BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á

KHOA: CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN

HỌC PHẦN: LẬP TRÌNH MẠNG

ĐỀ TÀI 006

ỨNG DỤNG CHAT GIỮA HAI MÁY UDP WINDOWSFORM

Sinh viên thực hiện	Lớp	Khóa
Hà Tiến Dũng	DCCNTT12.10.12	K12
Đinh Xuân Hiếu	DCCNTT12.10.12	K12
Nguyễn Văn Đạt	DCCNTT12.10.12	K12
Vũ Thanh Hải	DCCNTT12.10.12	K12

Bắc Ninh, năm 2023

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á

KHOA: CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN

HỌC PHẦN: LẬP TRÌNH MẠNG

ĐỀ TÀI 006

ỨNG DỤNG CHAT GIỮA HAI MÁY UDP WINDOWSFORM

STT	Sinh viên thực hiện	Mã sinh viên	Điểm bằng số	Điểm bằng chữ
1	Hà Tiến Dũng	20213409		
2	Đinh Xuân Hiếu	20213331		
3	Nguyễn Văn Đạt	20213571		
4	Vũ Thanh Hải	20213345		

CÁN BỘ CHẤM 1

(Ký và ghi rõ họ tên)

CÁN BỘ CHẤM 2

(Ký và ghi rõ họ tên)

MỤC LỤC

DANH MỤC CÁC TỪ VIẾT TẮT 1
DANH MỤC BẢNG BIỂU, SƠ ĐỒ 2
DANH MỤC HÌNH ẢNH 3
MỞ ĐẦU 4
CHƯƠNG 1 – CƠ SỞ LÝ THUYẾT 5
1.1 Tổng quản lập trình mạng 5
1.2 Phương pháp lập trình mạng 7
1.3 Phân tích yêu cầu đề tài 13
CHƯƠNG 2 – THIẾT KẾ CHƯƠNG TRÌNH ỨNG DỤNG CHAT GIỮA HAI MÁY UDP WINDOWS FORM 15
2.1 Sơ đồ kiến trúc tổng quát bài toán 15
2.2 Các chức năng 15
2.3 Thiết kế dữ liệu 16
CHƯƠNG 3 – CÀI ĐẶT VÀ THỬ NGHIỆM CHƯƠNG TRÌNH 18
3.1 Yêu cầu cài đặt 18
3.2 Các bước cài đặt và kết quả 18
3.3 Thử nghiệm chương trình 25
Hạn chế và hướng phát triển của đề tài. 34
Tài liệu tham khảo 34

DANH MỤC CÁC TỪ VIẾT TẮT

Chữ viết tắt	Giải thích
LAN	Local Area Network
MAN	Metropolitan
WAN	Wide Area Network
PAN	Personal Area Network
VPN	Vitual Private Network
SAN	Storage – Area Network
TCP/IP	Transmission Control Protocol / Internet Protocol
UDP	User Datagram Protocol
HTTP	HyperText Transfer Protocol

DANH MỤC BẢNG BIỂU, SƠ ĐỒ

Số hiệu	Tên	Trang
0	Bảng phần công công việc.	4
2	Sơ đồ kiến trúc tổng quát bài toán.	15
2.1	Sơ đồ hoạt động của chức năng chat của ứng dụng	16
3	Bảng thành phần của giao diện ứng dụng	18

DANH MỤC HÌNH ẢNH

Số hiệu	Tên	Trang
1	định dạng cấu trúc header của giao thức UDP	12
3.1	Giao diện ứng dụng chat	19
4	Mở ứng dụng ở máy chính	25
4.1	Mở ứng dụng ở máy ảo	25
5	Kiểm tra địa chỉ ip ở máy ảo	26
5.1	Kiểm tra địa chỉ ip ở máy chính	26
6	Nhập các thông số trên vào ứng dụng ở máy chính	27
6.1	Nhập các thông số trên vào ứng dụng ở máy ảo	27
7	Tin nhắn hiện lên khung chat sau khi gửi ở ứng dụng của máy chính	28
7.1	Tin nhắn hiện lên khung chat ở ứng dụng máy ảo khi nhận tin nhắn từ ứng dụng của máy chính	28
7.2	Tin nhắn hiện lên khung chat sau khi gửi ở ứng dụng của máy ảo	29
7.3	ở khung chat ứng dụng ở máy chính cũng sẽ hiện tin nhắn khi nhận được tin nhắn ở ứng dụng máy ảo	29
8	Soạn tin nhắn và click nút gửi của ứng dụng ở máy chính	30
8.1	Tin nhắn sau khi gửi sẽ hiện lên khung chat	30
8.2	ở khung chat ứng dụng ở máy ảo cũng sẽ hiện tin nhắn khi nhận được tin nhắn ở ứng dụng máy chính	31
9	Soạn tin nhắn và click nút gửi của ứng dụng ở máy ảo	31
9.1	Tin nhắn sau khi gửi sẽ hiện lên khung chat	32
9.2	ở khung chat ứng dụng ở máy chính cũng sẽ hiện tin nhắn khi nhận được tin nhắn ở ứng dụng máy ảo	32
10	Ấn nút Disconnect của ứng dụng ở máy chính	33
10.1	ở khung chat ứng dụng ở máy ảo sẽ nhận được thông báo sau khi ứng dụng ở máy chủ ấn Disconnect	33

MỞ ĐẦU

- LÝ DO CHỌN ĐỀ TÀI

Hiện nay, mạng Lan đã có những tiến bộ vượt bậc và ngày càng phổ biến hơn trong đời sống sinh hoat. Điều này làm cho nhu cầu liên lạc và trao đổi thông tin thông qua mạng Lan ngày càng lớn hơn. Chính vì vậy, chương trình Chat trên mạng Lan được xây dựng để đáp ứng phần nào những nhu cầu cấp thiết đó. Chính vì vậy nhóm em đã sẽ nghiên cứu và lập trình ứng dụng chat giữa hai máy qua mạng lan sử dụng giao thức UDP. Chương trình được xây dựng với khả năng gửi các đoạn văn bản qua lại giữa hai máy trong mạng lan.

- MỤC TIÊU CỦA BÀI TOÁN

Xây dựng chương trình chat giữa hai máy bằng mạng lan sử dụng giao thức UDP

Áp dụng cho mô hình Client – Sever

Sử dụng lớp hỗ trợ UdpClient

Sử dụng ngôn ngữ C# và giao diện WindowForm

- KẾT QUẢ ĐẠT ĐƯỢC CỦA BÀI TOÁN

Cài đặt và thử nghiệm thành công ứng dụng chat giữa hai máy qua mạng lan bằng giao thức udp.

Bảng 0: Bảng phần công công việc.

Công việc phân công		Thành viên
Chương 1	**Tổng quan lập trình mạng**	**Cả nhóm cùng tham gia**
	Phương pháp kỹ thuật lập mạng	**Hà Tiến Dũng, Nguyễn Văn Đạt**
	Phân tích yêu cầu đề tài	**Hà Tiến Dũng, Nguyễn Văn Đạt, Vũ Thanh Hải**
Chương 2	**Thiết kế sơ đồ kiến trúc tổng quát của bài toán**	**Nguyễn Văn Đạt, Hà Tiến Dũng, Đinh Xuân Hiếu**
	Mô tả các chức năng, và sơ đồ hoạt động	**Vũ Thanh Hải, Đinh Xuân Hiếu, Hà Tiến Dũng**
Chương 3	**Trình bày các yêu cầu cài đặt**	**Đinh Xuân Hiếu, Nguyễn Văn Đạt**
	Cài đặt và thử nghiệm chương trình	**Hà Tiến Dũng**

CHƯƠNG 1 – CƠ SỞ LÝ THUYẾT

1.1 Tổng quản lập trình mạng

- Lập trình mạng (Network Programing) Sử dụng ngôn ngữ lập trình để xây dựng chương trình mà máy tính có thể giao tiếp với nhau thông qua các dịch vụ mạng và truyền thông nhằm phát triển các ứng dụng doanh nghiệp hoạt động trên internet.

- Lập trình mạng được xây dựng dựa trên công thức: Lập trình mạng = Kiến thức mạng truyền thông + Mô hình lập trình mạng + Ngôn ngữ lập trình mạng. Các kiến thức cơ bản cần nắm rõ khi học lập trình mạng bao gồm: kiến thức về mạng và lập trình mạng cũng ngư ngôn ngữ lập trình. Hiện nay khi học lập trình mạng, các nhà lập trình thường sử dụng ngôn ngữ lập trình .NET

- Các mô hình lập trình mạng bao gồm:

- Mô hình Client - Server: là mô hình phổ biến nhất trong lập trình mạng. Trong mô hình này, có hai thành phần chính là Client và Server. Client là người dùng sử dụng ứng dụng và Server là máy chủ cung cấp dịch vụ cho Client.
- Mô hình Peer - to – Peer: là mô hình trong đó các máy tính kết nối với nhau để chia sẽ tài nguyên và dữ liệu. Trong mô hình này, không có máy chủ trung tâm nào và các máy tính được coi là bằng nhau.
- Mô hình Publish – Subscribe: là mô hình trong đó các ứng dụng đăng ký để nhận thông tin từ các ứng dụng khác. Các ứng dụng khác sẽ gửi thông tin đến các ứng dụng đã đăng ký.

- Lĩnh vực ứng dụng của lập trình mạng: Lập trình mạng có rất nhiều ứng dụng trong nhiều lĩnh vực khác nhau như:

- Công nghệ thông tin
- Khoa học máy tính
- Điện tử viễn thông
- Truyền thông và quảng cáo
- Tài chính và ngân hàng
- Y tế và y khoa

- Trong các lĩnh vực này lập trình mạng được sử dụng để phát triển các ứng dụng như:

- Ứng dụng web
- Ứng dụng di động
- Hệ thống quản lý cơ sở dữ liệu
- Hệ thống quản lý tài khoản ngân hàng
- Hệ thống quản lý bệnh viện và y tế

- Với sự phát triển của công nghệ thông tin, lập trình mạng đang trở thành một trong những lĩnh vực có nhu cầu tuyển dụng cao nhất.

- **Kiến thức mạng truyền thông :**

- Mạng máy tính : là những sợi liên kết của máy tính với nhau, qua các thiết bị kết nối, dựa vào cấu trúc tại môi trường truyền dẫn.
- Mạng Lan: Hoạt động với giao thức TCP/IP, phủ trong diện tích nhỏ, hay còn được biết là mạng cục bộ.
- Mạng Man: Khác với mạng Lan, phạm vi của mạng Man sẽ rộng lớn hơn, ví dụ như một thành phố, một doanh nghiệp. Nó được hình thành từ nhiều mạng Lan liên kết với nhau.
- Mạng Wan: Tương tự như mạng Lan nhưng phạm vi hoạt động rộng lớn hơn.
- Mạng Pan: Là mạng cá nhân, dùng để thực hiện truyền dữ liệu ở những thiết bị đơn như máy tính, điện thoại,…
- VPN: Kết nối internet được mã hóa giữa thiết bị người đùng và mạng. Kết nối được mã hóa giúp đảm bảo rằng dữ liệu nhạt cảm được truyền đi một cách an toàn. Nó ngăn những người không được phép truy cập vào lưu lượng truy cập.
- SAN: Mạng tốc độ cao chuyên dụng giúp các thiết bị lưu trữ có thể truy cập vào máy chủ bằng cách gắn bộ lưu trữ trực tiếp vào hệ điều hành.

- **Giao thức và dịch vụ mạng truyền thông:**

- TCP (Transmission Control Protocol): thiết lập kết nối giữa các máy tính để truyền dữ liệu. Nó chia nhỏ dữ liệu ra thành những gói (packet) và đảm bảo việc truyền dữ liệu thành công.
- IP (*Internet Protocol*): định tuyến (*route*) các gói dữ liệu khi chúng được truyền qua Internet, đảm bảo dữ liệu sẽ đến đúng nơi cần nhận.
- HTTP (*HyperText Transfer Protocol*): cho phép trao đổi thông tin (chủ yếu ở dạng siêu văn bản) qua Internet.
- FTP (*File Transfer Protocol*): cho phép trao đổi tập tin qua Internet.
- HTTPS: Là giao thức được đánh giá có khả năng an toàn và bảo mật dữ liệu tốt nhất hiện nay. Giao thức sẽ bảo vệ thông tin cá nhân tối ưu hơn khi truyền dữ liệu cá nhân đến các server.
- SNMP – Simple Network Management Protocol: Đây là một khung cơ bản được sử dụng để quản lý các thiết bị trên internet bằng cách sử dụng giao thức TCP/IP.
- DNS- Domain Name System: Hệ thống phân giải tên miền. Một địa chỉ IP được sử dụng để xác định duy nhất kết nối của một dịch vụ hosting với

internet. Tuy nhiên, người dùng thích sử dụng tên được cấu tạo từ chữ cái thay vì địa chỉ IP là các chữ số cho DNS đó.

- TELNET – Terminal Network: Giao thức này thiết lập kết nối giữa máy tính cục bộ và máy tính từ xa.
- SMTP (*Simple Mail Transfer Protocol*): cho phép gởi các thông điệp thư điện tử (*e-mail*) qua Internet.
- POP3 (*Post Office Protocol*, phiên bản 3): cho phép nhận các thông điệp thư điện tử qua Internet.
- MIME (*Multipurpose Internet Mail Extension*): một mở rộng của giao thức SMTP, cho phép gởi kèm các tập tin nhị phân, phim, nhạc,... theo thư điện tử.
- WAP (*Wireless Application Protocol*): cho phép trao đổi thông tin giữa các thiết bị không dây, như điện thoại di động.

1.2 Phương pháp lập trình mạng

- Các ngôn ngữ lập trình mạng phổ biến hiện nay là: Java, python, C, C++, C#,...

- Các ngôn ngữ lập trình Java, Python, C, C++, C# đều được sử dụng rộng rãi trong lập trình mạng. Mỗi ngôn ngữ có những ưu điểm và nhược điểm riêng.

- Java: được sử dụng rộng rãi trong lập trình mạng do tính bảo mật và khả năng xử lý đa luồng tốt, Tuy nhiên, java có thể chậm hơn so với các ngôn ngữ khác.
- Python: được sử dụng rộng rãi trong lập trình mạng do cú pháp đơn giản và dễ hiểu. Tuy nhiên python có thể chậm hơn so với các ngôn ngữ khác.
- C: được sử dụng rộng rãi trong lập trình mạng do tốc độ xử lý nhanh và hiệu suất cao. Tuy nhiên, C có cú pháp phức tạp hơn so với các ngôn ngữ khác.
- C++: được sử dụng rộng rãi trong lập trình mạng do tính bảo mật cao, tốc độ nhanh và khả năng xử lý đa luồng tốt. Tuy nhiên C++ có cú pháp phực tạp hơn so với những ngôn ngữ khác
- C# được sử dụng rộng rãi trong lập trình mạng do tính bảo mật cao và khả năng xử lý đa luồng tốt. Tuy nhiên, C# có thể chậm hơn so với các ngôn ngữ khác.

- Ở đề tài này chúng em chọn .Net của Microsoft vì nền tảng.NET có một số đặc điểm nổi bật như: Có thể sử dụng được nhiều ngôn ngữ và nhiều nền tảng. Khi sử dụng.NET có thể giảm thiểu đáng kể xung đột, tính bảo mật cao và khá an toàn. Hiệu suất tăng, chi phí sử dụng giảm và cũng vì hiện tại chúng em cũng đang học môn lập trình .Net việc sử dụng .Net sẽ giúp chúng em có kỹ năng về lập trình mạng hơn trong lập trình .Net.

- Các lớp trong .Net chúng em sử dụng ở đề tài này bao gồm:

- Lớp ứng dụng: Class Chat_UDP;
  - Là class của giao diện được thiết kế. Trong lớp này chứa code lập trình cho các button như button mở kết nối, button gửi,...
- Lớp thư viện:
  - using System; : thư viện cung cấp các lớp cơ bản để xử lý các tác vụ thông dụng như chuỗi, số, ngày, giờ và đối tượng.
  - using System.Collections.Generic; thư viện cung cấp các lớp để quản lý các tập hợp đối tượng.
  - using System.ComponentModel; : thư viện cung cấp các lớp để hỗ trợ việc phát triển các thành phần trong ứng dụng.
  - using System.Data; : thư viện cung cấp các lớp để xử lý dữ liệu và truy cập cơ sở dữ liệu.
  - using System.Drawing;: thư viện cung cấp các lớp để xử lý đồ họa và hình ảnh.
  - using System.Linq; : thư viên cung cấp các lớp và phương thức để thực hiện các hoạt động trên các tập hợp đối tượng.
  - using System.Net; : là một namespace trong .NET Framework, cung cấp các lớp và phương thức để thực hiện các hoạt động liên quan đến mạng, chẳng hạn như kết nối đến các máy chủ, truyền dữ liệu qua mạng, xử lý các giao thức mạng như HTTP, FTP, SMTP, POP3, và nhiều hơn nữa.

    + Các tác dụng của using System.Net bao gồm:

    1. Kết nối đến các máy chủ: System.Net cung cấp các lớp để thiết lập kết nối đến các máy chủ, chẳng hạn như lớp WebClient, lớp HttpWebRequest và lớp FtpWebRequest.

    2. Truyền dữ liệu qua mạng: System.Net cung cấp các lớp để truyền dữ liệu qua mạng, chẳng hạn như lớp WebClient, lớp HttpWebRequest và lớp FtpWebRequest.

    3. Xử lý các giao thức mạng: System.Net cung cấp các lớp để xử lý các giao thức mạng như HTTP, FTP, SMTP, POP3, và nhiều hơn nữa.

    4. Xác thực và quản lý phiên: System.Net cung cấp các lớp để xác thực và quản lý phiên, chẳng hạn như lớp CredentialCache và lớp CookieContainer.

    5. Bảo mật: System.Net cung cấp các lớp để bảo mật các kết nối mạng, chẳng hạn như lớp SslStream và lớp SecurityProtocolType.

+ System.Net được sử dụng để thực hiện các tác vụ liên quan đến mạng, chẳng hạn như:

1. Tải và tải lên tệp tin: Sử dụng lớp WebClient để tải và tải lên tệp tin từ một URL.

2. Gửi yêu cầu HTTP: Sử dụng lớp HttpWebRequest để gửi yêu cầu HTTP đến một máy chủ và nhận phản hồi.

3. Gửi email: Sử dụng lớp SmtpClient để gửi email từ ứng dụng của bạn.

4. Tải và tải lên tệp tin FTP: Sử dụng lớp FtpWebRequest để tải và tải lên tệp tin từ một máy chủ FTP.

5. Xử lý các giao thức mạng khác: Sử dụng các lớp như TcpClient, UdpClient và Socket để xử lý các giao thức mạng khác.

6. Xác thực và quản lý phiên: Sử dụng lớp CredentialCache để lưu trữ thông tin xác thực và lớp CookieContainer để quản lý các cookie.

7. Bảo mật: Sử dụng lớp SslStream để bảo mật các kết nối mạng và lớp SecurityProtocolType để xác định các giao thức bảo mật được sử dụng.

- using System.Net.Sockets; : là một namespace trong .NET Framework cung cấp các lớp để tạo và quản lý các kết nối mạng. Các lớp này cho phép ứng dụng tạo và quản lý các kết nối TCP/IP và UDP/IP.

  + Các lớp chính trong namespace System.Net.Sockets bao gồm:

  1. Socket: Lớp này đại diện cho một kết nối mạng. Nó cung cấp các phương thức để tạo, kết nối

  Bạn đã gửi, gửi và nhận dữ liệu trên kết nối mạng.

  2. TcpClient và TcpListener: Các lớp này cung cấp các phương thức để tạo và quản lý kết nối TCP/IP.

  3. UdpClient: Lớp này cung cấp các phương thức để tạo và quản lý kết nối UDP/IP.

  + System.Net.Sockets được sử dụng để tạo và quản lý các kết nối mạng. Các tác dụng của nó trong C# bao gồm:

  1. Tạo và quản lý kết nối mạng: System.Net.Sockets cung cấp các lớp và phương thức để tạo và quản lý các kết nối mạng, bao gồm TCP/IP và UDP/IP.

  2. Gửi và nhận dữ liệu trên kết nối mạng: System.Net.Sockets cung cấp các phương thức để gửi và nhận dữ liệu trên các kết nối mạng, bao gồm

các phương thức cho phép gửi và nhận dữ liệu theo các giao thức khác nhau như HTTP, FTP, SMTP, POP3, và IMAP.

3. Xử lý các sự kiện mạng: System.Net.Sockets cung cấp các lớp và phương thức để xử lý các sự kiện mạng, bao gồm các sự kiện như kết nối, ngắt kết nối, gửi và nhận dữ liệu.

4. Tích hợp với các ứng dụng mạng: System.Net.Sockets được sử dụng để tích hợp các ứng dụng mạng với các dịch vụ mạng khác nhau, bao gồm các dịch vụ web, dịch vụ email, và các dịch vụ khác.

- using System.Text; : thư viện cung cấp các lớp để xử lý chuỗi, bao gồm mã hóa, giải mã và định dạng chuỗi.
- using System.Threading.Tasks; : thư viện được hỗ trợ để lập trình đa luồng. Trong C#, không có module threading như trong Python. Tuy nhiên, C# cung cấp các công cụ để tạo và quản lý các luồng (threads) trong một chương trình.

  + Các tác dụng của thearding Task trong C# bao gồm:

  1. Tạo các công việc (tasks) để thực hiện các tác vụ đồng thời trong một luồng.

  2. Quản lý các công việc (tasks) bằng cách sử dụng các phương thức như Start(), Cancel(), Wait(), Result,...

  3. Đồng bộ hóa các công việc (tasks) bằng cách sử dụng các phương thức như ContinueWith(), WhenAll(), WhenAny(),...

  4. Xử lý các ngoại lệ (exceptions) trong các công việc (tasks) bằng cách sử dụng các phương thức như Exception, ContinueWith(),..
- using System.Windows.Forms; : là một phần của .NET Framework và cung cấp các lớp và điều khiển để tạo giao diện người dùng cho các ứng dụng Windows. Thư viện này bao gồm các lớp để tạo các điều khiển như nút, hộp văn bản, danh sách, bảng điều khiển và các lớp để quản lý các sự kiện và xử lý các tương tác người dùng.

  + Với thư viện System.Windows.Forms, bạn có thể tạo các ứng dụng Windows với các tính năng như:

  1. Tạo các điều khiển người dùng như nút, hộp văn bản, danh sách, bảng điều khiển và các menu.

  2. Xử lý các sự kiện và tương tác người dùng.

  3. Tạo các đối tượng đồ họa như hình ảnh và màu sắc.

  4. Tạo các ứng dụng đa nhiệm với nhiều cửa sổ và các tính năng khác.

- Các lớp hỗ trợ trong code đề tài:
    - UdpClient: lớp này hỗ trợ lập trình với Socket Udp. Lớp này được xây dựng trên lớp socket nhưng bị che bớt một số chi tiết của lớp socket và đơn giảng hóa việc gửi nhận dữ liệu.
    - Thread: được sử dụng để tạo và điều khiển các luồng trong một ứng dụng đa luồng. Bạn có thể sử dụng phương thức Thread.Start() để bắt đầu một luồng mới.
    - IPAddress: được sử dụng để đại diện cho một địa chỉ IP. Bạn có thể sử dụng phương thức IPAddress.Parse() để chuyển đổi một chuỗi biểu diễn của địa chỉ IP thành một đối tượng.
    - IPEndPoint: lớp này được sử dụng để đại diện cho một địa chỉ IP và một cổng kết nối.

*** Mô hình: TCP, UDP, HTTP.**

- Mô hình TCP/IP được sử dụng trong tất cả các ứng dụng mạng, mô hình này được xây dựng dựa trên 2 giao thức chính là giao thức TCP (Transmission Control Protocol) và giao thức IP (Internet Protocol). Trong đó:

- TCP : là một giao thức mạng dùng trong việc truyền dữ liệu qua một mạng khác, được đánh giá là rất quan trọng. Giao thức bao gồm những quy tắc và thứ tự quản lý việc truyền dữ liệu sao cho người dùng trên toàn thế giới dù ở đâu, trên nền tảng gì, phần mềm nào cũng đều được quyền thao tác theo cùng một cách thức tương tự nhau. Nó đảm bảo rằng dữ liệu sẽ được truyền tải đúng đích và đầy đủ, đồng thời kiểm soát được việc truyền tải dữ liệu trong trường hợp có sự cố.
- IP : là địa chỉ số có trên mọi thiết bị kết nối mạng để chia sẻ dữ liệu với nhau giao thức kết nối Internet. IP giữ vai trò gán địa chỉ thực hiện đưa những gói tin từ nguồn đến đích một cách chính xác và hiệu quả.

- UDP: là một giao thức cốt lõi của giao thức TCP/IP được sử dụng để thiết lập các kết nối có độ trễ thấp và giảm mất mát giữa các ứng dụng trên Internet. Giao thức UDP cung cấp hai dịch vụ không được cung cấp bởi lớp IP, đó là:

1. Cung cấp các port number để giúp phân biệt các yêu cầu khác nhau từ người dùng.
2. Đồng thời sử dụng một thuật toán checksum để xem dữ liệu có được toàn vẹn hay không.

+ UDP là một giải pháp được dùng thay cho giao thức TCP. Cả hai giao thức này đều hoạt động trên lớp IP và có lúc được gọi là UDP/IP hoặc TCP/IP. Mặc dù đều hoạt động trên giao thức IP, nhưng UDP và TCP có những khác biệt rất quan trọng. Cụ thể:

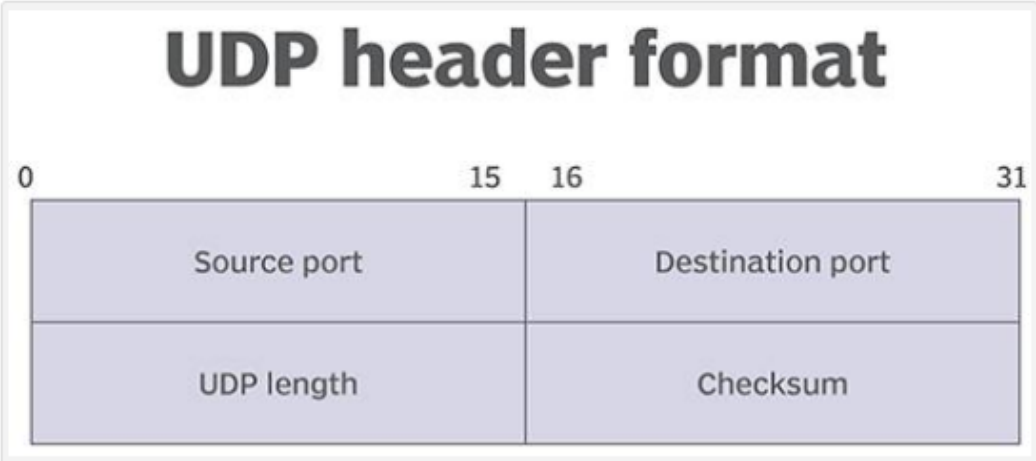
1. Giao thức TCP hỗ trợ giao tiếp máy chủ tới máy chủ (host-to-host). Được coi là phương thức vận chuyển bảo mật đáng tin cậy nhưng có tốc độ truyền chậm hơn. TCP chia nhỏ các tập dữ liệu lớn thành các gói dữ liệu riêng lẻ, đảm bảo lưu lượng và kiểm soát lỗi, thực hiện gửi lại các tập tin bị mất và gửi theo đúng thứ tự.
2. Giao thức UDP cho phép giao tiếp quy trình tới quy trình (process-to-process). UDP chỉ gửi gói tin được gọi là datagram, các gói tin này được truyền đi theo các đường dẫn khác nhau giữa người gửi và người nhận. Nghĩa là UDP có khả năng làm mất dữ liệu hoặc truyền dữ liệu không theo thứ tự, tuy nhiên lại sử dụng chi phí băng thông và độ trễ thấp hơn.

+ Các tính năng của giao thức UDP:

1. Cho phép các gói bị loại bỏ và nhận không theo thứ tự truyền đi ban đầu, điều này làm cho UDP phù hợp với các ứng dụng thời gian thực, trong đó ưu tiên tốc độ và độ trễ thấp.
2. Nó có thể được sử dụng cho các giao thức dựa trên giao dịch.
3. UDP có thể hữu ích khi có nhiều người truy cập, kết nối và khi không có nhu cầu điều chỉnh lỗi thời gian thực.

+ Cấu trúc header của giao thức UDP:

1. Source Port Number là cổng của người gửi.
2. Destination Port Number là cổng mà diagram được gửi đến.
3. UDP length là độ dài tính bằng byte của UDP header hoặc bất kỳ gói dữ liệu nào.
4. Checksum là thuật toán kiểm tra lỗi đảm bảo tính toàn vẹn của dữ liệu được sử dụng trong Ipv6 hoặc Ipv4.



Hình 1: định dạng cấu trúc header của giao thức UDP.

+ Giao thức UDP hoạt động như thế nào:

1. Giao thức UDP sử dụng IP để lấy datagram từ máy tính này sang máy tính khác. UDP hoạt động bằng cách thu thập dữ liệu trong gói UDP và thêm thông tin tiêu đề của riêng mình vào gói. Dữ liệu này bao gồm các port nguồn và đích để giao tiếp, độ dài gói và thuật toán kiểm tra checksum. Sau khi các gói UDP được gói gọn trong gói IP, chúng sẽ gửi đến các điểm đích được chỉ định.
2. Giao thức UDP sử dụng một mô hình truyền đơn giản không bao gồm các cuộc đối thoại bắt tay (handshaking dialogues) để cung cấp độ tin cậy, sắp đặt hoặc tính toàn vẹn dữ liệu. Do đó, dịch vụ của UDP không đáng tin cậy. Các gói có thể không xuất hiện, dữ liệu có thể thêm bản sao hoặc bị mất mát mà không có cảnh báo.

* Mặc dù phương pháp truyền dữ liệu của UDP không đảm bảo rằng dữ liệu được gửi sẽ đến đích, nhưng nó có chi phí thấp và phổ biến cho các dịch vụ. ứng dụng chấp nhận việc mất mát cho dịch vụ.

- HTTP : là giao thức truyền tải siêu văn bản. Đây là một giao thức ứng dụng được sử dụng thường xuyên nhất trong bộ các giao thức TCP/IP (gồm một nhóm các giao thức nền tảng cho internet). HTTP hoạt động dựa trên mô hình Client–Server. Các máy tính của người dùng sẽ đóng vai trò làm Client. Sau một thao tác nào đó của người dùng, các máy khách sẽ gửi yêu cầu đến Server và chờ đợi câu trả lời từ những máy chủ này(Sever).

1.3 Phân tích yêu cầu đề tài

1.Các yêu cầu cần thực hiện.

- Xây dựng chương trình Chat hoạt động theo mô hình Client-Server .
- Sử dụng lớp hỗ trợ UDP Client.
- Giao diện Windows form.
- Chat được trong mạng lan.

2.Phân tích yêu cầu.

- Các chức năng đưa ra:

+ CONNECT: dùng để kết nối 2 dối tượng với nhau khi nhập đầy đủ các thông tin của 2 đối tượng ở phần “PORT PC1, PORT PC2, NAME, IP CONNECT” thì có thể thực hiện kết nối giữa 2 đối tượng với nhau .

+ DISCONNECT: dùng để ngắt kết nối khi mà một trong 2 đối tượng đang liên kết muốn hủy liên kết để kết nối với một đối tượng khác bất kì.

+ SEND: dùng để gửi đi thông điệp đã nhập ở phần nhập dữ liệu chat sau đó thông điệp sẽ xuất lên boxchat của 2 đối tượng đang kết nối với nhau.

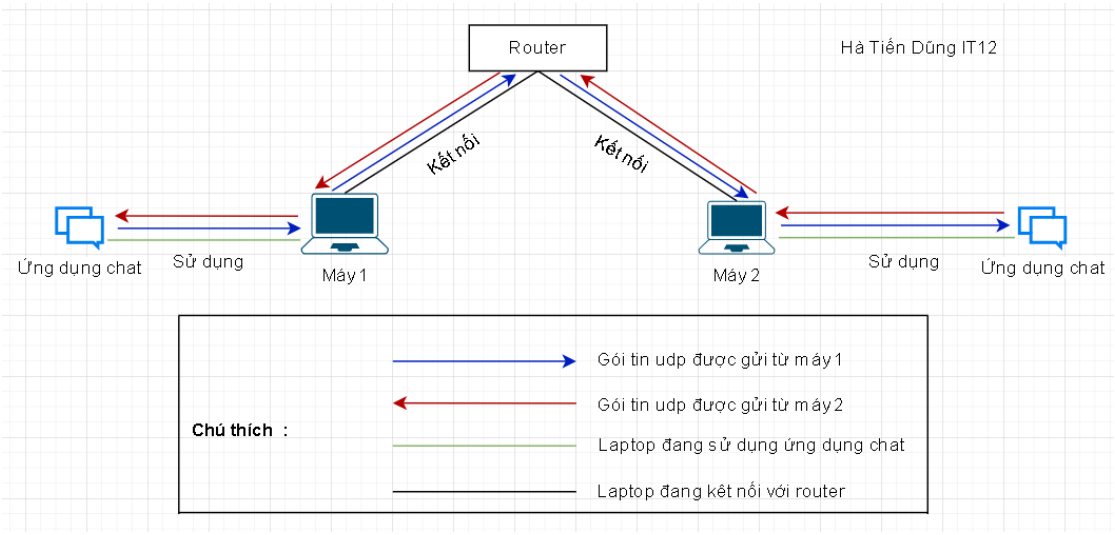
- Dữ liệu: Ứng dụng được thiết kế theo hướng là một ứng dụng chat trực tiếp giữa hai máy qua mạng lan. Dữ liệu được lấy trực tiếp khi người dùng sử dụng ứng dụng như nhập tên, nhập port, nhập ip để kết nối và nội dung chat. Giả sử máy 1 có tên là dungha khi gửi tin nhắn cho máy 2, máy 2 sẽ nhận được tin nhắn với nội dung là dungha : (nội dung tin nhắn).

3. Khả năng thực hiện.

- Xây dựng được giao diện ứng dụng bằng windows form với một giao diện đơn giản dễ dùng cho người sử dụng.
- Ứng dụng có thể chat qua mạng Lan.
- Ứng dụng đáp ứng yêu cầu của chủ đề được chọn sử dụng lớp hỗ trợ UdpClient.

CHƯƠNG 2 – THIẾT KẾ CHƯƠNG TRÌNH ỨNG DỤNG CHAT GIỮA HAI MÁY UDP WINDOWS FORM

2.1 Sơ đồ kiến trúc tổng quát bài toán



Sơ đồ 2 : Sơ đồ kiến trúc tổng quát bài toán.

- Khi hai thiết bị laptop hoặc pc kết nối với 1 Router và sử dụng ứng dụng. Người dùng ở máy 1 nhập port nhận và port nhận của máy 2, địa chỉ ip của máy 2 sau đó nhấn kết nối lúc này người dùng ở máy 2 chưa kết nối tới nên ứng dụng sẽ chờ kết nối của ứng dụng chat ở máy 2. Lúc này người dùng ở máy 2 cũng nhập port nhận và port nhận của máy 1, địa chỉ ip của máy 1 sau đó nhấn kết nối. Lúc này hai máy đã có thể gửi tin nhắn cho nhau.

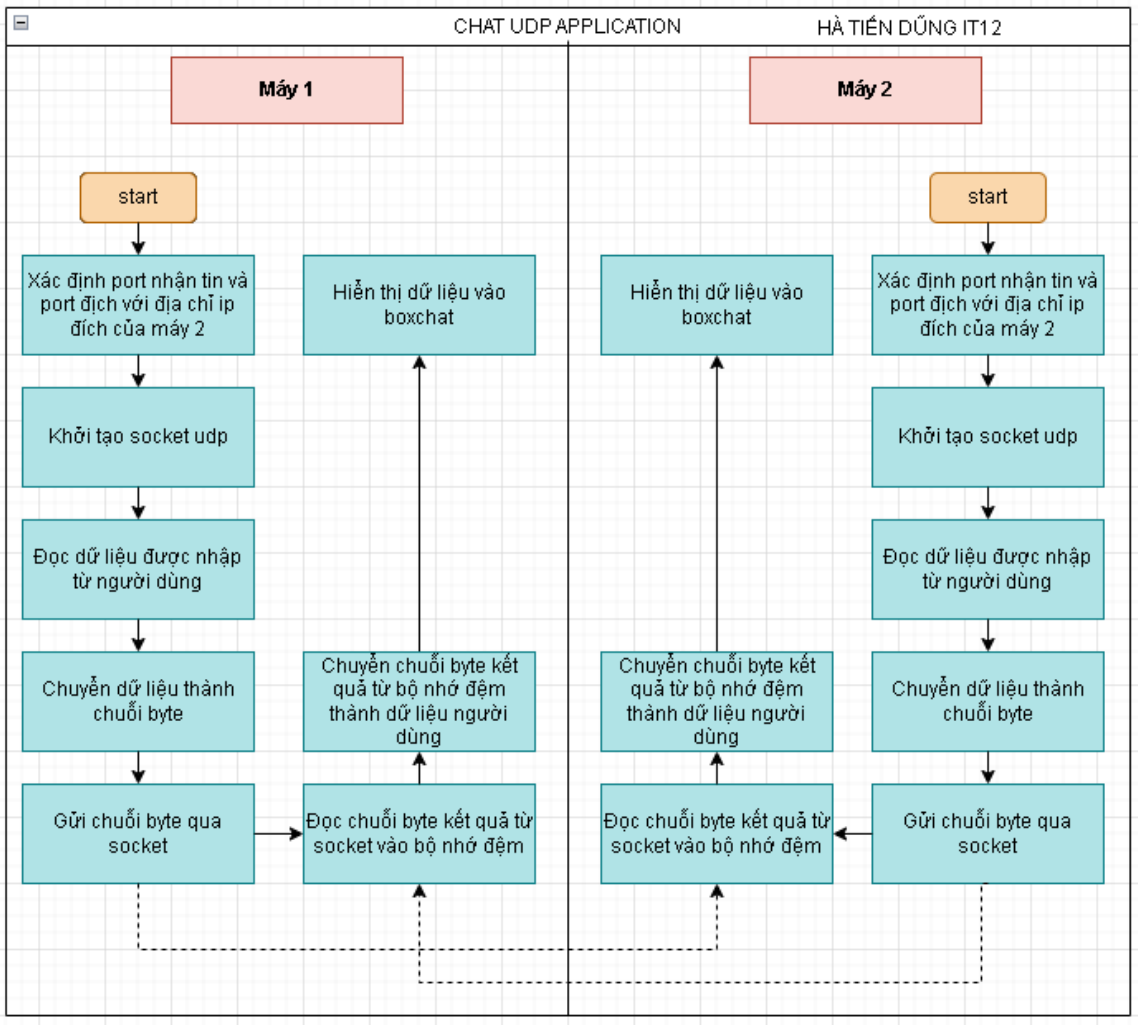
- Khi người dùng nhập một đoạn chat và ấn gửi ứng dụng sẽ chuyển chuỗi string đó thành chuỗi byte và gửi gói tin udp đó tới máy 2, đường đi của gói tin udp như sơ đồ trên. Sau khi máy 2 nhận được gói tin ứng dụng chat sẽ chuyển chuỗi byte đó thành chuỗi string và hiển thị vào khung chat để cho người dùng hiểu được. Quá trình gửi 1 đoạn chat ở máy 2 sang máy một cũng tương tự như trên.

2.2 Các chức năng

- Mô tả chức năng

+ Chức năng chat: Chức năng chat của ứng dụng gồm có phần kết nối, gửi và ngắt kết nối. Để có thể chat người dùng cần nhập đúng port và địa chỉ ip đích sau đó nhấn kết nối, lúc này người dùng có thể gửi bất kỳ đoạn văn bản nào tới ứng dụng chat của máy kia và cuối cùng khi không muốn chat nữa người dùng có thể chọn nút ngắt kết nối. Sau khi ngắt kết nối người dùng ứng dụng ở máy này cũng như máy bên kia sẽ không thể gửi đoạn văn bản tới được nữa.

- Sơ đồ hoạt động của chức năng



Sơ đồ 2.1 : Sơ đồ hoạt động của chức năng chat của ứng dụng

- Mô tả sơ đồ hoạt động của ứng dụng:

+ Người dùng ở máy 1 tiến hành nhập các thông số như là port nhận (cổng của người gửi), port gửi (cổng mà diagram được gửi đến), tên và địch chỉ ip của máy 2 và nhấn kết nối → lúc này chương trình sẽ khởi tạo socket udp và chờ đợi kết nối từ máy 2 → Sau khi máy 2 nhập đầy đủ thông số hợp lệ và nhấn kết nối → Lúc này hai máy có thể giao tiếp với nhau → Giả sử người dùng ở máy 1 nhập một đoạn tin nhắn và ấn gửi → chương trình sẽ đọc dữ liệu mà người dùng vừa nhập → sau đó mã hóa thành một chuỗi byte → và gửi chuỗi byte qua socket với chuỗi dữ liệu được mã hóa, độ rộng chuỗi, địa chỉ ip và port của máy 2 → lúc này gói tin được gửi đến máy 2 → chương trình ở máy 2 nhận chuỗi byte đó vào bộ nhớ đệm → sau đó mã hóa chuỗi byte thành dữ liệu người dùng có thể đọc được → cuối cùng hiện dữ liệu đó vào boxchat của người dùng ở máy 2.

+ Quá trình gửi tin nhắn từ máy 2 sang máy 1 cũng tương tự như trên → chương trình cũng sẽ đọc dữ liệu người dùng vừa nhập → sau đó mã hóa đoạn tin nhắn đó thành chuỗi byte → và gửi chuỗi byte qua socket với chuỗi được mã hóa, độ rộng

chuỗi, địa chỉ ip và port của máy 1 → lúc này gói tin được gửi đến máy 1 → chương trình ở máy 1 nhận chuỗi byte đó vào bộ nhớ đệm → sau đó mã hóa chuỗi byte thành dữ liệu người dùng có thể đọc được → cuối cùng hiện dữ liệu đố vào boxchat của người dùng ở máy 1.

+ Khi người dùng nhập một đoạn tin nhắn và ấn gửi → đoạn tin nhắn đó cũng sẽ hiện lên box chat của người dùng đó.

2.3 Thiết kế dữ liệu

- Ứng dụng được thiết kế theo hướng là một ứng dụng chat trực tiếp giữa hai máy qua mạng lan. Dữ liệu được lấy trực tiếp khi người dùng sử dụng ứng dụng như nhập tên, nhập port, nhập ip để kết nối và nội dung chat. Giả sử máy 1 có tên là dungha khi gửi tin nhắn cho máy 2, máy 2 sẽ nhận được tin nhắn với nội dung là dungha : (nội dung tin nhắn).

CHƯƠNG 3 – CÀI ĐẶT VÀ THỬ NGHIỆM CHƯƠNG TRÌNH

3.1 Yêu cầu cài đặt

Yêu cầu về phần cứng

- Bộ vi xử lý (CPU) : Tối thiểu Intel Core i3 hoặc AMD A6 trở lên.

- Bộ nhớ Ram: Tối thiểu là 2GB hoặc nhiều hơn để đảm bảo khả năng xử lý tốt của hệ thống và tránh trục trặc.

- Card đồ họa: Ứng dụng không có yêu cầu về card đồ họa.

- Bộ nhớ trong: Cần đủ không gian trên ổ cứng để cài đặt tối thiểu 30MB.

Yêu cầu về phần mềm

Hệ điều hành: Windows 7 trở lên để tránh vấn đề không tương thích.

Kết nối mạng: Cần có kết nối ổn định tránh gián đoạn.

3.2 Các bước cài đặt và kết quả

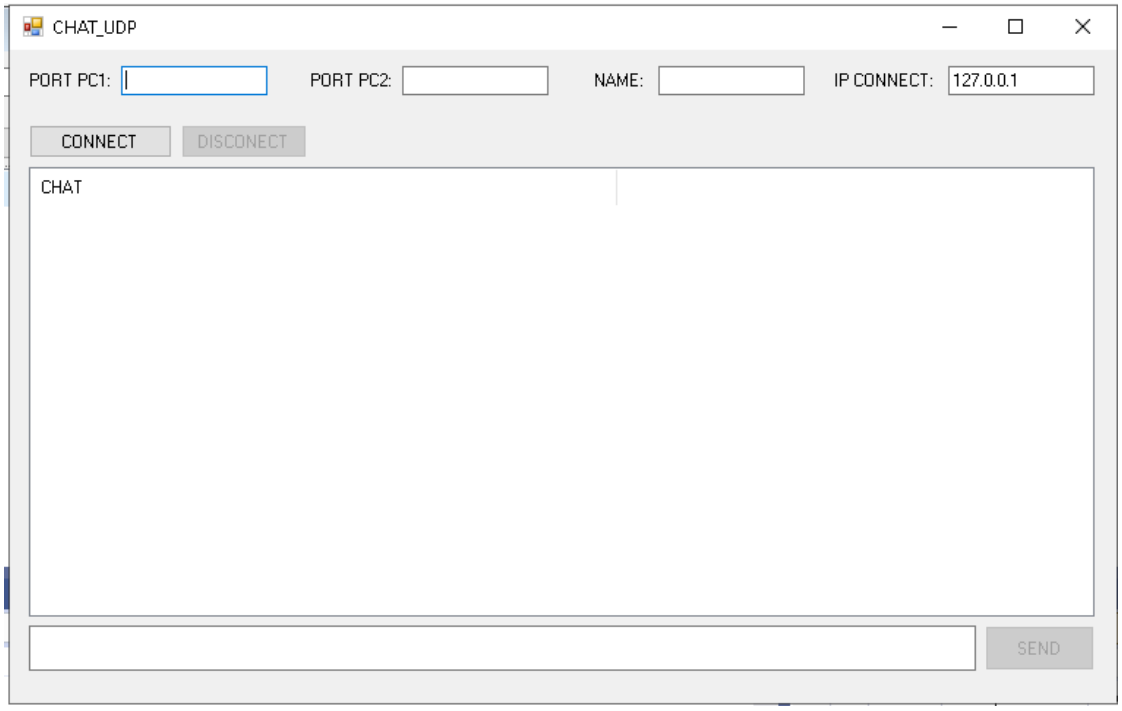
- Thiết kế giao diện của ứng dụng chat

+ Bảng thành phần giao diện ứng dụng chat:

Bảng 3: Bảng thành phần của giao diện ứng dụng

STT	Tên đối tượng	Kiểu	Ý nghĩa	Ghi chú
1	CHAT_UDP	Form	Form giao diện ứng dụng	
2	lbPR	Label	Chỉ chỗ nhập port để nhận tin nhắn	
3	lbPS	Label	Chỉ chỗ nhập port để gửi tin nhắn	
4	lbName	Label	Chỉ chỗ nhập tên	
5	lbIP	Label	Chỉ chỗ nhập địa chỉ ip	
6	btnConnect	Button	Nút để kết nối	
7	btnDisconnect	Button	Nút để ngắt kết nối	
8	btnSend	Button	Nứt để gửi tin nhắn	
9	txtPortR	Textbox	Chỗ nhập port nhận tin	
10	txtPortS	Textbox	Chỗ nhập port gửi tin	
11	txtName	Textbox	Chỗ nhập tên	
12	txtIP	Textbox	Chỗ nhập ip	
13	txtMSG	Textbox	Chỗ nhập đoạn tin nhắn	
14	lsvMSG	Listview	Khung chat	

+ Kết quả giao diện:



Hình 3.1 : Giao diện ứng dụng chat

- Các thư viện sử dụng trong ứng dụng

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Net.Sockets;
using System.Net;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Threading;
```

- Thiết lập cài đặt chức năng của ứng dụng chat

+ Khai báo một số biến và lớp:

```
string localPort = ""; // Port để nhận tin nhắn
string remotePort = ""; // Port để gửi tin nhắn
UdpClient APPCHAT = new UdpClient(); // Tạo một đối tượng UdpClient
Thread thr; // Tạo một luồng
bool exit = false;  // Kiểm tra trạng thái kết nối
delegate void ClearCacheReceivedData(string Data, string RemoteHost);
// Tạo một delegate dọn dẹp cache
```

+ Thiết lập hàm khởi tạo:

```
public CHAT_UDP()
{
InitializeComponent();
txtIP.Text = "127.0.0.1"; // Khi mở ứng dụng chỗ IP mặc định là 127.0.0.1
btnDisconnect.Enabled = false; // Khi chưa connect thì không click được
btnSend.Enabled = false; // Khi chưa connect thì không click được
}
```

- Mô tả: khi mở chương trình dòng nhập ip sẽ để mặc định là 127.0.0.1 và lúc này nút ngắt kết nối và nút gửi sẽ bị ẩn đi vì chưa mở kết nối.

+ Thiết lập cho nút Disconnect:

```
private void btnDisconnect_Click(object sender, EventArgs e)
{
txtIP.ReadOnly = false; // Ngắt kết nối textbox này được chỉnh sửa
txtPortR.ReadOnly = false; // Ngắt kết nối textbox này được chỉnh sửa
txtPortS.ReadOnly = false; // Ngắt kết nối textbox này được chỉnh sửa
txtName.ReadOnly = false; // Ngắt kết nối textbox này được chỉnh sửa
btnConnect.Enabled = true; // Ngắt kết nối nút này được click
btnDisconnect.Enabled = false; // Ngắt kết nối nút này không click được
btnSend.Enabled = false; // Ngắt kết nối nút này không click được
exit = true; // Ngắt kết nối không nhận tin nhắn
txtPortR.Text = "";
SentReport(); // Gửi thông báo đã rời khỏi phòng
}
```

- Mô tả:Khi chọn nút ngắt kết nối chương trình sẽ mở cho chỉnh sửa các ô thông tin như port nhận, port gửi, name , ip và lúc này nút mở kết nối sẽ được hiện còn nút ngắt kết nối và nút gửi tin nhắn sẽ bị ẩn đi. Sau đó ngắt kết nối không cho nhận tin nhắn và gọi một hàm SentReport() để gửi thông báo cho máy kia là đã rời khỏi phòng.

+ Thiết lập cho hàm SentReport():

```
private void SentReport()
{
byte[] msg; // khai báo 1 mảng nhị phân msg
msg = System.Text.Encoding.UTF8.GetBytes(txtName.Text + "Đã rời khỏi phòng!!");
APPCHAT.Send(msg, msg.Length, txtIP.Text, int.Parse(remotePort)); // Gửi thông
báo đã rời khỏi phòng đến địa chỉ ip và port từ xa
}
```

- Mô tả: ở hàm này đầu tiên khai báo một mảng byte có tên là msg → gán msg bằng chuỗi gồm tên và đoạn thông báo rời khỏi phòng đã được mã hóa thành chuỗi byte → Gọi Lớp socket Udp tên là APPCHAT gửi tới địa chỉ ip và port đã được chỉ định trước. cấu trúc gửi gồm chuỗi byte , độ rộng chuỗi byte, ip, port kết nối từ xa.

+ Thiết lập cho nút Connect:

```
private void btnConnect_Click(object sender, EventArgs e)
{
            if (txtPortS.Text == "" || txtPortR.Text == "" || txtName.Text == "" ||
txtIP.Text == "") // Kiểm tra người dùng có nhập đủ không
            {
                MessageBox.Show("Vui lòng nhập đầy đủ thông tin", "Thông báo",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            else
            {
                localPort = txtPortR.Text; // Gắn localPort = Port nhận người dùng
nhập
                remotePort = txtPortS.Text; // Gắn localPort = Port gửi người dùng
nhập
                try
                {
                    APPCHAT = new UdpClient(int.Parse(localPort));
                    thr = new Thread(Explore); // Khai báo luồng Explore để thiết
lập kết nối
                    thr.Start(); // Khởi chạy luồng
                }
                catch (Exception)
                {
                    MessageBox.Show("Vui lòng nhập Port mới cho PC2", "Thông báo",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
                thr = new Thread(Explore); // Khai báo luồng Explore để thiết lập
kết nối
                thr.Start();  // Khởi chạy luồng
                txtIP.ReadOnly = true; // Khi connect chỗ này không được nhập
                txtName.ReadOnly = true; // Khi connect chỗ này không được nhập
                txtPortR.ReadOnly = true; // Khi connect chỗ này không được nhập
                txtPortS.ReadOnly = true; // Khi connect chỗ này không được nhập
                btnConnect.Enabled = false; // Khi connect nút này không click được
                btnSend.Enabled = true; // Khi connect nút này click được
                btnDisconnect.Enabled = true; // Khi connect nút này click được
            }
        }
```

- Mô tả: Đầu tiên kiểm tra người dùng đã nhập đủ thông số cần thiết và hợp lệ chưa nếu chưa thì thông báo và nhập lại → gán localPort và remotePort bằng thông số của người dùng vừa nhập → Tạo một đối tượng UdpClient mới và truyền vào localPort → Tạo một Thread mới và truyền vào hàm Explore (hàm thiết lập kết nối udp) → Sau đó khởi chạy luồng → lúc này khi kết nối thành công người dùng sẽ không thể chỉnh sửa các thông số như port nhận, port gửi, tên, ip được nữa và nút ngắt kết nối và nút gửi tin nhắn sẽ hiện lên còn nút mở kết nối sẽ bị ẩn đi.

+ Thiết lập hàm Explore():

```
private void Explore() // Luồng kết nối Udp
        {
            IPAddress ip; // Khai báo 1 địa chỉ ip
            byte[] msg; // Khai báo 1 mạng nhị phân msg
            string str = "";
            ip = Dns.GetHostEntry(txtIP.Text).AddressList[0]; // Gán ip = địa
chỉ ip được người dùng nhập.
            IPEndPoint ipep = new IPEndPoint(ip, Convert.ToInt16(remotePort));
// Khai báo lớp đối tượng mới cho lớp IPEndPoint với port và ip đã chỉ định
            while (exit == false)
            {
                Application.DoEvents();
                if (APPCHAT.Available > 0)
                {
                    try // Exception chỉ cho kết nối 1 máy
                    {
                        msg = APPCHAT.Receive(ref ipep); // Nhận 1 mảng kiểu
byte chưa dữ liệu datagram
                        str = System.Text.Encoding.UTF8.GetString(msg);
                        ReceivedData(str, ipep.Address.ToString());// hàm nhận
tin nhắn
                    }
                    catch
                    {
                        MessageBox.Show("Chưa có ai kết nối tới");
                        thr.Abort();
                    }
                }
            }
        }
```

○ Mô tả: Khai báo một IPAddress có tên ip, một mảng byte[] có tên msg, một chuỗi str rỗng → Sau đó gán ip bằng một địa chỉ ip do người dùng nhập → Khai báo một IPEndPoint có tên ipep và truyền vào ip và port từ xa đã được định dạng do người dùng nhập → Tạo một vòng lặp với điều kiện exit == false (biến bool để kiểm tra trạng thái kết nối) → sau đó gán mạng byte bằng lớp UdpClient có tên APPCHAT và nhận gói dữ liệu datagram từ một địa chỉ từ xa → sau đó gán chuỗi str rỗng bằng mảng byte vừa nhận được đã được mã hóa thành chuỗi → và gọi hàm nhận dữ liệu ReceiveData và truyền vào chuỗi str và đại chỉ ip từ xa.

+ Thiết lập hàm ReceivedData():

```
private void ReceivedData(string Data, string RemoteHost) // Hàm nhận tin nhắn
{
        if (lsvMSG.InvokeRequired) // kiểm tra và xóa cache
        {
           ClearCacheReceivedData CCRD = new ClearCacheReceivedData(ReceivedData);
           lsvMSG.Invoke(CCRD, new object[] { Data, RemoteHost });
           return;
        }
            lsvMSG.Items.Add(Data); // Hiện thị tin nhắn vào hộp chat
}
```

- Mô tả: Hàm nhận dữ liệu đầu tiền kiểm tra luồng kết nối có khác với luồng kết nối vừa tạo trước đó không nếu khác thì làm mới → sau đó hiện thị dữ liệu vào boxchat.

+ Thiết lập cho nút Send:

```
private void btnSend_Click(object sender, EventArgs e)
{
     IPAddress ip; // Khai báo 1 địa chỉ ip
     if (!IPAddress.TryParse(txtIP.Text, out ip)) // Kiểm tra ip có hợp lệ không
     {
        MessageBox.Show("Hãy nhập chính xác ip của người nhận", "Thông báo",
MessageBoxButtons.OK, MessageBoxIcon.Error);
     }
     else
     {
        lsvMSG.Items.Add(txtName.Text + " [" + txtIP.Text + "] " + txtMSG.Text);
        SentData();// Hàm gửi tin nhắn
        txtMSG.Clear();// Xóa dữ liệu vừa nhập trong ô tin nhắn
     }
}
```

- Mô tả: Đầu tiên khai báo một IPAddress có tên ip → Sau đó kiểm tra ip có nhập chính xác không nếu có thì thông báo → nếu không thì hiện đoạn tin nhắn vào boxchat của người dùng gồm tên, ip, đoạn tin nhắn → Gọi hàm gửi tin nhắn SentData → Và xóa dữ liệu ở ô nhập tin nhắn.

+ Thiết lập cho hàm SentData():

```
private void SentData() // Hàm gửi tin nhắn
{
    byte[] msg; // Khai báo 1 mạng nhị phân msg
    msg = System.Text.Encoding.UTF8.GetBytes(txtName.Text + " [" + txtIP.Text + "] "
+ txtMSG.Text); // Chuyển chuỗi string thành chuỗi byte
    APPCHAT.Send(msg, msg.Length, txtIP.Text, int.Parse(remotePort)); // Gửi một một
gói dữ liệu UDP đến port và ip từ xa được chỉ định.
}
```

- Mô tả: Ở hàm gửi tin nhắn đầu tiên khai báo một mảng byte[] có tên msg → gắn msg bằng chuỗi ký tự gồm tên, ip, đoạn tin nhắn đã được mã hóa thành chuỗi byte → sau đó gọi lớp UdpClient APPCHAT gửi tin nhắn tới địa chỉ ip và port đã được chỉ định cấu trúc gửi bao gồm chuỗi byte, độ rộng chuỗi byte, địa chỉ ip, port từ xa.

+ Thiết lập một số hàm kiểm tra:

```
protected override void OnClosing(CancelEventArgs e)
{
     if(thr == null) // Trường hợp mới mở lên tắt liền, lúc này thr chưa
được gọi nếu không loại thr này sẽ bị carsh
     {
          exit = true;
     }
     else
     {
```

```
            if(thr.IsAlive) // Nếu thr đã chạy thì ta kiểm tra thr có còn
sống hay đang chạy không nếu còn thì ta hủy nó đi
            {
                thr.Abort();
            }
        }
}
```

- Hàm kiểm tra luồng kết nối.

```
private void txtPortS_KeyPress(object sender, KeyPressEventArgs e)
{
    if(!Char.IsDigit(e.KeyChar) && !Char.IsControl(e.KeyChar))
    {
        e.Handled = true;
        MessageBox.Show("Giá trị nhập không đúng định dạng", "Thông
báo", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

- Hàm kiểm tra port gửi có nhập hợp lệ không

```
private void txtPortR_KeyPress(object sender, KeyPressEventArgs e)
{
     if (!Char.IsDigit(e.KeyChar) && !Char.IsControl(e.KeyChar))
     {
         e.Handled = true;
         MessageBox.Show("Giá trị nhập không đúng định dạng", "Thông
báo", MessageBoxButtons.OK, MessageBoxIcon.Error);
     }
}
```

- Hàm kiểm tra port nhận có nhập hợp lệ không

```
private void txtIP_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!Char.IsDigit(e.KeyChar) && !Char.IsControl(e.KeyChar)) // Kiểm tra ip
nhập có đúng định dạng không
    {
        e.Handled = true;
        if (e.KeyChar == '.')
            e.Handled = false;
        else
            MessageBox.Show("Giá trị nhập không đúng định dạng", "Thông báo",
MessageBoxButtons.OK, MessageBoxIcon.Error);
     }
}
```

- Hàm kiểm tra ip nhập có đúng định dạng không

```
private void txtIP_Leave(object sender, EventArgs e)
{
    IPAddress ip;
    if (!IPAddress.TryParse(txtIP.Text, out ip)) // Kiểm tra ip có hợp lệ không
    {
          MessageBox.Show("Địa chỉ ip không chính xác", "Thông báo",
MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

- Hàm kiểm tra ip nhập có chính xác không

- Kết quả

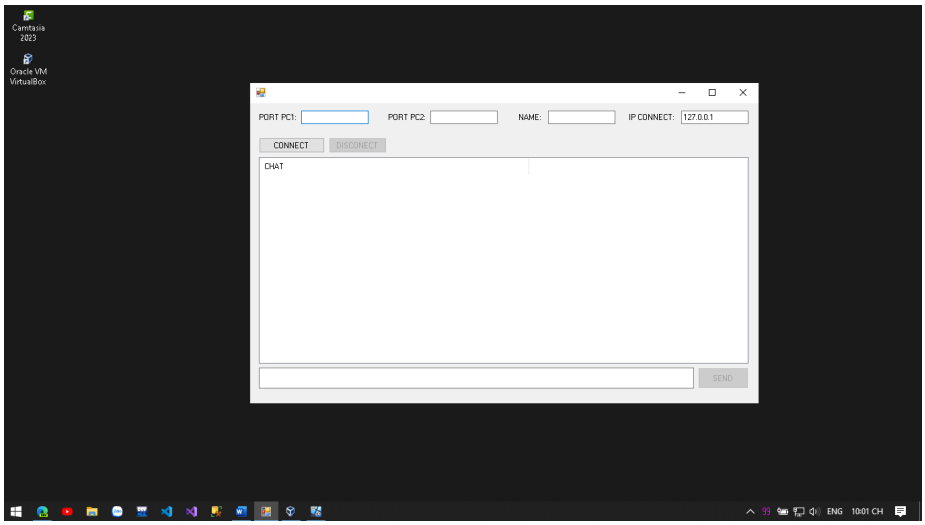
Từ các giai đoạn trên nhóm chúng em đã hoàn thành tạo ra được ứng dụng chat giữa hai máy sử dụng giao thức UDP và giao diện Windows Form cơ bản. Do kiến thức khi xây dựng ứng dụng thức tế khi đi làm còn hạn chế nên ứng dụng sẽ không tránh được một số thiếu sót. Chúng em rất vui khi thầy, cô đã đọc và góp ý cho chúng em nhận ra và rút kinh nghiệm cho những lần sau.

Cuối cùng nhóm em xin cảm ơn thầy Nguyễn Anh Thơ giảng viên môn lập trình mạng đã hướng dẫn chúng em chúng em thực hành đề tài này và quá trình giảng dạy rất dễ hiểu.

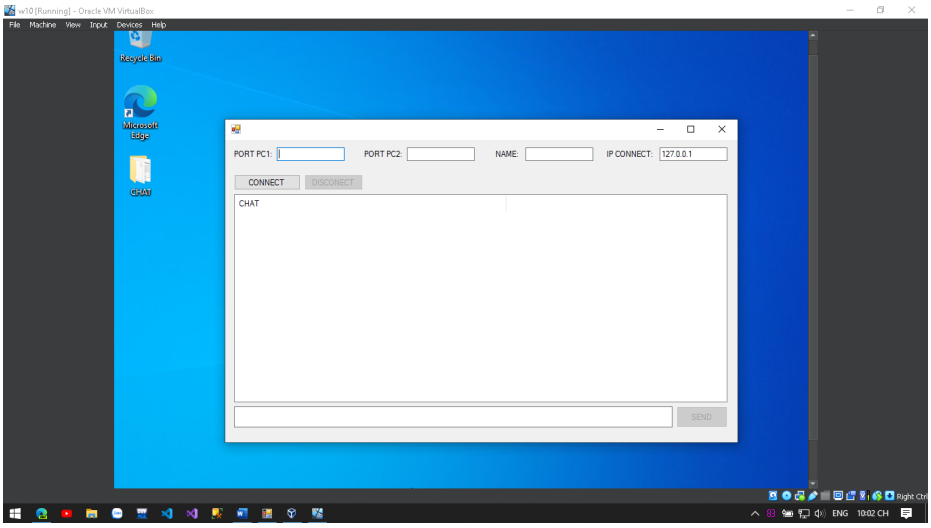
3.3 Thử nghiệm chương trình

- Thử nghiệm chat thử giữa 2 máy trong mạng lan.

- Mở 1 ứng dụng ở máy của mình và (Do không có 2 laptop riêng nên demo trên một máy ảo) 1 ứng dụng trên một máy ảo.

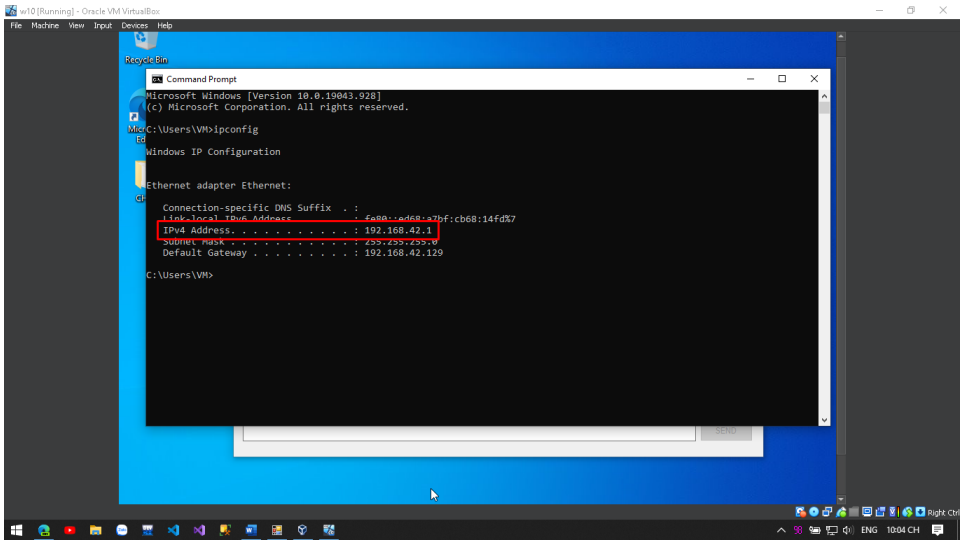


Hình 4 : Mở ứng dụng ở máy chính

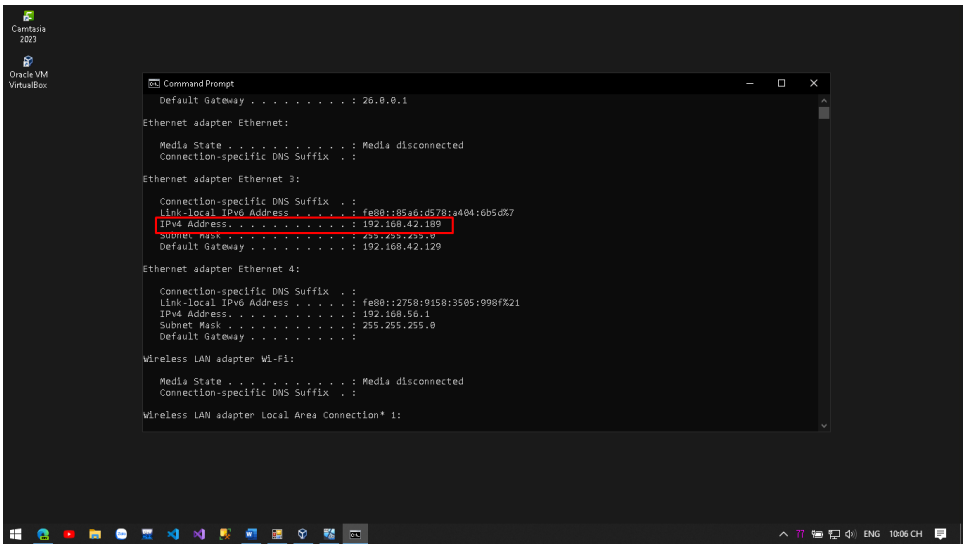


Hình 4.1 : Mở ứng dụng ở máy ảo

- Tiếp theo kiểm tra địa chỉ ipv4 của máy chính và máy ảo để tiến hành kết nối.



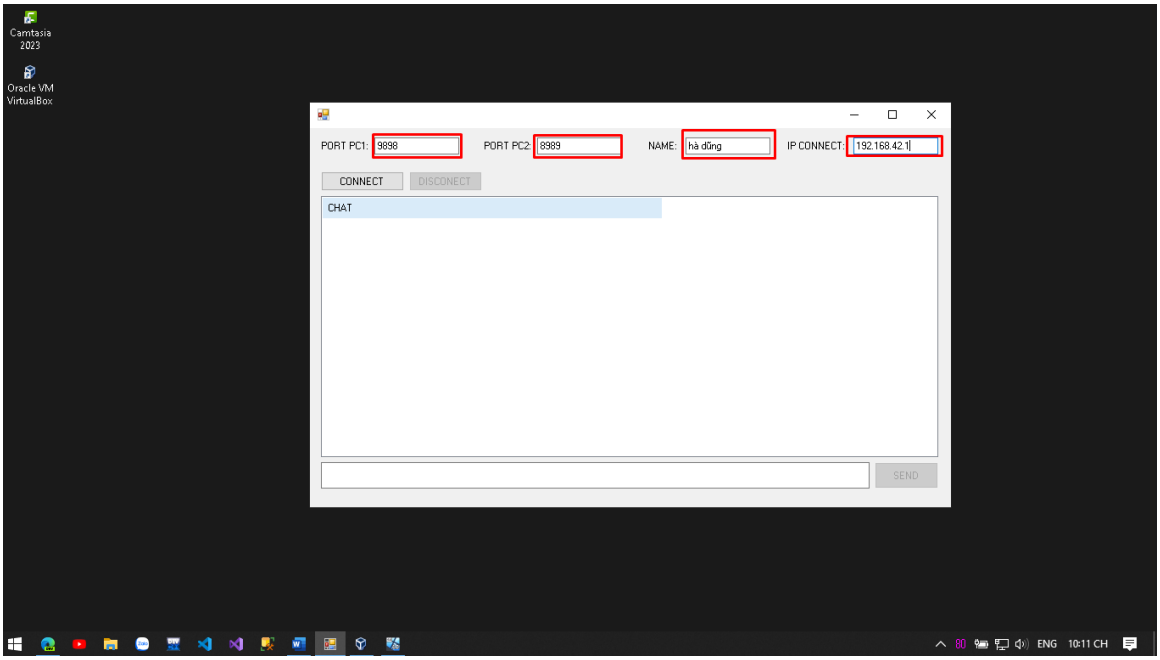
Hình 5 : Kiểm tra địa chỉ ip ở máy ảo



Hình 5.1 : Kiểm tra địa chỉ ip ở máy chính

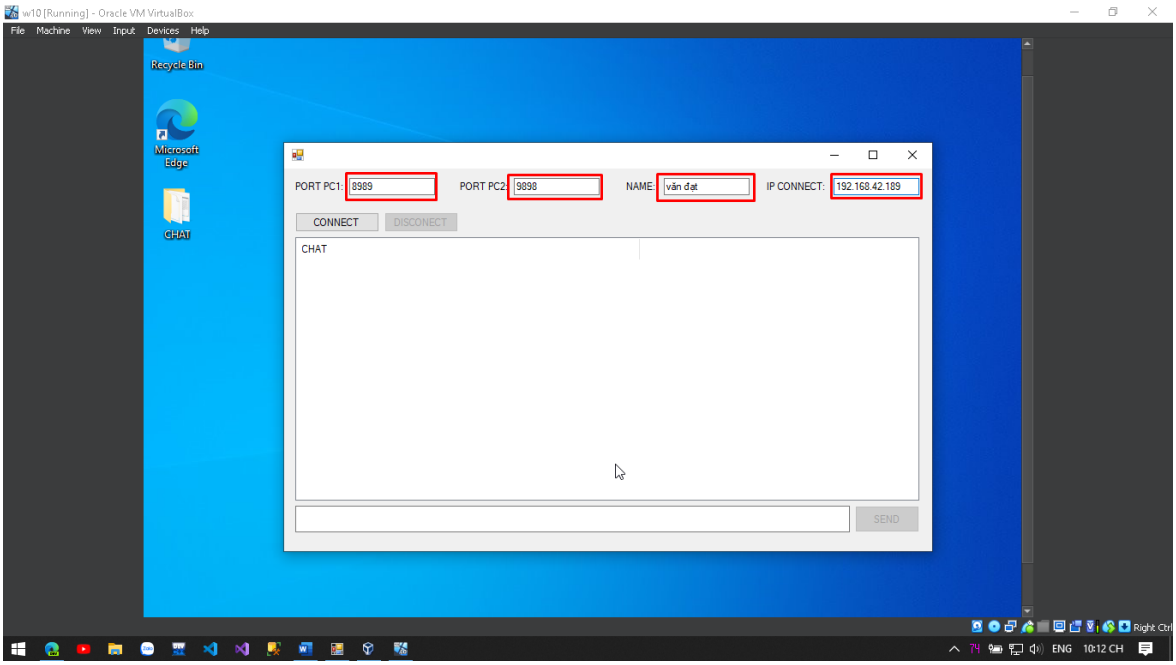
- Tiến hành nhập các thông số cần thiết cho ứng dụng ở máy chính và máy ảo để tiến hành kết nối.

+ Máy chính sử dụng port nhận : 9898 ; port gửi 8989 ; tên : hà dũng ; ip : 192.168.42.1



Hình 6 : Nhập các thông số trên vào ứng dụng ở máy chính

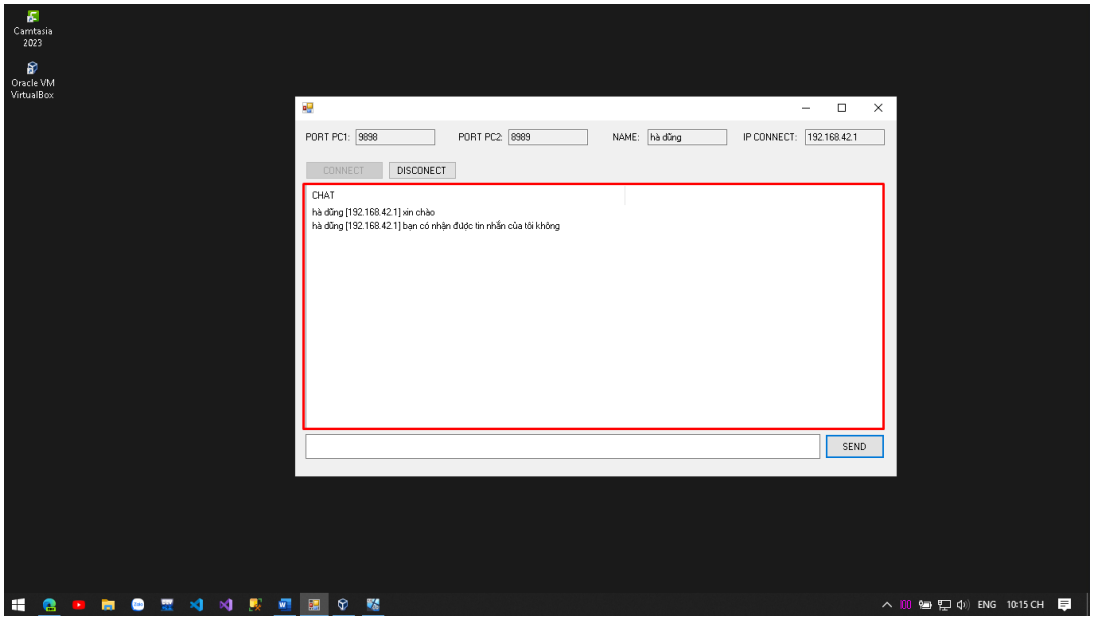
+ Máy ảo sử dụng port nhận : 8989 ; port gửi 9898 ; tên : văn đạt ; ip : 192.168.42.189



Hình 6.1 : Nhập các thông số trên vào ứng dụng ở máy ảo

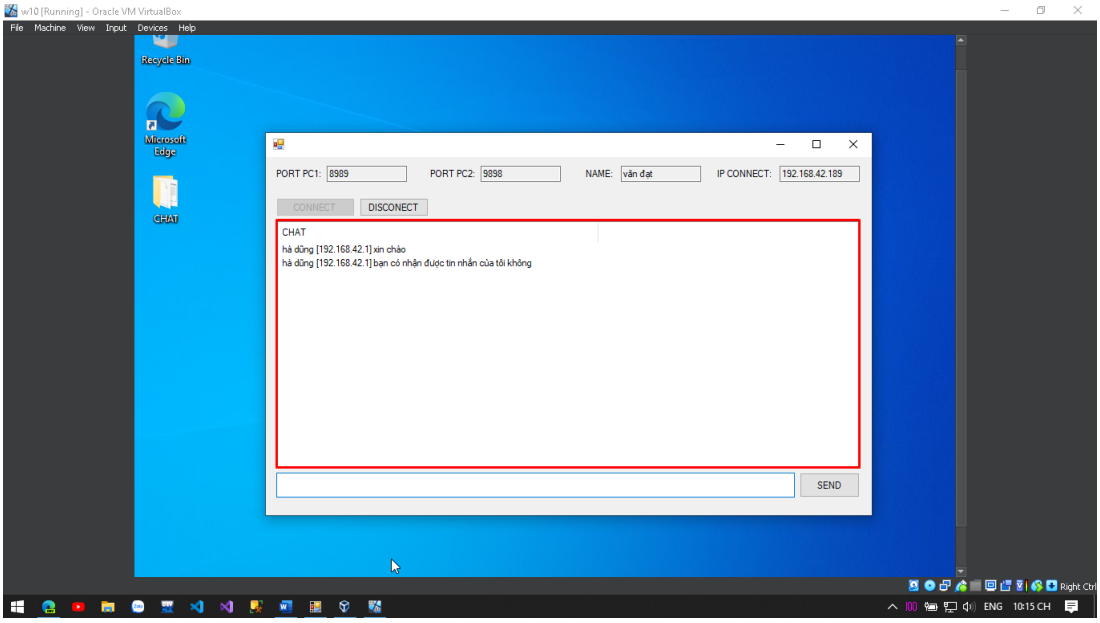
- Sau khi nhập xong tiến hành Connect và bắt đầu chat.

- Bên máy chính soạn tin và gửi.



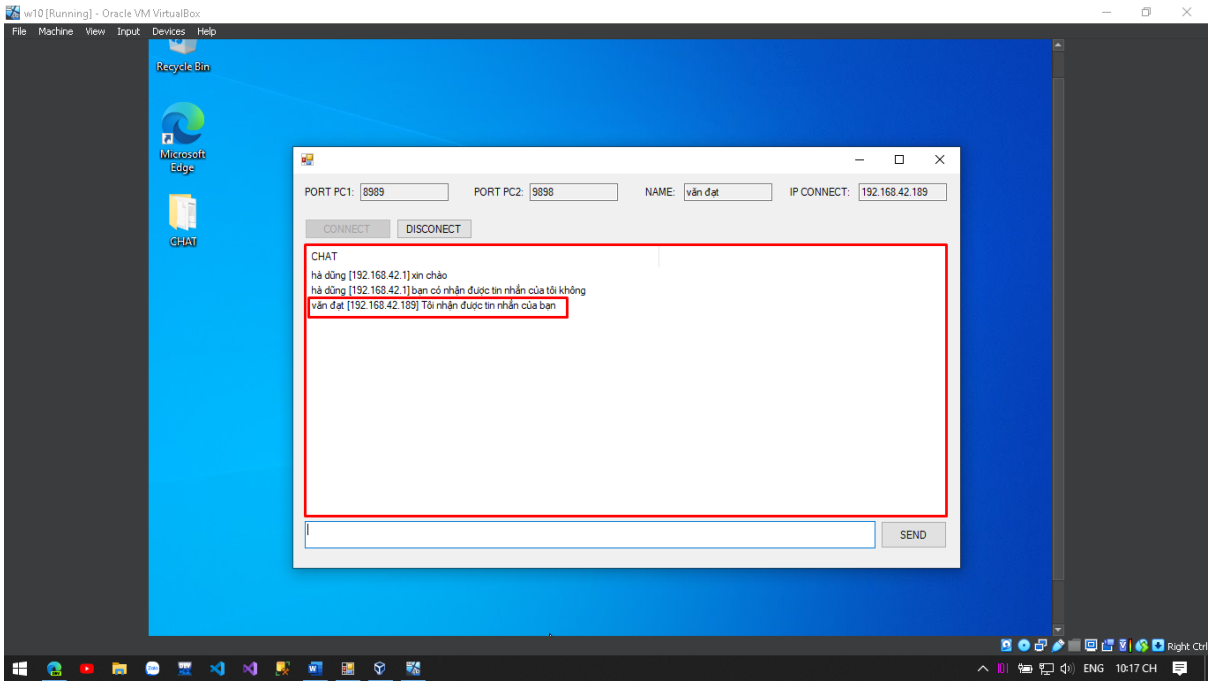
Hình 7 : Tin nhắn hiện lên khung chat sau khi gửi ở ứng dụng của máy chính

- Tin nhắn hiện bên khung chat máy ảo sau khi máy chủ gửi tin nhắn



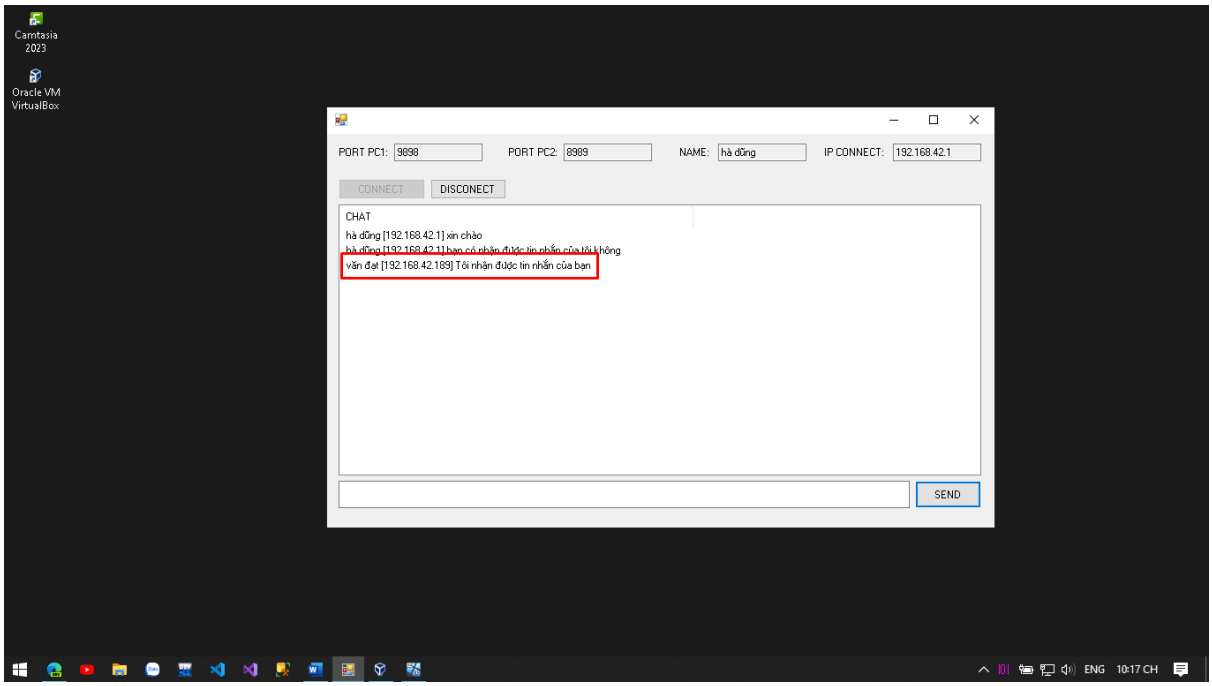
Hình 7.1 : Tin nhắn hiện lên khung chat ở ứng dụng máy ảo khi nhận tin nhắn từ ứng dụng của máy chính

- Bên máy ảo soạn tin và gửi



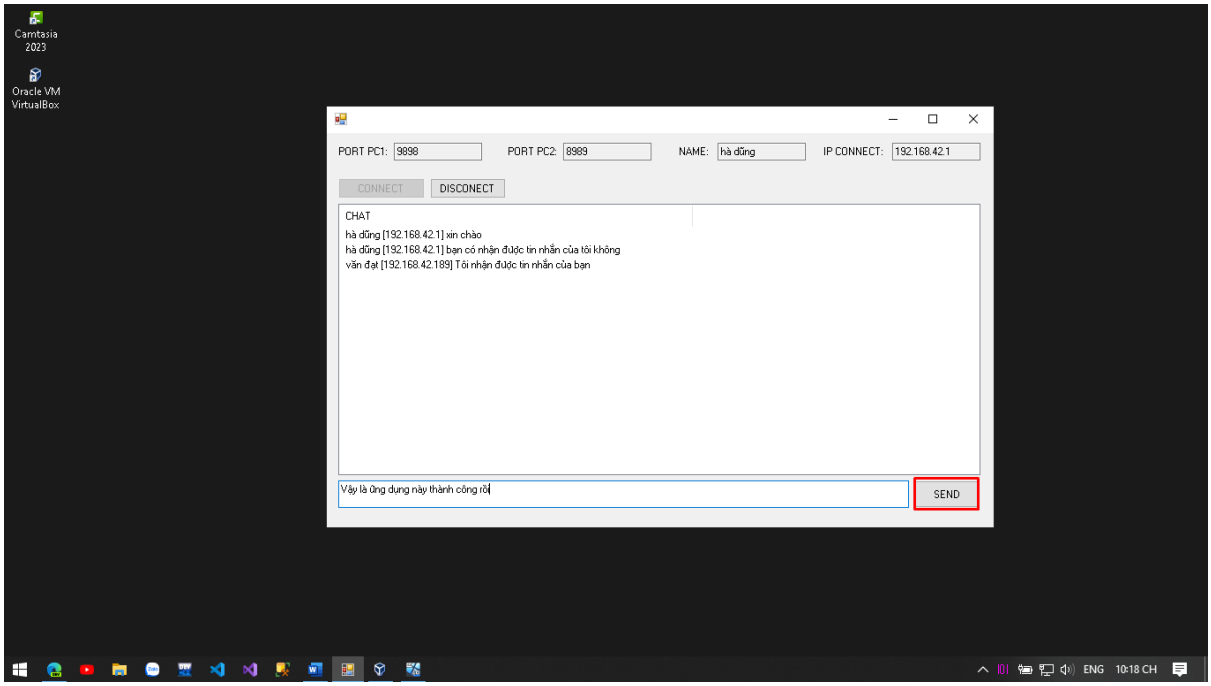
Hình 7.2 : Tin nhắn hiện lên khung chat sau khi gửi ở ứng dụng của máy ảo

- Tin nhắn hiện bên khung chat máy chính sau khi máy ảo gửi tin nhắn

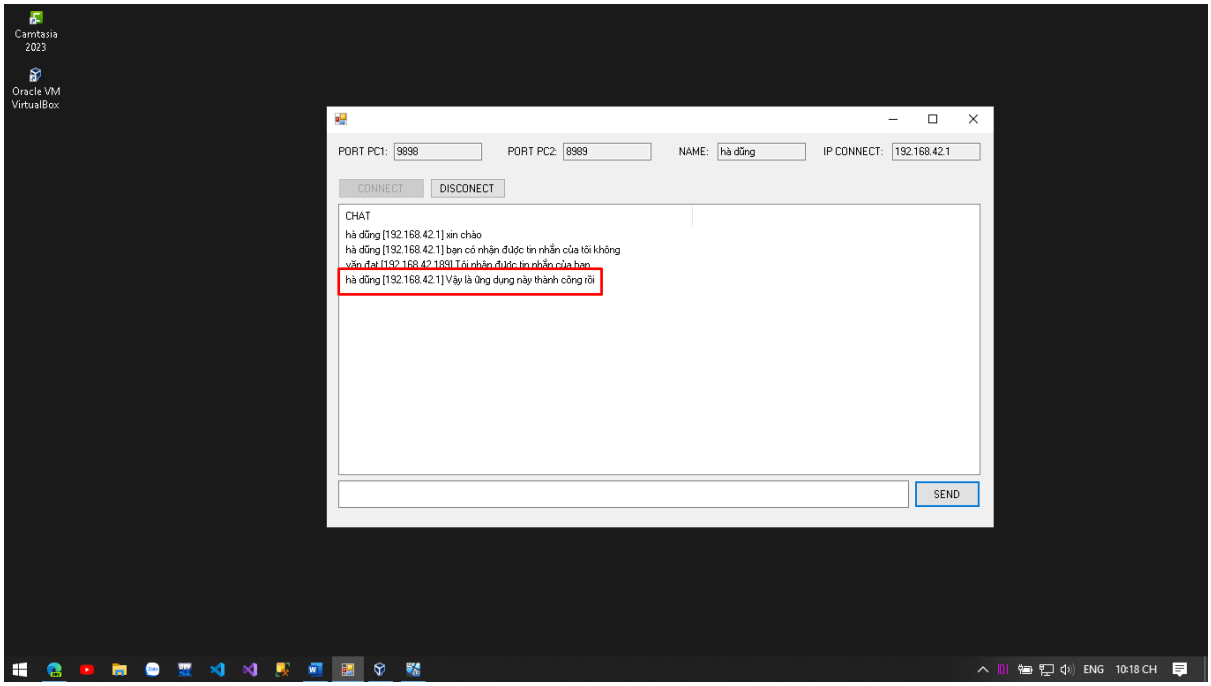


Hình 7.3 : ở khung chat ứng dụng ở máy chính cũng sẽ hiện tin nhắn khi nhận được tin nhắn ở ứng dụng máy ảo

- Soạn tin và nhấn gửi của ứng dụng ở máy chính

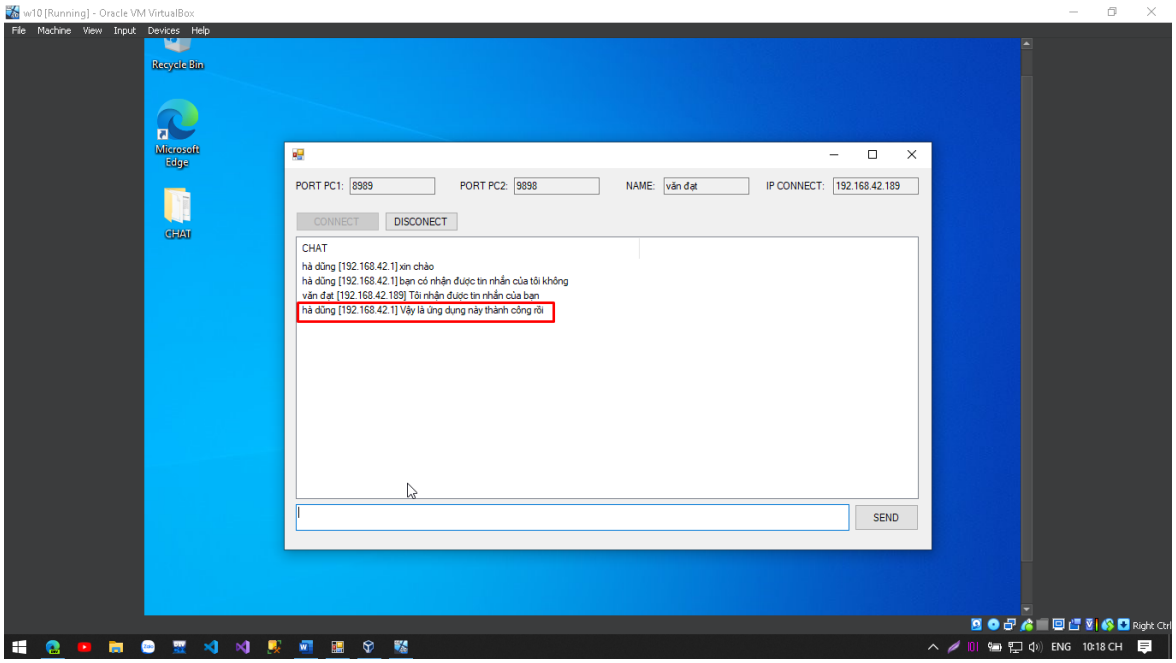


Hình 8 : Soạn tin nhắn và click nút gửi của ứng dụng ở máy chính



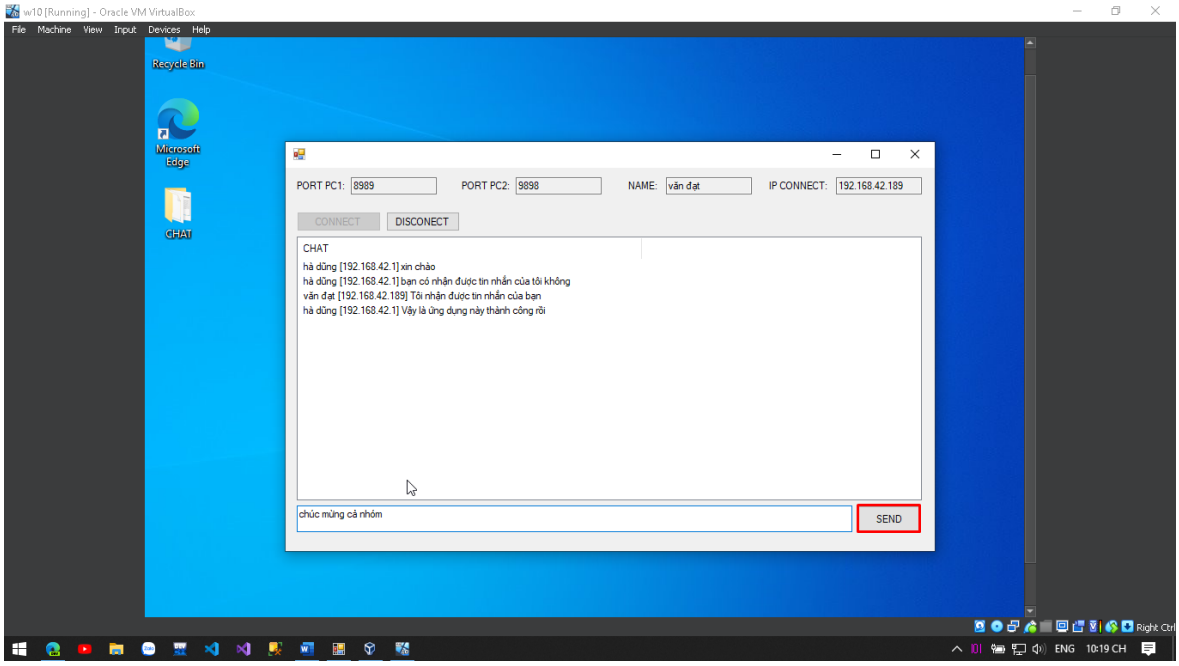
Hình 8.1 : Tin nhắn sau khi gửi sẽ hiện lên khung chat

- Tin nhắn hiện thi ở khung chat của ứng dụng ở máy ảo sau khi ấn gửi từ ứng dụng ở máy chính.

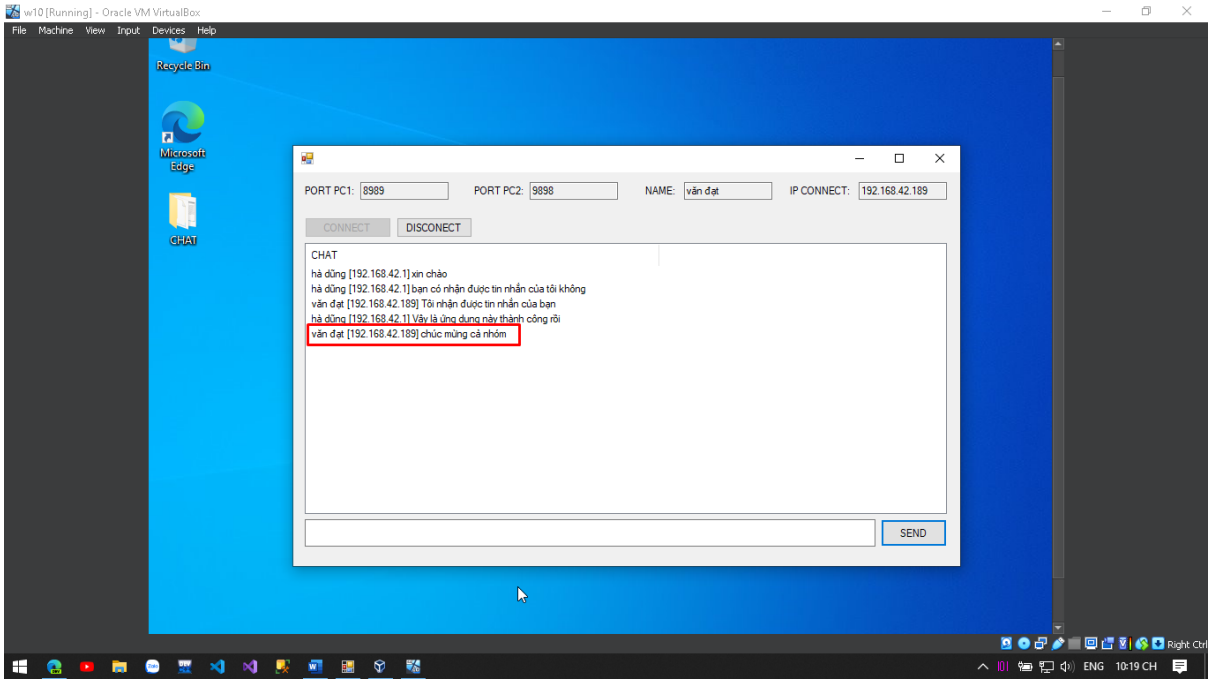


Hình 8.2: ở khung chat ứng dụng ở máy ảo cũng sẽ hiện tin nhắn khi nhận được tin nhắn ở ứng dụng máy chính

- Soạn tin và nhấn gửi của ứng dụng ở máy ảo

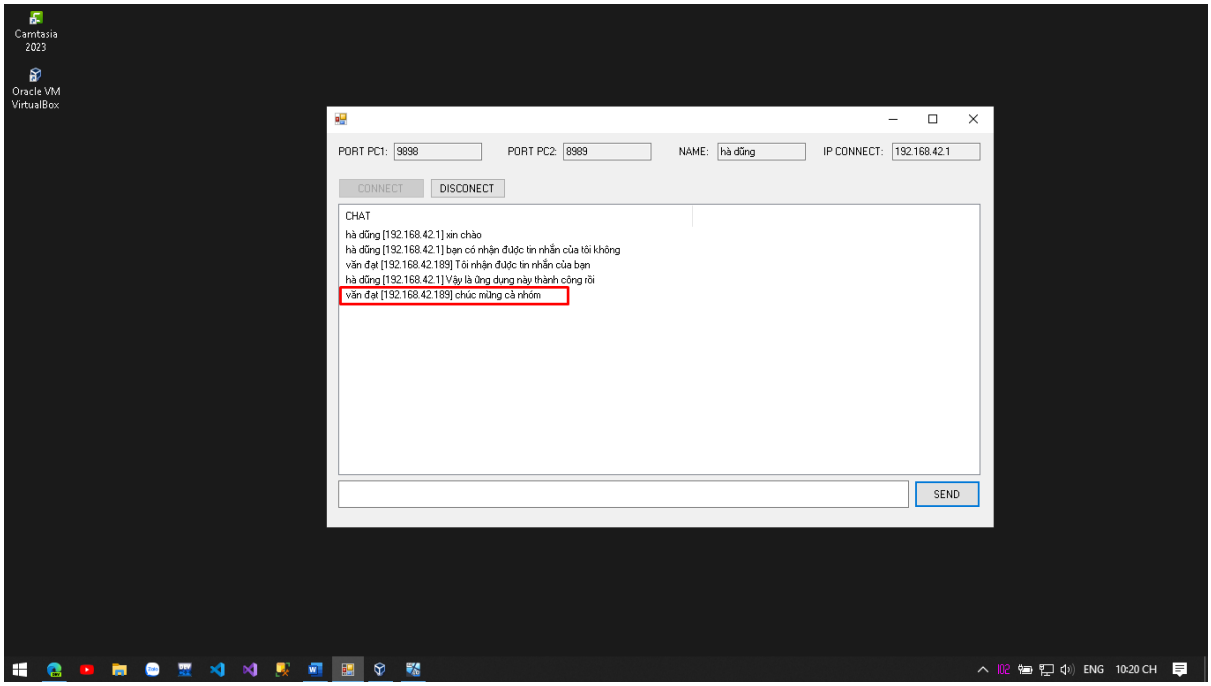


Hình 9: Soạn tin nhắn và click nút gửi của ứng dụng ở máy ảo



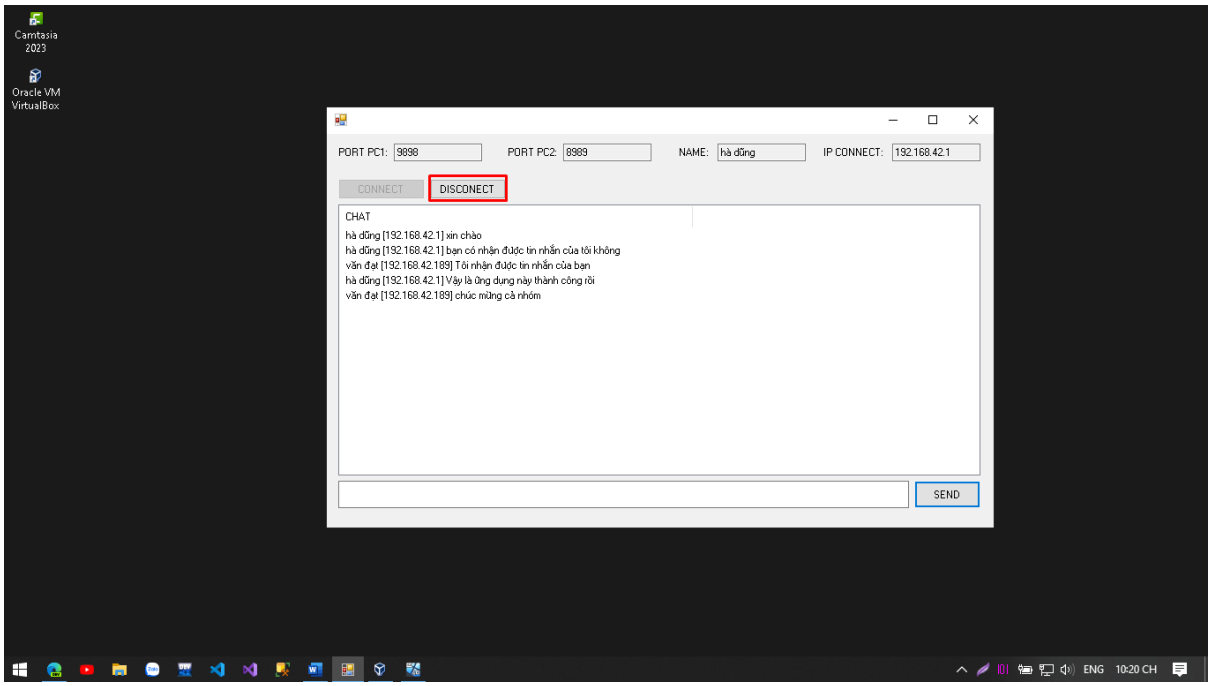
Hình 9.1 : Tin nhắn sau khi gửi sẽ hiện lên khung chat

- Tin nhắn hiện thi ở khung chat của ứng dụng ở máy chính sau khi ấn gửi từ ứng dụng ở máy ảo.



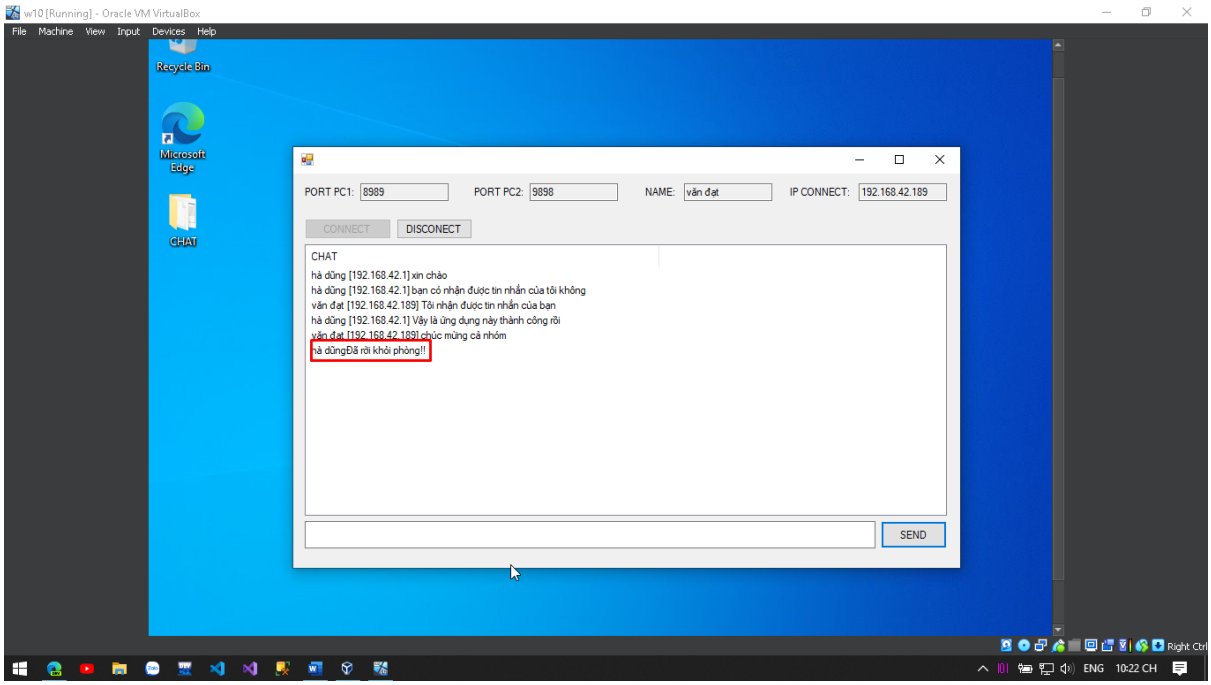
Hình 9.2: ở khung chat ứng dụng ở máy chính cũng sẽ hiện tin nhắn khi nhận được tin nhắn ở ứng dụng máy ảo

- Sau khi chat xong ấn nút Disconnect.



Hình 10 : Ấn nút Disconnect của ứng dụng ở máy chính

- Khi một máy ấn nút Disconnect thì máy kia sẽ nhận được thông báo máy đó đã rời khỏi phòng.



Hình 10.1: ở khung chat ứng dụng ở máy ảo sẽ nhận được thông báo sau khi ứng dụng ở máy chủ ấn Disconnect

Hạn chế và hướng phát triển của đề tài.

Chương trình ứng dụng trong quá trình lập trình có thể sẽ không tối ưu do chưa được tiếp cận với các dự án thực tế.

Nhóm chúng em sẽ cập nhật và nâng cấp lại hệ thống khi đã có nhiều kiến thức mới và được tiếp cận với nhiều thứ mới hơn sau này.

Tài liệu tham khảo

Một số kiến thức lý thuyết tham khảo qua công cụ BingAi
https://bkhost.vn/blog/giao-thuc-udp/
https://codelearn.io/sharing/lap-trinh-ung-dung-chat-noi-bo-don-gian-voi-socket
https://tuhocict.com/thuc-hanh-lap-trinh-socket-udp-co-ban