TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**

Logo, company name

Description automatically generated

**BÁO CÁO CUỐI KỲ**

**MÔN HỌC: LẬP TRÌNH MẠNG CĂN BẢN**

**Mã môn học: 502065**

Đề tài 10: **XÂY DỰNG CHƯƠNG TRÌNH QUẢN LÝ SINH VIÊN TRÊN MÔ HÌNH CLIENT-SERVER BẰNG JAVA**

**VỚI GIAO THỨC UDP**

*Người hướng dẫn*: **GV.Tống Thanh Văn**

*Người thực hiện*: **Trần Thu Hảo – 52100412**

**Trần Bảo Duy-52100403**

**Nguyễn Cao Minh-52100702**

*Khóa* **: 25**

**TP. HỒ CHÍ MINH, THÁNG 3 NĂM 2021**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**

Logo, company name

Description automatically generated

**BÁO CÁO CUỐI KỲ**

**MÔN HỌC: LẬP TRÌNH MẠNG CĂN BẢN**

**Mã môn học: 502065**

Đề tài 10: **XÂY DỰNG CHƯƠNG TRÌNH QUẢN LÝ SINH VIÊN TRÊN MÔ HÌNH CLIENT-SERVER BẰNG JAVA**

**VỚI GIAO THỨC UDP**

*Người hướng dẫn*: **GV.Tống Thanh Văn**

*Người thực hiện* **Trần Thu Hảo – 52100412**

**Trần Bảo Duy-52100403**

**Nguyễn Cao Minh-52100702**

*Khóa* **: 25**

**TP. HỒ CHÍ MINH, THÁNG 3 NĂM 2021**

**LỜI CẢM ƠN**

Chúng em xin cảm ơn các thầy cô Khoa Công Nghệ Thông Tin, Trường Đại Học Tôn Đức Thắng, lời cảm ơn chân thành nhất. Đặc biệt, chúng em xin bày tỏ lòng biết ơn sâu sắc với thầy Tống Thanh Văn, người đã truyền đạt kiến thức tận tâm trong quá trình giảng dạy và hướng dẫn cách trình bày đề tài nghiên cứu với đề tài là “Xây dựng chương trình quản lý sinh viên trên mô hình client-server bằng java với giao thức UDP”. Cảm ơn thầy Tống Thanh Văn đã tạo điều kiện cho chúng em nghiên cứu và tìm hiểu thêm về một số tài liệu trong quá trình học tập nhằm nâng cao hiểu biết về phương diện mới của các bài học, và cảm ơn thầy đã cung cấp thêm một số tài liệu hướng dẫn cách trình bày để báo cáo được hoàn chỉnh hơn.

Do kiến thức còn hạn chế nên không tránh khỏi những thiếu sót. Nhóm em mong nhận được sự thông cảm và góp ý kiến của quý thầy cô trong trường.

Cuối cùng em kính chúc quý thầy cô dồi dào sức khỏe, vạn sự bình an và thành công trong sự nghiệp. Em xin chân thành cảm ơn!

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Chúng tôi xin cam đoan đây là sản phẩm đồ án của riêng chúng tôi và được sự hướng dẫn của GV Tống Thanh Văn. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 5 tháng 4 năm 2023*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Trần Thu Hảo*

*Trần Bảo Duy*

*Nguyễn Cao Minh*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(ký và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(ký và ghi họ tên)

# **LỜI MỞ ĐẦU**

Sau khi tham khảo các tài liệu, cũng như áp dụng các kiến thức được thầy giảng dạy trên lớp lý thuyết và lớp thực hành và được sự hướng dẫn nhiệt tình của các thầy cô của Khoa Công Nghệ Thông Tin. Trong thời gian làm bài báo cáo, chúng em đã được tìm hiểu thêm vài kiến thức mới về môn Lập trình mạng căn bản và nó cũng rất quan trọng đối với ngành mà chúng em đang hướng đến. Chúng em gồm bạn Nguyễn Cao Minh, bạn Trần Thu Hảo và bạn Trần Bảo Duy cảm ơn các thầy cô đã giúp chúng em để chúng em có thể làm bài báo cáo một cách tốt nhất. Nếu có sai sót mong thầy cô bỏ qua và nhắc nhở để bài báo cáo sau được hoàn thiện hơn.

Quản lý sinh viên hay quản lý hồ sơ sinh viên là một công tác rất quan trọng và phức tạp mà bất kỳ đơn vị đào tạo nào cũng phải thực hiện. Bắt đầu từ việc làm thủ tục nhập học cho sinh viên, tiếp đó là theo dõi, quản lý thông tin về sinh viên trong suốt khóa học cũng như việc hệ thống hóa, lưu trữ và bảo quản thông tin kết quả của sinh viên một cách an toàn.

Việc quản lý này hầu như đã được thực hiện hầu hết ở các đơn vị đào tạo. Tuy nhiên, thông tin ở dạng này chỉ dừng ở mức lưu trữ và mang tính thủ công, chưa thực sự linh hoạt trong việc quản lý. Việc quản lý hồ sơ sinh viên cần tính đến các hoạt động tích cực hơn nữa, chẳng hạn như việc nhập xuất dữ liệu, thoe dõi thời khóa biểu, lịch học, bảng điểm và các thông tin sinh viên. Ngoài ra, để linh hoạt trong quản lý thì việc ứng dụng mô hình client-server sẽ giúp người quản lý có thể truy cập và quản lý sinh viên một cách dễ dàng và hiệu quả.

**MỤC LỤC**

[**LỜI MỞ ĐẦU** 6](#_Toc131451136)

[**DANH MỤC KÝ HIỆU VÀ CHỮ VIẾT TẮT** 9](#_Toc131451137)

[**DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ** 10](#_Toc131451138)

[**CHƯƠNG I: TỔNG QUAN** 12](#_Toc131451139)

[1.1 Tổng quan đề tài 12](#_Toc131451140)

[1.2 Mục tiêu 13](#_Toc131451141)

[1.3 Cấu trúc 13](#_Toc131451142)

[**CHƯƠNG II: CƠ SỞ LÝ THUYẾT** 15](#_Toc131451143)

[1.Tổng quan về mô hình Client-Server 15](#_Toc131451144)

[2.Sơ lược về giao thức UDP 15](#_Toc131451145)

[3.Sơ lược về thuật toán Vigenere 16](#_Toc131451146)

[4.Socket for Client 17](#_Toc131451147)

[4.1 Tổng quan về Socket 17](#_Toc131451148)

[4.1.1 Client Socket 17](#_Toc131451149)

[4.2 The Socket Class 18](#_Toc131451150)

[4.2.1 Xây dựng phương thức 18](#_Toc131451151)

[4.2.2 Nhập thông tin vào Socket 18](#_Toc131451152)

[4.2.3 Closing the Socket 21](#_Toc131451153)

[4.2.4 Tùy chon Socket 21](#_Toc131451154)

[4.3 Socket Exceptions 22](#_Toc131451155)

[5. Socket for server 23](#_Toc131451156)

[5.1 The constructors 23](#_Toc131451157)

[5.1.1 public ServerSocket(int port) throws BindException, IOException 23](#_Toc131451158)

[5.1.2 public ServerSocket(int port, int queueLength) throws IOException 23](#_Toc131451159)

[5.1.3 public ServerSocket() throws IOException 24](#_Toc131451160)

[5.2 Accepting and Closing Connections 24](#_Toc131451161)

[5.2.1 Accepting 24](#_Toc131451162)

[5.2.2 Closing 25](#_Toc131451163)

[5.3 The get Methods 26](#_Toc131451164)

[5.3.1 public InetAddress getInetAddress() 26](#_Toc131451165)

[5.3.2 public int getLocalPort( ) 26](#_Toc131451166)

[5.4 Socket Options 27](#_Toc131451167)

[5.5 The Object Methods 28](#_Toc131451168)

[5.6 Implementation 29](#_Toc131451169)

[**CHƯƠNG III: KẾT QUẢ THỰC NGHIỆM** 30](#_Toc131451170)

[1.Kịch bản thực hiện 30](#_Toc131451171)

[1.1 Giới thiệu chương trình 30](#_Toc131451172)

[1.2 Cách thức hoạt động 31](#_Toc131451173)

[1.3 Giao diện chương trình 31](#_Toc131451174)

[2. Yều cầu thực nghiệm 33](#_Toc131451175)

[3.Công cụ thực hiện 33](#_Toc131451176)

[4.Các bước thực hiện 33](#_Toc131451177)

[5.Kết luận 34](#_Toc131451178)

[**CHƯƠNG IV: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN** 34](#_Toc131451179)

[1.Kết luận 34](#_Toc131451180)

[2.Hướng phát triển 35](#_Toc131451181)

[**TÀI LIỆU THAM KHẢO** 36](#_Toc131451182)

# **DANH MỤC KÝ HIỆU VÀ CHỮ VIẾT TẮT**

UDP: User Datagram Protocol

HTTP: HyperText Transfer Protocol

FTP: File Transfer Protocol

SMTP: Simple Mail Transfer Protocol

# **DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ**

Hình 2.1: Mô hình Client-Server………………………………………………15

Hình 2.2: Ví dụ minh hoạ……………………………………………………...18

Hình 2.3: Ví dụ minh hoạ……………………………………………………...19

Hình 2.4: Ví dụ minh hoạ……………………………………………………...19

Hình 2.5: Ví dụ minh hoạ……………………………………………………...19

Hình 2.6: Mối liên hệ giữa Program và Socket thông qua

OutputStream và InputStream…………………………………………………20

Hình 2.7: Các hàm sử dụng Closing the Socket……………………………….21

Hình 2.8 Ví dụ minh hoạ………………………………………………………23

Hình 2.9: Ví dụ minh hoạ……………………………………………………...23

Hình 2.10: Ví dụ minh hoạ…………………………………………………….24

Hình 2.11: Máy chủ chấp nhận kết nối…………………………………….…..24

Hình 2.12 Ví dụ về public Socket accept() throws IOException………………25

Hình 2.13 Close………………………………………………………………..25

Hình 2.14 Shutdown………………………………………………………...…25

Hình 2.15: Ví dụ về public void close( ) throws IOException…………….…..26

Hình 2.16: Ví dụ về getInetAddress…………………………………………...26

Hình 2.17: Ví dụ về getLocalPort…………………………………………...…27

Hình 2.18 Tạo đối tượng Socket…………………………………………...…..29

Hình 2.19: Chấp nhận liên kết từ yêu cầu liên kết của client………………….29

Hình 2.20 Tạo luồng nhập đọc dữ liệu……………………………………...…29

Hình 2.21 Tạo luồng xuất gửi dữ liệu. ………………………………………...30

Hình 2.22 Các thao tác………………………………………………………....30

Hình 2.23 Đóng socket………………………………………………………...30

Hình 3.1: Giao diện Server………………………………………………….…31

Hình 3.2: Giao diện Client……………………………………………………..32

Hình 3.3: Giao diện Client……………………………………………….…….32

Hình 3.4: Sơ đồ thực hiện chương trình……………………………………….34

**DANH MỤC BẢNG**

Bảng 2.1: Socket Option…………………………………………………...…..28

Bảng 2.2 Object Method…………………………………………………….....29

**NỘI DUNG BÁO CÁO**

# **CHƯƠNG I: TỔNG QUAN**

## 1.1 Tổng quan đề tài

Xây dựng chương trình Quản lý Sinh Viên trên mô hình Client-Server bằng Java với giao thức UDP

Chương trình cho phép nhiều người dùng truy cập vào thông tin của sinh viên cùng lúc.

Một số tính năng chính của chương trình quản lý sinh viên có thể bao gồm:

-Thêm, xóa và sửa thông tin sinh viên: Người dùng có thể thêm mới thông tin của một sinh viên vào hệ thống, chỉnh sửa hoặc xóa thông tin của sinh viên đã có.

-Tìm kiếm thông tin sinh viên: Người dùng có thể tìm kiếm thông tin sinh viên theo tên, mã sinh viên hoặc một số thuộc tính khác.

-Hiển thị danh sách sinh viên: Chương trình có thể hiển thị danh sách sinh viên theo các tiêu chí khác nhau, ví dụ như theo lớp, khoa hoặc địa chỉ.

-Thống kê thông tin sinh viên: Chương trình có thể thống kê các thông tin về số lượng sinh viên theo lớp, giới tính, khoa hoặc một số thuộc tính khác.

Chương trình quản lý sinh viên trên mô hình client-server có thể được thiết kế như sau:

Server:

* Tạo cơ sở dữ liệu cho thông tin sinh viên, bao gồm các thông tin như tên, mã sinh viên, lớp, điểm số,…
* Nhận yêu cầu từ các client và thực hiện các hành động như tìm kiếm, thêm, sửa, xóa thông tin sinh viên trong cơ sở dữ liệu.
* Đảm bảo an toàn và bảo mật cho cơ sở dữ liệu.

Client:

* Hiển thị giao diện cho người dùng để tương tác với server.
* Gửi yêu cầu đến server để thực hiện các hành động trên cơ sở dữ liệu.
* Nhận kết quả trả về từ server và hiển thị cho người dùng.

Để triển khai chương trình này, ta có thể sử dụng các công nghệ như:

* Ngôn ngữ lập trình :Java
* Cơ sở dữ liệu: MySQL, MongoDB,…
* Ngôn ngữ lập trình client: HTML, CSS, JavaScript, Java…
* Giao thức truyền thông giữa client và server: HTTP, WebSocket,UDP…
* Framework và thư viện hỗ trợ: ExpressJS, ReactJS, AngularJS, VueJS, Socket.io,…

## 1.2 Mục tiêu

Mục tiêu của hệ thống quản lý sinh viên là quản lý thông tin sinh viên, giúp đơn giản hóa quá trình quản lý và giám sát sinh viên trong các trường đại học, cao đẳng, trung học phổ thông.

Các mục tiêu cụ thể của hệ thống quản lý sinh viên có thể bao gồm:

+Lưu trữ thông tin chi tiết của sinh viên, bao gồm tên, mã số sinh viên, lớp học, điểm số, v.v.

+Đảm bảo tính toàn vẹn của dữ liệu để tránh việc mất mát thông tin.

+Hỗ trợ cho quá trình quản lý, giám sát và đánh giá kết quả học tập của sinh viên.

+Tạo điều kiện thuận lợi cho việc phân tích và đánh giá hoạt động học tập của sinh viên để đưa ra các quyết định quản lý, đào tạo phù hợp.

+Cung cấp thông tin đầy đủ, chính xác và nhanh chóng về sinh viên cho các đơn vị liên quan như giảng viên, nhà trường, phụ huynh,...

+Tăng tính hiệu quả và độ chính xác trong quá trình quản lý và giám sát sinh viên, từ đó cải thiện chất lượng đào tạo và nâng cao uy tín của trường học.

## 1.3 Cấu trúc

-Cơ sở dữ liệu:

+Lưu trữ thông tin chi tiết của sinh viên, bao gồm tên, mã số sinh viên, lớp học, điểm số, v.v.

+Đảm bảo tính toàn vẹn của dữ liệu để tránh việc mất mát thông tin.

+Cung cấp các chức năng tìm kiếm, thêm, sửa, xóa thông tin sinh viên.

+Giao diện người dùng:

* ,Hiển thị giao diện cho người dùng để tương tác với hệ thống quản lý sinh viên.
* Cung cấp các chức năng tìm kiếm, thêm, sửa, xóa thông tin sinh viên.

-Xử lý logic:

+Điều khiển các chức năng của hệ thống quản lý sinh viên.

+Xử lý các yêu cầu từ người dùng và truy vấn cơ sở dữ liệu để lấy dữ liệu cần thiết.

-Lớp giao diện người dùng (Presentation layer):

+Lớp này có nhiệm vụ hiển thị thông tin cho người dùng, cho phép người dùng thao tác với hệ thống, như thêm, sửa, xóa, tìm kiếm thông tin sinh viên. Giao diện người dùng thường được thiết kế bằng các công nghệ web như HTML, CSS, JavaScript hoặc framework như React, Angular, VueJS, ...

-Bảo mật:

+Đảm bảo tính bảo mật của hệ thống quản lý sinh viên.

+Xác thực người dùng và quản lý quyền truy cập của từng người dùng.

-Phía Client:

Phía client là phần giao diện người dùng, chịu trách nhiệm tương tác với người dùng để thu thập thông tin về sinh viên, hiển thị thông tin của sinh viên, cập nhật thông tin sinh viên, và gửi yêu cầu đến phía server để lấy dữ liệu hoặc cập nhật dữ liệu. Phía client thường được phát triển dưới dạng một ứng dụng desktop, ứng dụng web hoặc ứng dụng di động.

-Phía Server:

Phía server là phần xử lý logic của hệ thống, chịu trách nhiệm lưu trữ và quản lý cơ sở dữ liệu về sinh viên. Phía server nhận các yêu cầu từ phía client, xử lý yêu cầu đó và trả lại kết quả cho phía client. Phía server thường được phát triển dưới dạng một ứng dụng web hoặc một hệ thống ứng dụng enterprise.

# **CHƯƠNG II: CƠ SỞ LÝ THUYẾT**

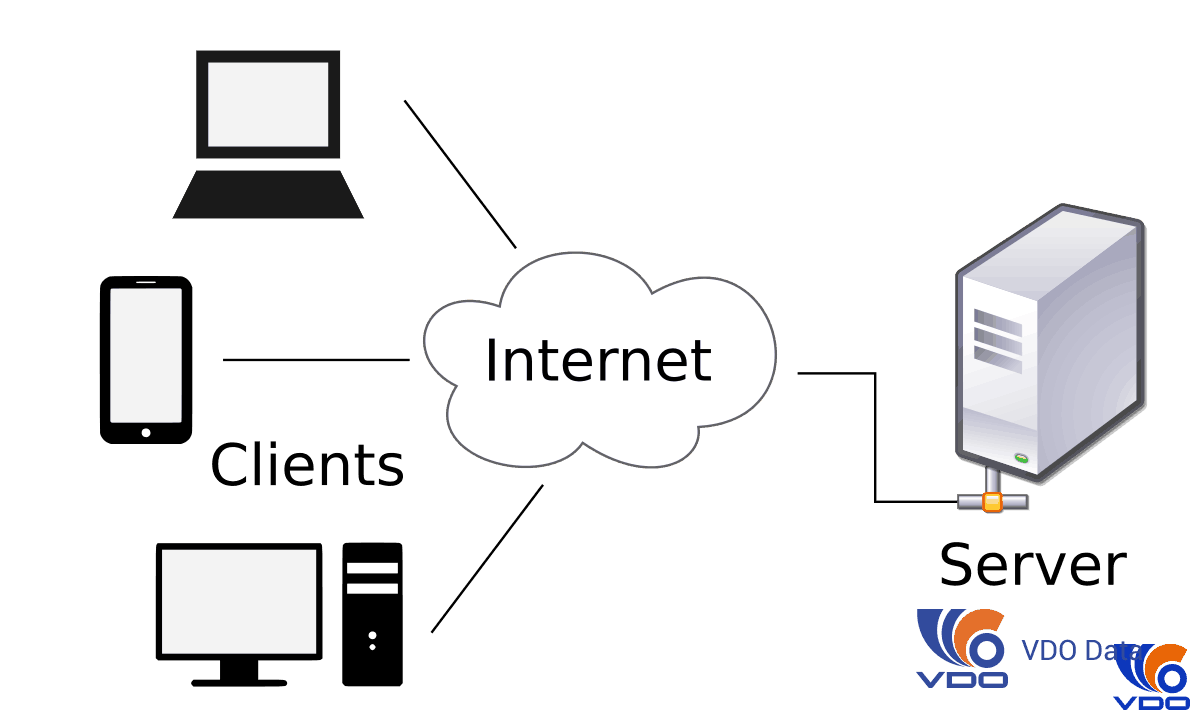
## 1.Tổng quan về mô hình Client-Server

Mô hình client-server là một mô hình phân tán trong đó các ứng dụng phần mềm được phân chia thành hai phần riêng biệt: client và server.

Client là một ứng dụng phần mềm được cài đặt trên máy tính hoặc thiết bị người dùng cuối, nó tạo ra các yêu cầu (request) đến server để yêu cầu dữ liệu hoặc thực hiện một tác vụ nào đó.

Server là một ứng dụng phần mềm chạy trên một máy chủ (server) có thể đáp ứng các yêu cầu từ các client. Server có nhiệm vụ xử lý các yêu cầu từ client và trả về kết quả cho client tương ứng.

Mô hình client-server thường được sử dụng trong các ứng dụng phần mềm có tính chất phân tán như hệ thống quản lý cơ sở dữ liệu, trò chuyện trực tuyến, mạng xã hội, email, và các ứng dụng web. Trong mô hình này, client và server liên lạc với nhau thông qua giao thức truyền thông như HTTP, FTP, SMTP, hay các giao thức truyền thông khác.



Hình 2.1: Mô hình Client-Server

## 2.Sơ lược về giao thức UDP

UDP (User Datagram Protocol) là một giao thức truyền dữ liệu không đồng bộ, không đảm bảo tính toàn vẹn và độ tin cậy trong quá trình truyền tải dữ liệu giữa các máy tính trên mạng. UDP là một giao thức tầng trên (transport layer protocol) trong mô hình OSI (Open Systems Interconnection).

UDP thường được sử dụng cho các ứng dụng cần tốc độ truyền tải nhanh và độ trễ thấp, như video streaming, game online, VoIP (Voice over Internet Protocol), DNS (Domain Name System), v.v.

So với TCP (Transmission Control Protocol), UDP không có quá trình bắt tay giữa các máy tính để thiết lập một kết nối trước khi truyền dữ liệu, điều này giúp giảm thiểu độ trễ. Tuy nhiên, vì không có quá trình kiểm tra tính toàn vẹn và độ tin cậy trong quá trình truyền tải dữ liệu, các gói tin dữ liệu có thể bị mất hoặc đến nơi không đúng thứ tự.

Mặc dù UDP không được sử dụng để truyền tải dữ liệu nhạy cảm và yêu cầu tính toàn vẹn cao, nhưng nó vẫn có ứng dụng trong một số trường hợp đặc biệt, nơi mà tốc độ truyền tải dữ liệu và độ trễ là yếu tố quan trọng hơn so với tính toàn vẹn và độ tin cậy, ví dụ như truyền tải các gói tin dữ liệu chứa các thông tin liên quan đến video game.

## 3.Sơ lược về thuật toán Vigenere

-Thuật toán Vigenere là một phương pháp mã hóa thông điệp bằng cách sử dụng một chuỗi các chữ cái được gọi là "khóa". Thuật toán Vigenere sử dụng phép cộng modulo để mã hóa thông điệp. Mỗi ký tự của thông điệp được thay thế bằng một ký tự trong khóa. Khóa được lặp lại nhiều lần để tạo ra độ dài bằng hoặc lớn hơn độ dài của thông điệp.

-Việc mã hóa và giải mã thông điệp bằng thuật toán Vigenere đều dựa trên việc sử dụng bảng mã Vigenere, một ma trận đơn giản được sử dụng để thay thế ký tự. Bảng này được tạo bằng cách viết các chữ cái từ A đến Z theo thứ tự trong hàng ngang và cột dọc, với các chữ cái được đánh số từ 0 đến 25.

Cách thức hoạt động của thuật toán Vigenere như sau:

* Bước 1: Tạo ra một bảng mã hoán vị (tabula recta) bằng cách xếp chồng các bảng Caesar (bảng mã dịch chuyển) theo các giá trị của các kí tự khóa.
* Bước 2: Chọn một khóa (key) là một chuỗi kí tự để mã hóa văn bản cần được bảo mật.
* Bước 3: Chuyển đổi văn bản cần mã hóa và khóa sang các số tương ứng trong bảng mã hoán vị.
* Bước 4: Sử dụng các giá trị tương ứng từ bảng mã hoán vị để thực hiện phép toán XOR giữa từng cặp số tương ứng của văn bản và khóa.
* Bước 5: Chuyển đổi các kết quả của phép XOR trở lại thành các kí tự tương ứng trong bảng mã hoán vị để tạo thành văn bản mã hóa.

-Tuy nhiên, thuật toán Vigenere không được coi là an toàn bởi vì nó có thể bị phá vỡ bằng cách sử dụng các kỹ thuật phân tích tần suất và các kỹ thuật mã hóa khác. Do đó, hiện nay nó ít được sử dụng trong thực tế và đã được thay thế bởi các thuật toán mã hóa hiện đại hơn và an toàn hơn.

## 4.Socket for Client

### 4.1 Tổng quan về Socket

Trong Java, các lớp URL và URLConnections cung cấp một cơ chế nâng cao để truy cập tài nguyên trên Internet. Tuy nhiên, đôi khi các chương trình của bạn cần giao tiếp với nhau qua mạng cục bộ, chẳng hạn khi bạn muốn viết một ứng dụng Server-Client. TCP (Transmission Control Protocol) cung cấp một kênh giao tiếp đáng tin cậy mà các ứng dụng Server-Client sử dụng để giao tiếp với nhau. Để có thể giao tiếp với nhau qua TCP, Server-Client được thiết lập để kết nối với nhau thông qua Socket. Vậy Socket là gì?

Socket là điểm cuối của liên kết giao tiếp hai chiều giữa hai chương trình đang chạy trên mạng. Lớp Socket được sử dụng để biểu diễn kết nối giữa máy khách và máy chủ. Nó có thể thực hiện bảy hoạt động cơ bản sau:

- Kết nối với một máy từ xa.

- Gửi dữ liệu.

- Nhận dữ liệu.

- Đóng một mối liên hệ.

- Liên kết với một cổng.

- Nghe xem có nhận được dữ liệu không.

- Chấp nhận kết nối từ các máy từ xa trên cổng giới hạn

### 4.1.1 Client Socket

Khi kết nối được thiết lập, các chủ thể địa phương và từ xa lấy các luồng đầu vào và đầu ra từ socket và sử dụng các luồng đó để gửi dữ liệu cho nhau.

Khi việc truyền dữ liệu hoàn tất, một hoặc cả hai bên sẽ đóng kết.

### 4.2 The Socket Class

### 4.2.1 Xây dựng phương thức

Socket class trong Java là một lớp đối tượng cung cấp các phương thức để tạo, thiết lập và quản lý kết nối mạng giữa các máy tính thông qua các giao thức mạng như TCP hoặc UDP.

Trong Java, lớp đối tượng Socket được định nghĩa trong gói java.net, và các phương thức của lớp Socket được sử dụng để tạo, thiết lập, và điều khiển kết nối socket. Một số phương thức quan trọng của Socket class trong Java bao gồm:

* Socket(): phương thức tạo một socket mới
* bind(): phương thức gán một địa chỉ IP và cổng cho socket
* connect(): phương thức thiết lập kết nối đến một socket khác
* getInputStream(): phương thức trả về một luồng đầu vào cho socket
* getOutputStream(): phương thức trả về một luồng đầu ra cho socket
* close(): phương thức đóng kết nối socket

Ngoài ra, Java còn cung cấp lớp ServerSocket để tạo các socket server và các phương thức để chấp nhận kết nối đến từ client.

Các lớp Socket và ServerSocket trong Java có thể được sử dụng để tạo các ứng dụng mạng đa luồng, trong đó mỗi kết nối sẽ được quản lý bởi một luồng riêng biệt. Các tính năng này giúp cho Socket class trong Java trở thành một công cụ mạnh mẽ cho việc phát triển các ứng dụng mạng trên nền tảng Java.

### 4.2.2 Nhập thông tin vào Socket

- **public InetAddress getInetAddress()**

Phương thức getInetAddress() này trả về địa chỉ của máy tính khác mà socket này được kết nối.

Graphical user interface, text

Description automatically generated with medium confidence

Hình 2.2: Ví dụ minh hoạ.

- **public int getPort()**

Phương thức này cho ta biết Socket đã được kết nối hoặc sẽ được vào cổng nào trên máy chủ từ xa

Graphical user interface, text, application, Word

Description automatically generated

Hình 2.3: Ví dụ minh hoạ.

- **public int getLocalPort()**

Phương thức này trả về cổng mà socket bị ràng buộc trên máy local

Graphical user interface, text, application, Word

Description automatically generated

Hình 2.4: Ví dụ minh hoạ.

- **public InetAddress getLocalAddress()**

Phương thức này báo cho ta biết một Socket bị ràng buộc với giao diện mạng nào

Text

Description automatically generatedHình 2.5: Ví dụ mình họa

- **GetInputStream and GetOutputStream**

Graphical user interface, text, application, chat or text message

Description automatically generated

Hình 2.6: Mối liên hệ giữa Program và Socket thông qua OutputStream và

InputStream.

**- public InputStream getInputStream() throws IOException**

phương thức này trả về một dòng đầu vào có thể đọc dữ liệu từ socket vào một chương trình

-**public OutputStream getOutputStream( ) throws IOException**

Phương thức này trả về dòng đầu ra của socket. Output stream được kết nối với input stream của socket remote.

### 4.2.3 Closing the Socket

Graphical user interface, text, application, email

Description automatically generated

Hình 2.7: Các hàm sử dụng trong Closing the Socket.

**- public boolean isClosed()**

Phương thức này trả về giá trị true nếu socket đã đóng, trả về giá trị false nếu không.

- **public boolean isConnected()**

Phương thức isConnected () của lớp Java Socket trả về trạng thái kết nối của socket. Phương thức này sẽ tiếp tục trả về trạng thái kết nối trước khi bị đóng.

- **Half – closed sockets**

Khi chúng ta truyền dữ liệu lên máy chủ và không biết lượng dữ liệu truyền đi là bao nhiêu, nếu chúng ta đóng socket khi truyền dữ liệu thì chúng ta sẽ bị ngắt kết nối với máy chủ ngay lập tức. Tại thời điểm này, chúng ta sẽ không thể nhận được phản hồi từ máy chủ

Trong trường hợp này chúng ta sẽ Sử dụng phương pháp half – close , nghĩa là chỉ đóng luồng đầu ra của một ổ cắm để cho biết rằng dữ liệu yêu cầu được gửi đến máy chủ đã kết thúc, nhưng luồng đầu vào phải được giữ ở trạng thái mở.

• public void shutdownInput( ) throws IOException.

• public void shutdownOutput( ) throws IOException.

### 4.2.4 Tùy chon Socket

- **TCP\_NODELAY**

public void setTcpNoDelay(boolean on) throws SocketException.

public boolean getTcpNoDelay() throws SocketException.

* Nó sẽ Vô hiệu hóa thuật toán của Nagle.
* Hợp lệ cho client Sockets

- **SO\_LINGER**

public void setSoLinger(boolean on, int seconds) throws SocketException.23

public int getSoLinger() throws SocketException.

* Chỉ định thời gian chờ kéo dài khi đóng cửa.
* Hợp lệ cho client Sockets

- **SO\_TIMEOUT**

public void setSoTimeout(int milliseconds) throws SocketException

public int getSoTimeout() throws SocketException

+ public String toString()

+Phương pháp tostring() tạo ra một chuỗi như sau:

* *Socket[[addr=www.oreilly.com/198.112.208.11,port=80,localport=50055](http://addr%3Dwww.oreilly.com/198.112.208.11,port=80,localport=50055)]*
* Socket[[*addr=www.oreilly.com/198.112.208.11,port=80,*
* *localport=50055*](*http://addr%3Dwww.oreilly.com/198.112.208.11,port=80,localport=50055*)]

+Chỉ định thời gian chờ cho các hoạt động của socket chặn.

+Có giá trị cho tất cả các socket: Socket, ServerSocket, DatagramSocket.

### 4.3 Socket Exceptions

Hầu hết các phương pháp của lớp Socket được tuyên bố là throw IOException hoặc lớp con của nó, java.net.SocketException.

**+public class SocketException extends IOException**

Phân lớp của SocketException

* **public class BindException extends SocketException**
* **public class ConnectException extends SocketException**
* **public class NoRouteToHostException extends SocketException**

Phương thứcnày trả về địa chỉ của hệ thống đang được kết nối.

**• public SocketAddress getLocalSocketAddress( )**

Phương thức này trả về địa chỉ nơi kết nối

## 5. Socket for server

### 5.1 The constructors

Có 4 constructor public ServerSocket:

### 5.1.1 public ServerSocket(int port) throws BindException, IOException

Constructor này tạo một socket cho server trên cống xác định. Nếu port bằng 0, hệ thống chọn một cổng ngẫu nhiên cho ta. Cổng do hệ thống chọn đôi khi được gọi là cổng vô danh vì ta không biết số hiệu cổng. Với các server, các cổng vô danh không hữu ích lắm vì các client cần phải biết trước cổng nào mà nó nối tới (giống như người gọi điện thoại ngoài việc xác định cần gọi cho ai cần phải biết số điện thoại để liên lạc với người đó)

Graphical user interface, text, application

Description automatically generated

Hình 2.8 Ví dụ minh hoạ

### 5.1.2 public ServerSocket(int port, int queueLength) throws IOException

Mở socket máy chủ trên cổng được chỉ định với độ dài hàng đợi người dùng chọn

Graphical user interface, text, application, email

Description automatically generated

Hình 2.9: Ví dụ minh hoạ.

### 5.1.3 public ServerSocket() throws IOException

Graphical user interface, text, application, Word

Description automatically generated

Hình 2.10: Ví dụ minh hoạ.

### 5.2 Accepting and Closing Connections

### 5.2.1 Accepting

Với một socket được đánh dấu là đang lắng nghe, máy chủ có thể chấp nhận kết nối:

Graphical user interface, text, application, Word

Description automatically generated

Hình 2.11: Máy chủ chấp nhận kết nối.

Khi một kết nối được chấp nhận, một bộ mô tả mới sẽ được trả về (cùng với một socket mới). Một đối tượng ServerSocket hoạt động trong một vòng lặp chấp nhận các liên kết. Mỗi lần lặp nó gọi phương thức accept(). Phương thức này trả về một đối tượng Socket biểu diễn liên kết giữa client và server. Tương tác giữ client và server được tiến hành thông qua socket này. Khi giao tác hoàn thành, server gọi phương thức close() của đổi tượng socket. Nếu client ngắt liên kết trong khi server vẫn đang hoạt động, các luồng vào ra kết nối server với client sẽ đưa ra ngoại lệ InterruptedException trong lần lặp tiếp

theo .Khi bước thiết lập liên kết hoàn thành, và ta sẵn sàng để chấp nhận liên kết, cần gọi phương thức accept() của lớp ServerSocket. Phương thức này phong tỏa; nó dừng quá trình xử lý và đợi cho tới khi client được kết nối. Khi client thực sự kết nối, phương thức accept() trả về đối tượng Socket. Ta sử dụng các phương thức getlnputStream() và getOutputStream() để truyền tin với client.

Graphical user interface, text, application, email

Description automatically generatedHình 2.12 Ví dụ về public Socket accept() throws IOException.

### 5.2.2 Closing

Khi một socket không còn được quan tâm, nó có thể bị loại bỏ bằng cách áp dụng một giá trị đóng vào bộ mô tả:

Graphical user interface

Description automatically generated with medium confidence

Hình 2.13 Close.

Nếu dữ liệu được liên kết với một socket hứa hẹn phân phối đáng tin cậy (ví dụ: socket luồng) khi *close()* diễn ra, hệ thống sẽ tiếp tục cố gắng truyền dữ liệu. Tuy nhiên, nếu sau một khoảng thời gian khá dài mà dữ liệu vẫn không được phân phối, nó sẽ bị loại bỏ. Nếu bạn không sử dụng bất kỳ dữ liệu đang chờ xử lý nào, bạn có thể thực hiện *shutdown()* trên socket trước khi đóng nó:

Graphical user interface, application

Description automatically generated

Hình 2.14 Shutdown

Trong đó how là 0 nếu bạn không còn quan tâm đến việc đọc dữ liệu, là 1 nếu không còn dữ liệu nào được gửi hoặc là 2 nếu không có dữ liệu nào được gửi hoặc nhận.

Nếu đã hoàn thành công việc với một ServerSocket, ta cần phải đóng nó lại, đặc biệt nếu chương trình của ta tiếp tục chạy. Điều này nhằm tạo điều kiện cho các chương trình khác muốn sử dụng nó. Đóng một ServerSocket không đồng nhất với việc đóng một Socket. Lớp ServerSocket cung cấp một số phương thức cho ta biết địa chỉ cục bộ và cổng mà trên đó đối tượng server đang hoạt động. Các phương thức này hữu ích khi ta đã mở một đồi tượng server socket trên một cổng vô danh và trên một giao tiếp mạng không.

Graphical user interface, text, application, email

Description automatically generated

Hình 2.15: Ví dụ về public void close( ) throws IOException.

### 5.3 The get Methods

### 5.3.1 public InetAddress getInetAddress()

Phương thức này trả về địa chỉ được sử dụng bởi server (localhost). Nếu localhost có địa chỉ IP, địa chỉ này được trả về bởi phương thức InetAddress.getLocalHost()

Graphical user interface, text, application, Word

Description automatically generated

Hình 2.16 Ví dụ về getInetAddress.

### 5.3.2 public int getLocalPort( )

Các contructor ServerSocket cho phép ta nghe dữ liệu trên cống không định trước bằng cách gán số 0 cho cống. Phương thức này cho phép ta tìm ra cống mà server đang nghe.

Text

Description automatically generated

Hình 2.17: Ví dụ về getLocalPort

### 5.4 Socket Options

Các socket option này có thể ảnh hưởng đến hành vi của Sockets layer, Transport layer, và IP layer. Việc sử dụng các socket option phổ biến nhất là tăng hiệu suất của kết nối, nhưng chúng cũng được sử dụng rộng rãi trong các tình huống khác. Chúng ta xem xét rất nhiều socket option có sẵn và cung cấp các ví dụ cho mỗi socket option.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Vị trí** | **Tên Option** | **Kiểu dữ liệu** | **Get /Set** | **Mô tả Option** |
| **SOL\_ SOCKET** | SO\_ACCEPTCONN | BOOL | Get | Socket có lắng nghe không? |
|  | SO\_BROADCAST | BOOL | Get/Set | Cho phép phát tin nhắn  broadcast trên socket. |
|  | SO\_DONTLINGER | BOOL | Get/Set | Cho phép hoặc vô hiệu hóa trả về ngay lập tức từ  closeSocket () |
|  | SO\_KEEPALIVE | BOOL | Get/Set | Gửi tin nhắn keep-alive |
|  | SO\_LINGER | struct  linger | Get/Set | Cho phép hoặc vô hiệu hóa  trả về ngay lập tức từ  closeSocket () |
|  | SO\_OOBINLINE | BOOL | Get/Set | Out-of-band data nằm trong data stream thông thường |
|  | SO\_REUSEADDR | BOOL | Get/Set | Cho phép hoặc vô hiệu hóa  việc tái sử dụng một socket  bị ràng buộc |
|  | SO\_SECURE | DWORD | Get/Set | Cho phép hoặc vô hiệu hóa  mã hóa SSL trên socket |
|  | SO\_SNDBUF | Int | Get/Set | Kích thước của bộ đệm  được phân bổ để gửi dữ liệu |
|  | SO\_TYPE | Int | Get | Kiểu socket |
| **IPPROTO TCP** | TCP\_NODELAY | BOOL | Get/Set | Bật/Tắt thuật toán NAGLE |
| **IPPROTO IP** | IP\_MULTICAST\_TTL | Int | Get/Set | Thời gian để duy trì một  gói phát đa hướng |
|  | IP\_MULTICAST\_IF | unsigned long | Get/Set | Địa chỉ giao diện phát đa  hướng đi |
|  | IP\_ADD\_MEMBERS  HIP | struct ip\_mreg | Get/Set | Thêm socket vào một  nhóm phát đa hướng |
|  | IP\_DROP\_MEMBERS  HIP | struct | Get/Set | Loại bỏ socket từ một  nhóm phát đa hướng |

Bảng 2.1: Socket Option.

### 5.5 The Object Methods

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên phương thức** | **Mô tả** |
| 1 | public Socket accept() throws IOException | Phương thức này chặn cho đến khi  hoặc một client kết nối với server  trên cổng được chỉ định hoặc socket  time-out giả định rằng giá trị time-  out được thiết lập theo phương thức  setSoTimeout(). Nếu không thì  phương thức này chặn vô thời hạn |
| 2 | public InetAddress getInetAddress() | Phương thức này trả về địa chỉ của  máy tính khác mà socket này được  kết nối. |
| 3 | public int getPort() | Trả về cổng socket được liên kết  với máy từ xa |
| 4 | public int getLocalPort() | Trả về cổng socket được liên kết  với máy cục bộ. |
| 5 | Public SocketAddressgetRemoteSocketAddress() | Trả về địa chỉ của socket từ xa. |
| 6 | public InputStream getInputStream()  throws IOException | Trả về input stream của socket.  Input stream được kết nối với  output stream của socket từ xa. |
| 7 | public OutputStream getOutputStream() throws IOException | Trả về output stream của socket.  Output stream được kết nối với  luồng đầu vào của socket từ xa. |
| 8 | public void close() throws IOException | Đóng socket, giúp socket object này  không còn có khả năng kết nối lại  với bất kỳ server nào. |
| 9 | public void setSocketTimeot(int ,timeout) | Đặt giá trị time-out cho server  socket trong bao lâu để chờ client  accept(). |

Bảng 2.2 Object Method.

### 5.6 Implementation

Bước 1: Tạo một đối tượng ServerSocket  
Timeline

Description automatically generated with low confidence

Hình 2.18: Tạo đối tượng Socket.

Bước 2: Tạo một đối tượng Socket bằng cách chấp nhận liên kết từ yêu cầu liên kết của client. Sau khi chấp nhận liên kết, phương thức accept() trả về đối tượng Socket thể hiện liên kết giữa Client và Server.

Graphical user interface, text, application, Word

Description automatically generated

Hình 2.19: Chấp nhận liên kết từ yêu cầu liên kết của client.

Bước 3: Tạo một luồng nhập để đọc dữ liệu từ client.



Hình 2.20 Tạo luồng nhập đọc dữ liệu.

Bước 4: Tạo một luồng xuất để gửi dữ liệu trở lại cho server.

Graphical user interface, text

Description automatically generated

Hình 2.21 Tạo luồng xuất gửi dữ liệu.

Bước 5: Thực hiện các thao tác vào ra với các luồng nhập và luồng xuất.

Graphical user interface, text, application, email

Description automatically generatedHình 2.22 Các thao tác.

Bước 6: Đóng socket sau khi đã truyền tin xong. Việc đóng socket cũng đồng nghĩa với việc đóng các luồng

Graphical user interface, text, application

Description automatically generated

Hình 2.23 Đóng socket

# **CHƯƠNG III: KẾT QUẢ THỰC NGHIỆM**

## 1.Kịch bản thực hiện

### 1.1 Giới thiệu chương trình

Chương trình quản lý hệ thống sinh viên là một ứng dụng phần mềm cho phép các trường đại học, cao đẳng, trung học chuyên nghiệp và các trung tâm đào tạo khác quản lý thông tin liên quan đến sinh viên. Chương trình giúp cho việc quản lý các thông tin liên quan đến sinh viên, như thông tin cá nhân, học phí, học tập, điểm số và các hoạt động khác của sinh viên, trở nên dễ dàng hơn

### 1.2 Cách thức hoạt động

Để thực hiện chương trình, đầu tiên cần kết nối máy với server khi đó người dùng mới có thể đăng nhập vào hệ thống.

Server và Client được thực hiện kết nối với nhau. Server sẽ nhận thông tin từ Client, sau đó tiến hành xử lý thông tin trả về Client.

Client:

+Nhập vào tên địa chỉ, cổng kết nối với Server, nếu không thành công thì thông báo nhập lại còn thành công thì kết nối.

+Sau đó vào giao diện cho nhập thông tin gồm họ tên, lớp,ID,giới tính,…cần lưu gửi lên Server.

+ Nhập kết quả Server trả về và hiển thị lên giao diện.

Server:

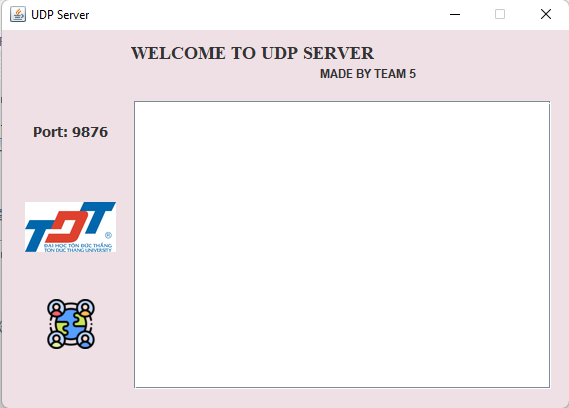
+ Đọc nội dung từ Client gửi lên, kiểm tra file có tồn tại không nếu không thì tạo file, mã hóa dữ liệu theo thuật toán Vigenere trước khi vào file.

+ Sau đó đọc dữ liệu từ file và giải mã với key được nhập trên giao diện Server, tìm các chữ “key” trong file vừa lưu, gửi các vị trí xuất hiện từ “key” tìm được về cho Client và hiển thị lên giao diện Server.

+ Lưu trữ dữ liệu trên CSDL trong quá trình Client gửi lên.

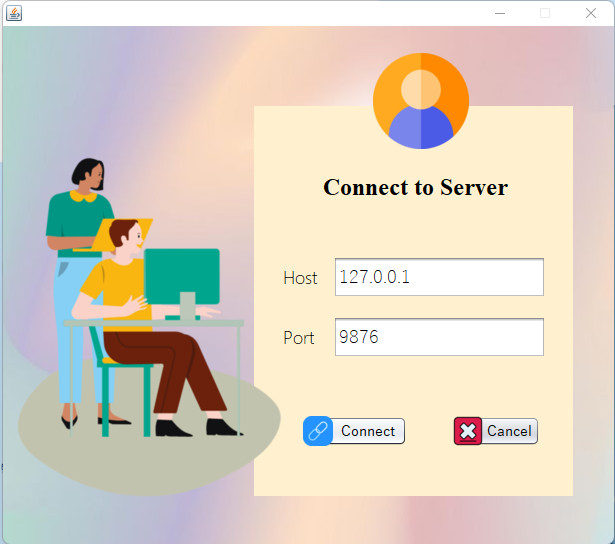
### 1.3 Giao diện chương trình

-Giao diện Server



Hình 3.1: Giao diện Server

-Giao diện Client



Hình 3.2: Giao diện Client

Graphical user interface

Description automatically generated

Hình 3.3: Giao diện Client

## 2. Yều cầu thực nghiệm

Yêu cầu thực nghiệm trong hệ thống quản lý sinh viên có thể bao gồm những yêu cầu sau:

-Độ chính xác: Hệ thống phải đảm bảo tính chính xác cao trong việc lưu trữ và xử lý thông tin sinh viên, bao gồm thông tin về học tập, điểm số, tiến độ học tập, thời khóa biểu, v.v.

-Tính bảo mật: Hệ thống phải đảm bảo tính bảo mật cao cho thông tin sinh viên, bảo vệ thông tin cá nhân và thông tin học tập của sinh viên khỏi các mối đe dọa an ninh mạng.

-Tính khả chuyển: Hệ thống phải được thiết kế để dễ dàng triển khai và sử dụng trên nhiều nền tảng và thiết bị khác nhau, từ máy tính để bàn đến thiết bị di động.

-Tính thân thiện với người dùng: Hệ thống phải đảm bảo tính thân thiện với người dùng, dễ sử dụng và có giao diện trực quan, giúp người dùng dễ dàng truy cập thông tin và thực hiện các chức năng quản lý sinh viên.

-Tính linh hoạt: Hệ thống phải có tính linh hoạt cao để có thể mở rộng và thích nghi với các yêu cầu và nhu cầu của người dùng trong tương lai.

-Tính đáp ứng: Hệ thống phải đảm bảo tính đáp ứng cao, có khả năng xử lý một lượng lớn thông tin và truy cập nhanh chóng và chính xác.

-Tính ổn định: Hệ thống phải đảm bảo tính ổn định cao, đáp ứng được một lượng lớn người dùng và các yêu cầu của họ mà không bị gián đoạn hoặc chậm trễ.

## 3.Công cụ thực hiện

- Ngôn ngữ lập trình: Để viết phần mềm client và server, chúng em sử dụng ngôn ngữ lập trình Java

- Cơ sở dữ liệu: Để lưu trữ thông tin sinh viên, chúng em sử dụng Mysql

-Trình web server: Để triển khai phần mềm server, chúng em dùng Xampp triển khai chương trình

-Công cụ phát triển: Netbeans IDE

## 4.Các bước thực hiện

Diagram

Description automatically generated

Hình 3.4: Sơ đồ thực hiện chương trình

Bước 1: Client kết nối Server bằng cách nhập địa chỉ (host) và cổng kết nối (port), nếu không thành công thì nhập lại.

Bước 2: Sau khi kết nối thành công, hệ thống sẽ hiện lên giao diện để người dùng nhập thông tin như họ tên, mssv, lớp,…

Bước 3: Khi nhập xong thông tin sinh viên, người dùng có thể chọn 1 trong 4 tính năng của hệ thống là thêm, xóa, tìm kiếm hoặc cập nhật thông tin sinh viên.

Bước 5: Server sẽ nhận thông tin từ phía Client và gửi về cơ sở dữ liệu.

Bước 6: Server nhận dữ liệu từ cơ sở dữ liệu và trả lại kết qua cho Client.

## 5.Kết luận

Để thiết kế hệ thống này, chúng tôi đã sử dụng giao thức UDP để truyền tải dữ liệu giữa client và server. Giao thức UDP được chọn vì tính đơn giản và hiệu quả của nó. Hệ thống được lập trình bằng ngôn ngữ Java và sử dụng socket để tạo kết nối giữa client và server.

# **CHƯƠNG IV: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

## 1.Kết luận

Trong báo cáo này, chúng tôi đã trình bày về hệ thống quản lý sinh viên được thiết kế và phát triển bằng giao thức UDP để quản lý thông tin sinh viên của một trường đại học. Hệ thống này cho phép quản lý thông tin cá nhân của sinh viên, thông tin khóa học, kết quả học tập, v.v.

Hệ thống hoạt động trên mô hình client-server, trong đó các máy khách (clients) được kết nối với máy chủ (server) để truy cập vào cơ sở dữ liệu sinh viên. Giải pháp này cho phép đáp ứng nhanh chóng các yêu cầu từ các máy khách và đồng thời đảm bảo an toàn và bảo mật cho dữ liệu của sinh viên.

Báo cáo cũng đề cập đến việc xây dựng giao diện đơn giản và thân thiện với người dùng, giúp người sử dụng dễ dàng truy cập và tìm kiếm thông tin sinh viên.

Tóm lại, hệ thống quản lý sinh viên trên mô hình client-server là một giải pháp hiệu quả và tiện ích cho việc quản lý thông tin sinh viên trong các tổ chức giáo dục

## 2.Hướng phát triển

**-** Cải thiện hiệu suất hệ thống: Tối ưu hóa hệ thống và tăng tốc độ truy cập đến cơ sở dữ liệu sinh viên, đặc biệt là khi có nhiều máy khách cùng truy cập vào cùng một lúc.

- Tăng cường tính bảo mật: Cập nhật hệ thống bảo mật, bao gồm cả cơ chế xác thực và phân quyền truy cập, để đảm bảo rằng chỉ các người dùng được ủy quyền mới có thể truy cập và sửa đổi thông tin sinh viên.

- Phát triển tính năng đa ngôn ngữ: Thêm tính năng hỗ trợ đa ngôn ngữ để hệ thống có thể được sử dụng ở nhiều quốc gia và với nhiều ngôn ngữ khác nhau.

- Cải thiện giao diện người dùng: Tăng tính thẩm mỹ và dễ sử dụng của giao diện người dùng, giúp người dùng có trải nghiệm tốt hơn khi sử dụng hệ thống.

# **TÀI LIỆU THAM KHẢO**

**TIẾNG VIỆT**

[1] [Client Server là gì? Tìm hiểu mô hình Client Server từ A - Z (fptcloud.com)](https://fptcloud.com/client-server/)

[2] [5 Phút Tìm Hiểu Về Mô Hình Client - Server (codelearn.io)](https://codelearn.io/sharing/tim-hieu-ve-mo-hinh-client-server)

[3] [Giao thức UDP là gì? UDP hoạt động như thế nào? - VinaHost](https://blog.vinahost.vn/giao-thuc-udp-la-gi)

[4] [UDP là gì? Các ứng dụng và tính năng phổ biến của UDP - Blog BizCloud](https://bizcloud.vn/blog/udp-la-gi-cac-ung-dung-va-tinh-nang-pho-bien-cua-udp.html)

[5] [UDP và TCP - Sự khác biệt của 2 giao thức internet phổ biến (vietnix.vn)](https://vietnix.vn/udp-va-tcp/)

[6] [Mật mã Vigenère – Wikipedia tiếng Việt](https://vi.wikipedia.org/wiki/M%E1%BA%ADt_m%C3%A3_Vigen%C3%A8re)

[7] [Bài 12: Mã hóa Vigenère - Blog Nam Phạm](https://nam.name.vn/bai-12-ma-hoa-vigenere.html)

[8] [Hướng dẫn mã hóa mật mã vigenère cipher và playfair cipher bằng tay (webaffiliatevn.com)](https://webaffiliatevn.com/huong-dan-ma-hoa-mat-ma-vigenere-cipher-va-playfair-cipher/)

[9] [Socket là gì? Kiến thức nền tảng về giao thức TCP/IP và UDP (fptcloud.com)](https://fptcloud.com/socket-la-gi/#:~:text=Socket%20%C4%91ang%20l%C3%A0%20m%E1%BB%99t%20giao,tr%C3%B2%20chuy%E1%BB%87n%20c%C3%B9ng%20v%E1%BB%9Bi%20nhau.)

[10] [Socket là gì? - Tìm hiểu khái niệm các loại socket phổ biến hiện nay (teky.edu.vn)](https://teky.edu.vn/blog/socket-la-gi/)

[11] [Java - Sockets (viblo.asia)](https://viblo.asia/p/java-sockets-WAyK8x7kKxX)

[12] [Lập trình Socket trong Java - Nâng cấp kỹ năng lập trình Java của bạn (t3h.com.vn)](https://t3h.com.vn/tin-tuc/lap-trinh-socket-trong-java)