**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

A red circle with a star and a book

Description automatically generated----------**🙠🙣✪🙡🙣**----------

**BÀI BÁO CÁO CUỐI KỲ**

**MÔN HỌC: CÁC HỆ THỐNG PHÂN TÁN**

**Đề tài: Xây dựng ứng dụng các client nhập vào các số ngẫu nhiên từ 1-10, Server xuất ra tổng các giá trị mà server xác nhận được nhập vào từ client**

**GVHD : LÊ NGỌC BẢO**

**Sinh viên : NGUYỄN HỮU HIẾU**

**Lớp : D20CQCNPM02-N**

**MSSV : N20DCCN102**

**TP.THỦ ĐỨC, tháng 1 năm 2024**

**NHẬN XÉT CỦA GVHD**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**MỤC LỤC**

[**I.** **TÊN ĐỀ TÀI VÀ THÀNH VIÊN NHÓM:** 4](#_Toc155128956)

[**1.** **Tên đề tài** 4](#_Toc155128957)

[**2.** **Thành viên nhóm** 4](#_Toc155128958)

[**II.** **GIỚI THIỆU** 4](#_Toc155128959)

[**1.** **Miêu tả đề tài** 4](#_Toc155128960)

[**2.** **Mục tiêu** 5](#_Toc155128961)

[**III.** **PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG** 6](#_Toc155128962)

[**1.** **Sơ lược lý thuyết** 6](#_Toc155128963)

[**a.** **Khái niệm** 6](#_Toc155128964)

[**b.** **Tại phía server (server-side):** 6](#_Toc155128965)

[**c.** **Tại phía client (client-side):** 6](#_Toc155128966)

[**2.** **Thiết Kế Ứng Dụng Client-Server:** 7](#_Toc155128967)

[**3.** **Phân Tích Cấu Trúc và Giao Tiếp với Socket:** 7](#_Toc155128968)

[**4.** **Mô Tả Giao Thức Truyền Thông sử Dụng Socket:** 7](#_Toc155128969)

[**IV.** **XÂY DỰNG ỨNG DỤNG** 8](#_Toc155128970)

[**1.** **Hoạt động của Client** 8](#_Toc155128971)

[**2.** **Hoạt động của Server** 14](#_Toc155128972)

[**V.** **CHẠY THỰC NGHIỆM** 16](#_Toc155128973)

[**VI.** **TỔNG KẾT** 20](#_Toc155128974)

# **TÊN ĐỀ TÀI VÀ THÀNH VIÊN NHÓM:**

## **Tên đề tài**

Xây dựng ứng dụng các client nhập vào các số ngẫu nhiên từ 1-10, Trường hợp server không xác nhận phải nhập lại giá  trị; Server xuất ra tổng các giá trị  mà server xác nhận được nhập vào từ client

## **Thành viên nhóm**

Nhóm 13 gồm cách thành viên:

Phạm Văn Thành N20DCCN072

Võ Thành Đạt N20DCCN015

Nguyễn Văn Hoài Nam N20DCCN043

Nguyễn Viết Sĩ N20DCCN059

Nguyễn Hữu Hiếu N20DCCN102

# **GIỚI THIỆU**

## **Miêu tả đề tài**

Đề tài của chúng tôi xoay quanh việc phát triển một ứng dụng client-server có khả năng nhập và xác nhận các số ngẫu nhiên từ 1 đến 10. Trong mô hình này, người dùng có thể tương tác với server bằng cách nhập giá trị, và sau đó, server sẽ xác nhận giá trị đó. Nếu xác nhận thành công, quá trình tiếp tục; ngược lại, server yêu cầu người dùng nhập lại giá trị. Điều này tạo ra một hệ thống linh hoạt và chính xác, giúp đảm bảo tính chính xác và đáng tin cậy trong việc xác nhận thông tin nhập từ phía người dùng.

Mục tiêu của dự án là tạo ra một ứng dụng client-server hiệu quả, giúp người dùng tương tác với server một cách thuận tiện và linh hoạt. Qua đó, chúng tôi đặt ra mục tiêu xây dựng một giao diện đơn giản, nhưng đầy đủ chức năng, giúp người dùng dễ dàng thực hiện các thao tác nhập và xác nhận giá trị.

Ngoài ra, chúng tôi hướng đến mục tiêu tối ưu hóa trải nghiệm người dùng thông qua việc thiết kế giao diện thân thiện và dễ sử dụng. Mục tiêu không chỉ là cung cấp một ứng dụng có chức năng, mà còn là tạo ra một trải nghiệm mượt mà và thân thiện, giúp tăng cường sự tương tác và hài lòng của người dùng.

Cuối cùng, mục tiêu của chúng tôi không chỉ là xây dựng một ứng dụng có tính ứng dụng cao trong việc nhập và xác nhận số, mà còn là tạo ra một sản phẩm có khả năng mở rộng và tích hợp vào nhiều hệ thống khác nhau đòi hỏi sự tương tác giữa client và server. Chúng tôi tin rằng tính linh hoạt này sẽ giúp ứng dụng trở nên hữu ích và có giá trị thực tế trong nhiều ngữ cảnh ứng dụng khác nhau.

## **Mục tiêu**

Mục tiêu của dự án này là xây dựng một ứng dụng client-server linh hoạt và mạnh mẽ, giúp người dùng tương tác một cách thuận lợi với server thông qua việc nhập và xác nhận các số ngẫu nhiên từ 1 đến 10. Chúng tôi đặt ra mục tiêu tạo ra một giao diện đơn giản, nhưng hiệu quả, giúp người dùng dễ dàng thực hiện các thao tác nhập và xác nhận giá trị mà không gặp phải sự phức tạp không cần thiết.

Mục tiêu chính của ứng dụng là đảm bảo tính chính xác và đáng tin cậy trong quá trình xác nhận các giá trị từ phía server. Điều này có thể đảm bảo rằng thông tin nhập từ người dùng được xử lý chính xác, và nếu có sự không nhất quán, hệ thống có thể yêu cầu người dùng nhập lại giá trị một cách linh hoạt.

Ngoài ra, chúng tôi cũng hướng đến việc tối ưu hóa trải nghiệm người dùng thông qua giao diện đơn giản và thân thiện. Mục tiêu không chỉ là cung cấp một ứng dụng có chức năng, mà còn là tạo ra một trải nghiệm mượt mà và dễ sử dụng. Điều này có thể tăng cường sự tương tác và sự hài lòng của người dùng, làm cho ứng dụng trở nên hấp dẫn và thân thiện hơn.

Mục tiêu cuối cùng là tạo ra một ứng dụng có khả năng mở rộng, có thể được tích hợp vào nhiều hệ thống khác nhau đòi hỏi sự tương tác giữa client và server. Chúng tôi hiểu rằng tính linh hoạt của ứng dụng sẽ đóng vai trò quan trọng trong việc áp dụng nó vào các tình huống thực tế và đa dạng, từ ứng dụng di động đến các hệ thống máy chủ lớn. Điều này giúp tạo ra một sản phẩm có giá trị thực tế và khả năng triển khai rộng lớn trong nhiều lĩnh vực ứng dụng khác nhau.

# **PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG**

## **Sơ lược lý thuyết**

### **Khái niệm**

·        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.

·        Một chức năng khác của socket là giúp các tầng TCP hoặc TCP Layer định danh ứng dụng mà dữ liệu sẽ được gửi tới thông qua sự ràng buộc với một cổng port (thể hiện là một con số cụ thể), từ đó tiến hành kết nối giữa client và server.

### **Tại phía server (server-side):**

·        Thông thường, một chương trình server chạy trên một máy tính cụ thể, chương trình này có một ổ cắm (Server Socket), ổ cắm được ràng buộc bởi cổng (Port number) cụ thể. Các chương trình phục vụ (Server program) chỉ chờ đợi, lắng nghe tại ổ cắm (Server Socket) các Client để thực hiện một yêu cầu kết nối.Quá trình đọc dữ liệu từ server

### **Tại phía client (client-side):**

·        Các Client biết tên máy của máy tính mà trên đó chương trình chủ (server) đang chạy và số cổng mà chương trình chủ lắng nghe. Để thực hiện một yêu cầu kết nối, Client cố gắng tạo ra cuộc gặp với máy chủ trên máy tính của chương trình chủ và cổng. Các Client cũng cần phải tự định danh chính nó với server để gắn với một cổng địa phương cái sẽ được sử dụng trong suốt quá trình kết nối này, thông thường nó được gán bởi hệ điều hành.

## **Thiết Kế Ứng Dụng Client-Server:**

Hệ thống client-server của chúng tôi được xây dựng trên cơ sở sử dụng socket, cung cấp một giao thức liên lạc đa chiều giữa client và server. Cả hai đều sử dụng socket để thiết lập kết nối và truyền thông dữ liệu. Phía client sẽ khởi tạo một socket và kết nối đến server thông qua một địa chỉ và cổng xác định

## **Phân Tích Cấu Trúc và Giao Tiếp với Socket:**

* Client:
  + Socket Kết Nối: Sử dụng socket để kết nối với server thông qua địa chỉ và cổng.
  + Gửi Dữ Liệu: Thực hiện gửi dữ liệu nhập từ người dùng tới server qua socket.
* Server:
  + Chấp Nhận Kết Nối: Sử dụng socket để chấp nhận kết nối từ client và tạo ra một socket mới cho tương tác.
  + Nhận và Xử Lý Dữ Liệu: Sử dụng socket để nhận dữ liệu từ client, xác nhận giá trị, và gửi phản hồi.

Giao tiếp giữa client và server thông qua socket giúp xác định rõ ràng quy trình truyền thông và đồng bộ hoạt động. Các socket đóng vai trò là giao diện để truyền thông và truyền tải dữ liệu giữa các thành phần của hệ thống.

## **Mô Tả Giao Thức Truyền Thông sử Dụng Socket:**

Giao thức truyền thông dựa trên socket được xác định bởi việc sử dụng TCP/IP, cung cấp một cách an toàn và ổn định để truyền tải dữ liệu giữa client và server. Mỗi thông điệp từ client đến server và ngược lại được đóng gói trong các gói tin và truyền qua socket. Các mã trạng thái và lỗi được định rõ để quản lý quá trình truyền thông và đảm bảo độ tin cậy.

Tổng quan, thiết kế hệ thống client-server với sử dụng socket giúp tạo ra một môi trường tương tác mạnh mẽ và linh hoạt, cung cấp các công cụ cần thiết để quản lý kết nối và truyền tải dữ liệu giữa các thành phần của hệ thống.

# **XÂY DỰNG ỨNG DỤNG**

## **Hoạt động của Client**

A screenshot of a computer

Description automatically generated

Bên phái Client sẽ nhập địa chị IP và Port để kết nối với server. Nếu nhập trống thông tin thì sẽ gửi thông báo “ Vui lòng nhập đầy đủ thông tin.

Sau khi nhập đầy đủ thông tin thì sẽ kết nối với server nếu sai IP và PORT thì sẽ gửi thông báo “ Lỗi kết nối!!”. Nếu thành công thì sẽ thực hiện đoạn code sau:

A computer screen shot of a computer code

Description automatically generated

Đoạn code trên khởi tạo 1 đối tượng socket với IP và Port đã nhập.Tạo một đối tượng DataInputStream để đọc dữ liệu từ InputStream của socket. Nó được sử dụng để nhận dữ liệu từ server. Tạo một đối tượng DataOutputStream để ghi dữ liệu vào OutputStream của socket. Nó được sử dụng để gửi dữ liệu đến server. Lưu trữ các đối tượng Socket, DataInputStream, và DataOutputStream vào đối tượng clientThread thông qua các setter. Điều này cho phép các thông tin này được chia sẻ và sử dụng trong toàn bộ ứng dụng.

Sau đó sẽ hiển thị giao diện cho người dùng nhập thông tin cá nhân.

A screenshot of a computer

Description automatically generated

Yêu cầu phải nhập đầy đủ thông tin không là sẽ ngay lập tức báo lỗi. Sau khi đã điền đầy đủ thông tin thì sẽ nhấn button Bắt Đầu.

Sau khi nhấn button bắt đầu chương trình sẽ thực hiện đoạn code sau:

A screenshot of a computer

Description automatically generated

* Đầu tiên sẽ đóng giao diện nhập thông tin trước đó lại.
* Sau đó lấy mssv và name từ ô nhập text đã được mã hóa Base64 gửi đến server.
* Nhận thông tin từ server thông qua DataInputStream của đối tượng clientThread.
* Giải mã thông tin Option và Code Room từ chuỗi Base64.
* Cập nhật thông tin về mã số sinh viên và tên cho client hiện tại
* Đóng cửa sổ nhập thông tin để mở cửa sổ phòng chat.
* Nếu Option (tmp) là "0", hiển thị giao diện của chat với việc là người đầu tiên vào thì sẽ không cần nhập mã phòng vào thẳng phòng chat.
* Ngược lại, hiển thị giao diện của ChatRoomCode tức là không phải người đầu tiên vào và phải yêu cầu nhập mã phòng mới được vào.
* Nếu là Option là “0” thì sẽ hiển thị cửa sổ chat và thực hiện các công việc sau:

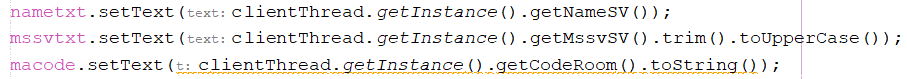
A close-up of a white background

Description automatically generated

‘’another’’ và ‘’me’’ là hai đối tượng SimpleAttributeSet, được sử dụng để định dạng văn bản trong StyledDocument. SimpleAttributeSet chứa các thuộc tính như màu sắc và căn lề.

StyleConstants.setAlignment: Đặt căn lề cho văn bản. Ở đây, cả hai đối tượng another và me đều được đặt căn lề sang trái (StyleConstants.ALIGN\_LEFT).

StyleConstants.setForeground: Đặt màu chữ cho văn bản. another có màu đen (Color.BLACK), me có màu xanh lam (Color.BLUE).



Thực hiện việc gán các thông tin tên, mã số sinh viên, mã code phòng hiển thị lên cửa sổ khung chat.

A screenshot of a computer code

Description automatically generated

Đoạn mã này tạo một luồng (thread) để lắng nghe và hiển thị tin nhắn từ server trong ứng dụng chat.

Tạo một đối tượng Thread mới và truyền vào một đối tượng Runnable để thực hiện logic trong phương thức run.

Trong phương thức run, sử dụng vòng lặp vô hạn (while (true)) để liên tục lắng nghe tin nhắn từ server.

Sử dụng clientThread.getInstance().getDin().readUTF() để đọc một dòng tin nhắn từ DataInputStream liên kết với Socket của client.

Nếu tin nhắn không rỗng, nó sẽ được thêm vào StyledDocument (doc) để hiển thị trên giao diện. another là một SimpleAttributeSet được sử dụng để định dạng văn bản.

Bắt các ngoại lệ BadLocationException và IOException để xử lý các vấn đề có thể xảy ra khi thêm tin nhắn vào StyledDocument hoặc khi đọc tin nhắn từ DataInputStream.

Gọi phương thức start() để bắt đầu thực thi luồng. Khi luồng này được bắt đầu, nó sẽ chạy trong nền và liên tục lắng nghe tin nhắn từ server.

Tới công đoạn nhập và gửi tin nhắn từ mình đi. Sau khi nhập đúng yêu cầu là các số từ 1-10 chương trình sẽ thực hiện công việc:

A screenshot of a computer code

Description automatically generated

Sử dụng clientThread.getInstance().getDout().writeUTF(msg) để gửi tin nhắn (msg) đến server thông qua DataOutputStream liên kết với Socket.

Xóa nội dung của ô nhập tin nhắn để chuẩn bị cho tin nhắn mới.

String[] strTemps = msg.split("\n");: Phân tách tin nhắn thành mảng các phần tử dựa trên ký tự xuống dòng \n.

Nếu tin nhắn có ít nhất một phần tử:

* if (!strTemps[0].trim().equals("")): Nếu phần tử đầu tiên không rỗng, thêm tin nhắn vào StyledDocument với định dạng của đối tượng me (văn bản của chính người gửi).
* else: Nếu phần tử đầu tiên rỗng, thêm "Tôi:" vào StyledDocument với định dạng của đối tượng me.

Nếu tin nhắn có nhiều hơn một phần tử:

* Duyệt qua từng phần tử từ phần tử thứ hai trở đi (for (int i = 1; i < strTemps.length; i++)).
* Nếu phần tử không rỗng, thêm " " (khoảng trắng) và nối với phần tử vào StyledDocument với định dạng của đối tượng me.

* Ngược lại nếu  Option khác “0” thì sẽ hiển thị cửa sổ nhập mã code sau đó mới vào được chương trình.

A white background with text

Description automatically generated

Kiểm tra xem đã điền đúng mã không hay chưa. Nếu đúng rồi thì xác nhận đã vào phòng chat. Gủi tin nhắn “SUCCESS” đến server.

## **Hoạt động của Server**

* Kết nối và lắng nghe
* Server bắt đầu bằng việc tạo một Server Socket trên một cổng cụ thể (ví dụ: 9090).

A computer code with text

Description automatically generated

* Server lắng nghe và chờ các kết nối từ client sử dụng serverSocket.accept(). Khi một client kết nối, server tạo một Socket mới để xử lý giao tiếp với client đó.

A computer code with black and white text

Description automatically generated

* Thiết lập luồng cho mỗi client
* Đối với mỗi kết nối được chấp nhận, server tạo một đối tượng ServerThread mới (hoặc luồng khác tương tự) để xử lý giao tiếp với client cụ thể đó.
* Luồng này được khởi chạy bằng cách gọi start().

A computer code with black and white text

Description automatically generated

* Xử lý dữ liệu từ client
* Trong mỗi ServerThread, server đọc dữ liệu từ client thông qua DataInputStream.
* Dữ liệu này được hiểu là một số (float) và được cộng vào biến sum của server.

A computer code with text

Description automatically generated

* Gửi dữ liệu cho tất cả Client
* Sau khi nhận được số từ client, server gửi lại thông điệp đó cho tất cả các client đã kết nối thông qua phương thức sendToAllClients.

A computer code with text

Description automatically generated

* Ngoài ra, server có thể cập nhật ghi chú hoặc tệp log (ví dụ: logs.dat) để theo dõi hoạt động.

A computer screen shot of a computer code

Description automatically generated

# **CHẠY THỰC NGHIỆM**

* Giao diện màn hình chính của Server

A screenshot of a computer

Description automatically generated

* Trên server nhập địa chỉ port 9090 để bắt đầu

A screenshot of a computer

Description automatically generated

* Phía client nhập địa chỉ Ip và port để kết nối tới server

A screenshot of a computer

Description automatically generated

* Server thông báo kết nối

A screenshot of a computer

Description automatically generated

* Sau khi kết nối thành công, tiến hành nhập mã sinh viên và họ tên

A screenshot of a computer

Description automatically generated

* Giao diện chat

A screenshot of a computer

Description automatically generated

* Khi client nhập lần lượt các số 3, 4, 7 thì server tính toán tổng

A screenshot of a computer

Description automatically generated

* Nút Reset có chức năng tính lại tổng từ đầu

A screenshot of a computer

Description automatically generated

* Nút Clear xóa dữ liệu đã nhập trong ô Tính Tổng

A screenshot of a computer

Description automatically generated

* Nút Close đóng kết nối, không cho các kết nối mới tới server

A screenshot of a computer

Description automatically generated

* Khi client thoát, trên server hiển thị thông báo

A screenshot of a computer

Description automatically generated

# **TỔNG KẾT**

Trong quá trình phân tích và thiết kế hệ thống client-server của chúng tôi, chúng tôi đã hướng đến mục tiêu xây dựng một ứng dụng linh hoạt, hiệu quả và đáng tin cậy. Sự tương tác giữa client và server được xây dựng dựa trên kiến trúc socket, cung cấp một cơ chế giao tiếp mạnh mẽ và đồng bộ.

Kiến trúc của ứng dụng đã được thiết kế sao cho phần client chịu trách nhiệm thu thập thông tin từ người dùng và gửi yêu cầu đến server thông qua socket. Server, từ phía mình, xác nhận giá trị và gửi phản hồi dựa trên kết quả xác nhận hoặc yêu cầu nhập lại từ client. Sự tương tác giữa hai bên diễn ra thông qua việc sử dụng socket và giao thức truyền thông, cung cấp một cơ sở cho tính đồng bộ và ổn định của hệ thống.

Giao thức truyền thông, được xác định chủ yếu qua socket, đảm bảo tính nhất quán và độ tin cậy của quá trình tương tác. Mã trạng thái và mã lỗi được định rõ để quản lý tình huống ngoại lệ và bảo đảm tính ổn định của ứng dụng trong mọi trường hợp.

Tóm lại, sự linh hoạt, đáng tin cậy và hiệu quả của hệ thống client-server này không chỉ mang lại trải nghiệm người dùng tốt mà còn đáp ứng được các yêu cầu và thách thức của việc xây dựng một ứng dụng tương tác đa chiều trong môi trường mạng.