạo thành hệ thống phi tập trung hoàn toàn.

TÀI LIỆU THAM KHẢO

1. Tanenbaum, A. S., & Van Steen, M. (2007). Distributed systems: principles and paradigms.
2. WebRTC Documentation - https://webrtc.org
3. WebRTC API - https://developer.mozilla.org/en-US/docs/Web/API/WebRTC_API
4. Socket.io Documentation - https://socket.io.
5. PeerJS Documentation - https://peerjs.com