**TRƯỜNG ĐẠI HỌC PHENIKAA**

**KHOA CÔNG NGHỆ THÔNG TIN**



BÀI TẬP LỚN

Đề tài: Xây dựng chương trình chat sử dụng giao thức TCP

**Học phần : Lập trình hướng đối tượng-1-2-22(N03)**

**Giảng viên : TS. Trần Đăng Hoan**

**Sinh viên : Nguyễn Thị Tâm - msv : 21012892**

**Phạm Thành Trung - msv : 21012899**

**Nguyễn Nam Hải - msv : 21012056**

**Hà Nội, Tháng 04 Năm 2023**

**MỤC LỤC**

[Bảng phân chia công việc 3](#_Toc133249849)

[TỔNG QUAN VỀ ĐỀ TÀI 4](#_Toc133249850)

[Lý do thực hiện đề tài 4](#_Toc133249851)

[Phương pháp triển khai đề tài 4](#_Toc133249852)

[Kết cấu của đồ án 4](#_Toc133249853)

[CƠ SỞ LÝ THUYẾT 5](#_Toc133249854)

[Giao thức TCP 5](#_Toc133249855)

[Mô hình Client/Server 7](#_Toc133249856)

[Cơ chế Socket trong Java 9](#_Toc133249857)

[1.1. Khái quát về Socket 9](#_Toc133249858)

[1.2. Cơ chế Socket 9](#_Toc133249859)

[1.3. Mô hình truyền tin socket 10](#_Toc133249860)

[1.4. Một số hàm cơ bản trong socket 12](#_Toc133249861)

[THIẾT KẾ VÀ XÂY DỰNG HỆ THỐNG 14](#_Toc133249862)

[Phân tích yêu cầu 14](#_Toc133249863)

[Phân tích các chức năng 14](#_Toc133249864)

[Thiết kế kế chương trình 15](#_Toc133249865)

[1.1. Thiết kế giao diện 15](#_Toc133249866)

[1.2. Xây dựng các chức năng 16](#_Toc133249867)

[TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ 17](#_Toc133249868)

[Môi trường triển khai 17](#_Toc133249869)

[Kết quả các chức năng của chương trình 17](#_Toc133249870)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 20](#_Toc133249871)

[Những kết quả đạt được 20](#_Toc133249872)

[Những vấn đề tồn tại 20](#_Toc133249873)

[Hướng phát triển 20](#_Toc133249874)

[TÀI LIỆU THAM KHẢO 21](#_Toc133249875)

**Danh sách hình ảnh**

Hình 1 - Quá trình bắt tay 3 bước …………………………………………... 4

Hình 2 - Minh họa việc truyền dữ liệu bởi giao thức TCP ………………….. 5

Hình 3 - Bảng thông tin điều khiển gói tin TCP ...………………………….. 5

Hình 4 - Mô hình client/server ……………………………………………… 5

Hình 5 - Client gửi yêu cầu kết nối tới Server ……………………………… 9

Hình 6 - Server đồng ý kết nối và tiếp tục lắng nghe ……………………….. 9

Hình - Mô hình truyền tin socket ……………………………………….... 9

Hình 8 - Mô hình tương tác giữa client/server qua socket TCP ………..….. 10

Hình 9 - Thiết kế …………………………………………………………... 14

Hình 10 - Kết quả thực nghiệm ……………………………………………. 16

# Bảng phân chia công việc

|  |  |
| --- | --- |
| Thành viên | Công việc |
| Nguyễn Thị Tâm - 21012892 | - Viết báo cáo  - Phân chia nhiệm vụ  - Tìm hiểu về Socket  - Tìm hiểu và code file thư viện ChatMessageSocket |
| Phạm Thành Trung - 21012899 | - Tổng hợp code  - Thiết kế và xây dựng hệ thống  - Triển khai và đánh giá kết quả  - Tìm hiểu và code file Server |
| Nguyễn Nam Hải - 21012056 | - Thiết kế slide thuyết trình  - Tìm hiểu giao thức TCP  - Tìm hiểu mô hình Client-Server  - Tìm hiểu và code file Client |

# TỔNG QUAN VỀ ĐỀ TÀI

## Lý do thực hiện đề tài

Ngày nay, nhu cầu về công nghệ thông tin trong đời sống là đa dạng. Việc mở rộng các hệ thống truyền thông và ngày có nhiều máy vi tính kết nối vào mạng Internet. Với việc ứng dụng giao thức TCP làm cho hệ thống mạng ngày càng rộng hơn và phát triển vượt bật. Bên cạnh việc phát triển của thư điện tử bằng nhiều dịch vụ khác nhau (gmail, yahoo,MSM trong mạng điện thoại di động…), việc Chat trực tuyến và gửi flie trực tiếp trên Internet cũng là nhu cầu không thể thiếu.

Với thực tế như vậy, nhóm chúng em đã nghiên cứu và xây dựng mô hình Chat sử dụng giao thức TCP.

## Phương pháp triển khai đề tài

Ngôn ngữ lập trình: Java

Chương trình soạn thảo và build : NetBeans IDE

Sử dụng: Giao thức TCP

Sử dụng Socket trong Java

Sử dụng JavaSwing để thiết kế ứng dụng

## Kết cấu của đồ án

Gồm ba chương:

Chương 1: Cơ sở lý thuyết

Chương 2: Thiết kế và xây dựng hệ thống

Chương 3: Triển khai và đánh giá kết quả

# 

# CƠ SỞ LÝ THUYẾT

## Giao thức TCP

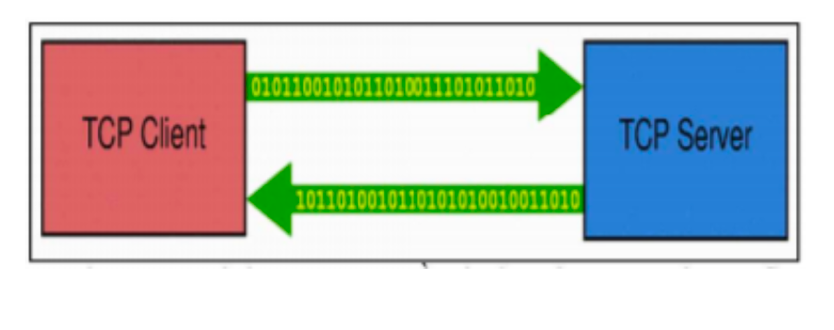
Giao thức TCP là giao thức truyền thông có kết nối (connection oriented) và tin cậy (reliable). Truyền thông có kết nối có nghĩa là trước khi truyền dữ liệu, giao thức TCP gửi và giao thức TCP nhận phải thiết lập kênh truyền trước khi gửi/nhận dữ liệu và, trong suốt quá trình trao đổi đữ liệu hai giao thức gửi/nhận phải duy trì kênh truyền đã được thiết lập.

Quá trình thiết lập kênh truyền được thực hiện cụ thể bởi một quá trình gọi là quá trình bắt tay ba bước (three-way handshake) như sau:

|  |  |
| --- | --- |
| 1. Trước hết giao thức TCP phía máy gửi phải yêu cầu thiết lập một kênh truyền bằng cách gửi một phân đoạn (segment) gọi là SYN đến giao thức TCP phía máy nhận.  2. Giao thức TCP phía máy nhận trả lại một phân đoạn (segment) để xác nhận đã nhận được thành công.  3. Giao thức TCP phía máy gửi tiếp tục gửi một phân đoạn ACK để xác nhận và sau đó là quá trình gửi dữ liệu. | Hình 1 : Quá trình bắt tay 3 bước |

Truyền thông tin cậy có nghĩa là nút gửi biết được gói tin đã được gửi đến đích hay không. Nếu không đến được, nút gửi phải gửi lại gói tin. Nếu gói tin đã được gửi thành công, nút gửi gửi gói tin tiếp theo. Cụ thể của quá trình này được thực hiện bằng cách mỗi gói tin được đánh một số thứ tự. Mỗi khi nhận được gói tin, nút nhận phải gửi một phản hồi xác nhận (acknowledgment) cho nút gửi để thông báo đã nhận được thành công gói tin.

Giao thức TCP sử dụng truyền thông dạng luồng dữ liệu hay dãy các bytes (byte-stream) liên tục trên kênh truyền, được mô tả như hình vẽ bên dưới:



Hình 2 : Minh họa việc truyền dữ liệu bởi giao thức TCP

Mỗi gói tin TCP (hay còn gọi là segment) gồm một số thông tin điều khiển ở phần đầu được mô tả ở bảng dưới đây:

|  |  |
| --- | --- |
| **Thông tin điều khiển** | **Mô tả** |
| Số cổng nguồn(Source Port) | Số hiệu cổng của tiến trình gửi |
| Số cổng đích (Destination Port) | Số hiệu cổng của tiến trình nhận |
| Số hiệu gói tin (Sequence Number) | Số thứ tự byte đầu tiên của gói tin TCP |
| Số hiệu xác nhận (Acknowledgment Number) | Số thứ tự của byte đầu tiên của gói tin TCP mà giao thức TCP gửi chờ nhận từ phía nhận |
| Kích thước bộ đệm (Window) | Kích thước hiện tại của bộ đệm TCP phía máy gửi dùng để chứa gói tin TCP nhận được. |
| Thông tin kiểm tra lỗi (TCP Checksum) | Dùng để kiểm tra tính toàn vẹn của phần đầu và dữ liệu của gói tin TCP. |

Để kết thúc kết nối, giao thức TCP cũng sử dụng cơ chế bắt tay 3 bước. Điều này đảm bảo cả hai giao thức TCP (gửi và nhận) đã kết thúc truyền dữ liệu và tất cả các dữ liệu đã được gửi và nhận thành công.

## Mô hình Client/Server

Mô hình client-server là một kiến trúc phần mềm trong đó các ứng dụng được chia thành hai phần chính: client và server. Client là phần yêu cầu dịch vụ, và server là phần cung cấp dịch vụ. Trong mô hình này, client gửi yêu cầu đến server, server xử lý yêu cầu và trả lại kết quả cho client.

Mô hình client-server cho phép nhiều người dùng truy cập vào các tài nguyên chung từ xa. Ví dụ như trong trường hợp các trang web, các trình duyệt web (client) yêu cầu các trang web từ máy chủ web (server) và server trả về các trang đó cho client để hiển thị.

Mô hình client-server cũng có thể được sử dụng để chia sẻ tài nguyên khác như máy in, dữ liệu và phần mềm. Sự phân tách giữa client và server cung cấp tính bảo mật và kiểm soát truy cập vào các tài nguyên chung.

Mô hình truyền tin client/server hướng tới việc cung cấp dịch vụ. Quá trình trao đổi dữ liệu bao gồm:

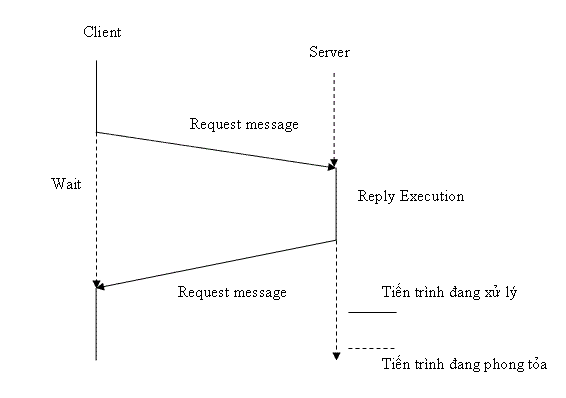
1. Truyền một yêu cầu từ tiến trình client tới tiến trình server

2. Yêu cầu được server xử lý

3. Truyền đáp ứng cho client

Mô hình truyền tin này liên quan đến việc truyền thông điệp và một dạng đồng bộ hóa cụ thể giữa client và server. Tiến trình server phải nhận thức được thông điệp được yêu cầu ở bước một ngay khi nó đến và hành động phát ra yêu cầu trong client phải được tạm dừng (bị phong tỏa) và buộc tiến trình client ở trạng thái chờ cho tớ khi nó nhận được đáp ứng do server gửi về ở bước ba.

Mô hình client/server thường được cài đặt dựa trên các thao tác cơ bản là gửi (send) và nhận (receive).



Hình 4 : Mô hình client/server

Quá trình giao tiếp 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 (blocked):

Trong 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 tiến trình cũng sẽ bị tạm ngừng cho tới khi có dữ liệu gửi tới.

Chế độ không bị phong tỏa (non-blocked).

Trong chế độ này, khi tiến trình client hay 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.

## Cơ chế Socket trong Java

### Khái quát về Socket

Socket là một khái niệm trong lập trình mạng. Nó cung cấp một giao diện để các ứng dụng có thể gửi và nhận dữ liệu thông qua mạng.

Các socket được sử dụng trong kết nối giữa hai thiết bị hoặc ứng dụng khác nhau trên mạng internet. Một socket bao gồm địa chỉ IP và số cổng, giúp xác định máy tính và dịch vụ mà một chương trình muốn kết nối tới. Tùy theo loại protocol (TCP hoặc UDP) mà socket được sử dụng, việc kết nối giữa client và server có thể là liên tục và duy trì trong một khoảng thời gian lâu hay chỉ truyền tải dữ liệu trong một khoảng thời gian rất ngắn.

### Cơ chế Socket

Socket được sử dụng để thiết lập kết nối mạng giữa hai máy tính. 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.

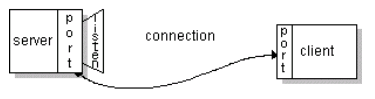
Thông thường thì server sẽ chạy trên một máy đặc biệt và có một socket giới hạn trong 1 Portnumber đặc biệt.

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.



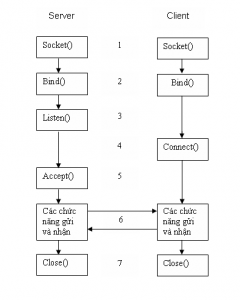
Hình 5 : Client gửi yêu cầu kết nối tới Server

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.

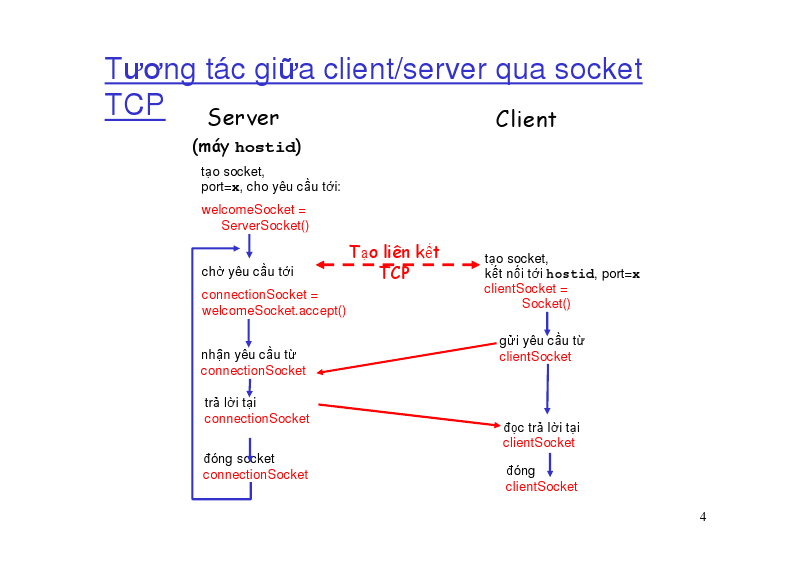


Hình 6 : Server đồng ý kết nối và tiếp tục lắng nghe.

### Mô hình truyền tin socket



Hình 7 : Mô hình truyền tin socket



Hình 8 : Mô hình tương tác giữa client/server qua socket TCP

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

- Kết nối với một máy ở xa (chuẩn bị để gửi và nhận dữ liệu)

- Gửi dữ liệu

- Nhận dữ liệu

- Ngắt liên kêt

- 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 cổng (full-duplex)-các host có thể nhận và gửi dữ liệu đồng thời. Ý nghĩa của dữ liệu phụ thuộc vào giao thức.

Khi việc truyền dữ liệu hoàn thành, một hoặc cả hai phía ngắt liên kết. Một số giao thức, như HTTP, đòi hỏi mỗi liên kết phải bị đóng sau mỗi khi yêu cầu được phục vụ. Các giao thức khác, chẳng hạn FTP, cho phép nhiều yêu cầu được xử lý trong một liên kết đơn.

### Một số hàm cơ bản trong socket

*Class mô tả về socket*

* Tạo một socket (cấu tử):

Socket(InetAddress address, int port)

Socket(String host, int port)

Socket(InetAddress address, int port, InetAddress, localAddr, int localPort)

Socket(String host, int port, InetAddress, localAddr, int localPort)

Socket()

* Lấy thông tin về một socket

InetAddress getInetAddress() : trả về địa chỉ mà socket kết nối đến

int getPort() : trả về số hiệu cổng mà socket kết nối đến.

InetAddress getLocalAddress() : trả về địa chỉ cục bộ.

int getLocalPort() : trả về số hiệu cổng cục bộ.

* Sử dụng Streams

public OutputStream getOutputStream() throws IOException: Trả về một output stream cho việc viết các byte đến socket này.

public InputStream getInputStream() throws IOException : Trả về một input stream cho việc đọc các byte từ socket này.

*Class mô tả ServerSocket*

* Tạo một ServerSocket

ServerSocket(intport) throws IOException

ServerSocket(intport, intbacklog) throws IOException ServerSocket(intport, intbacklog, InetAddressbindAddr) throws IOException

* Các phương thức trong ServerSocket

Socket accept() throws IOException: lắng nghe một kết nối đến socket này và có chấp nhận nó hay không

void close() throws IOException: Ðóng socket.

InetAddress getInetAddress() : trả về địa chỉ cục bộ của socket

int getLocalPort() : trả về port mà server đang lắng nghe

# THIẾT KẾ VÀ XÂY DỰNG HỆ THỐNG

## Phân tích yêu cầu

Yêu cầu bài toán: Xây dựng chương trình chat sử dụng giao thức TCP

*Xây dựng chương trình Server*

- Tạo một TCP Socket và gắn vào một cổng

- Xây dựng một chương trình Server cho phép Client kết nối tới Server

- Chờ và lắng nghe yêu cầu kết nối từ Client

- Chấp nhận kết nối và nhận Socket tương ứng

- Truyền nhận thông tin qua các luồng nhận/gửi dữ liệu của socket

- Đóng kết nối

*Xây dựng chương trình Client*

- Tạo một TCP Socket với địa chỉ IP và số cổng mà chương trình Server đang chạy

- Thiết lập kết nối tới Server

- Trao đổi dữ liệu với Server

- Gửi/ nhận thông điệp với Server

- Đóng kết nối

*Xây dựng thư viện chứa luồng nhận và luồng gửi dùng chung cho Client và Server*

*Thiết kế giao diện hiển thị khung chat phía Client*

- Sử dụng giao thức TCP

- Yêu cầu giao diện:

+ Chat trực tiếp giữa Client và Server khi được kết nối

+ Trao đổi dữ liệu với nhau nhờ vào Stream nhận và Stream gửi

## Phân tích các chức năng

Chức năng của chương trình được thể hiện qua chương trình Client và Server

Chức năng Chat:

- Chuỗi ký tự nhập từ bàn phím

- Ấn nút “Send” để gửi chuỗi ký tự vừa nhập để giao tiếp với nhau giữa Client và Server

Xây dựng chương trình Client: kết nối đến Server để trao đổi thông tin và truyền tải dữ liệu

Xây dựng chương trình Server để kết nối với Client:

- Xử lý các yêu cầu kết nối: lắng nghe yêu cầu kết nối của Client

- Xử lý các thông điệp yêu cầu từ Client: chấp nhận hoặc từ chối kết nối

## Thiết kế kế chương trình

### Thiết kế giao diện

|  |  |
| --- | --- |
| Giao diện Server | Giao diện Client |
| port: số hiệu cổng kết nối  start: bắt đầu lắng nghe kết nối  send: thông tin chat sẽ được gửi | host: địa chỉ IP  port: số hiệu cổng kết nối  connect: gửi kết nối đến Server  send: thông tin chat sẽ được gửi |

### Xây dựng các chức năng

*Xây dựng chương trình Client (sử dụng lớp java.net.Socket):*

- Mở một socket nối kết đến Server đã biết địa chỉ IP/localhost và số hiệu cổng(2003)

- Lấy Stream nhận và Stream gửi được gán với socket

- Trao đổi dữ liệu với Server nhờ vào các Stream nhận và Stream gửi

- Dùng giao thức TCP để kiểm tra dữ liệu trao đổi với Server

*Xây dựng chương trình Server (Server phải luôn giữ kết nối):*

- Xử lý các yêu cầu nối kết:

+ Lắng nghe yêu cầu nối kết

+ Chấp nhận một yêu cầu nối kết

- Tạo kênh giao tiếp ảo với Client

- Xử lý các thông điệp yêu cầu từ Client:

+ Chờ nhận thông điệp của các Client

+ Phân tích và xử lý yêu cầu

+ Gửi thông điệp trả lời cho Client

# TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ

## Môi trường triển khai

Hệ điều hành MacOS Ventura, Window 10, Window 11

Netbeans

JDK 19

Mạng LAN

## Kết quả các chức năng của chương trình

|  |  |
| --- | --- |
| Giao diện Server | Giao diện Client |
| click "Start"      click "Send" | click "Connect”    click "Send" |

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Những kết quả đạt được

- Mô phỏng được mô hình Client-Server

- Thực hiện được yêu cầu bài đề ra (ứng dụng chat)

- Dùng giao thức TCP trong truyền dữ liệu

- Dùng socket để lập trình

## Những vấn đề tồn tại

- Chương trình Chat còn đơn giản

- Giao diện sơ sài

- Chỉ thực hiện được kết nối giữa Client-Server

## Hướng phát triển

- Hoàn thiện giao diện đẹp và phù hợp với người dùng

- Mở rộng ứng dụng trong chương trình Chat như :

+ Xử lý truyền file thông qua chương trình Chat

+ Xây dựng hệ cơ sở dữ liệu quản lý thông tin đăng nhập (username, password …)

- Phát triển trên mạng rộng ngoài LAN

# 

# TÀI LIỆU THAM KHẢO

[1] Herbert Schildt. “Java: A Beginner's Guide 9th Edition” [Accessed January 7 2022].

[2] Herbert Schildt. “Java: The Complete Reference, 12th Edition”  [Accessed December 12 2021].

[3] GP Coder. “Xây dựng ứng dụng Client-Server với Socket trong Java” [Post on 26/03/2018]. <https://gpcoder.com/3679-xay-dung-ung-dung-client-server-voi-socket-trong-java/>

[4] “Network Socket” [Last edited on 26/01/2023]. <https://en.wikipedia.org/wiki/Network_socket>