

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**  
**KHOA ĐÀO TẠO SAU ĐẠI HỌC**



**BÁO CÁO**  
**CÁC HỆ THỐNG PHÂN TÁN**

**ĐỀ TÀI**

**ỨNG DỤNG GỬI VÀ NHẬN FILE GIỮA SERVER VÀ CLIENT**

**Giảng viên hướng dẫn:** TS Kim Ngọc Bách

**Lớp:** M24CQHT02-B

**Nhóm:** 4

**Danh sách học viên:** B24CHHT071 - Nguyễn Văn Hiếu  
B24CHHT074 - Phạm Văn Hoàng  
B24CHHT085 - Nguyễn Khắc Minh

*Hà Nội – 2025*

# PHÂN CHIA CÔNG VIỆC

<b>Nguyễn Khắc Minh</b>	Tìm hiểu cấu trúc và cách hoạt động của giao thức TCP/IP. Phân tích và thiết kế hệ thống phía Server
<b>Phạm Văn Hoàng</b>	Tìm hiểu về đặc điểm và tính chất của hệ thống phân tán. Phân tích nghiệp vụ giữa Client và Server. Kiểm thử và đánh giá kết quả.
<b>Nguyễn Văn Hiếu</b>	Tìm hiểu tính bảo mật và xác thực trong hệ thống phân tán. Phân tích và thiết kế hệ thống phía Client.

# MỤC LỤC

<b>1. Mở đầu:</b>	4
1.1 Đặt vấn đề:	4
1.2 Lý do chọn đề tài:	4
<b>2. Cơ sở lý thuyết:</b>	5
2.1. Giới thiệu về Socket Programming trong Java:	5
2.2. Giao thức TCP/IP:	5
2.2.1 Cấu trúc và hoạt động của TCP/IP:	5
2.2.2 Ứng dụng:	5
2.2.3 Ưu điểm của TCP/IP:	5
2.2.4 Ví dụ ứng dụng của TCP/IP:	6
2.3. Tổng quan hệ thống phân tán	6
2.4 Bảo mật và xác thực trong hệ phân tán:	7
2.4.1 Bảo mật trong hệ thống phân tán:	7
2.4.2 Xác thực trong hệ thống phân tán:	7
2.5 Một số kiến trúc phổ biến(Client-server, P2P):	8
2.5.1 Mô hình Client Server:	8
2.5.2 Mô hình Peer to peer (P2P):	9
<b>3. Phân tích và thiết kế hệ thống:</b>	11
3.1 Mục tiêu:	11
3.2. Mô tả nghiệp vụ:	11
3.2.1 Mô tả người dùng:	11
3.2.2 Các chức năng chính:	12
3.2.3 Yêu cầu phi chức năng:	12
3.3 Phân tích hệ thống:	13
3.3.1 Biểu đồ use case:	13
3.3.2 Mô tả các use case:	13
3.4. Thiết kế hệ thống:	15
3.4.1 Kiến trúc hệ thống:	15
3.4.2 Đặc điểm phân tán:	15
3.4.3 Khái quát tổng quan hệ thống:	16

3.4.4 Mô tả các luồng xử lý giữa Client và Server: .....	16
3.5 Công nghệ sử dụng: .....	20
3.6 Giao diện người dùng: .....	20
<b>4. Kết luận:</b> .....	23
4.1 Đáp ứng mô hình hệ thống phân tán: .....	23
4.2 Giao tiếp từ xa (Remote Communication):.....	24
<b>5. Danh mục tài liệu tham khảo:</b> .....	26

## 1. Mở đầu:

### 1.1 Đặt vấn đề:

Trong bối cảnh công nghệ ngày nay, việc chia sẻ và truyền tải dữ liệu giữa các hệ thống ngày càng trở nên quan trọng. Các ứng dụng mạng như gửi và nhận file từ client đến server đóng vai trò chủ chốt trong việc giúp các tổ chức và cá nhân truyền tải thông tin một cách nhanh chóng và an toàn. Các giao thức và công nghệ mạng luôn được phát triển để đáp ứng nhu cầu này, và Java Swing là một công cụ mạnh mẽ để xây dựng các ứng dụng giao diện người dùng (GUI) trong môi trường Java.

Tuy nhiên, việc xây dựng một ứng dụng có khả năng gửi và nhận file hiệu quả, dễ sử dụng và bảo mật cao luôn là một thách thức đối với các lập trình viên. Đặc biệt là khi cần kết hợp với Java Swing để xây dựng giao diện, việc quản lý việc truyền tải file giữa client và server đòi hỏi người phát triển phải hiểu rõ các nguyên lý lập trình mạng và các thuật toán nén, mã hóa dữ liệu nếu cần thiết.

Vì vậy, việc nghiên cứu và xây dựng một ứng dụng gửi và nhận file giữa client và server trong Java Swing là một đề tài có tính ứng dụng cao, không chỉ giúp người dùng dễ dàng chia sẻ dữ liệu mà còn góp phần nâng cao kiến thức về lập trình mạng và giao diện người dùng.

### 1.2 Lý do chọn đề tài:

- **Nhu cầu thực tế:** Trong thời đại số hóa ngày nay, việc truyền tải file giữa các hệ thống, đặc biệt là qua mạng internet, là nhu cầu thiết yếu của nhiều tổ chức, doanh nghiệp cũng như cá nhân. Việc có một ứng dụng có thể gửi và nhận file dễ dàng sẽ tiết kiệm thời gian, công sức và chi phí cho người sử dụng.
- **Ứng dụng Java Swing:** Java Swing là một công nghệ phổ biến để xây dựng giao diện người dùng trong môi trường Java. Việc sử dụng Swing giúp người phát triển dễ dàng tạo ra các giao diện đẹp mắt và dễ sử dụng. Việc tích hợp công nghệ này với việc gửi và nhận file tạo ra một ứng dụng có tính tiện dụng cao.
- **Khả năng mở rộng và bảo mật:** Khi xây dựng ứng dụng gửi và nhận file, có thể tích hợp thêm các tính năng như mã hóa dữ liệu, nén file, hoặc kiểm tra tính toàn vẹn của dữ liệu. Đây là một phần quan trọng trong việc bảo mật thông tin khi truyền tải qua mạng, đồng thời nâng cao kiến thức về lập trình mạng và bảo mật.
- **Kỹ năng lập trình mạng:** Việc xây dựng một hệ thống client-server sử dụng Java (cả Swing và các thư viện mạng của Java như java.net) giúp người học và lập trình viên nâng cao kỹ năng lập trình mạng, làm quen với các giao thức truyền tải dữ

liệu như TCP/IP và UDP, và hiểu rõ hơn về cách thức giao tiếp giữa các ứng dụng mạng.

- **Tính ứng dụng cao:** Đề tài này không chỉ dừng lại ở lý thuyết mà còn có tính ứng dụng thực tế cao. Nó có thể được áp dụng vào nhiều loại ứng dụng khác nhau, từ các phần mềm chia sẻ tài liệu trong doanh nghiệp, đến các hệ thống gửi nhận file trong các dịch vụ web.

## 2. Cơ sở lý thuyết:

### 2.1. Giới thiệu về Socket Programming trong Java:

- Lập trình socket trong Java cho phép các chương trình khác nhau giao tiếp với nhau qua mạng, dù chúng đang chạy trên cùng một máy hay trên các máy khác nhau. Bài viết này mô tả một thiết lập Máy khách và Máy chủ một chiều rất cơ bản, trong đó Máy khách kết nối, gửi tin nhắn đến máy chủ và máy chủ hiển thị chúng bằng kết nối socket.

### 2.2. Giao thức TCP/IP:

- Giao tiếp TCP/IP là tập hợp các giao thức mạng cho phép các thiết bị truyền thông với nhau trên Internet hoặc mạng cục bộ. TCP/IP là viết tắt của Transmission Control Protocol/Internet Protocol, đóng vai trò như một bộ quy tắc chuẩn hóa cách dữ liệu được chia nhỏ, định địa chỉ, truyền đi và nhận lại.

#### 2.2.1 Cấu trúc và hoạt động của TCP/IP:

- **Giao thức TCP:** Đảm bảo việc truyền dữ liệu đáng tin cậy bằng cách chia dữ liệu thành các gói, sắp xếp lại thứ tự, kiểm tra lỗi và truyền lại nếu cần.

- **Giao thức IP:** Chịu trách nhiệm định địa chỉ và định tuyến các gói dữ liệu, đảm bảo chúng đến đúng đích.

- **Mô hình TCP/IP:** Thường được mô tả như một mô hình bốn lớp, bao gồm lớp ứng dụng, lớp truyền tải, lớp internet và lớp truy cập mạng.

#### 2.2.2 Ứng dụng:

- TCP/IP được sử dụng rộng rãi trong các ứng dụng Internet như duyệt web (HTTP), gửi email (SMTP, POP3, IMAP), truyền file (FTP) và nhiều ứng dụng khác.

#### 2.2.3 Ưu điểm của TCP/IP:

- **Tính phổ biến:** TCP/IP là giao thức mạng được sử dụng rộng rãi nhất, tương thích với nhiều hệ điều hành, phần cứng và thiết bị.

- **Khả năng mở rộng:** TCP/IP được thiết kế để dễ dàng mở rộng khi mạng lưới phát triển.
- **Độ tin cậy:** TCP/IP cung cấp cơ chế kiểm tra lỗi và truyền lại dữ liệu, đảm bảo dữ liệu được truyền đi một cách đáng tin cậy.
- **Tính linh hoạt:** TCP/IP hỗ trợ nhiều loại dịch vụ mạng khác nhau.

#### 2.2.4 Ví dụ ứng dụng của TCP/IP:

- **Truy cập internet:** TCP/IP là nền tảng cho việc truy cập và sử dụng internet, cho phép người dùng duyệt web, gửi email, xem video, v.v.
- **Kết nối mạng nội bộ:** TCP/IP cũng được sử dụng trong các mạng LAN (Local Area Network) để kết nối các thiết bị trong một khu vực hạn chế.
- **Ứng dụng di động:** Các ứng dụng như Zalo, Facebook, và nhiều ứng dụng khác sử dụng TCP/IP để giao tiếp và truyền dữ liệu.

#### 2.3. Tổng quan hệ thống phân tán

- **Định nghĩa:** hệ thống phân tán là hệ thống phần mềm mà các thành phần cấu tạo nên nó nằm ở trên các máy tính khác nhau được kết nối thành mạng lưới. *Nhìn chung, các định nghĩa hệ thống phân tán đều đề cập đến hai đặc điểm quan trọng, đặc điểm thứ nhất đó là tập hợp các phần tử tính toán có thể hoạt động độc lập với nhau và đặc điểm thứ hai là chúng cần phải cộng tác để giải quyết một nhiệm vụ chung.*

- Đặc điểm chính của hệ thống phân tán:

- + **Tính phân tán:** Các thành phần của hệ thống (máy tính, phần mềm, dữ liệu) được phân bố trên nhiều máy tính vật lý khác nhau.
- + **Tính đồng bộ:** Các máy tính trong hệ thống phải phối hợp với nhau để đảm bảo tính nhất quán của dữ liệu và hoạt động của hệ thống.
- + **Tính độc lập:** Mỗi máy tính trong hệ thống có thể hoạt động độc lập, nhưng vẫn có thể giao tiếp và trao đổi thông tin với các máy tính khác.
- + **Tính trong suốt:** Người dùng không cần biết chi tiết về cấu trúc phân tán của hệ thống, họ chỉ cần tương tác với hệ thống như một thực thể duy nhất.
- + **Khả năng mở rộng:** Hệ thống phân tán có thể dễ dàng mở rộng bằng cách thêm các máy tính mới vào hệ thống.
- + **Độ tin cậy cao:** Nếu một máy tính trong hệ thống bị lỗi, các máy tính khác vẫn có thể tiếp tục hoạt động, đảm bảo tính liên tục của hệ thống.
- + **Hiệu suất cao:** Các tác vụ có thể được phân chia và xử lý song song trên nhiều máy tính, giúp tăng tốc độ xử lý.

+ **Tiết kiệm chi phí:** Sử dụng các máy tính rẻ hơn thay vì một máy tính mạnh mẽ duy nhất có thể tiết kiệm chi phí.

## 2.4 Bảo mật và xác thực trong hệ phân tán:

- Bảo mật và xác thực trong hệ thống phân tán là hai khía cạnh quan trọng để đảm bảo tính toàn vẹn, bảo mật và tin cậy của hệ thống. Hệ thống phân tán, với đặc thù dữ liệu và xử lý được phân chia trên nhiều máy tính, đối mặt với nhiều thách thức về bảo mật hơn so với hệ thống tập trung.

### 2.4.1 Bảo mật trong hệ thống phân tán:

- **Mã hóa dữ liệu:** Sử dụng các giao thức mã hóa mạnh mẽ như AES-256 để bảo vệ dữ liệu khi lưu trữ và truyền tải giữa các nút.

- **Kiểm soát truy cập:** Hạn chế quyền truy cập vào các tài nguyên hệ thống, chỉ cho phép người dùng và dịch vụ được ủy quyền mới có thể truy cập và thao tác.

- **Phòng chống tấn công:** Triển khai các biện pháp phòng chống tấn công như tấn công từ chối dịch vụ (DDoS), tấn công SQL injection, tấn công phát lại.

- **Phát hiện và phản ứng:** Thiết lập cơ chế phát hiện các hoạt động bất thường và có khả năng phản ứng nhanh chóng khi có sự cố xảy ra.

- **Bảo mật mạng:** Sử dụng các giao thức bảo mật như HTTPS để bảo vệ dữ liệu truyền tải trên mạng, và các kỹ thuật tường lửa để ngăn chặn truy cập trái phép.

### 2.4.2 Xác thực trong hệ thống phân tán:

#### - Xác thực người dùng:

Đảm bảo rằng người dùng đăng nhập vào hệ thống là đúng người và có quyền truy cập, bằng cách sử dụng các phương pháp xác thực như tên đăng nhập/mật khẩu, xác thực đa yếu tố (MFA).

#### - Xác thực dịch vụ:

Đảm bảo rằng các dịch vụ giao tiếp với nhau là hợp lệ, bằng cách sử dụng các giao thức xác thực như OAuth, JWT.

#### - Xác thực giữa các nút:

Đảm bảo rằng các nút trong hệ thống phân tán giao tiếp với nhau một cách an toàn, bằng cách sử dụng các giao thức như TLS/SSL.

#### - Giao thức xác thực:

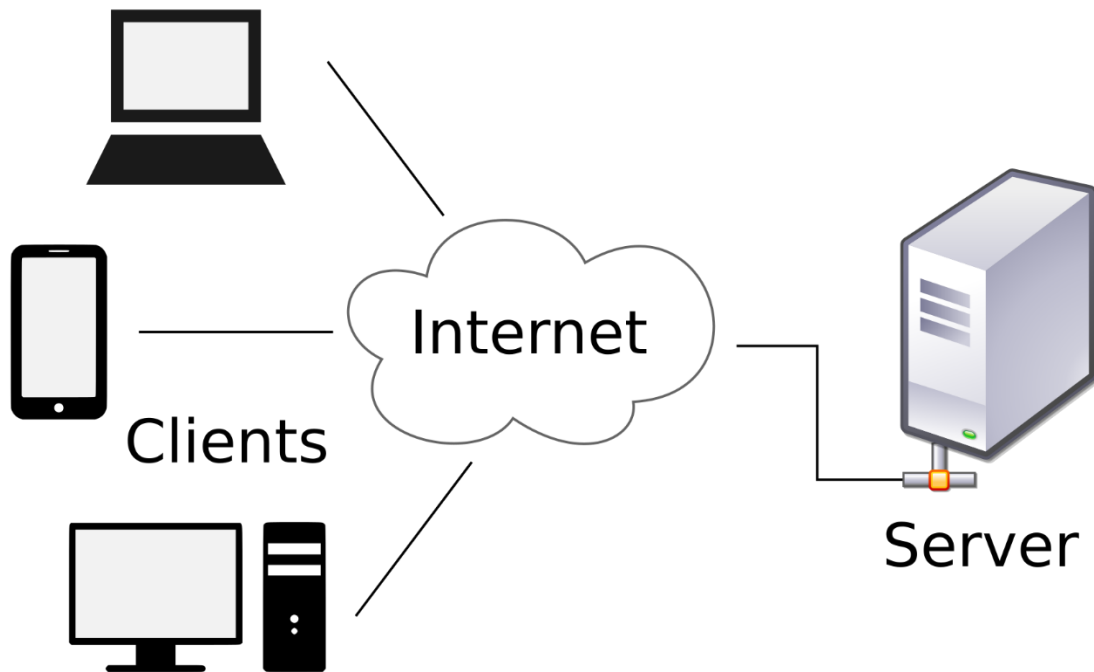
Sử dụng các giao thức như PAP và CHAP để đảm bảo việc xác thực diễn ra an toàn và tin cậy.



## 2.5 Một số kiến trúc phổ biến(Client-server, P2P):

### 2.5.1 Mô hình Client Server:

- Là một mô hình mạng máy tính gồm hai thành phần chính là client và server. Client sẽ là bên yêu cầu dịch vụ cài đặt cũng như lưu trữ tài nguyên từ server. Khi client gửi yêu cầu dữ liệu đến server qua Internet, server sẽ xử lý yêu cầu và gửi các gói dữ liệu cho client.



*Mô hình Client-server(Wikipedia)*

- Ưu điểm:

Tất cả dữ liệu, tài nguyên được lưu trữ và quản lý tập trung trên Server. Điều này giúp doanh nghiệp dễ dàng tổ chức, kiểm soát hệ thống.

- Nhược điểm:

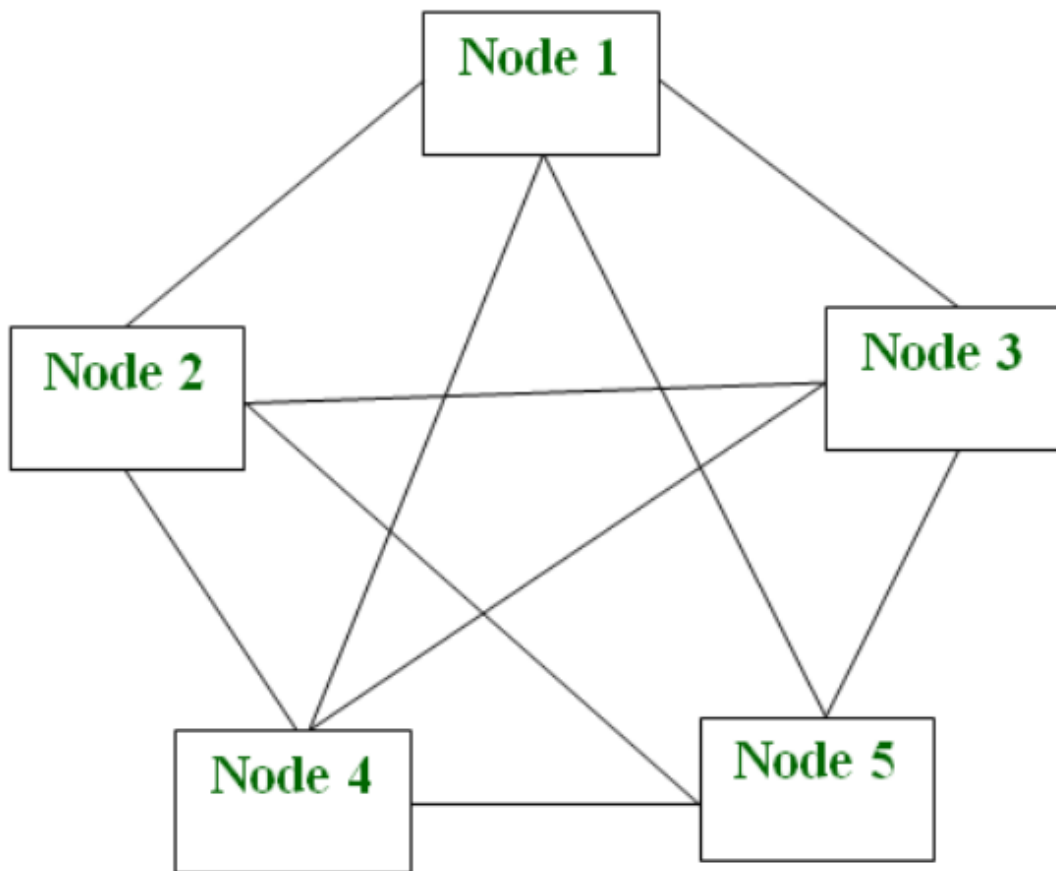
- **Phụ thuộc vào Server:** Nếu Server gặp sự cố, toàn bộ hệ thống có thể bị gián đoạn. Khi Server bị lỗi phần cứng hoặc bị tấn công, Client không thể truy cập dữ liệu hoặc tài nguyên. Do đó, nên triển khai biện pháp sao lưu và hệ thống dự phòng (backup và failover).
- **Chi phí đầu tư ban đầu cao:** Doanh nghiệp cần đầu tư vào phần cứng mạnh mẽ và phần mềm chuyên dụng cho Server. Chi phí mua máy chủ, thiết lập hệ thống mạng, phần mềm quản lý Server có thể cao hơn so với mô hình mạng

ngang hàng (Peer-to-Peer). Sử dụng dịch vụ đám mây để giảm bớt chi phí đầu tư ban đầu.

- **Yêu cầu kỹ thuật cao:** Để vận hành hệ thống hiệu quả, doanh nghiệp cần đội ngũ IT có kinh nghiệm để quản lý, bảo trì Server. Quản trị viên mạng phải đảm bảo Server luôn hoạt động ổn định, được cập nhật kịp thời.

#### 2.5.2 Mô hình Peer to peer (P2P):

- Là một kiến trúc ứng dụng phân tán nhằm phân vùng nhiệm vụ hoặc khối lượng công việc giữa các peer. Các peer là những thiết bị tham gia trong ứng dụng có đặc quyền như nhau. Chúng tạo thành một mạng lưới các node ngang hàng. Các peer tạo ra một phần tài nguyên của chúng, chẳng hạn như processing power, lưu trữ đĩa hoặc băng thông mạng, có sẵn cho những participant khác mà không cần sự điều phối trung tâm của server hoặc host ổn định. Các peer vừa là nhà cung cấp vừa là người tiêu thụ tài nguyên.



*Mô hình P2P(Geeksforgeek)*

- Ưu điểm:

- Không cần sử dụng tới máy chủ.
- Mỗi một thiết bị máy tính là một người dùng quản lý riêng.
- Không yêu cầu bất kỳ các kiến thức kỹ thuật chuyên ngành phức tạp nào.
- Một mạng P2P thích hợp với môi trường gia đình và doanh nghiệp nhỏ.
- Sử dụng ít lưu lượng truy cập mạng.

- Nhược điểm:

- Các thông tin trên máy không thể thực hiện sao lưu tập trung.

- Việc cho phép truy cập cùng một lúc bởi nhiều thiết bị máy tính làm giảm hiệu suất hoạt động.
- Các tệp không được sắp xếp khoa học mà được lưu trữ trên máy tính cá nhân gây khó khăn trong việc xác định vị trí của chúng.
- Việc đảm bảo an toàn cho hệ thống mạng là việc của tất cả người dùng.
- Chỉ cung cấp một số quyền cơ bản và không có bảo mật nâng cao.

### **3. Phân tích và thiết kế hệ thống:**

#### **3.1 Mục tiêu:**

- Hệ thống được xây dựng nhằm mục đích hỗ trợ việc gửi và nhận file giữa Client và Server thông qua giao thức TCP. Thông qua ứng dụng giao diện đồ họa được phát triển bằng Java Swing:
- Đơn giản hóa quy trình chia sẻ file qua mạng cục bộ hoặc mạng nội bộ giữa các máy tính.
- Cung cấp giao diện người dùng thân thiện, trực quan thông qua Java Swing, giúp người dùng dễ dàng thao tác gửi hoặc nhận file mà không cần thao tác dòng lệnh.
- Xây dựng kiến trúc Client - Server có thể mở rộng, hỗ trợ xử lý các tác vụ truyền/nhận file với kích thước khác nhau một cách ổn định.
- Tạo nền tảng để mở rộng hệ thống trong tương lai, như: thêm tính năng bảo mật, xác thực người dùng, truyền file qua Internet, ...

#### **3.2. Mô tả nghiệp vụ:**

##### **3.2.1 Mô tả người dùng:**

- Người dùng hệ thống là những cá nhân có nhu cầu chia sẻ file giữa các máy tính với nhau trong môi trường mạng nội bộ (LAN). Họ không cần hiểu sâu về kỹ thuật, chỉ cần thao tác qua giao diện đồ họa để gửi hoặc nhận file một cách nhanh chóng.
- Có hai vai trò chính trong hệ thống:
  - Client: Là người dùng sử dụng ứng dụng có giao diện để thực hiện việc gửi và nhận file.
  - Server: Là máy chủ có trách nhiệm tiếp nhận file từ client, lưu trữ và phản hồi yêu cầu gửi file về cho client khi được yêu cầu.

### 3.2.2 Các chức năng chính:

- Hệ thống bao gồm Server và Client, về phía Client có các chức năng sau:

- Dễ dàng chọn và gửi file bất kỳ (văn bản, hình ảnh, nén, v.v.) lên phía Server.
- Xem danh sách các file hiện đang có trên Server.
- Chọn và tải các file từ Server về máy tính cá nhân.

- Phía Server có các chức năng sau:

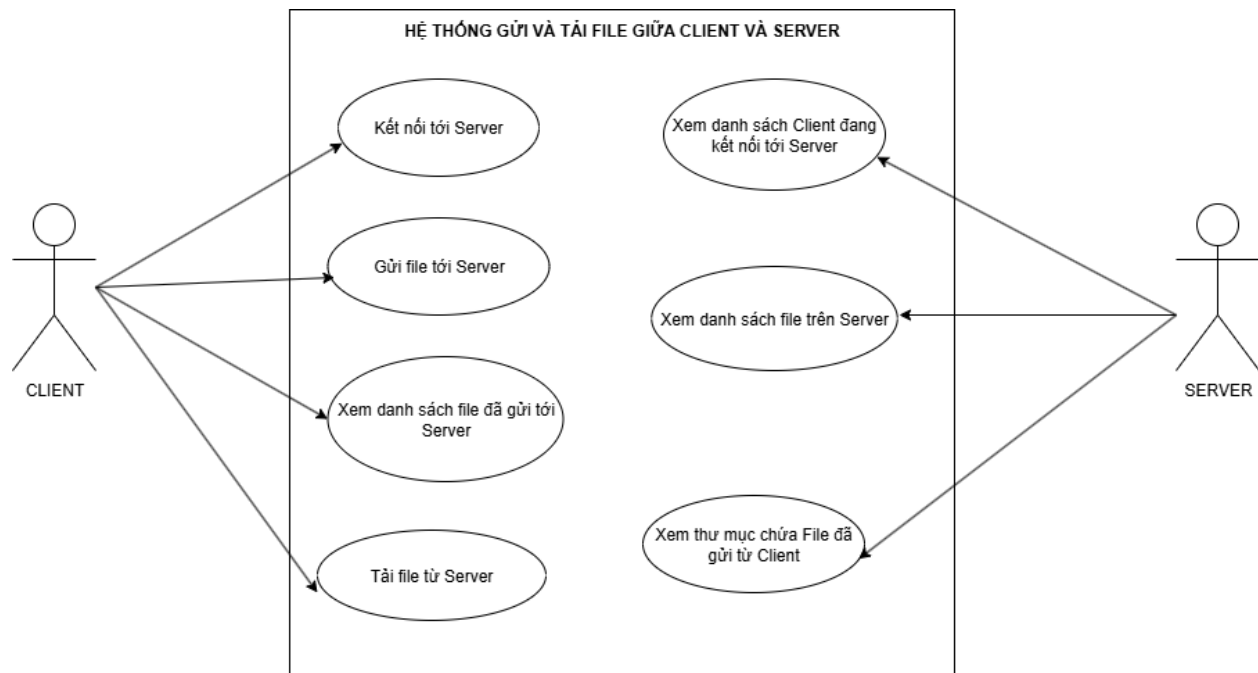
- Xem danh sách Client đang kết nối với Server
- Xem danh sách các file tương ứng với Client đã gửi cho Server
- Truy cập vào thư mục chứa các file mà Client đã gửi cho Server.

### 3.2.3 Yêu cầu phi chức năng:

- Giao diện người dùng thân thiện, dễ sử dụng, phát triển bằng Java Swing.
- Hệ thống truyền file phải ổn định, không làm hỏng dữ liệu, kể cả khi file có kích thước lớn (vài trăm MB đến GB).
- Phản hồi nhanh khi thực hiện thao tác gửi hoặc tải file.
- Hệ thống hoạt động trong mạng LAN mà không cần cấu hình phức tạp.
- Có thể mở rộng để hỗ trợ nhiều client cùng kết nối đến server trong tương lai.

### 3.3 Phân tích hệ thống:

#### 3.3.1 Biểu đồ use case:



#### 3.3.2 Mô tả các use case:

##### - Phía Client:

Use case: Kết nối tới Server	
Mô tả	Client kết nối tới Server
Tác nhân	Client
Điều kiện	Client phải khai báo địa chỉ IP của Server
Luồng sự kiện	1. Client khai báo địa chỉ IP của Server 2. Client nhập tên của mình 3. Client ấn Connect để kết nối tới Server

Use case: Gửi file tới Server	
Mô tả	Client chọn file từ máy tính và gửi file tới Server
Tác nhân	Client
Điều kiện	Client đã kết nối với Server

Luồng sự kiện	<ol style="list-style-type: none"> <li>1. Client chọn nút File để mở thư mục file trong máy tính</li> <li>2. Client chọn 1 file mà ấn Open.</li> <li>3. Màn hình Client sẽ hiển thị trạng thái upload File tới Server.</li> </ol>
---------------	---

Use case: Xem danh sách file đã gửi tới Server	
Mô tả	Client xem danh sách file đã được gửi tới Server
Tác nhân	Client
Điều kiện	Client đã kết nối với Server
Luồng sự kiện	Ở tab File on Server để xem danh sách file đã gửi cho Server

Use case: Tải file từ Server	
Mô tả	Client tải file từ Server
Tác nhân	Client
Điều kiện	Client đã kết nối với Server
Luồng sự kiện	Ở tab File on Server, Client bấm vào icon mũi tên tương ứng với tải file từ Server về

#### - Phía Server:

Use case: Xem danh sách Client đang kết nối tới Server	
Mô tả	Server xem được danh sách những client đang kết nối tới Server
Tác nhân	Server
Điều kiện	Cần có client đã kết nối với Server
Luồng sự kiện	Ở màn hình chính sẽ hiển thị tên Client đang kết nối với Server, cụ thể tab Name là tên Client.

Use case: Xem danh sách file trên Server	
Mô tả	Server xem được danh sách file được gửi từ Client
Tác nhân	Client

Điều kiện	Client phải gửi file cho Server trước
Luồng sự kiện	Ở màn hình chính, tab Status hiển thị danh sách file mà Client gửi cho Server

Use case: Xem thư mục chứa file đã gửi từ Client	
Mô tả	Server truy cập thư mục chứa tất cả các file mà các Client gửi cho Server
Tác nhân	Server
Điều kiện	Client đã gửi file cho Server trước.
Luồng sự kiện	Truy cập thư mục chứa file

### 3.4. Thiết kế hệ thống:

#### 3.4.1 Kiến trúc hệ thống:

- Hệ thống được xây dựng theo mô hình phân tán kiểu Client - Server, trong đó:

- Client là ứng dụng có giao diện người dùng (GUI) được xây dựng bằng Java Swing, chạy trên các máy trạm của người dùng cuối. Client có thể chọn file và gửi đến Server, hoặc yêu cầu tải file từ Server về máy tính của mình.
- Server là một ứng dụng chạy độc lập, có chức năng lắng nghe kết nối từ các Client, tiếp nhận file và lưu trữ. Server có thể được triển khai trên máy chủ cục bộ hoặc máy tính bất kỳ trong mạng LAN.

#### 3.4.2 Đặc điểm phân tán:

- Tính phân tán: Các thành phần Client và Server chạy trên các máy vật lý khác nhau, sử dụng Socket TCP (giao thức kết nối tin cậy, tuần tự) để giao tiếp qua mạng.

- Tính minh bạch trong việc truy cập: Gửi và tải file đều sử dụng một giao diện GUI thống nhất mà không cần quan tâm đến các chi tiết kỹ thuật như vị trí vật lý của Server hay cách dữ liệu được truyền đi trong mạng.

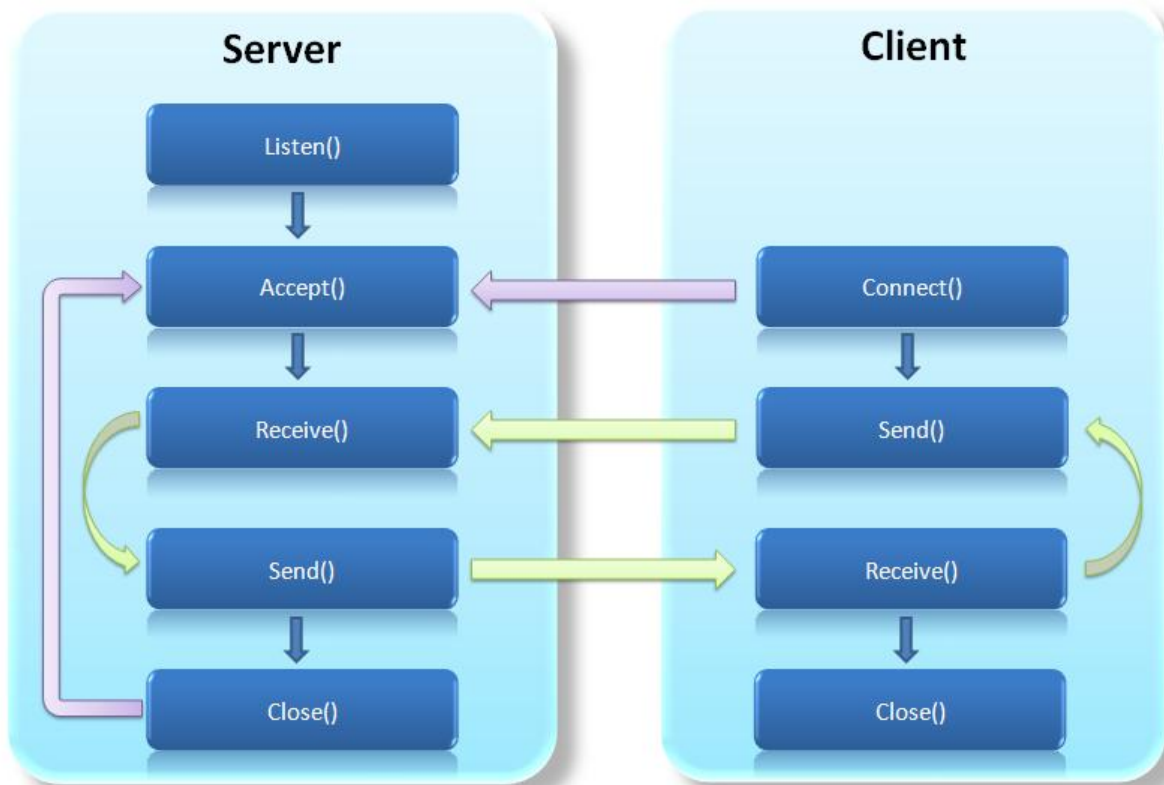
- Tính đồng thời: Nhiều client có thể hoạt động đồng thời.

- Khả năng truyền dữ liệu: File được chia nhỏ thành các byte rồi truyền qua socket; bên Server sẽ nhận và ghép lại.

- Không chia sẻ bộ nhớ: Các thành phần giao tiếp chỉ qua truyền dữ liệu, không chia sẻ tài nguyên bộ nhớ.



### 3.4.3 Khái quát tổng quan hệ thống:



*Mô hình kiến trúc TCP socket*

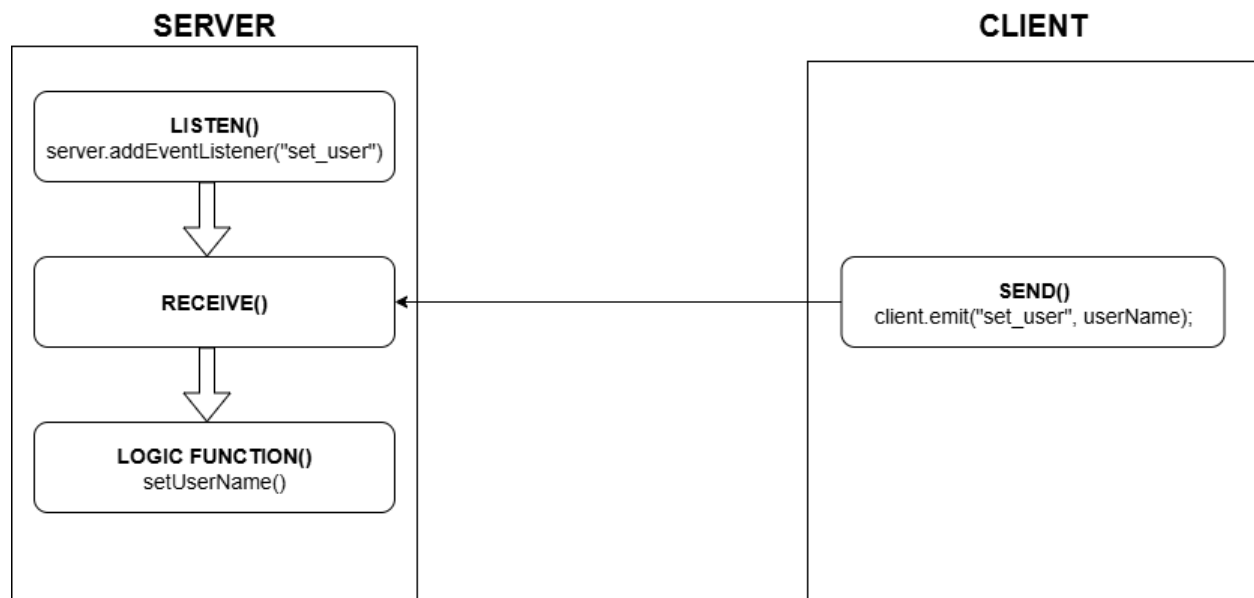
### 3.4.4 Mô tả các luồng xử lý giữa Client và Server:

#### a. Thiết lập kết nối Socket.IO:



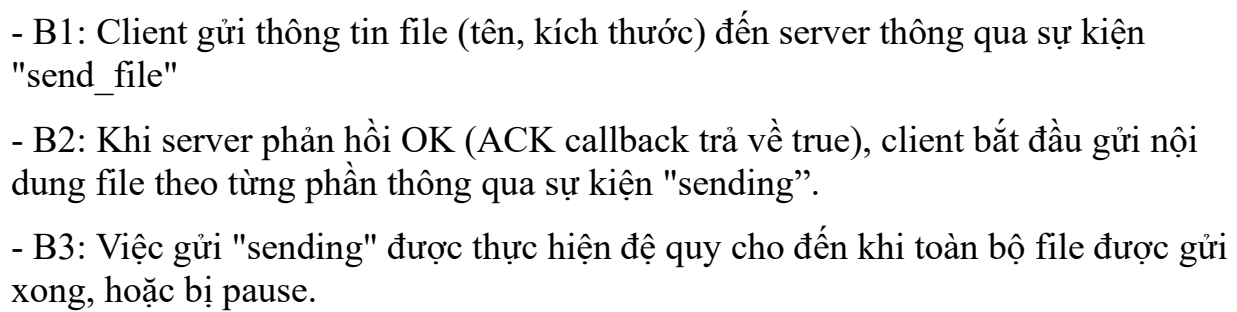
**Client = IO.socket("http://" + IP + ":" + DEFAULT\_PORT);**  
 VD: IP là IP Server: 192.168.100.49  
 DEFAULT\_PORT: 9999

### b. Thiết lập tên Client:



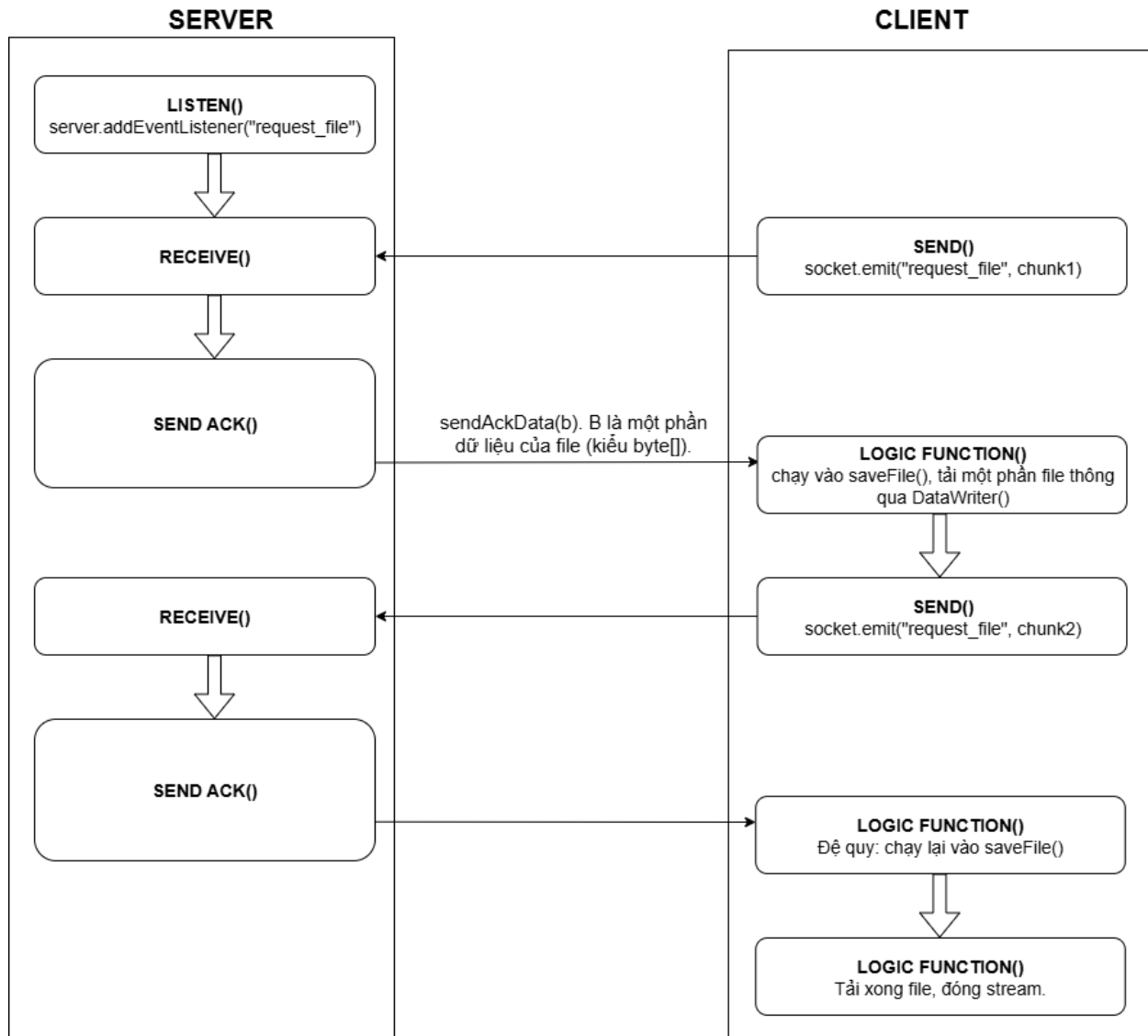
- B1: Phía client gửi sự kiện “set\_user”. Trong đó userName là tên của Client.
- B2: Phía Server đăng ký một sự kiện thông qua server.addEventListener() để Server lắng nghe. Khi server đã nhận được sẽ hiển thị tên client đó trên GUI.

### c. Gửi file tới Server:



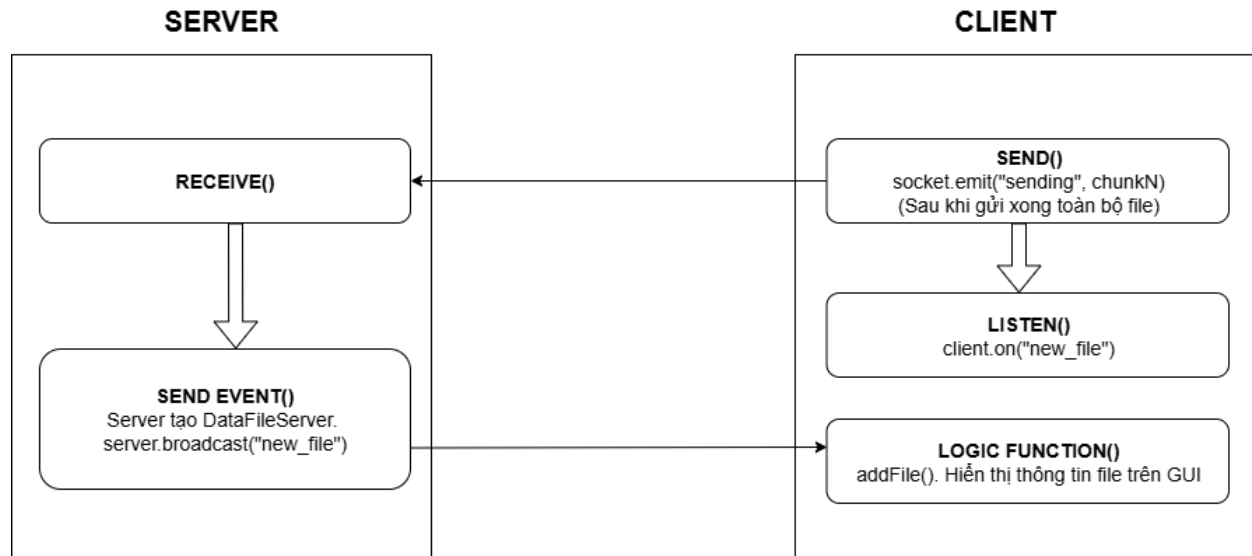
- B4: Nếu thành công và không bị pause → tiếp tục gửi phần kế tiếp (gọi lại `sendingFile`). Nếu `status.isPause()` → ngừng và set biến `pause = true`.

**d. Client tải file từ Server:**



- B1: Gửi yêu cầu đến server qua event `request_file`.
- B2: Server phản hồi lại một phần dữ liệu của file (kiểu `byte[]`).
- B3: Client tải file về local thông qua `DataWriter`.

**e. Hiển thị danh sách file có trên Server.**



- B1: Sau khi nhận được file gửi từ Client thành công, server tạo một object chứa thông tin file mới (DataFileServer data). Sau đó gửi broadcast đến tất cả client đang kết nối rằng: “Đã có file mới vừa được upload thành công!”

(Cụ thể: `server.getBroadcastOperations().sendEvent("new_file", data)`)

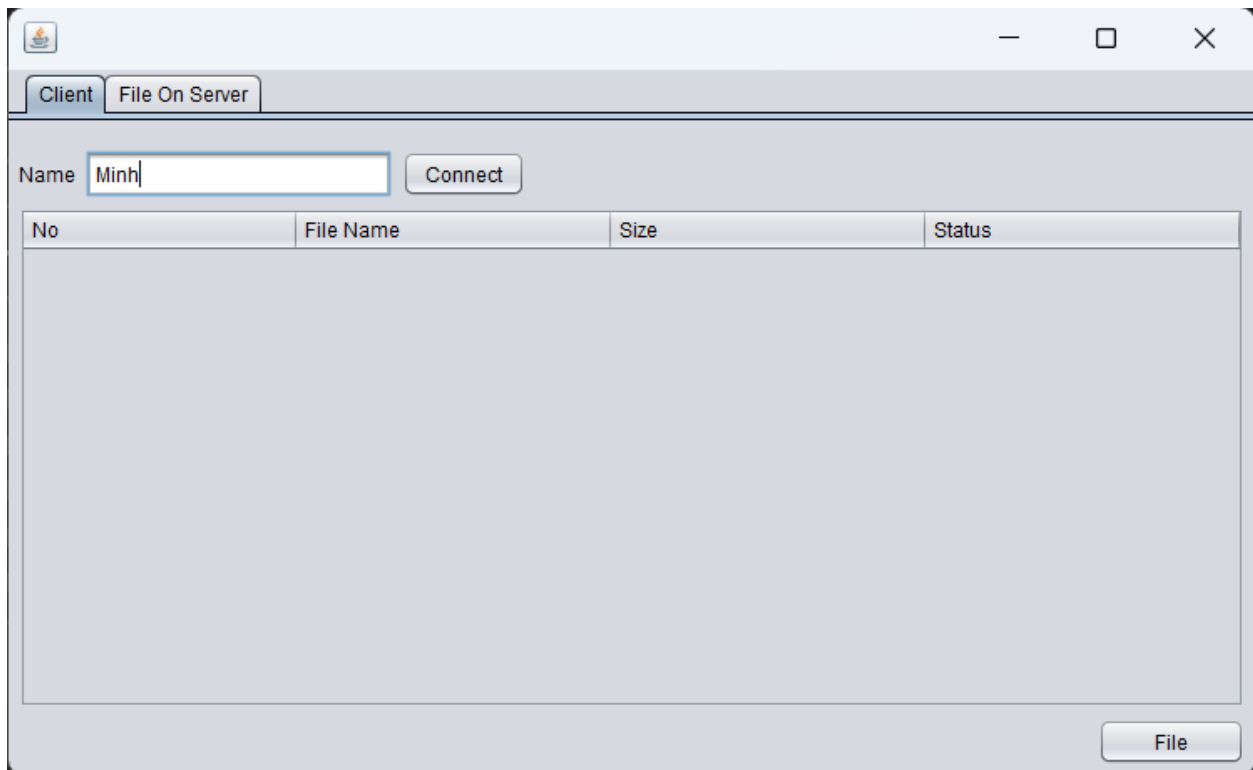
- B2: Phía client nhận được thông báo “new\_file” và tự động cập nhật danh sách file trên giao diện thông qua hàm `addFile()`

### 3.5 Công nghệ sử dụng:

- Ngôn ngữ lập trình: Java
- Giao diện: Java Swing
- Giao thức: TCP Socket
- IDE: IntelliJ

### 3.6 Giao diện người dùng:

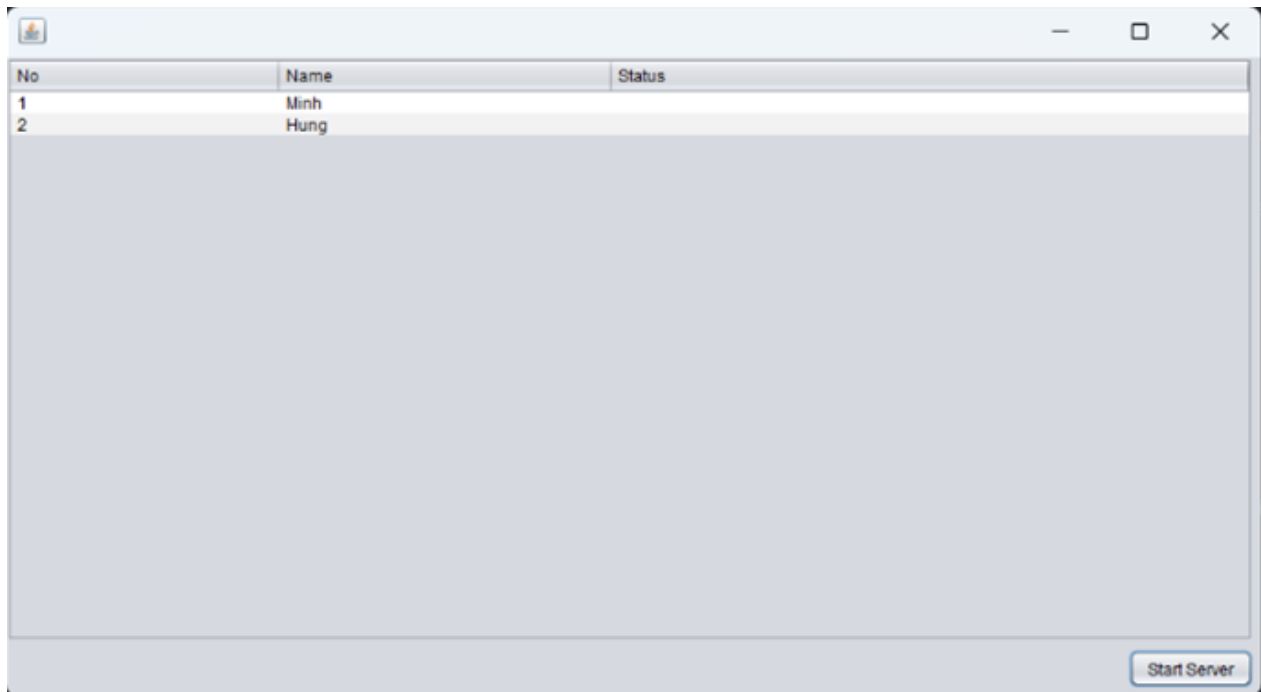
- Phía Client: Điền tên Client và chọn Connect.



The screenshot shows a window titled "Client" with a tab labeled "File On Server". Below the tab, there is a text input field labeled "Name" containing the text "Minh", followed by a "Connect" button. Below this is a table with four columns: "No", "File Name", "Size", and "Status". The table is currently empty. At the bottom right of the window is a "File" button.

No	File Name	Size	Status
----	-----------	------	--------

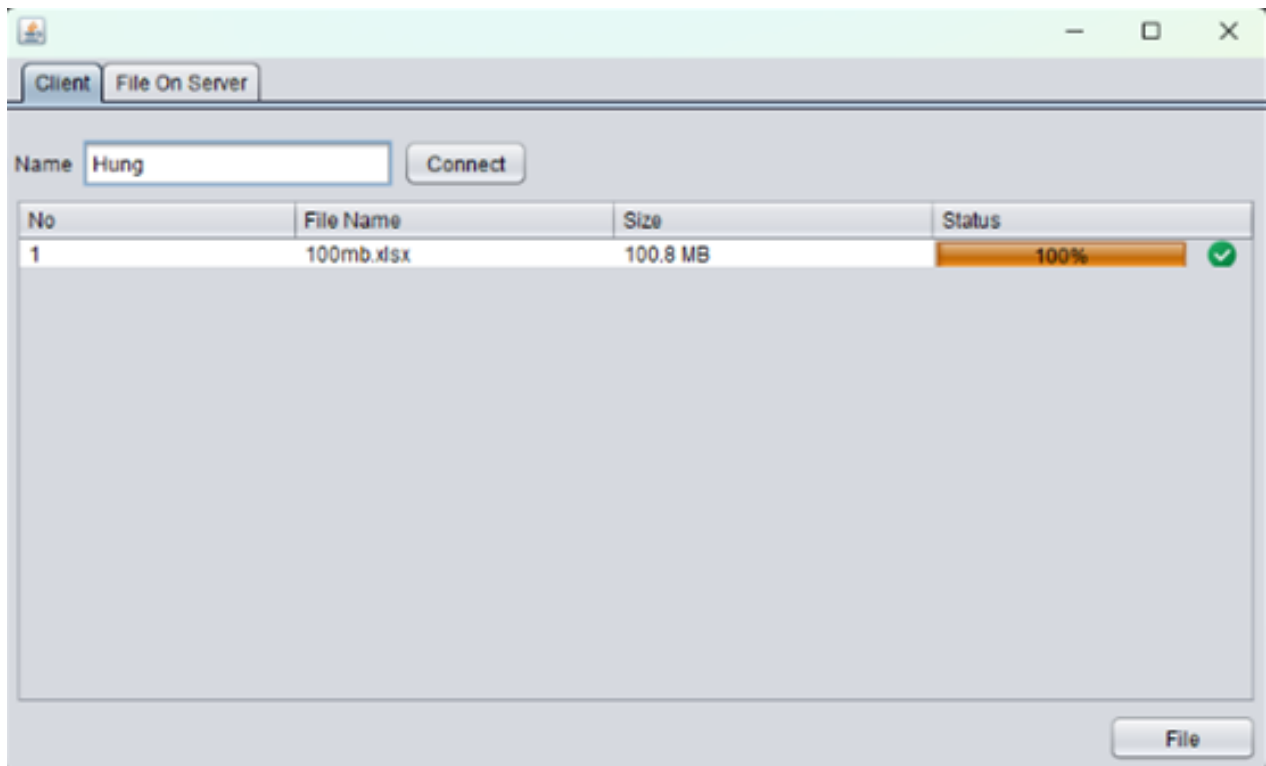
- Phía Server: Hiện thị danh sách các client đang kết nối



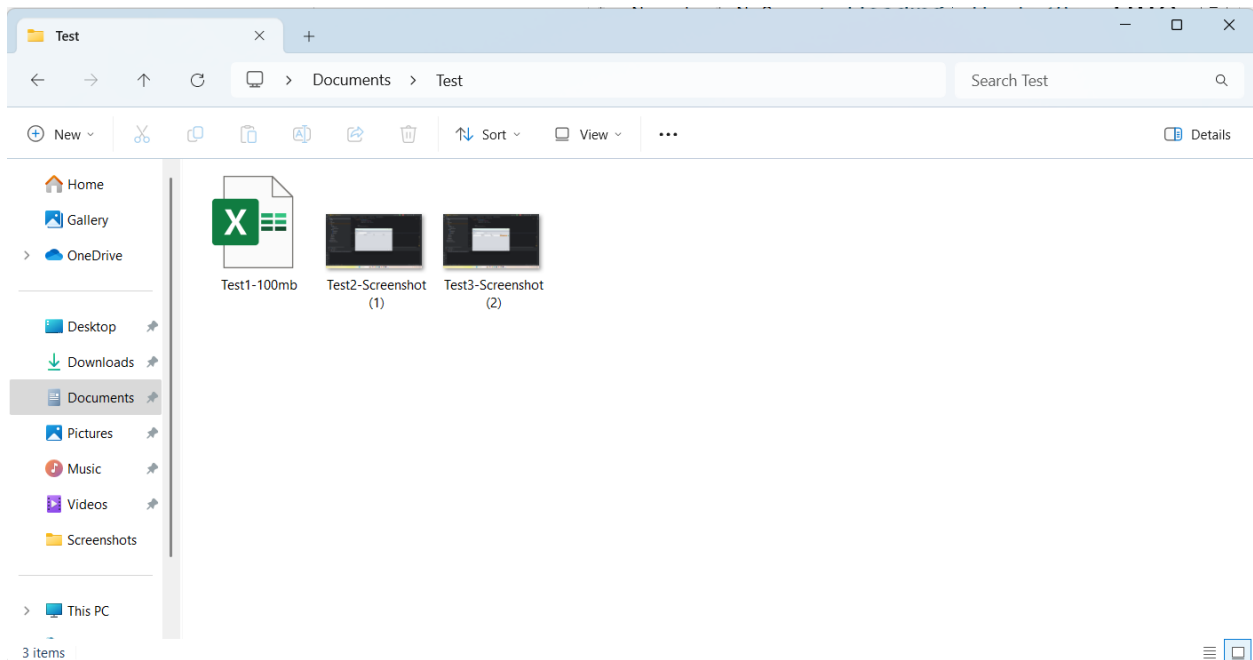
The screenshot shows a window titled "Server" with a table displaying a list of connected clients. The table has three columns: "No", "Name", and "Status". There are two rows of data: the first row has "1" in the "No" column and "Minh" in the "Name" column; the second row has "2" in the "No" column and "Hung" in the "Name" column. At the bottom right of the window is a "Start Server" button.

No	Name	Status
1	Minh	
2	Hung	

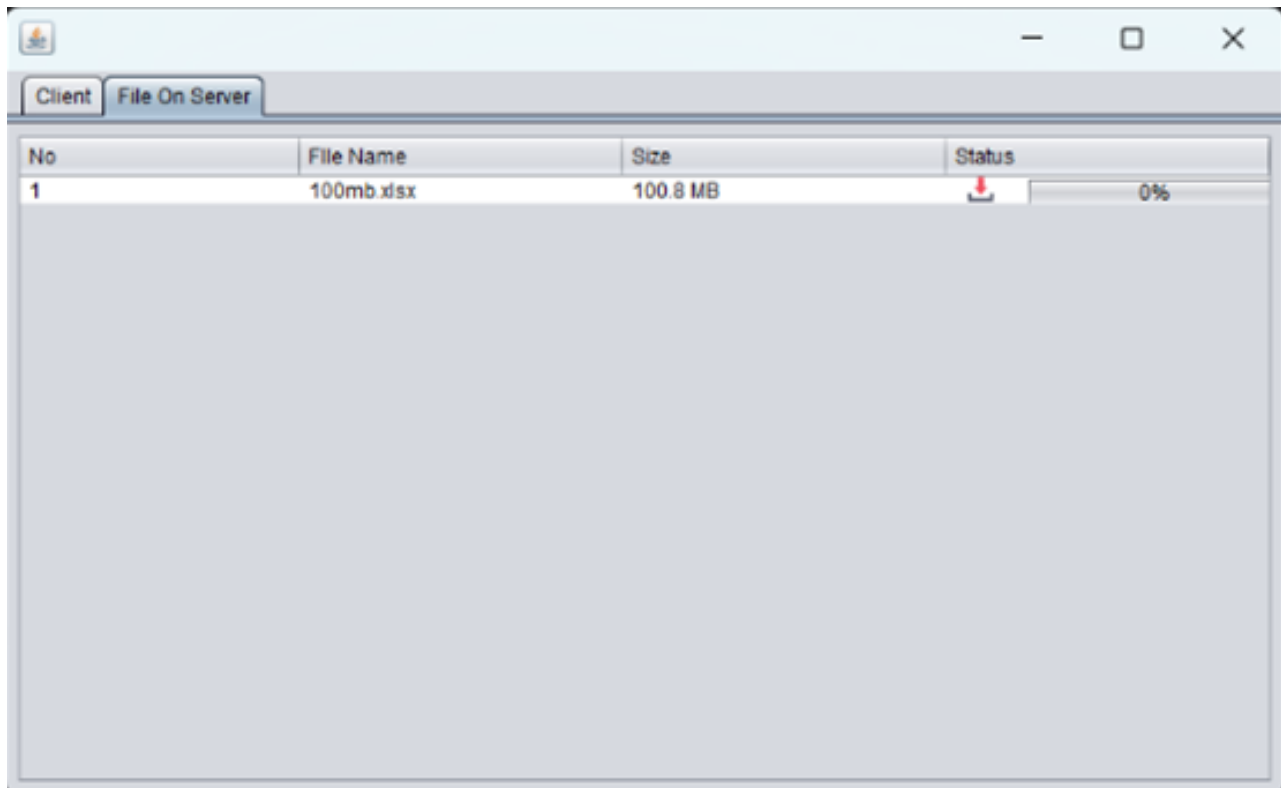
- Phía client, cụ thể là Client tên là Hung sẽ gửi file tới Server:



- Phía Server nhận được file và lưu trữ ở local của server:



- Client tên Minh hoàn toàn có thể xem và tải về file trên Server mà Client Hung vừa upload lên.



#### 4. Kết luận:

- Hệ thống đã xây dựng và triển khai thành công một hệ thống phân tán sử dụng mô hình Client-Server với ngôn ngữ lập trình Java và kỹ thuật Socket Programming. Hệ thống cho phép các máy Client thực hiện các thao tác upload và download tệp tin từ Server thông qua giao tiếp mạng TCP/IP. Đồng thời, hệ thống cũng cung cấp một giao diện người dùng đồ họa (GUI) giúp người dùng tương tác dễ dàng và trực quan hơn.

##### 4.1 Đáp ứng mô hình hệ thống phân tán:

- Hệ thống đã thể hiện đầy đủ các đặc điểm cơ bản của một hệ thống phân tán, cụ thể như sau:

- **Tính phân tán:** Hệ thống bao gồm nhiều thực thể phân tán về mặt vật lý, trong đó các máy Client có thể ở các vị trí khác nhau trong mạng nhưng vẫn có thể kết nối tới Server để thực hiện thao tác truyền tải dữ liệu.
- **Tính đồng thời:** Server được thiết kế có khả năng xử lý nhiều yêu cầu từ các Client khác nhau thông qua cơ chế đa luồng (multi-threading). Điều này



giúp đảm bảo rằng các Client có thể hoạt động độc lập và đồng thời mà không làm gián đoạn hoặc ảnh hưởng lẫn nhau.

- **Tính minh bạch trong truy cập:** Người dùng tương tác với hệ thống thông qua giao diện đồ họa mà không cần quan tâm đến các chi tiết kỹ thuật như vị trí vật lý của Server hay cách dữ liệu được truyền đi trong mạng.
- **Tính mở:** Hệ thống sử dụng các chuẩn mở như TCP/IP và được lập trình bằng Java – một ngôn ngữ lập trình phổ biến, giúp hệ thống có khả năng mở rộng và tích hợp với các thành phần khác trong tương lai.

#### 4.2 Giao tiếp từ xa (Remote Communication):

Giao tiếp giữa Client và Server được xây dựng dựa trên giao thức TCP, đảm bảo tính tin cậy trong truyền dữ liệu. Việc sử dụng Java Socket cho phép triển khai cơ chế kết nối mạng dễ dàng, hiệu quả và phù hợp với kiến trúc phân tán.

Thông qua các socket, Server có thể nhận yêu cầu từ các Client, xác định hành động cần thực hiện (upload/download), sau đó phản hồi dữ liệu về phía Client. Quá trình truyền tải dữ liệu đã được kiểm tra với các tệp có kích thước khác nhau, cho thấy hệ thống có thể xử lý hiệu quả mà không gặp lỗi trong quá trình truyền.

Mặc dù hệ thống mới chỉ triển khai ở mức cơ bản, nhưng cũng đã áp dụng các nguyên tắc cơ bản của bảo mật hệ thống phân tán, bao gồm:

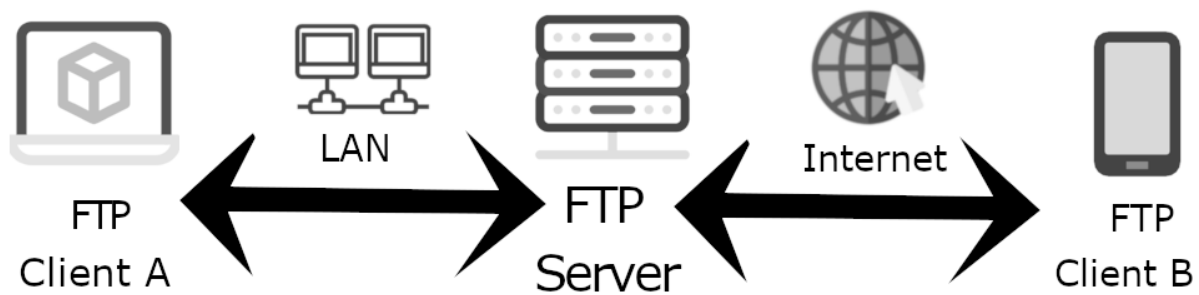
- Xác minh kết nối từ Client, đảm bảo rằng chỉ những Client hợp lệ mới được phép truy cập Server.
- Kiểm soát quyền truy cập vào tệp tin (Client chỉ có thể truy cập những tệp đã được chia sẻ hợp lệ).

Trong các phiên bản tương lai, hệ thống hoàn toàn có thể mở rộng thêm các cơ chế như mã hóa dữ liệu, xác thực người dùng qua tài khoản mật khẩu, hoặc sử dụng giao thức bảo mật SSL/TLS.

Hệ thống được thiết kế dựa trên kiến trúc Client-Server, một trong những kiến trúc phổ biến và đơn giản nhất trong hệ thống phân tán. Trong mô hình này:

- Client là thành phần chủ động gửi yêu cầu (upload/download) và nhận phản hồi.
- Server là thành phần lắng nghe các kết nối đến, xử lý các yêu cầu từ nhiều Client đồng thời, và thực hiện các hành động tương ứng.

Kiến trúc này phù hợp với quy mô triển khai hiện tại và có khả năng mở rộng lên các mô hình phức tạp hơn như Peer-to-Peer trong tương lai.



*Mô hình và tính năng của hệ thống đang xây dựng kiến trúc và tính năng tương tự như mô hình FTP Server cũng sử dụng kiến trúc Client-Server (Wikipedia)*

## **5. Danh mục tài liệu tham khảo:**

- <https://cloud.z.com/vn/en/news/ftp-server/>
- <https://itnavi.com.vn/blog/use-case-la-gi>
- <https://www.studocu.vn/vn/document/truong-dai-hoc-bach-khoa-ha-noi/he-thong-phan-tan/de-cuong-he-thong-phan-tan/113629800>
- <https://www.tpisoftware.vn/blog/enterprise-management/distributed-system>
- <https://fptcloud.com/socket-la-gi/>