ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN PBL4 HỆ ĐIỀU HÀNH & MẠNG

ĐỀ TÀI 203:
"XÂY DỰNG ỨNG DỤNG REMOTE FOLDER TRÊN CLOUD COMPUTING THEO MÔ HÌNH CLIENT – SERVER"

GIẢNG VIÊN HƯỚNG DẪN
Ths. Trần Hồ Thủy Tiên
Ths. Nguyễn Song Tùng - VNPT - IT3

SINH VIÊN THỰC HIỆN
SV. Huỳnh Văn Quân
SV. Trịnh Xuân Phúc
SV. Trần Hưng Dân
Lớp: 18TCLC_DT1
Nhóm: PBL4_04

Đà Nẵng, ngày 28 tháng 12 năm 2020

LỜI NÓI ĐẦU

Trong những năm gần đây, Đà Nẵng ngày càng phát triển mạnh và trở thành một trung tâm kinh tế, dịch vụ, công nghệ lớn. Do đó, việc phát triển các hệ thống, các ứng dụng nhằm tối ưu trong công việc, quản lý hiệu quả con người, cơ sở vật chất… là một điều hết sức cần thiết. Vì thế, để áp dụng những kiến thức đã học vào thực tế, chúng em đã cùng nhau lên ý tưởng và xây dựng một ứng dụng Remote Folder nhằm hướng tới xây dựng hệ thống quản lý và lưu trữ dữ liệu tối ưu, giải quyết được nhiều vấn đề trong thực tiễn. Trong quá trình thực hiện đồ án, có nhiều kiến thức về lập trình, giải thuật,… mà có thể chúng em chưa nắm được nên sai sót là điều khó tránh khỏi. Kính mong quý thầy cô tận tình góp ý để đồ án ngày một hoàn thiện hơn.

Chúng em xin chân thành cảm ơn sự hướng dẫn tận tình của quý thầy cô là giảng viên trong khoa trong suốt quá trình thực hiện đồ án này!

MỤC LỤC

GIỚI THIỆU

Vấn đề thực tế

Các thiết bị không đủ bộ nhớ để lưu trữ dữ liệu, cần một nơi có dung lượng lớn để lưu trữ.

Khi lưu trữ dữ liệu trên máy tính cá nhân, không thể truy cập dữ liệu mọi lúc mọi nơi.

Khi lưu trữ dữ liệu ở một nơi có nhiều rủi ro xảy ra như máy bị hư ổ cứng, bị nhiễm virus,… gây ảnh hưởng đến dữ liệu.

Khi làm việc chung trong cùng 1 tổ chức, cần một nơi lưu trữ dữ liệu nội bộ chung.

Vấn đề bao quát

- Cần xây dựng một ứng dụng quản lý và lưu trữ dữ liệu từ xa.
- Ứng dụng đáp ứng được đồng thời các yếu tố như tính tối giản, đầy đủ các chức năng cơ bản đối với file và folder,…
- Ứng dụng phải đáp ứng được các nhu cầu thực tế, giao diện đơn giản, giúp người dùng có thể tương tác dễ dàng.

Cách giải quyết

- Thiết kế và xây dựng một ứng dụng Remote Folder.

Thông tin đề tài

Mô tả đề tài: Remote folder là ứng dụng được thiết kế và xây dựng chạy trên môi trường Cloud computing. Thông qua ứng dụng này người dùng có thể lưu trữ các file, các folder trên Server. Cung cấp đầy đủ các chức năng như download, upload, delete, và xem toàn bộ nội dung của folder.

Yêu cầu đề tài:

Chương trình được thiết kế và xây dựng theo mô hình Client-Server, áp dụng kỹ thuật lập trình socket, lập trình multithread và semaphore để điều khiển các thao tác.

CLIENT

- Client chạy trên shell thông qua interpreter để thực thi các thao tác như: Connect đến Server; Upload filename; Download filename; Delete filename; DIR.
- Client cũng cần thống kê tốc độ upload/download dữ liệu để đánh giá.
- Thông tin thống kê phải được lưu ra file text.
- Thiết kế giao diện đồ họa.

SERVER

Server được đặt trên Host free (trao đổi trực tiếp với GVHD).

Server luôn rà soát để nhận ra và chấp nhận kết nối đến Client và điều khiển các lệnh, gửi đến Client.

Server điều khiển được nhiều Client kết nối đến Server.

Ý nghĩa lý luận và thực tiễn của đề tài

- Giúp sinh viên ứng dụng các kiến thức đã học vào thực tế, hình thành nên tư duy khi thiết kế và xây dựng ứng dụng hoàn chỉnh, tạo tiền đề để thực hiện những dự án lớn hơn.
- Giải quyết được các vấn đề lưu trữ dữ liệu trong cuộc sống, tạo ra một hệ thống quản lý tối ưu và chất lượng nhất.

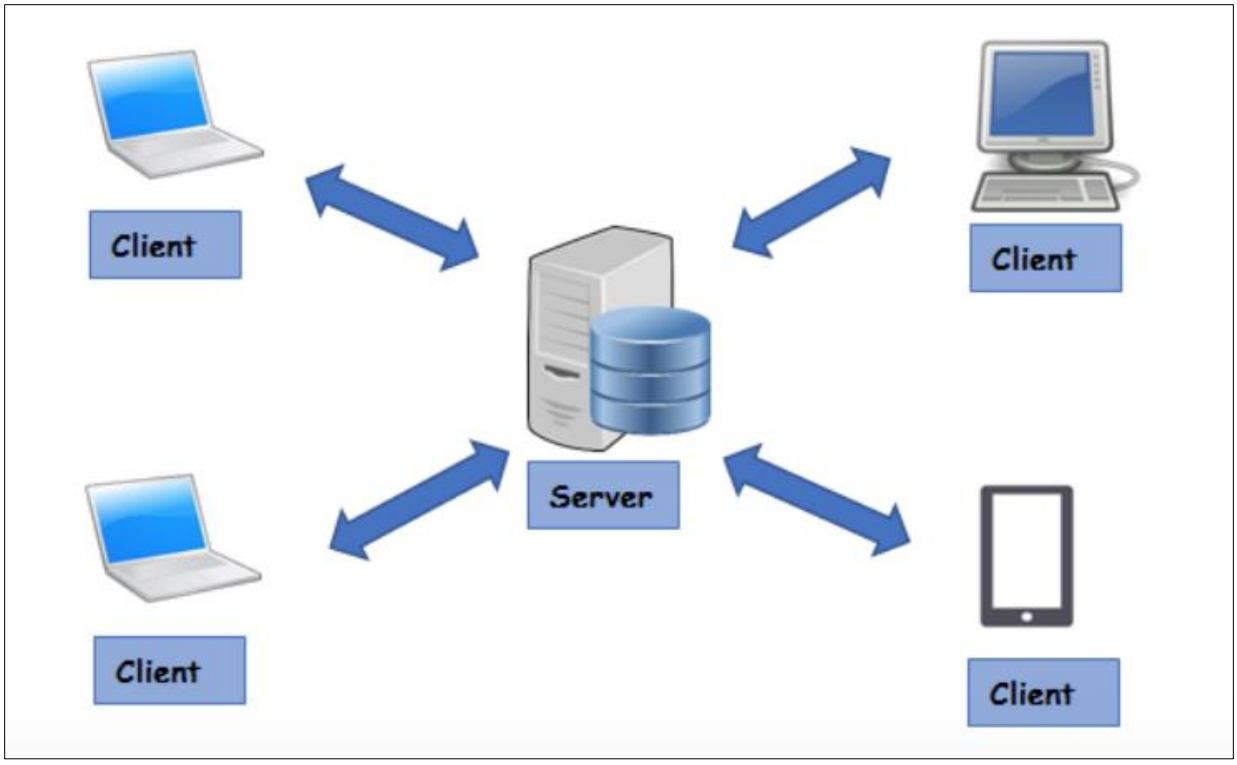
CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

1.1. Mô Hình Client - Server

1.1.1. Khái niệm

Khi sử dụng máy tính mỗi chúng ta đều có nhu cầu kết nối thu thập và chia sẻ thông tin. Để đáp ứng nhu cầu đó mạng máy tính hay hệ thống mạng (computer network) ra đời.

- Máy tính đóng vai trò là máy chủ – **Server**: Là máy tính có khả năng cung cấp tài nguyên và các dịch vụ đến các máy trạm khác trong hệ thống mạng. Server đóng vai trò hỗ trợ cho các hoạt động trên máy trạm client diễn ra hiệu quả hơn.
- Máy tính đóng vai trò là máy trạm – **Client**: Với vai trò là máy trạm, chúng sẽ không cung cấp tài nguyên đến các máy tính khác mà chỉ sử dụng tài nguyên được cung cấp từ máy chủ. Một client trong mô hình này có thể là một server cho mô hình khác, tùy thuộc vào nhu cầu sử dụng của người dùng.
- Mô hình mạng **Client - Server** là mô hình mạng máy tính trong đó các máy tính con được đóng vai trò như một máy khách, chúng làm nhiệm vụ gửi yêu cầu đến các máy chủ. Để máy chủ xử lý yêu cầu và trả kết quả về cho máy khách đó.

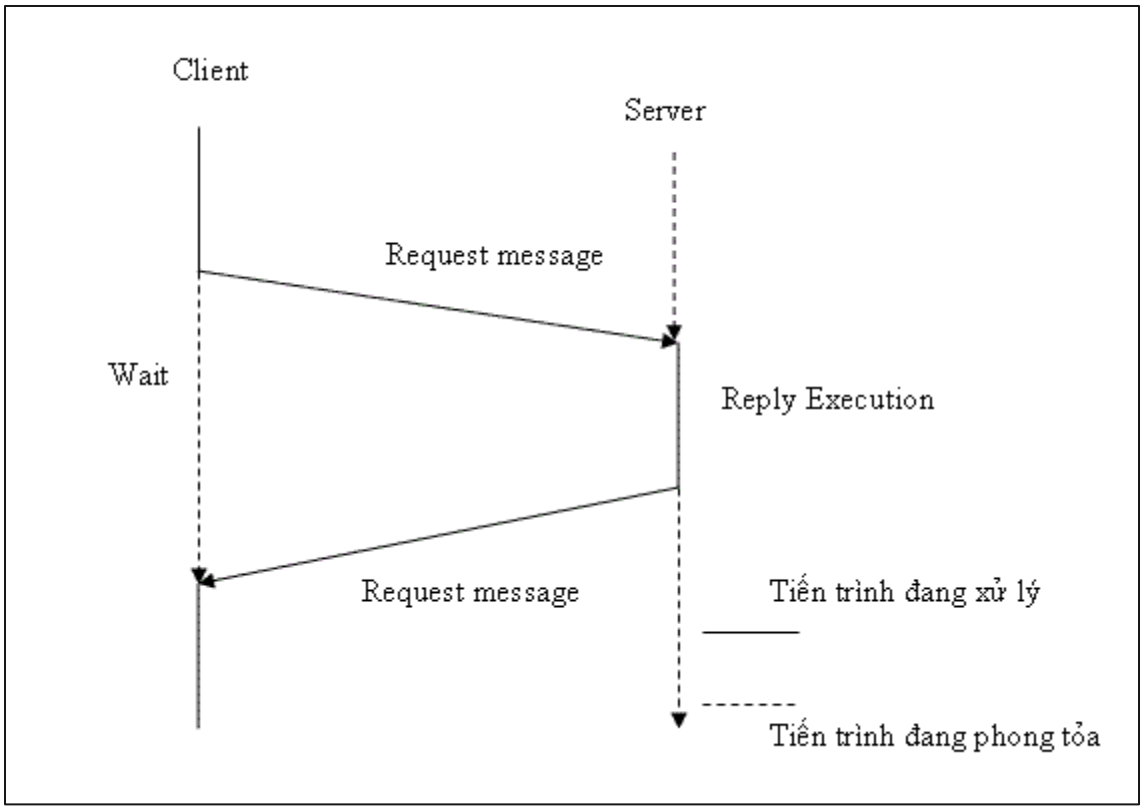


Hình 1.1: Mô hình client - server

1.1.2. Nguyên lý hoạt động

Trong mô hình Client - Server, Server chấp nhận tấy cả các yêu cầu hợp lệ từ mọi nơi khác nhau trên mạng, sau đó trả kết quả về máy tính đã gửi yêu cầu.

Máy tính được coi là máy khách khi chúng làm nhiệm vụ gửi yêu cầu đến các máy chủ và đợi câu trả lời được gửi về.



Hình 1.2: Nguyên lý hoạt động Client - Server

Để máy khách và máy chủ có thể giao tiếp được với nhau thì giữa chúng phải có một chuẩn nhất định, và chuẩn đó được gọi là giao thức. Một số giao thức chuẩn được sử dụng rộng rãi hiện nay như TCP/IP, OSI, ISDN, X.25, LAN-to-LAN,.. Khi đó, nếu máy khách muốn lấy được thông tin từ máy chủ, chúng phải tuân theo một giao thức mà máy chủ đó đưa ra. Nếu yêu cầu đó được chấp nhận thì máy chủ sẽ thu thập thông tin và trả về kết quả cho máy khách yêu cầu. Bởi thông thường, server luôn trong trạng thái sẵn sàng nhận yêu cầu từ các client, nên chỉ cần client gửi tín hiệu và chấp nhận yêu cầu là server sẽ trả về kết quả trong thời gian ngắn nhất có thể.

Quá trình giao tiếp giữa Client – Server:

- Bước 1: Truyền một yêu cầu từ tiến trình Client tới tiến trình Server.
- Bước 2: Yêu cầu được Server xử lý.
- Bước 3: Truyền đáp ứng cho Client.

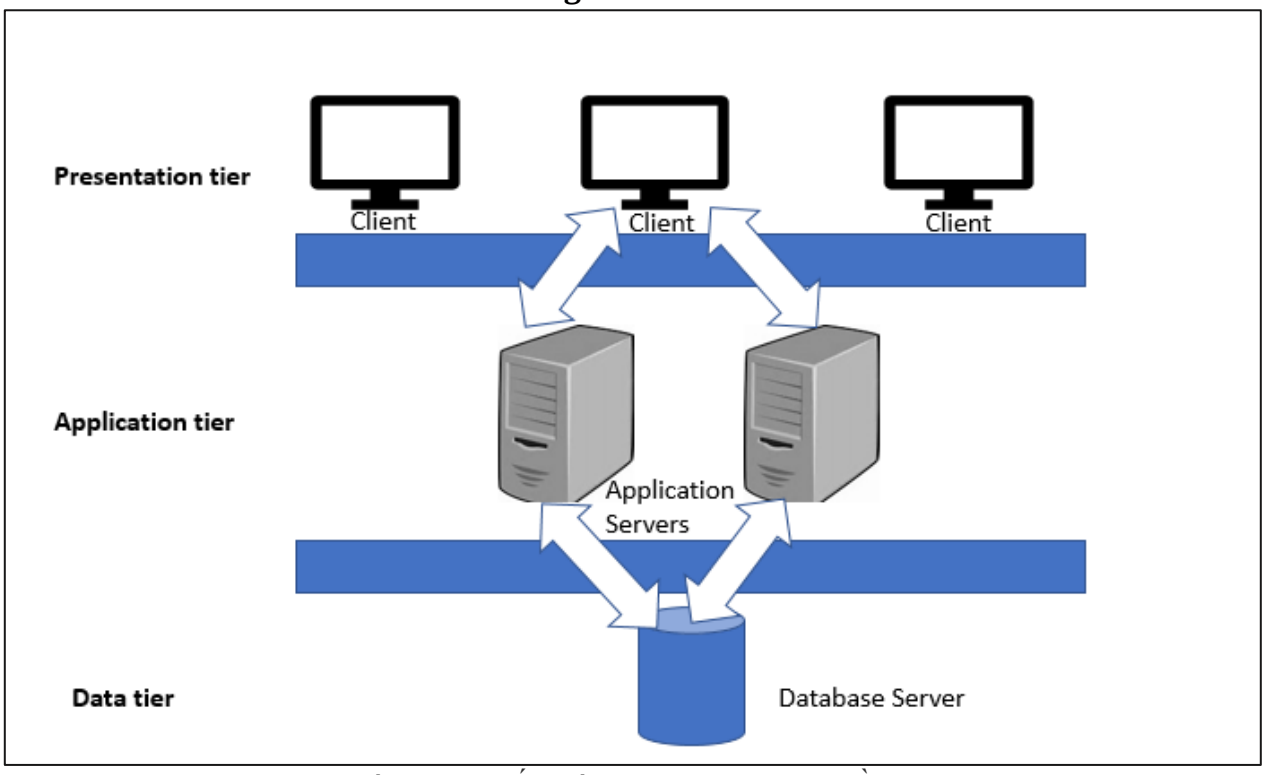
Quá trình giao tiếp giữa Client và Server có thể diễn ra theo một trong hai chế độ: bị phong tỏa (blocked) và không bị phong tỏa (non-blocked):

Chế độ bị phong tỏa: Khi tiến trình Client hoặc Server phát ra lệnh gửi dữ liệu (send), việc thực thi của tiến trình sẽ bị tạm ngừng cho tới khi tiến trình nhận phát ra lệnh nhận dữ liệu (receive). Tương tự đối với tiến trình nhận dữ liệu, nếu tiến trình nào đó (Client hoặc Server) phát ra lệnh nhận dữ liệu, mà tại thời điểm đó chưa có dữ liệu gửi tới thì việc thực thi của tiền trình cũng sẽ bị tạm ngừng cho tới khi có dữ liệu gửi tới.

Chế độ không bị phong tỏa: Khi tiến trình Client hoặc Server phát ra lệnh gửi dữ liệu thực sự, việc thực thi của tiến trình vẫn được tiến hành mà không quan tâm đến việc có tiến trình nào phát ra lệnh nhận dữ liệu đó hay không. Tương tự cho trường hợp nhận dữ liệu, khi tiến trình phát ra lệnh nhận dữ liệu, nó sẽ nhận dữ liệu hiện có, việc thực thi của

tiến trình vẫn được tiến hành mà không quan tâm đến việc có tiến trình nào phát ra lệnh gửi dữ liệu tiếp theo hay không.

1.1.3. Kiến trúc Client – Server ba tầng



Hình 1.3: Kiến trúc Client – Server ba tầng

Theo kiến trúc ba tầng, một ứng dụng được chia thành ba tầng tách biệt nhau về mặt logic. Tầng đầu tiên là tầng trình diễn thường bao gồm các giao diện đồ họa. Tầng thứ hai, còn được gọi là tầng trung gian hay tầng tác nghiệp. Tầng thứ ba chứa dữ liệu cần cho ứng dụng.

Tầng thứ ba về cơ bản là chương trình thực hiện các lời gọi hàm để tìm kiếm dữ liệu cần thiết. Tầng thứ ba chứa dữ liệu cần thiết cho ứng dụng. Dữ liệu này có thể bao gồm bất kỳ nguồn thông tin nào, bao gồm cơ sở dữ liệu như Oracale, SQL Server hoặc tài liệu XML. Tầng trình diễn nhận dữ liệu và định dạng nó để hiển thị. Sự tách biệt giữa chức năng xử lý với giao diện đã tạo nên sự linh hoạt cho việc thiết kế ứng dụng. Nhiều giao diện người dùng được xây dựng và triển khai mà không làm thay đổi logic ứng dụng.

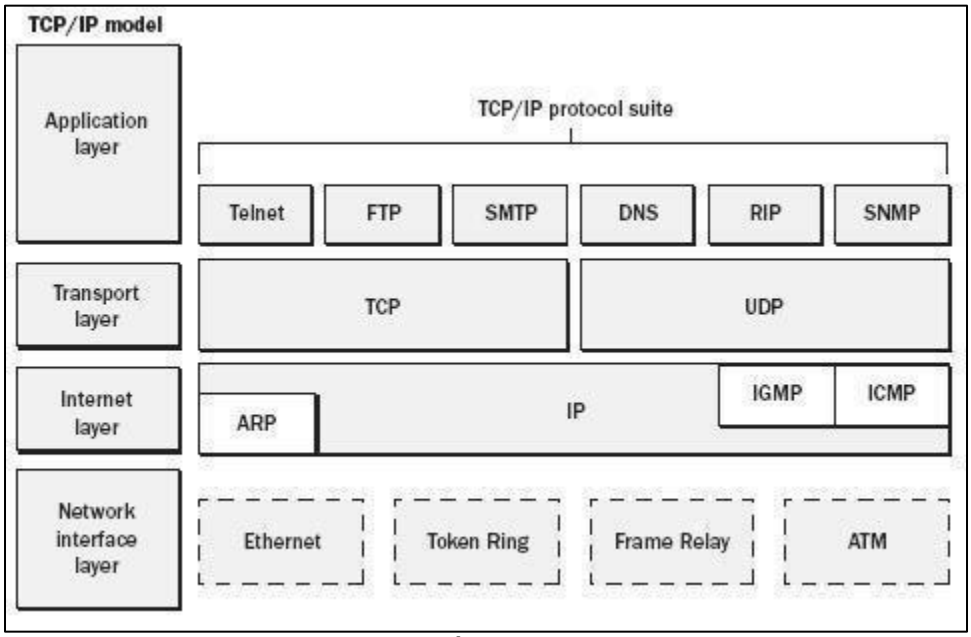
1.2. Giao Thức TCP/IP

1.2.1. Khái niệm

TCP/IP là tên chung cho một tập hợp hơn 100 giao thức được sử dụng để kết nối các máy tính vào mạng, trong đó hai giao thức chính là TCP (Transmission Control Protocol) và IP (Internet Protocol).

Trong phạm vi Internet, thông tin không được truyền tải như một dòng riêng biệt từ máy tính này tới máy tính khác. Thay vào đó, dữ liệu được chia thành những gói nhỏ gọi là packet.

Các packet này được gửi trên mạng máy tính. Công việc của IP là chuyển chúng đến các máy tính ở xa. Tại trạm cuối, TCP nhận các packet và kiểm tra lỗi. Nếu một lỗi xuất hiện, TCP yêu cầu gói riêng biệt đó phải được gửi lại. Chỉ khi tất cả các packet đã nhận được là đúng, TCP sẽ sử dụng số thứ tự để tạo lại thông tin ban đầu.



Hình 1.4: Sơ đồ giao thức TCP/IP

1.2.2. Giao thức IP (Internet Protocol – Giao thức liên mạng)

Giao thức IP là một giao thức hướng dữ liệu được sử dụng bởi các máy chủ nguồn và đích để truyền dữ liệu trong một liên mạng chuyển mạch gói.

Dữ liệu trong một liên mạng IP được gửi theo các khối được gọi là các gói (packet hoặc datagram). Cụ thể, IP không cần thiết lập các đường truyền trước khi một máy chủ gửi các gói tin cho một máy khác mà trước đó nó chưa từng liên lạc với.

Giao thức IP cung cấp một dịch vụ gửi dữ liệu không đảm bảo (còn gọi là cố gắng cao nhất), nghĩa là nó hầu như không đảm bảo gì về gói dữ liệu. Gói dữ liệu có thể đến nơi mà không còn nguyên vẹn, nó có thể đến không theo thứ tự (so với các gói khác được gửi giữa hai máy nguồn và đích đó), nó có thể bị trùng lặp hoặc bị mất hoàn toàn. Nếu một phần mềm ứng dụng cần được bảo đảm, nó có thể được cung cấp từ nơi khác, thường từ các giao thức giao vận nằm phía trên IP.

Các thiết bị định tuyến liên mạng chuyển tiếp các gói tin IP qua các mạng tầng liên kết dữ liệu được kết nối với nhau. Việc không có đảm bảo về gửi dữ liệu có nghĩa rằng các chuyển mạch gói có thiết kế đơn giản hơn. (Lưu ý rằng nếu mạng bỏ gói tin, làm đổi thứ tự hoặc làm hỏng nhiều gói tin, người dùng sẽ thấy hoạt động mạng trở nên kém đi. Hầu hết các thành phần của mạng đều cố gắng tránh để xảy ra tình trạng đó. Đó là lý do giao thức này còn được gọi là cố gắng cao nhất. Tuy nhiên, khi lỗi xảy ra không thường xuyên sẽ không có hiệu quả đủ xấu đến mức người dùng nhận thấy được.)

Giao thức IP rất thông dụng trong mạng Internet công cộng ngày nay. Giao thức tầng mạng thông dụng nhất ngày nay là IPv4; đây là giao thức IP phiên bản 4. IPv6 được

đề nghị sẽ kế tiếp IPv4: Internet đang hết dần địa chỉ IPv4, do IPv4 sử dụng 32 bit để đánh địa chỉ (tạo được khoảng 4 tỷ địa chỉ); IPv6 dùng địa chỉ 128 bit, cung cấp tối đa khoảng 3.4×1038 địa chỉ. Các phiên bản từ 0 đến 3 hoặc bị hạn chế, hoặc không được sử dụng. Phiên bản 5 được dùng làm giao thức dòng (stream) thử nghiệm. Còn có các phiên bản khác, nhưng chúng thường dành là các giao thức thử nghiệm và không được sử dụng rộng rãi.

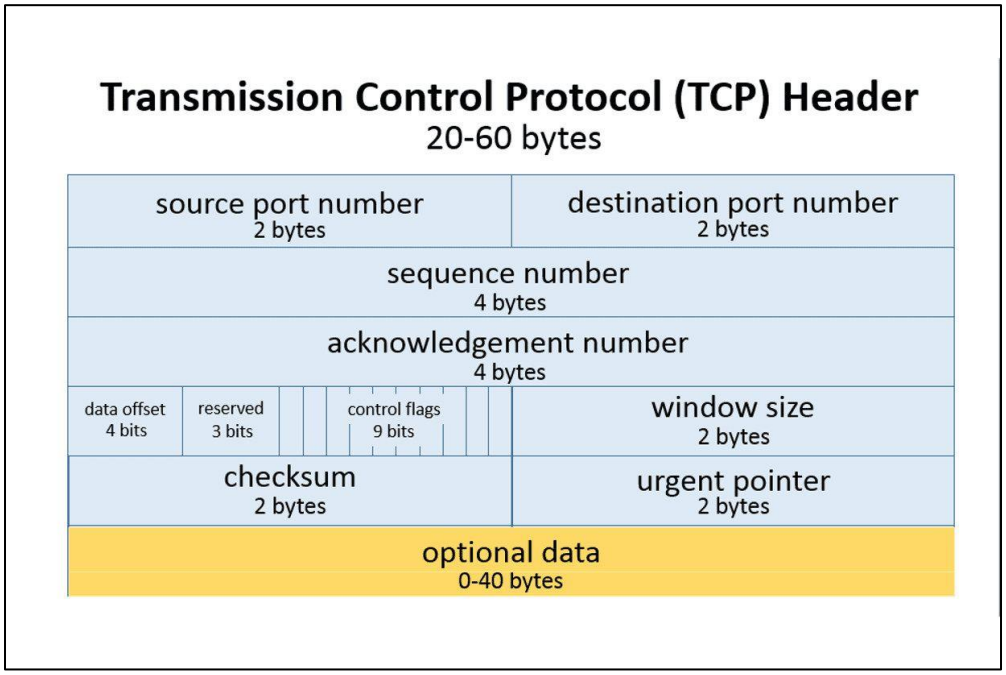
1.2.3. Giao thức TCP (Transmission Control Protocol)

TCP là một trong các giao thức cốt lõi của bộ giao thức TCP/IP. Sử dụng TCP, các ứng dụng trên các máy chủ được nối mạng có thể tạo các "kết nối" với nhau, mà qua đó chúng có thể trao đổi dữ liệu hoặc các gói tin. Giao thức này đảm bảo chuyển giao dữ liệu tới nơi nhận một cách đáng tin cậy và đúng thứ tự. TCP còn phân biệt giữa dữ liệu của nhiều ứng dụng (chẳng hạn, dịch vụ Web và dịch vụ thư điện tử) đồng thời chạy trên cùng một máy chủ.

TCP hỗ trợ nhiều giao thức ứng dụng phổ biến nhất trên Internet và các ứng dụng kết quả, trong đó có WWW, thư điện tử và Secure Shell.

Trong bộ giao thức TCP/IP, TCP là tầng trung gian giữa giao thức IP bên dưới và một ứng dụng bên trên. Các ứng dụng thường cần các kết nối đáng tin cậy kiểu đường ống để liên lạc với nhau, trong khi đó, giao thức IP không cung cấp những dòng kiểu đó, mà chỉ cung cấp dịch vụ chuyển gói tin không đáng tin cậy. TCP làm nhiệm vụ của tầng giao vận trong mô hình OSI đơn giản của các mạng máy tính.

Các ứng dụng gửi các dòng gồm các byte 8- bit tới TCP để chuyển qua mạng. TCP phân chia dòng byte này thành các đoạn (segment) có kích thước thích hợp (thường được quyết định dựa theo kích thước của đơn vị truyền dẫn tối đa (MTU) của tầng liên kết dữ liệu của mạng mà máy tính đang nằm trong đó). Sau đó, TCP chuyển các gói tin thu được tới giao thức IP để gửi nó qua một liên mạng tới mô đun TCP tại máy tính đích. TCP kiểm tra để đảm bảo không có gói tin nào bị thất lạc bằng cách gán cho mỗi gói tin một "số thứ tự" (sequence number). Số thứ tự này còn được sử dụng để đảm bảo dữ liệu được trao cho ứng dụng đích theo đúng thứ tự. Mô đun TCP tại đầu kia gửi lại "tin báo nhận" (acknowledgement) cho các gói tin đã nhận được thành công; một "đồng hồ" (timer) tại nơi gửi sẽ báo time-out nếu không nhận được tin báo nhận trong khoảng thời gian bằng một round-trip time (RTT), và dữ liệu (được coi là bị thất lạc) sẽ được gửi lại. TCP sử dụng checksum (giá trị kiểm tra) để xem có byte nào bị hỏng trong quá trình truyền hay không; giá trị này được tính toán cho mỗi khối dữ liệu tại nơi gửi trước khi nó được gửi, và được kiểm tra tại nơi nhận.



Hình 1.5: Cấu trúc header của TCP

1.2.4. Tầng ứng dụng (Application)

Tầng ứng dụng đảm nhận vai trò giao tiếp dữ liệu giữa 2 máy khác nhau thông qua các dịch vụ mạng khác nhau (duyệt web, chay hay các giao thức trao đổi dữ liệu SMTP, SSH, FTP…). Dữ liệu khi đến được tầng 4 sẽ được định dạng để kết nối theo kiểu Byte nối Byte. Các thông tin định tuyến tại đây sẽ giúp xác định đường đi đúng của một gói tin.

1.2.5. Tầng giao vận (Transport)

Chịu trách nhiệm duy trì liên lạc đầu cuối trên toàn mạng.

Tầng này có 2 giao thức chính là TCP và UDP.

TCP sẽ đảm bảo chất lượng truyền gửi gói tin, nhưng tốn khá nhiều thời gian để kiểm tra đầy đủ thông tin từ thứ tự dữ liệu cho đến việc kiểm soát vấn đề tắc nghẽn lưu lượng dữ liệu. Trái với TCP, UDP có thấy tốc độ truyền tải nhanh hơn nhưng lại không đảm bảo được chất lượng dữ liệu được gửi đi (tức là nó không quan tâm dữ liệu có đến được đích hay không).

1.2.6. Tầng mạng (Internet)

Tầng này chịu trách nhiệm xử lý quá trình truyền gói tin trên mạng:

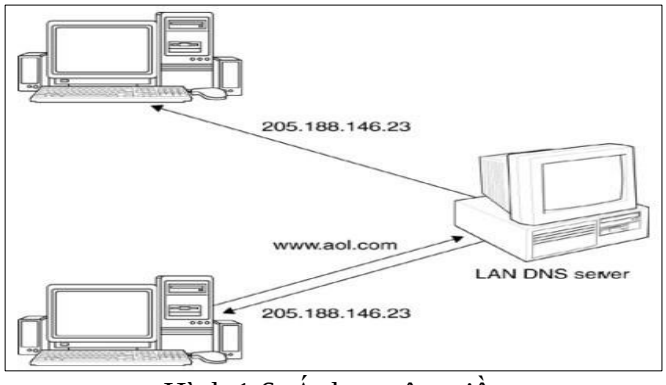
- Định tuyến: tìm tuyến qua các nút trung gian để gửi dữ liệu từ nguồn tới đích.
- Chuyển tiếp: chuyển tiếp gói tin từ cổng nguồn tới cổng đích theo tuyến đường.
- Định địa chỉ : định danh cho các nút mạng.
- Đóng gói dữ liệu: nhận dữ liệu từ giao thức ở trên, chèn thêm phần Header chứa thông tin của tầng mạng và tiếp tục được chuyển đến tầng tiếp theo.

Đảm bảo chất lượng dịch vụ (Quality of Service là tập hợp các kĩ thuật cho phép cấp phát các tài nguyên một cách thích hợp cho các loại dữ liệu khác nhau, từ đó có thể

đảm bảo chất lượng dịch vụ mạng cho các loại dữ liệu này): đảm bảo các thông số phù hợp của đường truyền theo từng dịch vụ.

Các giao thức của tầng này bao gồm: IP (Internet Protocol - giao thức được sử dụng rộng rãi trong mọi hệ thống mạng trên phạm vi toàn thế giới), ICMP (Internet Control Message Protocol), IGMP (Internet Group Message Protocol).

Mọi máy tính trong môi trường Internet đều được xác định bởi địa chỉ IP 4-byte mà được viết dạng chấm như 128.250.25.158, trong đó mỗi byte là một giá trị không dấu từ 0 đến 255. Nhưng cách đặt tên cho địa chỉ như vậy thì không thân thiện bởi vì nó chẳng nói lên được bất cứ điều gì về nội dung và thật khó để nhớ những dãy byte đó. Do đó Hệ thống phân giải tên miền DNS (Domain Name System) ra đời. Nó chuyển đổi" (ánh xạ) các con số địa chỉ IP khô khan thành những ký tự ABC thân thiện hơn. Nhờ DNS nên ta không cần phải nhớ địa chỉ IP để vào website mà chỉ cần nhớ tên miền của nó. Ví dụ như www.google.com.vn



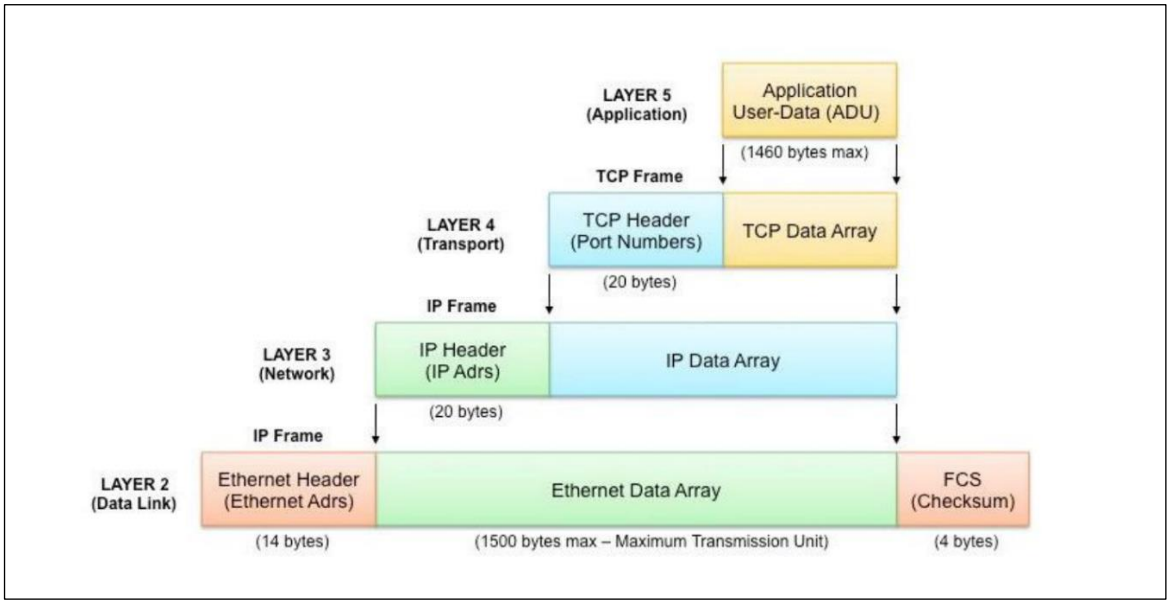
Hình 1.6: Ánh xạ tên miền

Thông thường, mỗi máy tính chỉ có một địa chỉ IP. Tuy nhiên, các máy tính thường cần kết nối và cung cấp hơn một loại kiểu dịch vụ hoặc để thông báo cho nhiều máychủ/máy tính cùng thời điểm. Ví dụ, trong một máy tính đang làm việc, có thể có nhiều phiên FTP, các kết nối web, và các chương trình chat và tất cả chương trình này cùng chạy một thời điểm. Vậy để phân biệt các dịch vụ này, ta có khái niệm về cổng (port), là một điểm truy cập logic được biểu diễn bởi một số nguyên 16-bit gắn cho mỗi tiến trình mạng. Và mỗi tiến trình mạng đều được gắn một cổng duy nhất.

1.2.7. Tầng vật lí (Physical)

Tầng vật lý (còn được gọi là tầng liên kết dữ liệu) là tầng thấp nhất trong mô hình TCP/IP. Tầng này chịu trách nhiệm truyền dữ liệu giữa hai thiết bị trong cùng một mạng. Tại đây, các gói dữ liệu được đóng vào khung (gọi là Frame) và được định tuyến đi đến đích đã được chỉ định ban đầu.

1.2.8. Cách thức hoạt động

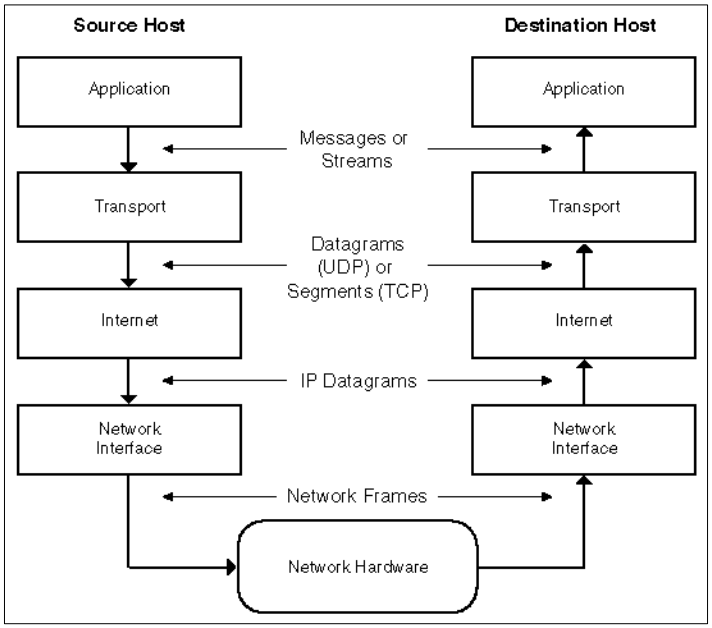


Hình 1.7: Mô hình hoạt động của TCP/IP

Khi truyền dữ liệu , quá trình tiến hành từ tầng trên xuống tầng dưới, qua mỗi tầng dữ liệu được thêm vào thông tin điều khiển gọi là Header. Khi nhận dữ liệu thì quá trình xảy ra ngược lại, dữ liệu được truyền từ tầng dưới lên và qua mỗi tầng thì phần header tương ứng sẽ được lấy đi và khi đến tầng trên cùng thì dữ liệu không còn phần header nữa.

Ở đây, IP có vai trò quan trọng, nó cho phép các gói tin được gửi đến đích đã định sẵn, bằng cách thêm các thông tin dẫn đường (chính là Header) vào các gói tin để các gói tin được đến đúng đích đã định sẵn ban đầu.

Giao thức TCP đóng vai trò kiểm tra và đảm bảo sự an toàn cho mỗi gói tin khi đi qua mỗi trạm. Trong quá trình này, nếu giao thức TCP nhận thấy gói tin bị lỗi, một tín hiệu sẽ được truyền đi và yêu cầu hệ thống gửi lại một gói tin khác.

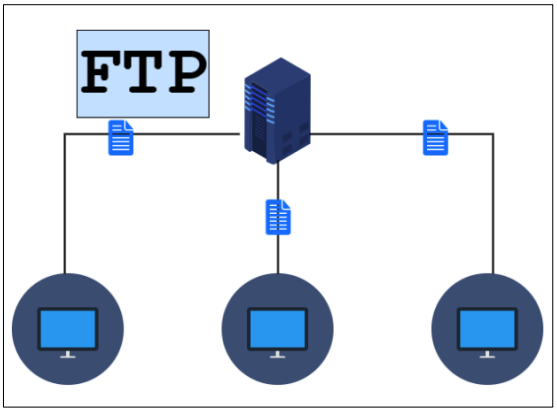


Hình 1.8: Cấu trúc dữ liệu qua các tầng

1.3. Giao Thức FTP (File Transfer Protocol)

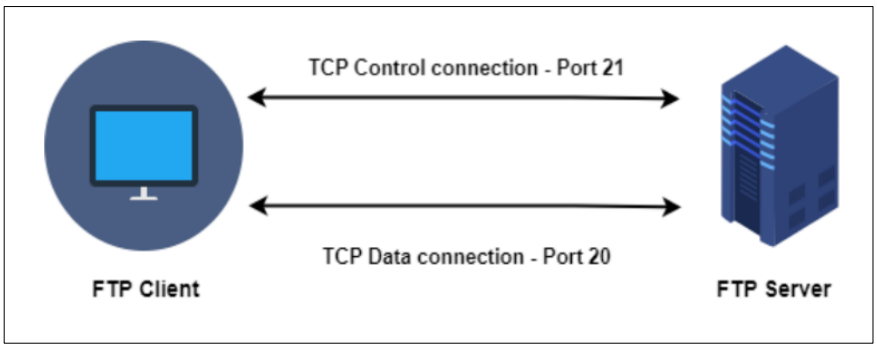
1.3.1. Khái niệm

FTP - File Transfer Protocol (Giao thức truyền tải tập tin) được dùng trong việc trao đổi dữ liệu trong mạng thông qua giao thức TCP/IP, thường hoạt động trên 2 cổng là 20 và 21. Với giao thức này, các máy client trong mạng có thể truy cập đến máy chủ FTP để gửi hoặc lấy dữ liệu. Điểm nổi bật là người dùng có thể truy cập vào máy chủ FTP để truyền và nhận dữ liệu dù đang ở xa.



Hình 1.9: Minh hoạ giao thức FTP

1.3.2. Mô hình và nguyên lý hoạt động của FTP



Hình 1.10: Kết nối trong giao thức FTP

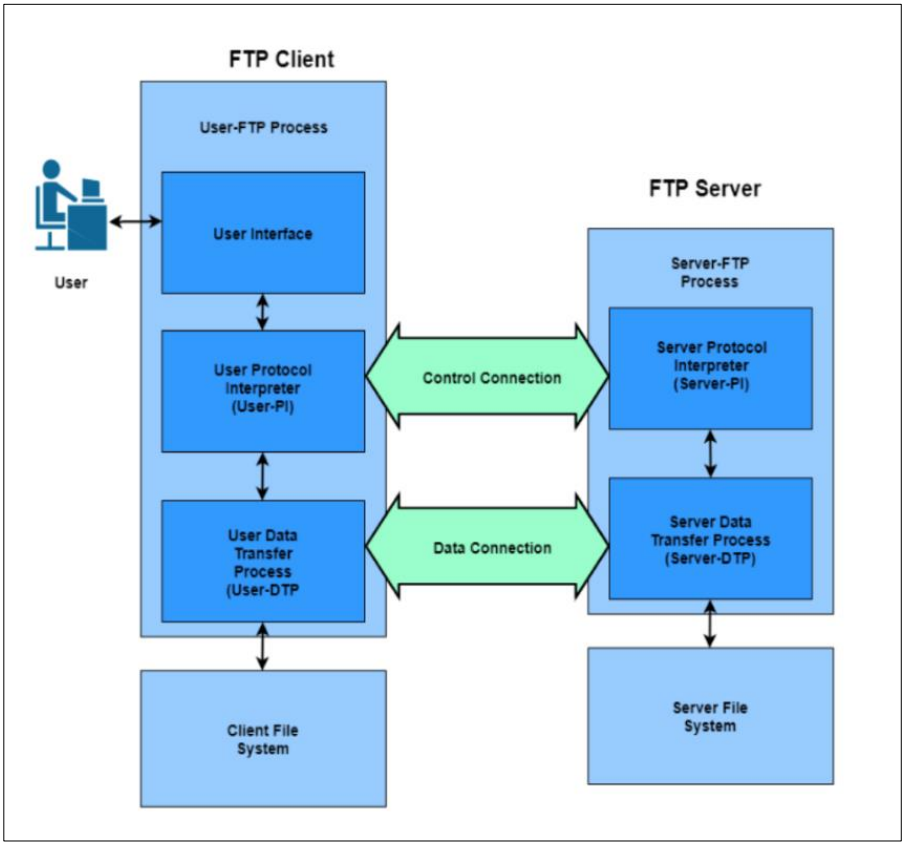
Giống như hầu hết các giao thức TCP/IP, FTP dựa trên mô hình Client – Server. Tuy nhiên, khác với các ứng dụng khác chạy trên nền TCP/IP, FTP cần tới 2 kết nối TCP:

- Control connection (sử dụng port 21 – trên server): Đây là kết nối TCP logic chính được tạo ra khi phiên làm việc được thiết lập. Nó được thực hiện giữa các quá trình điều khiển. Nó được duy trì trong suốt phiên làm việc và chỉ cho các thông tin điều khiển đi qua như lệnh hay response(phản hồi)
- Data connection (sử dụng port 20 – trên server): Kết nối này sử dụng các quy tắc rất phức tạp vì các loại dữ liệu có thể khác nhau. Nó được thực hiện giữa các quá trình truyền dữ liệu. Kết nối này mở khi có lệnh chuyển tệp và đóng khi tệp truyền xong.

Do chức năng điều khiển và dữ liệu được truyền tải bằng cách sử dụng các kênh riêng biệt nên mô hình FTP chia mỗi thiết bị thành 2 phần giao thức logic chịu trách nhiệm cho mỗi kết nối ở trên:

• Protocol interpreter (PI): Là thành phần quản lý kênh điều khiển, phát và nhận lệnh và trả lời.

• Data transfer process (DTP): chịu trách nhiệm gửi và nhận dữ liệu giữa client và server.



Hình 1.11: Sơ đồ minh hoạ mô hình FTP

Phía server:

• Server Protocol Interpreter (Server-PI) : Chịu trách nhiệm quản lí Control Connection trên Server. Nó lắng nghe yêu cầu kết nối hướng từ User trên cổng 21. Khi kết nối được thiết lập, nó nhận lệnh từ User-PI, gửi phản hồi và quản lí tiến trình truyền dữ liệu trên Server.

• Server Data Transfer Process (Server-DTP) : chịu trách nhiệm nhận và gửi file từ User-DTP. Server-DTP vừa làm nhiệm vụ thiết lập Data Connection và lắng nghe Data Connection của User thông qua cổng 20. Nó tương tác với Server File System trên hệ thống cục bộ để đọc và chép file.

Phía client:

• User Interface: Đây là chương trình được chạy trên máy tính, nó cung cấp giao diện xử lí cho người dùng, chỉ có trên phía Client. Nó cho phép người dùng sử dụng những lệnh đơn giản để điều khiển các session FTP, từ đó có thể theo dõi được các thông tin và kết quả xảy ra trong quá trình.

• User Protocol Interpreter (User-PI): Chịu trách nhiệm quản lí Control Connection phía Client. Nó khởi tạo phiên kết nối FTP bằng việc phát hiện ra Request tới Server-PI.

Sau khi kết nối được thiết lập, nó xử lí các lệnh nhận được trên User Interface, gửi chúng tới Server-PI rồi đợi nhận Response trở lại. Nó cũng quản lí các tiến trình trên Client.

- User Data Transfer Process (User-DTP): Có nhiệm vụ gửi hoặc nhận dữ liệu từ Server-DTP. User-DTP có thể thiết lập hoặc lắng nghe DataConnection từ Server thông qua cổng 20. Nó tương tác với Client File System trên Client để lưu trữ file.

1.3.3. Nguyên lí hoạt động

Cần có 2 kết nối TCP trong phiên làm việc của FTP: TCP Data connection trên cổng 20, TCP Control connection trên cổng 21.

- Control connection : luôn được mở ở mọi thời điểm khi dữ liệu hoặc lệnh được gửi.
- Data connection : chỉ được mở khi có trao đổi dữ liệu thực.

Trình tự chung của FTP hoạt động như sau:

- Bước 1: FTP Client mở Control connection đến FTP server (trên port 21) và chỉ định 1 cổng trên Client để Server gửi lại phản hồi. Đường kết nối này dùng để truyền lệnh và không phải là dữ liệu. Control connection sẽ mở trong suốt thời gian của phiên làm việc (telnet giữa 2 hệ thống)
- Bước 2: Client chuyển tiếp thông tin như username, password tới Server để thực hiện xác thực (authentication). Server sẽ trả lời bằng mã chấp nhận hay từ chối của các request.
- Bước 3: Client gửi thêm các lệnh với tên tệp, kiểu dữ liệu, … để vận chuyển, thêm luồng dữ liệu(tức là chuyển tập tin từ máy khách đến máy chủ hoặc ngược lại). Server sẽ phản hồi với mã (reply code) chấp nhận hoặc từ chối.
- Bước 4: Khi dữ liệu đã sẵn sàng, 2 bên sẽ mở kết nối TCP trên cổng 20.
- Bước 5: Dữ liệu có thể được vận chuyển giữa Client và Server trên cổng 20. Dữ liệu vận chuyển được mã hóa theo 1 số định dạng bao gồm NVT-ASCII hoặc nhị phân(binary)
- Bước 6: Khi quá trình vận chuyển dữ liệu được hoàn thành, phiên làm việc của FTP Server sẽ đóng lại Data Connection trên cổng 20. Nhưng vẫn giữ Control Connection trên công 21.
- Bước 7: Control connection có thể được sử dụng để thiết lập truyền dữ liệu khác hoặc đóng liên kết.

1.3.4. Kênh dữ liệu trong FTP

Kênh điều khiển được tạo ra giữa Server-PI và User-PI, sử dụng quá trình thiết lập kết nối và chứng thực được duy trì trong suốt phiên kết nối FTP. Các lệnh và các hồi đáp được trao đổi giữa bộ phận PI (Protocol Interpreter) qua kênh điều khiển, những dữ liệu thì không.

Mỗi khi cần phải truyền dữ liệu giữa các server và client, một kênh dữ liệu cần phải được tạo ra. Kênh dữ liệu kết nối bộ phận User-DTP và Server-DTP. Kết nối này cần thiết cho cả hoạt động truyền file trực tiếp (gửi hoặc nhận một file) cũng như đối với việc truyền dữ liệu ngầm, như là yêu cầu một danh sách file trong thư mục nào đó trên server.

Để tạo ra kênh dữ liệu, FTP sử dụng 2 phương thức khác nhau: Normal (Active) Data Connections (mặc định) và Passive Data Connections. Khác biệt giữa 2 phương thức này là phía Client hay bên Server đưa ra yêu cầu khởi tạo kết nối.

- Phương thức tạo kết nối bị động (Active Data Connections)

Phương thức tạo kết nối dữ liệu bình thường hay còn gọi là Kết nối kênh dữ liệu ở dạng chủ động.

Phía Server-DTP tạo kênh dữ liệu bằng cách mở một cổng kết nối tới User-DTP. Server sử dụng cổng đặc biệt được dành riêng cho kết nối dữ liệu là cổng số 20. Trên máy Client, cổng mặc định được sử dụng chính là cổng được sử dụng để kết nối điều khiển, nhưng Server sẽ thường chọn mỗi cổng khác nhau cho mỗi chuyển giao.

- Phương thức tạo kết nối bị động (Passive Data Connections)

Server sẽ chấp nhận 1 yêu cầu kết nối dữ liệu được khởi tạo từ Client.

Server sẽ trả lời lại phía Client với địa chỉ IP cũng như địa chỉ cổng mà Server sẽ sử dụng. Sau đó phía Server-DTP lắng nghe trên cổng này một kết nối TCP đến từ User-DTP.

Theo mặc định, phía Client sẽ sử dụng cùng cổng mà nó sử dụng cho Control Connection như trong trường hợp chủ động. Tuy nhiên, trong phương pháp này, Client cũng có thể chọn sử dụng một cổng khác cho Data Connection nếu cần thiết.

1.4. Cơ chế socket trong Java

1.4.1. Khái quát về Socket

Socket là giao diện lập trình ứng dụng mạng được dùng để truyền và nhận dữ liệu trên internet. Giữa hai chương trình chạy trên mạng cần có một liên kết giao tiếp hai chiều, hay còn gọi là two-way communication để kết nối 2 process trò chuyện với nhau. Điểm cuối (endpoint) của liên kết này được gọi là socket. Các lớp Socket được ràng buộc với một cổng port (thể hiện là một con số cụ thể) để các tầng TCP (TCP Layer) có thể định danh ứng dụng mà dữ liệu sẽ được gửi tới

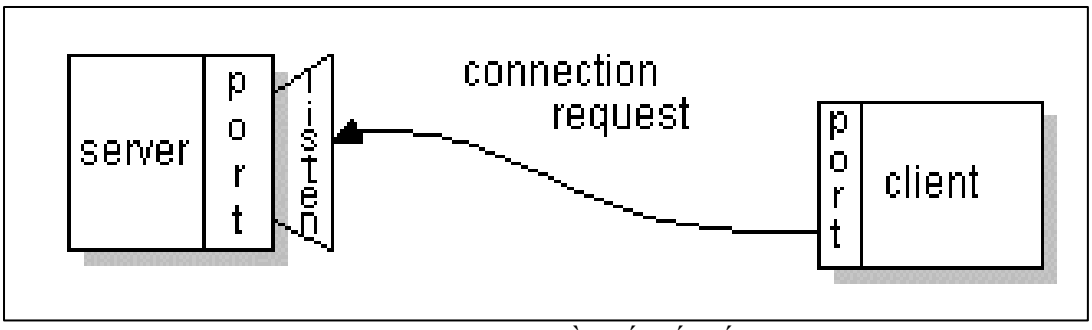
1.4.2. Cơ chế Socket

Một socket là một điểm cuối của thông tin hai chiều liên kết giữa hai chương trình đang chạy trên mạng. Những lớp socket được dùng để đại diện cho kết nối giữa một chương trình client và một chương trình server.

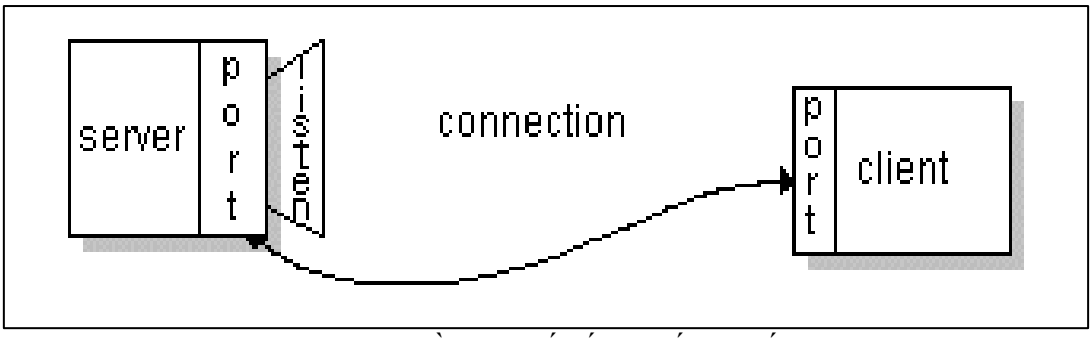
Trong Java gói Java.net cung cấp hai lớp Socket và ServerSocket để thực hiện kết nối giữa client và server. Thông thường thì server sẽ chạy trên một máy đặc biệt và có một socket giới hạn trong 1 Portnumber đặc biệt.

Ở phía client, client được biết hostname của máy mà server đang chạy và port number mà server đang lắng nghe. Để tạo một yêu cầu kết nối client sẽ thử hẹn gặp server ở trên máy của server thông qua port number. Client cũng cần xác định chính nó với server thông qua local port number. Nếu mọi thứ tốt đẹp thì server sẽ đồng ý kết nối. khi đồng ý

kết nối thì server sẽ tạo ra một socket mới để nói chuyện với client và cũng tạo ra một socket khác để tiếp tục lắng nghe

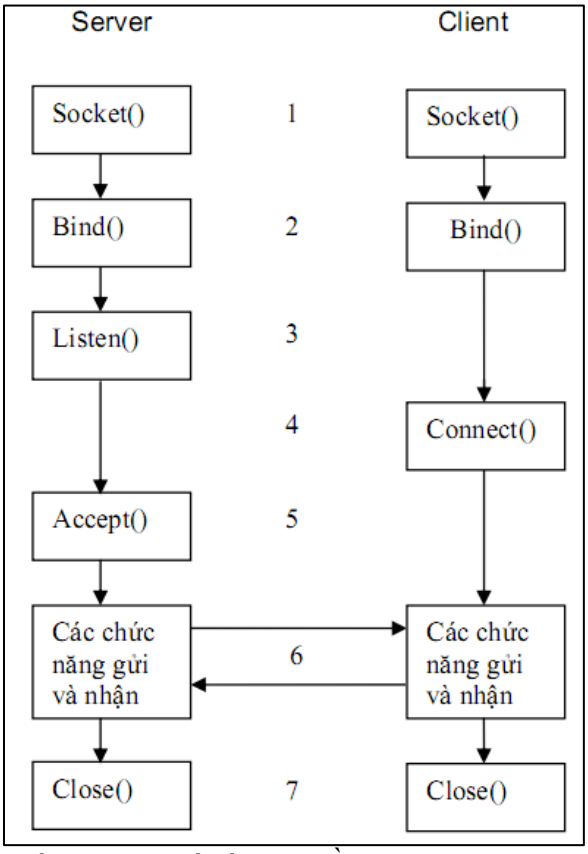


Hình 1.12: Client gửi yêu cầu kết nối đến Server



Hình 1.13: Server đồng ký kế nối và tiếp tục lắng nghe

1.4.3. Mô hình truyền tin Socket



Hình 1.14: Mô hình truyền tin trong Socket

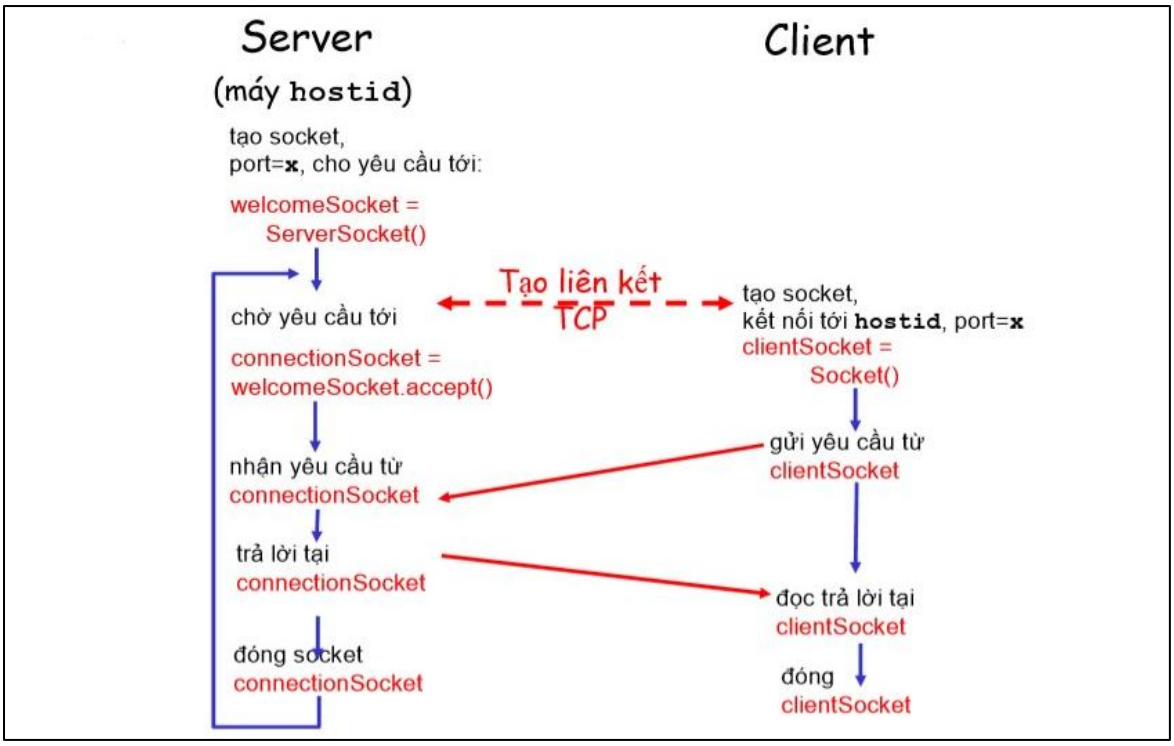
Một socket có thể thực hiện bảy thao tác cơ bản:

- Kết nối với một máy ở xa (ví dụ, chuẩn bị để gửi và nhận dữ liệu).
- Gửi dữ liệu.
- Nhận dữ liệu.
- Ngắt liên kết.
- Gán cổng.
- Nghe dữ liệu đến.
- Chấp nhận liên kết từ các máy ở xa trên cổng đã được gán

Lớp Socket của Java được sử dụng bởi cả client và server, có các phương thức tương ứng với bốn thao tác đầu tiên. Ba thao tác cuối chỉ cần cho server để chờ các client liên kết với chúng. Các thao tác này được cài đặt bởi lớp ServerSocket. Các socket cho client thường được sử dụng theo mô hình sau:

- Một socket mới được tạo ra bằng cách sử dụng hàm Socket().
- Socket cố gắng liên kết với một host ở xa.

Mỗi khi liên kết được thiết lập, các host ở xa nhận các luồng vào và luồng ra từ socket, và sử dụng các luồng này để gửi dữ liệu cho nhau. Kiểu liên kết này được gọi là song công (full-duplex)-các host có thể nhận và gửi dữ liệu đồng thời. Ý nghĩa của dữ liệu phụ thuộc vào giao thức.



Hình 1.15: Tương tác giữa Client - Server qua socket

1.5. Lập Trình Đa Luồng

1.5.1. Các khái niệm

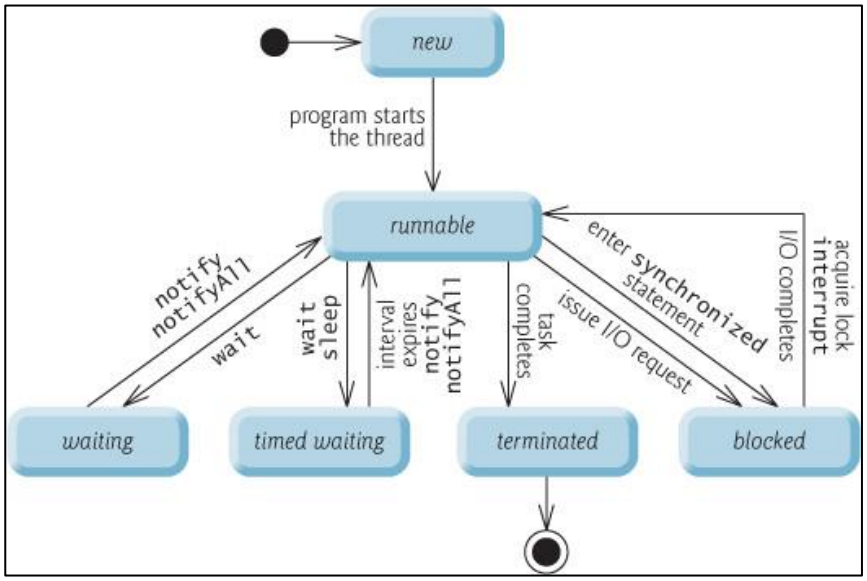
Thread (luồng) về cơ bản là một tiến trình con (sub-process). Một đơn vị xử lý nhỏ nhất của máy tính có thể thực hiện một công việc riêng biệt. Trong Java, các luồng được quản lý bởi máy ảo Java (JVM).

Multi-thread (đa luồng) là một tiến trình thực hiện nhiều luồng đồng thời. Một ứng dụng Java ngoài luồng chính có thể có các luồng khác thực thi đồng thời làm ứng dụng chạy nhanh và hiệu quả hơn.

1.5.2. Vòng đời của luồng trong Java

Vòng đời của thread trong java được kiểm soát bởi JVM. Java định nghĩa các trạng thái của luồng trong các thuộc tính static của lớp **Thread.State**:

- **NEW** : Đây là trạng thái khi luồng vừa được khởi tạo bằng phương thức khởi tạo của lớp Thread nhưng chưa được start(). Ở trạng thái này, luồng được tạo ra nhưng chưa được cấp phát tài nguyên và cũng chưa chạy. Nếu luồng đang ở trạng thái này mà ta gọi các phương thức ép buộc stop,resume,suspend … sẽ là nguyên nhân sảy ra ngoại lệ IllegalThreadStateException .
- **RUNNABLE** : Sau khi gọi phương thức start() thì luồng test đã được cấp phát tài nguyên và các lịch điều phối CPU cho luồng test cũng bắt đầu có hiệu lực. Ở đây, chúng ta dùng trạng thái là Runnable chứ không phải Running, vì luồng không thực sự luôn chạy mà tùy vào hệ thống mà có sự điều phối CPU khác nhau.
- **WAITING** : Thread chờ không giới hạn cho đến khi một luồng khác đánh thức nó.
- **TIMED_WAITING** : Thread chờ trong một thời gian nhất định, hoặc là có một luồng khác đánh thức nó.
- **BLOCKED**: Đây là 1 dạng của trạng thái "Not Runnable", là trạng thái khi Thread vẫn còn sống, nhưng hiện tại không được chọn để chạy. Thread chờ một monitor để unlock một đối tượng mà nó cần.
- **TERMINATED** : Một thread ở trong trạng thái terminated hoặc dead khi phương thức run() của nó bị thoát.



Hình 1.16: Vòng đời của một Thread trong Java.

1.5.3. Cách tạo luồng trong Java

1.5.3.1. Tạo luồng bằng cách extend từ lớp Thread

Để tạo luồng bằng cách tạo lớp kế thừa từ lớp Thread:

1. Khai báo 1 lớp mới kế thừa từ lớp Thread.

2. Override lại phương thức run ở lớp này, những gì trong phương thức run sẽ được thực thi khi luồng bắt đầu chạy. Sau khi luồng chạy xong tất cả các câu lệnh trong phương thức run thì luồng cũng tự hủy.

3. Tạo 1 thể hiện (hay 1 đối tượng) của lớp ta vừa khai báo.

4. Sau đó gọi phương thức start() của đối tượng này để bắt đầu thực thi luồng.

1.5.3.2. Tạo luồng bằng cách implement từ Interface Runnable

Để tạo luồng bằng cách hiện thực từ Interface Runnable:

1. Khai báo 1 lớp mới implements từ Interface Runnable.

2. Hiện thực phương thức run() ở lớp này, những gì trong phương thức run() sẽ được thực thi khi luồng bắt đầu chạy. Sau khi luồng chạy xong tất cả các câu lệnh trong phương thức run thì luồng cũng tự hủy.

3. Tạo 1 thể hiện (hay 1 đối tượng) của lớp ta vừa khai báo.

4. Tạo 1 thể hiện của lớp Thread bằng phương thức khởi tạo.

5. Gọi phương thức start() của đối tượng t1.

CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

2.1. Phân tích yêu cầu

2.1.1. Phạm vi

Hệ thống giúp thực hiện quản lý dữ liệu và thực hiện gửi dữ liệu một cách nhanh chóng và tiện lợi. Ứng dụng thuộc loại Desktop App, chạy trên nền tảng Java, môi trường Windows và Linux, sử dụng ngôn ngữ tiếng Việt. Hiện tại ứng dụng chỉ có một đối tượng người dùng phía Client.

2.1.2. Yêu cầu chức năng

Bảng 2.1: Yêu cầu chức năng phía Client và Server

Server	- Đặt trên host free. - Server luôn rà soát để nhận và chấp nhận kết nối đến Client và điều khiển các lệnh, gửi đến Client. - Server điều khiển được nhiều Client kết nối đến Server.
Client	- Thực hiện các chức năng: Kết nối đến Server, Upload filename, Download filename, Delete filename; DIR. - Client cần thống kê tốc độ upload/download dữ liệu để đánh giá. - Thông tin thống kê lưu ra file text. - Thiết kế giao diện đồ họa.

2.1.2.1. Mô tả chức năng

- Kết nối: Chức năng kết nối giúp Client kết nối đến Server thông qua socket. Sau khi kết nối và xác thực, người dùng có thể thực hiện các chức năng khác. Để thực hiện kết nối, Client phải có địa chỉ IP của Server.
- Upload filename: Chức năng giúp Client gửi dữ liệu lên Server. Client có thể gửi nhiều files, gửi folder. Để thực hiện gửi, Client gửi yêu cầu lên Server, Server và Client cùng thiết lập một luồng kết nối mới để gửi dữ liệu. Sau khi hoàn tất, luồng gửi sẽ đóng lại.
- Download filename: Chức năng giúp Client tải dữ liệu từ Server. Client có thể tải nhiều files, folders. Để thực hiện tải, Client gửi yêu cầu lên Server, Server phản hồi cổng khởi tạo luồng gửi dữ liệu, Client khởi tạo socket và kết nối đến Server. Sau khi hoàn thành, luồng tải sẽ đóng lại.
- Delete filename: Chức năng giúp Client xóa dữ liệu đã tải lên Server. Để thực hiện, Client gửi yêu cầu và đường dẫn lên Server. Server sẽ thực hiện xóa dữ liệu và phản hồi lại phía Client.
- Thống kê thông số tải: Chức năng giúp Client thống kê một vài thông số khi tải lên/ tải xuống. Thông tin thống kê được lưu trữ ở file text. Mục đích của việc thống kê là giúp đánh giá được tốc độ, thời gian,… gửi file. Từ đó đưa ra những giải pháp nhằm tối ưu hiệu suất quá trình gửi file.

2.1.3. Yêu cầu phi chức năng

2.1.3.1. Thiết kế và phát triển

- Backend: Java.
- Ngôn ngữ: tiếng Việt.

- Ứng dụng gồm 2 phía: Client và Server.
- Kết nối cơ sở dữ liệu MySQL thông qua XAMPP.
- Xử lý đa truy cập.
- Mã nguồn phải clean code để dễ bảo trì và phát triển.

2.1.3.2. Giao diện người dùng

Thiết kế dạng Dashboard, có bảng hiển thị danh mục file, folder ở phía Server. Giao diện có các nút bấm chức năng được kích hoạt khi nhấn vào. Yêu cầu giao diện rõ ràng, dễ sử dụng với người mới tiếp cận.

2.1.3.3. Về tính sẵn sàng của hệ thống

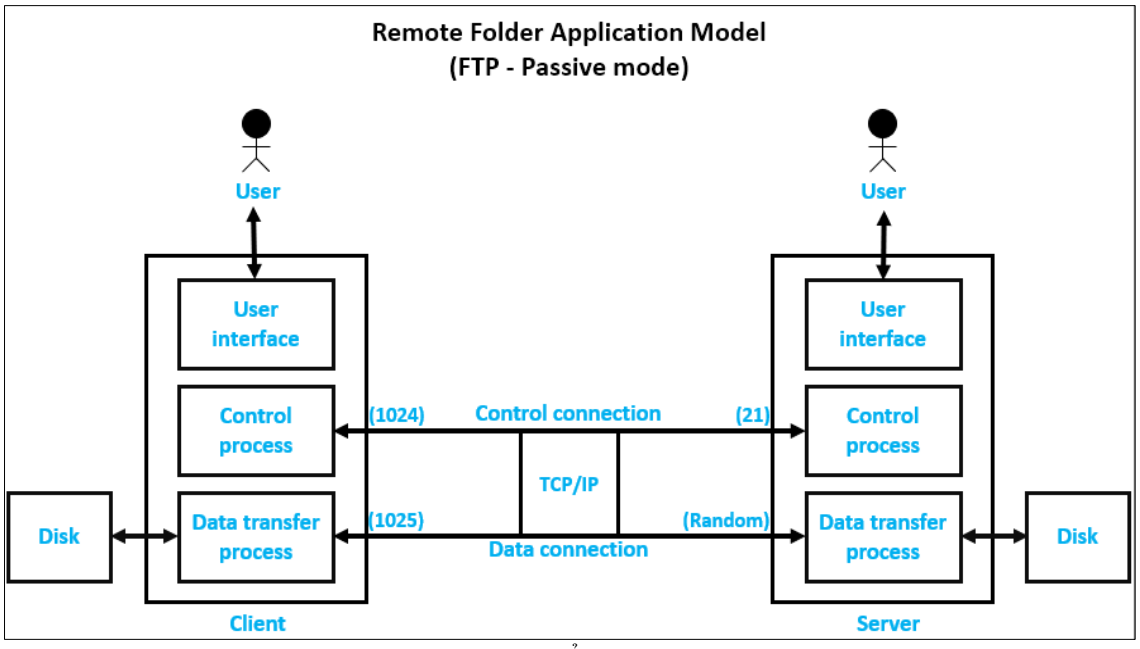
- Đảm bảo Server luôn được bật để các Client kết nối đến.
- Thời gian hệ thống phục vụ: 24/7.

2.1.3.4. Yêu cầu về độ tin cậy của hệ thống

- Hệ thống phải xử lý tình huống và đưa ra kết quả đúng như quy trình nghiệp vụ.

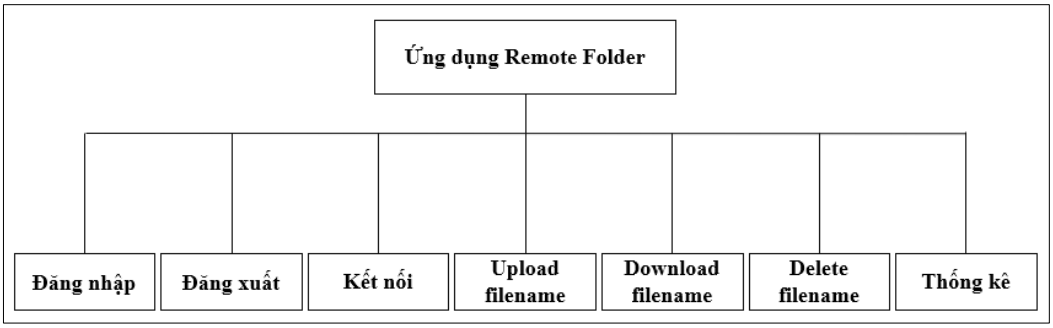
2.2. Thiết kế hệ thống

2.2.1. Mô hình tổng quát ứng dụng



Hình 2.1: Mô hình tổng quát ứng dụng

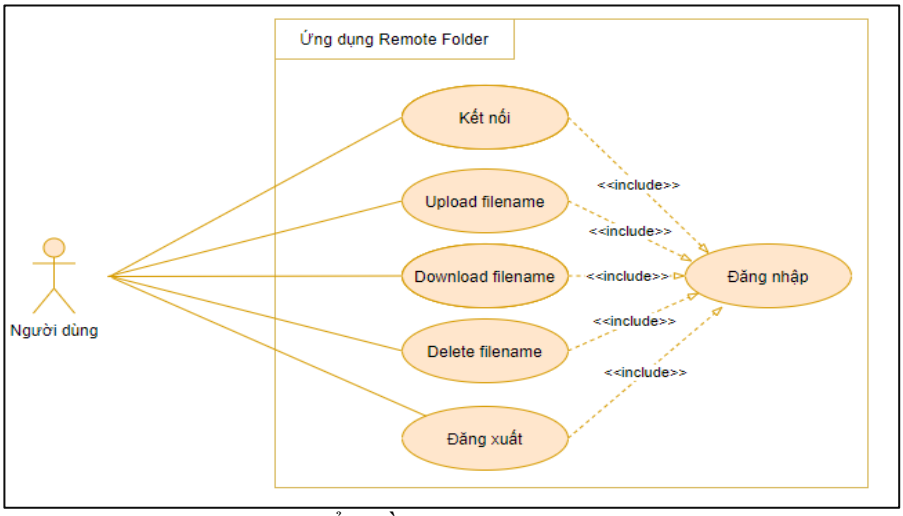
2.2.2. Sơ đồ cây phân rã chức năng



Hình 2.2 Sơ đồ phân rã chức năng

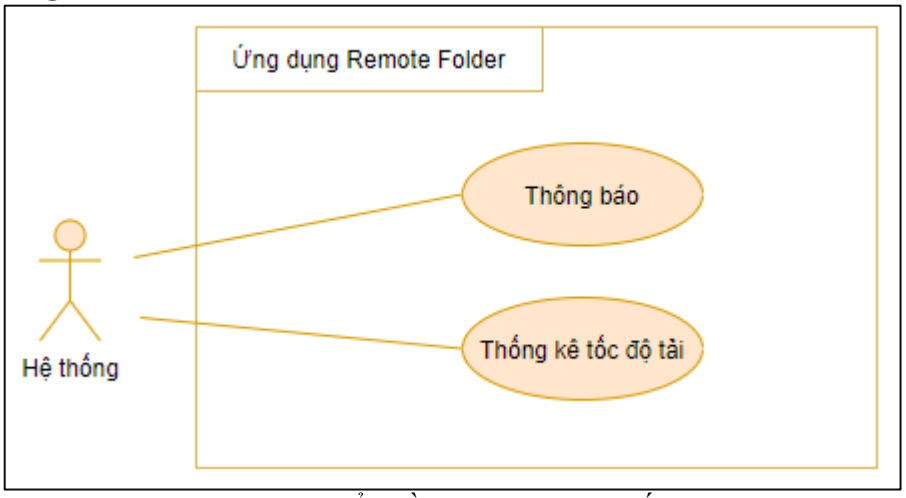
2.2.3. Biểu đồ ca sử dụng

2.2.3.1. Người dùng



Hình 2.3: Biểu đồ ca sử dụng người dùng

2.2.3.2. Hệ thống



Hình 2.4: Biểu đồ ca sử dụng hệ thống

2.2.4. Đặc tả ca sử dụng

2.2.4.1. Ca sử dụng đăng nhập

Bảng 2.2: Ca sử dụng đăng nhập

Mã ca sử dụng	UC-01		
Tác nhân	Người dùng		
Mô tả	Cho phép người dùng đăng nhập vào hệ thống		
Điều kiện trước	Chưa ở trạng thái đăng nhập ứng dụng		
Điều kiện sau	Người dùng đăng nhập thành công		
Kịch bản		**Hành động của tác nhân**	**Hành động của hệ thống**
	1	Người dùng khởi động ứng dụng.	

	2		Hiển thị trang đăng nhập.
	3	Nhập vào địa chỉ IP của Server, tên đăng nhập và mật khẩu.	
	4		Lấy thông tin, tiến hành kiểm tra và gởi yêu cầu về server, chuyển đến trang thông tin nếu thông tin đăng nhập là chính xác.

2.2.4.2. Ca sử dụng tải filename lên server

Bảng 2.3: Ca sử dụng tải filename lên server.

Mã ca sử dụng	UC-02		
Tác nhân	Người dùng		
Mô tả	Cho phép người dùng tải dữ liệu lên server.		
Điều kiện trước	Đã thực hiện đăng nhập hệ thống.		
Điều kiện sau	Tải file thành công.		
Kịch bản		**Hành động của tác nhân**	**Hành động của hệ thống**
	1	Người dùng nhấn vào nút Upload.	
	2		Hiển thị trang upload.
	3	Người dùng chọn Upload Files/ Upload Folder để hiển thị hộp thoại chọn dữ liệu.	
	4		Hiển thị hộp thoại chọn dữ liệu.
	kjhh5	Người dùng chọn dữ liệu.	
	6		Hiển thị dữ liệu được chọn vào bảng.
	7	Nhấn nút tải lên	
	8		Tải dữ liệu lên server.

2.2.4.3. Ca sử dụng tải xuống file name

Bảng 2.4: Ca sử dụng tải xuống file name

Mã ca sử dụng	UC-03
Tác nhân	Người dùng
Mô tả	Cho phép người dùng tải dữ liệu về máy.
Điều kiện trước	Đã thực hiện đăng nhập hệ thống.

Điều kiện sau	Tải dữ liệu thành công.		
Kịch bản		**Hành động của tác nhân**	**Hành động của hệ thống**
	1	Người dùng chọn dữ liệu cần tải.	
	2	Nhấn nút Download.	
	3		Tải xuống và thông báo thành công khi tải xuống thành công.

2.2.4.4. Ca sử dụng xóa dữ liệu

Bảng 2.5: Ca sử dụng xóa dữ liệu

Mã ca sử dụng	UC-04		
Tác nhân	Người dùng		
Mô tả	Cho phép xóa dữ liệu đã tải lên server.		
Điều kiện trước	Đã thực hiện đăng nhập hệ thống.		
Điều kiện sau	Xóa dữ liệu thành công.		
Kịch bản		**Hành động của tác nhân**	**Hành động của hệ thống**
	1	Người dùng chọn dữ liệu cần xóa.	
	2	Nhấn nút Delete.	
	3		Hệ thống hiện thị hộp thoại xác nhận.
	4	Người dùng xác nhận.	
	5		Hệ thống nhận xác nhận từ người dùng và thực hiện thao tác.

2.2.4.5. Ca sử dụng đăng xuất

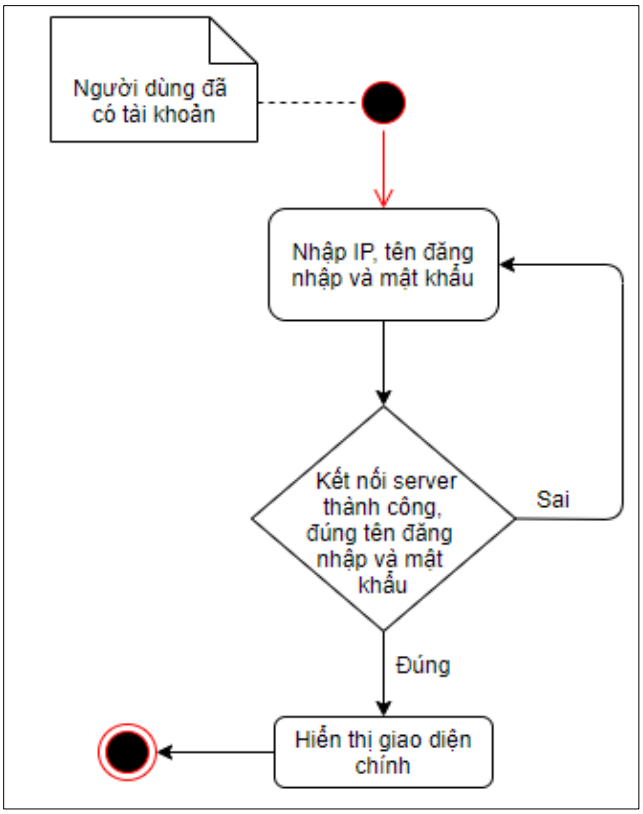
Bảng 2.6: Ca sử dụng đăng xuất

Mã ca sử dụng	UC-05		
Tác nhân	Người dùng		
Mô tả	Cho phép đăng xuất khỏi hệ thống.		
Điều kiện trước	Đã thực hiện đăng nhập hệ thống.		
Điều kiện sau	Đăng xuất hệ thống thành công.		
Kịch bản		**Hành động của tác nhân**	**Hành động của hệ thống**

	1	Người dùng nhấn chọn nút Log out.	
	2		Hiển thị hộp thoại xác nhận.
	3	Người dùng xác nhận.	
	4		Hệ thống nhận xác nhận từ người dùng và thực hiện thao tác.

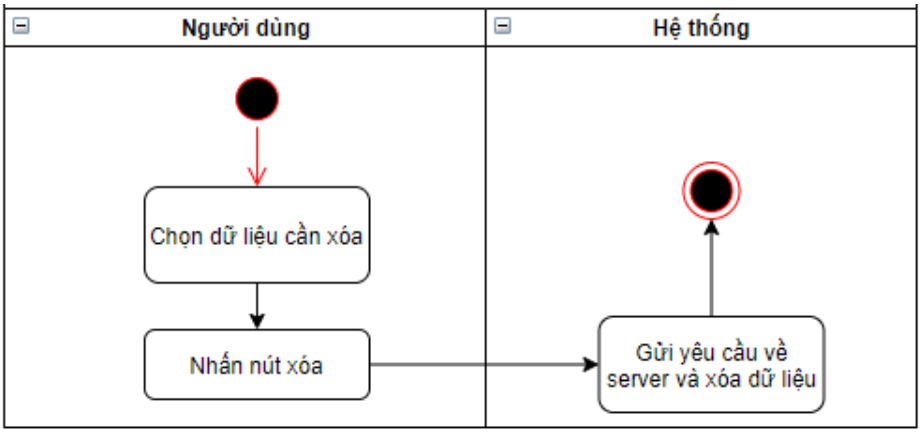
2.2.5. Biểu đồ hoạt động

2.2.5.1. Biểu đồ hoạt động chức năng đăng nhập



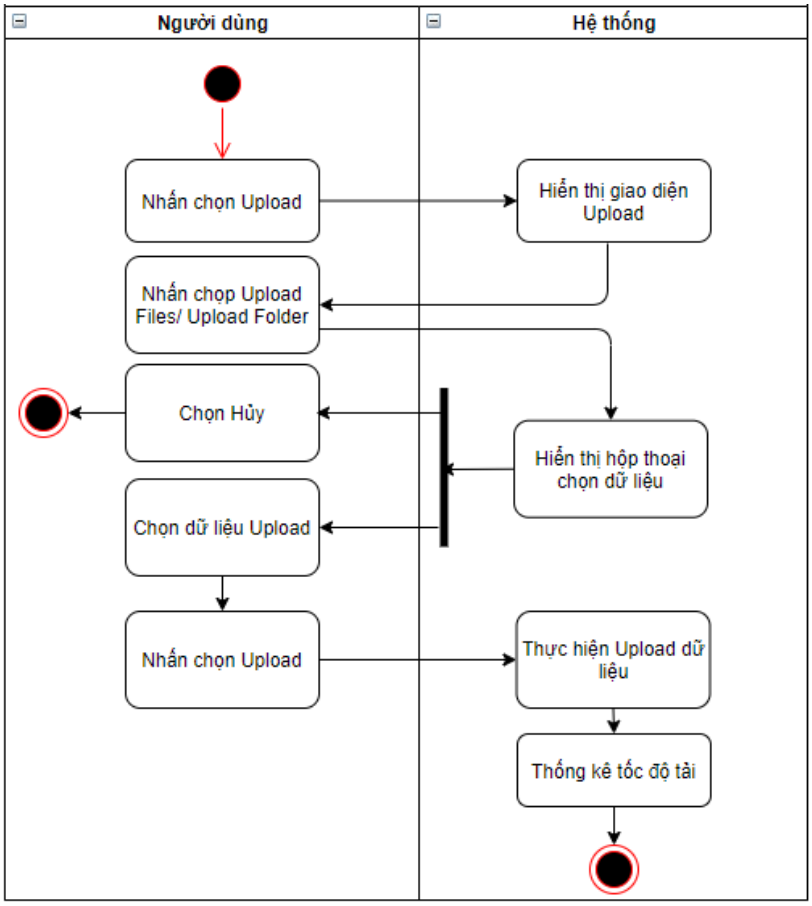
Hình 2.5: Biểu đồ hoạt động chức năng đăng nhập

2.2.5.2. Biểu đồ hoạt động chức năng xóa



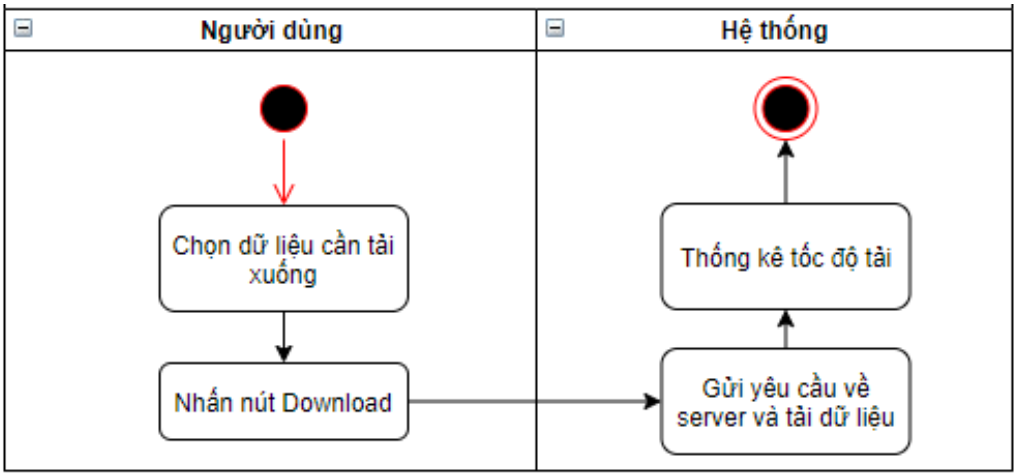
Hình 2.6: Biểu đồ hoạt động chức năng xóa

2.2.5.3. Biểu đồ hoạt động chức năng tải lên



Hình 2.7 Biểu đồ hoạt động chức năng tải lên

2.2.5.4. Biểu đồ hoạt động chức năng tải xuống

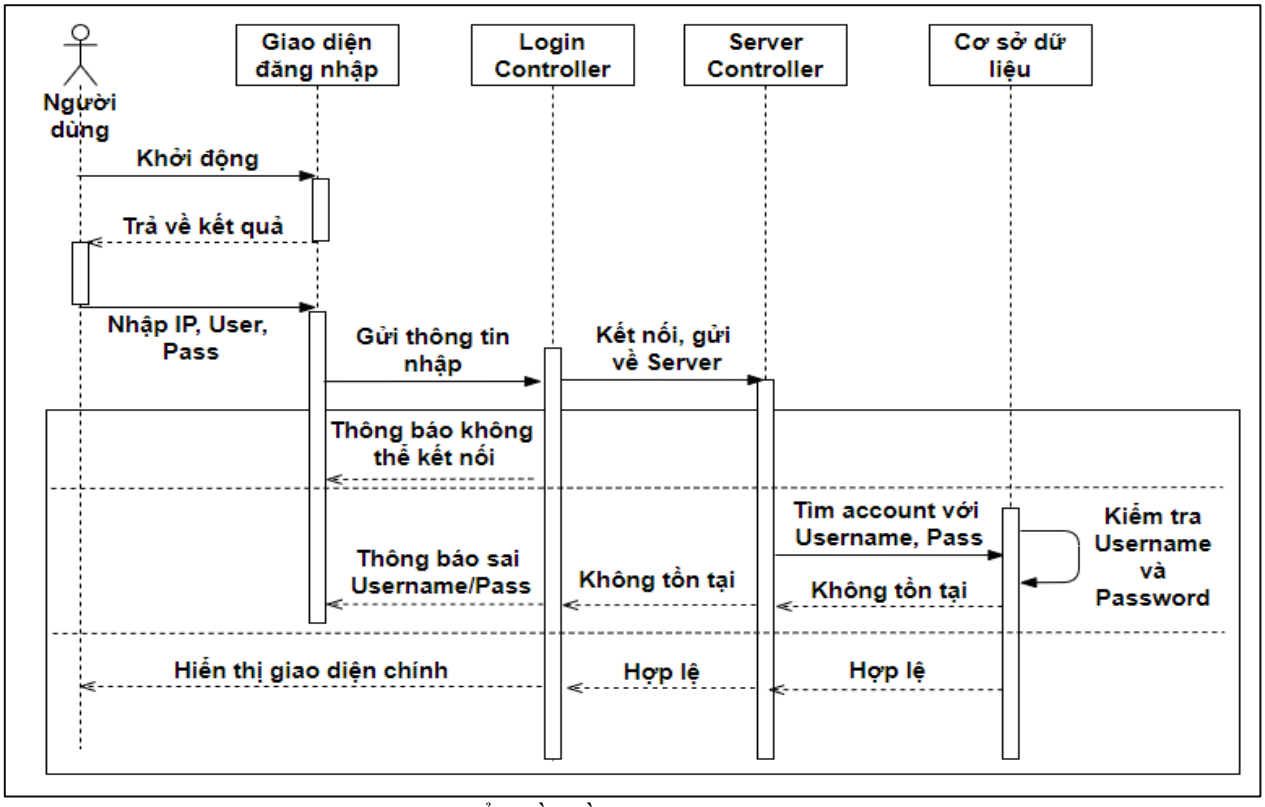


Hình 2.8: Biểu đồ hoạt động chức năng tải xuống

2.2.6. Biểu đồ tuần tự

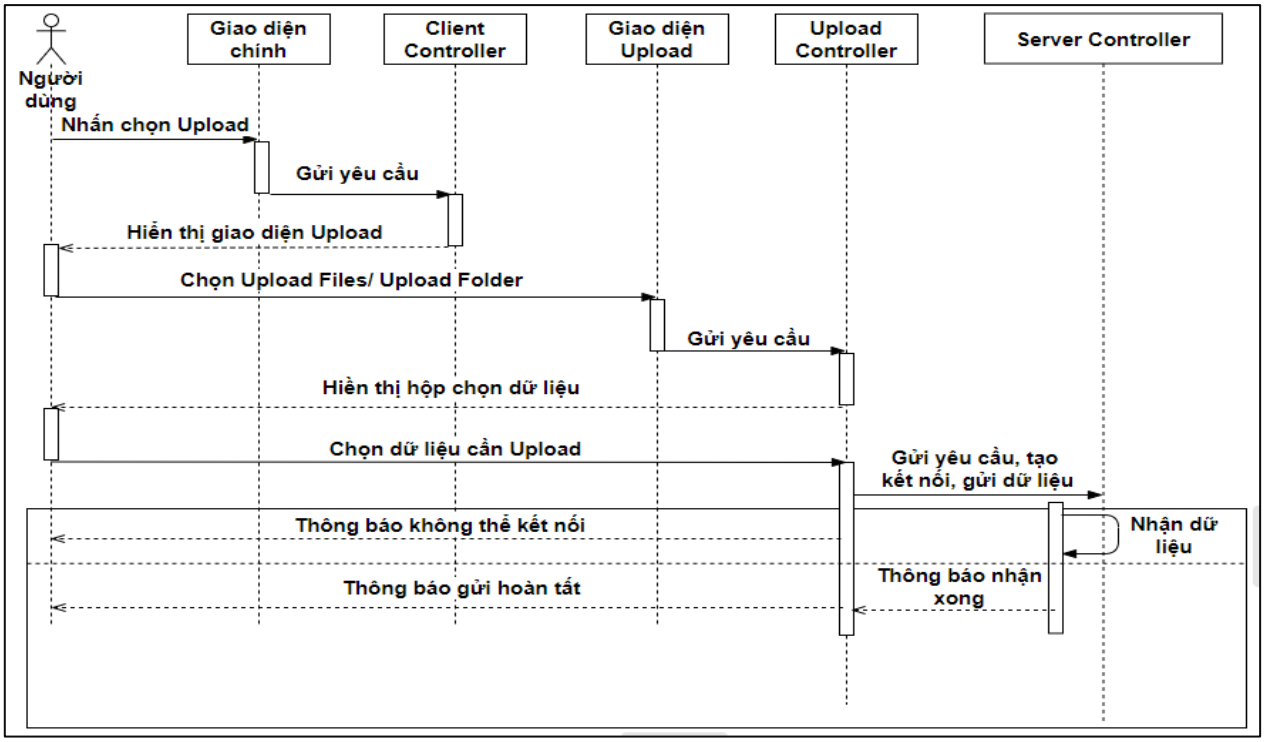
Một số biểu đồ tuần tự sau sẽ cho thấy rõ hơn sự tương tác cơ bản giữa các thành phần hệ thống ở một số chức năng quan trọng.

2.2.6.1. Biểu đồ tuần tự chức năng đăng nhập



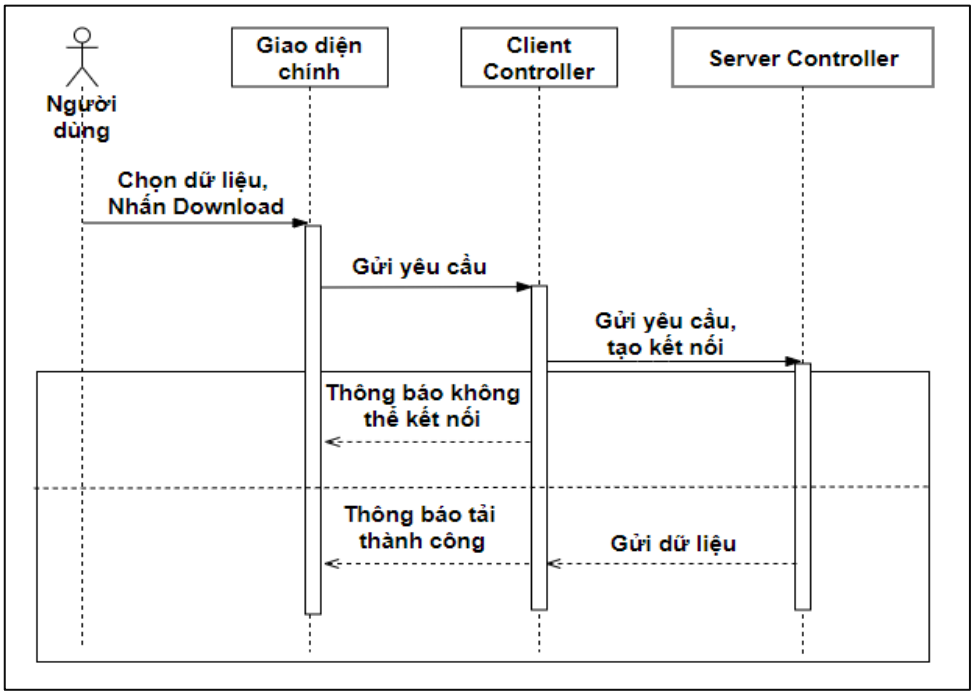
Hình 2.9 Biểu đồ tuần tự chức năng đăng nhập

2.2.6.2. Biểu đồ tuần tự chức năng tải lên



Hình 2.10: Biểu đồ tuần tự chức năng tải lên

2.2.6.3. Biểu đồ tuần tự chức năng tải xuống



Hình 2.11: Biểu đồ tuần tự chức năng tải xuống

2.2.7. Thiết kế cơ sở dữ liệu

2.2.7.1. Thực thể Account

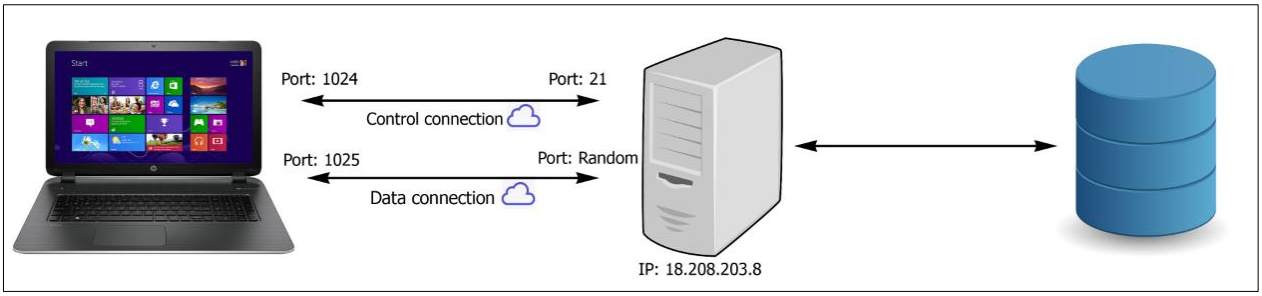
Trong đồ án này, chúng em chưa phát triển cơ sở dữ liệu nên cơ sở dữ liệu hiện tại chỉ có một thực thể Account, mục đích là để lưu trữ và quản lý tài khoản người dùng.

Bảng 2.7: Thực thể Account

Tên cột	Giải thích	Kiểu dữ liệu	Ghi chú
id	Mã tài khoản	int	Khóa chính, tự tăng
username	Tên đăng nhập	TEXT	Duy nhất
password	Mật khẩu	TEXT	
name	Tên chủ tài khoản	TEXT	
idFolder	Mã tham chiếu đến folder lưu trữ dữ liệu người dùng.	int	

CHƯƠNG 3: TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ

3.1. Triển khai



Hình 3.1: Mô hình triển khai

3.1.1. Phía Server

Ở phía Server, nhóm đã triển khai các hoạt động sau:

- Tìm kiếm dịch vụ về Cloud Computing.
- Xây dựng máy ảo hệ điều hành Window Server 2012.
- Cấu hình port cho máy ảo.
- Cài đặt máy ảo Java để có thể chạy file .JAR và XAMPP để sử dụng cơ sở dữ liệu MySQL.
- Tiến hành kiểm tra và chạy thử nghiệm việc truyền file có hoạt động tốt không.

3.1.2. Phía Client

Ở phía Client, nhóm đã triển khai các hoạt động sau:

- Cài đặt máy ảo Java để có thể chạy file .JAR.
- Cấu hình port cho máy khách.
- Xây dựng và kiểm tra các tính năng chạy ở phía máy khách.

3.1.3. Giao diện

Để xây dựng giao diện hiện đại, thân thiện người dùng, nhóm đã sử dụng công cụ JavaFX.

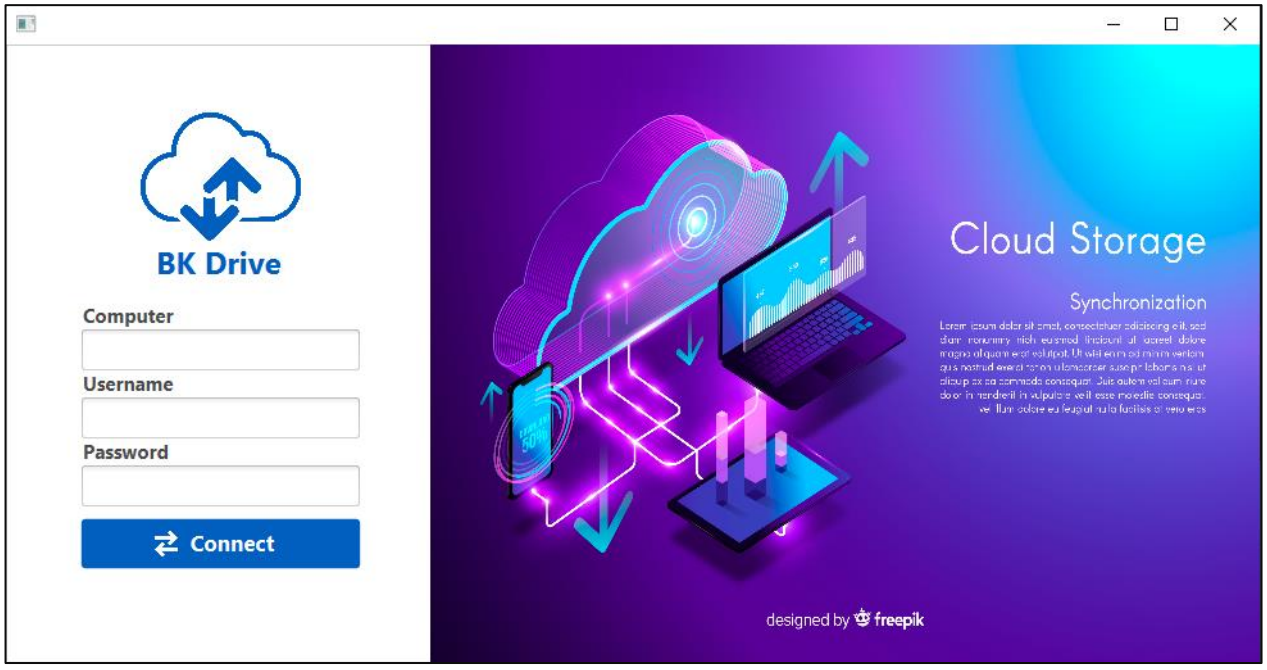
JavaFX là một bộ đồ họa và các gói media cho phép các nhà phát triển thiết kế, tạo ra, kiểm tra, gỡ lỗi và triển khai các ứng dụng có hoạt động liên tục trên nhiều nền tảng khác nhau.

Thư viện JavaFX được viết như là một Java API, mã ứng dụng JavaFX có thể tham chiếu các API từ bất kỳ thư viện Java. Các ứng dụng JavaFX có thể sử dụng các thư viện Java API để truy cập hệ thống và kết nối với các ứng dụng trung gian dựa trên máy chủ.

Giao diện của các ứng dụng JavaFX có thể được tùy chỉnh với mã CSS. Các nhà phát triển có thể tách riêng giao diện người dùng (UI) và login back-end, sau đó có thể phát triển khía cạnh trình bày của giao diện trong ngôn ngữ kịch bản FXML và sử dụng mã Java cho phần logic.

3.2. Kết quả

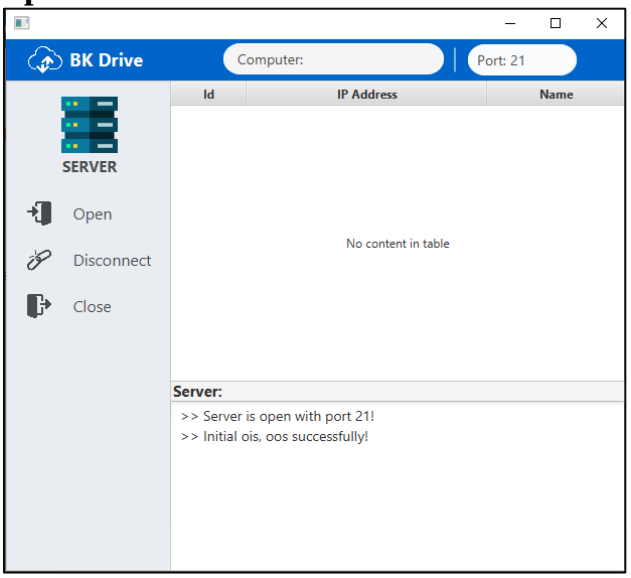
3.2.1. Giao diện đăng nhập



Hình 3.1: Giao diện đăng nhập

Giao diện trên là giao diện đăng nhập khi người dùng muốn đăng nhập vào hệ thống. Người dùng bắt buộc nhập đầy đủ IP Server, tên đăng nhập và mật khẩu. Nếu không kết nối được đến Server thì hệ thống hiển thị thông báo. Nếu sai tên đăng nhập hoặc mật khẩu, hệ thống sẽ thông báo. Trong trường hợp đăng nhập thành công, hệ thống hiển thị giao diện chính.

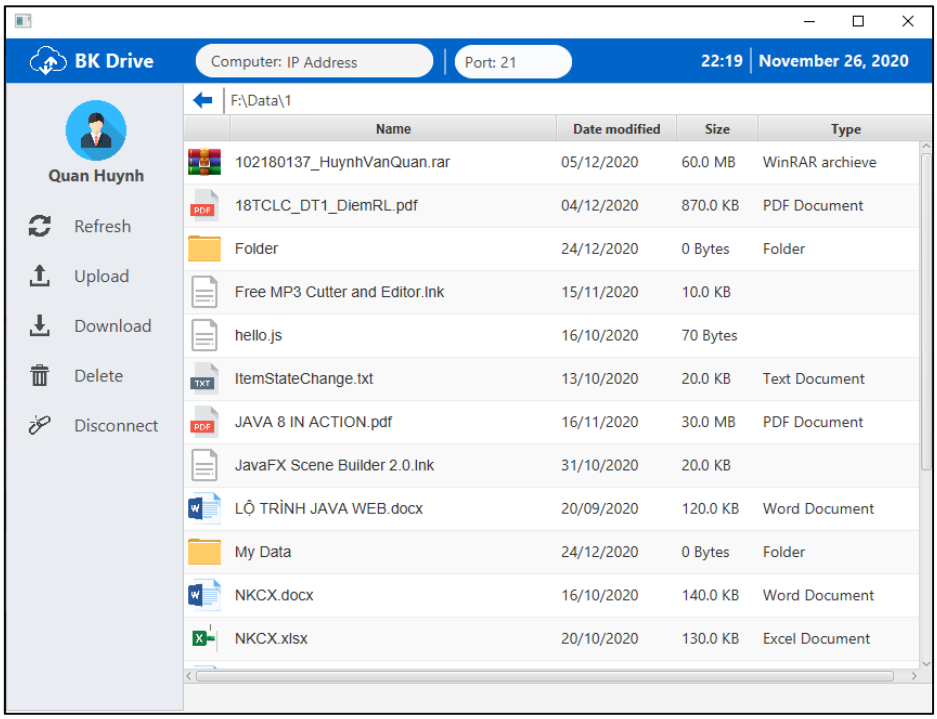
3.2.2. Giao diện chính phía Server



Hình 3.2: Giao diện chính phía Server

Giao diện trên là giao diện chính phía Server. Giao diện có chức năng mở kết nối, đóng kết nối và thống kê danh sách những Client đang kết nối đến Server.

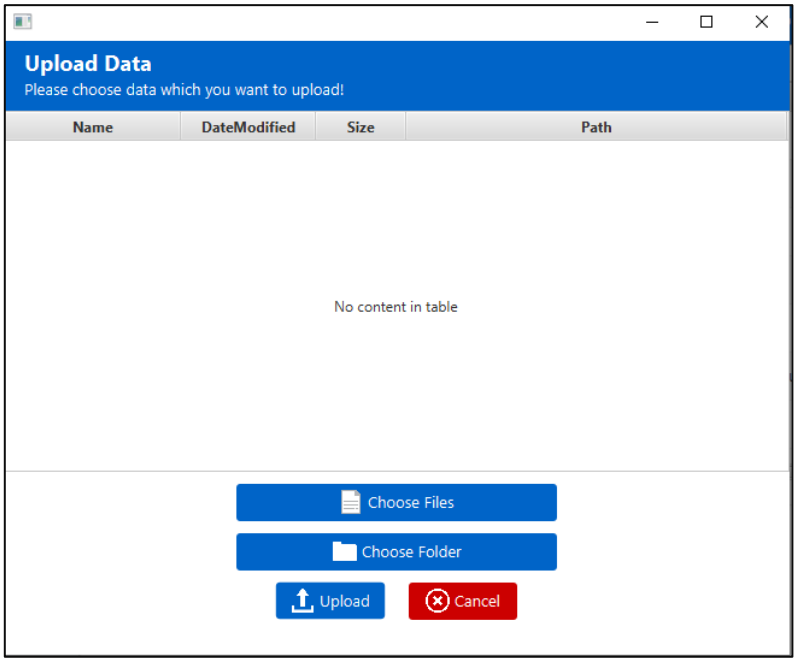
3.2.3. Giao diện chính phía Client



Hình 3.3: Giao diện chính phía Client

Giao diện trên là giao diện chính phía Client. Giao diện gồm các nút bấm chức năng, đường dẫn, một bảng thống kê file và folder có trong folder lưu trữ ở phía server của chủ tài khoản.

3.2.4. Giao diện Upload dữ liệu



Hình 3.4: Giao diện Upload dữ liệu

Giao diện trên là giao diện tải dữ liệu lên Server. Người dùng có thể thực hiện gửi nhiều files hoặc 1 folder tại một thời điểm lên Server. Phần bảng sẽ thống kê thông tin files hoặc folder mà người dùng chọn. Khi nhấn nút Upload, dữ liệu sẽ được tải lên Server và phần thông tin sẽ bị xóa đi khi nhấn nút. Nếu người dùng không muốn thao tác thì thoát khỏi giao diện hiện tại với nút Cancel.

3.3. Đánh giá kết quả

3.3.1. Chức năng

Nhóm đã xây dựng được đầy đủ các chức năng mà đồ án đề ra.

Chức năng kết nối: Xây dựng được chức năng kết nối theo như yêu cầu, có thể thực hiện kết nối nhanh. Hệ thống phản hồi đầy đủ khi gặp sự cố về kết nối hoặc kết nối thành công.

Chức năng tải lên: Xây dựng được chức năng tải lên theo như yêu cầu. Chức năng có thể thực hiện tải nhiều file tại một thời điểm hoặc một folder tại một thời điểm; có thể tải file với dung lượng lớn nhiều GB.

Chức năng tải xuống: Xây dựng được chức năng tải xuống theo như yêu cầu. Chức năng có thể thực hiện tải một file tại một thời điểm hoặc một folder tại một thời điểm; có thể tải với dung lượng lớn nhiều GB.

Chức năng thống kê: Xây dựng được chức năng thống kê theo như yêu cầu. Chức năng có thể thống kê các thông tin như tên, kích thước dữ liệu,… khi thực hiện tải lên hoặc tải xuống dữ liệu. Thông tin thống kê được lưu ở file text.

Những điều chưa làm được: Chưa sử dụng semaphore để quản lý các luồng theo như yêu cầu.

3.3.2. Giao diện

Giao diện thân thiện, rõ ràng, dễ sử dụng, đáp ứng yêu cầu đề ra.

CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

4.1. Kết luận

Nền công nghiệp 4.0 đang diễn ra rất mạnh mẽ. Công nghệ thông tin đang phát triển ngày càng mạnh và đóng nhiều vai trò quan trọng trong đời sống. Đặc biệt, với sự bùng nổ thông tin như hiện nay thì việc sử dụng những ứng dụng như Remote Folder để lưu trữ, trao đổi dữ liệu là hết sức cần thiết.

Do đó, chúng em đã cố gắng liên kết những kiến thức đã học, vận dụng kiến thức đa môn để cùng nhau xây dựng nên ứng dụng. Chúng em hy vọng ứng dụng sẽ góp phần giải quyết được phần nào những vấn đề đang đặt ra hiện nay.

4.2. Hạn chế

Do thời gian thực hiện đề tài tương đối hạn chế và chúng em cũng không thể tránh được những thiếu sót nhất định như chương trình có một số chức năng chưa hoàn thiện, giao diện cũng chưa thực sự được làm tốt nhất,…Trên đây là những mặt tồn tại lớn khi chúng em làm đồ án này. Và chắc chắn rằng sau này khi có đầy đủ kiến thức chuyên môn, chúng em nhất định sẽ tiến hành tối ưu và ngày một hoàn thiện sản phẩm hơn.

4.3. Hướng phát triển

Để chương trình hoạt động tối ưu hơn, chúng em cần phải sử thuật toán gửi file tốt hơn, thông tin cần được bảo mật cao hơn. Đồng thời, giải thuật cần phải được tối ưu để thao tác nhanh hơn. Bên cạnh đó, chúng em còn có ý tưởng phát triển thêm nhiều chức năng trong tương lai như đăng ký, quản lý đa truy cập tốt hơn, thực hiện chính xác hơn các chức năng, xử lý được các trường hợp ngoại lệ và phải triển giao diện thật thân thiện, hoàn chỉnh. Cuối cùng, chúng em luôn tìm kiếm sự tối ưu nhất có thể và hướng đến một chương trình thông minh tự động.

TÀI LIỆU THAM KHẢO

[1] - PGS. TS. Nguyễn Thanh Bình, Phân tích và thiết kế giải thuật, Nhà xuất bản giáo dục Việt Nam.

[2] - TS. Đặng Hoài Phương, Slide bài giảng môn Lập trình hướng đối tượng.

[3] - TS.Trương Ngọc Châu – Nguyễn Đức Thuần, Phương pháp giải bài tập cơ sở dữ liệu quan hệ, Nhà xuất bản khoa học và kỹ thuật.

[4] - Ths. Mai Văn Hà, Slide bài giảng môn Lập trình mạng.

[5] - T.S Nguyễn Thế Xuân Ly, Slide bài giảng môn Mạng máy tính.

[6] - Tanenbaum & Wetherall - Computer Networks Fifth Edition.

[7] - David Reily & Michael Reily - Java Network Programming & Distributed Computing.

[8] - Đại học FPT, Nhập môn cơ sở dữ liệu, Nhà xuất bản Bách khoa Hà Nội.

[9] - Website:

- www.qhonline.info.
- www. facebook.com.
- www.sinhvienit.net.
- www.vn-zoom.com.
- www.slideshare.net.