**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗‗**

****

**LẬP TRÌNH TRỰC QUAN**

**BÁO CÁO ĐỒ ÁN CHAT TRONG MẠNG LAN**

**NHÓM MAV:**

TRẦN LÊ NGỌC MAI 17520106  
 HÀ THỊ ANH 17520230  
 LÊ BÁ VƯƠNG 17521277

* **LỚP:  IT008.J11**

**Chương 1: MỞ ĐẦU**

* 1. **Lý do chọn đề tài:**

Hiện nay, mạng LAN đã có những tiến bộ vượt bậc và ngày càng phổ biến hơn trong đồi 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.

* 1. **Mục đích:**

Xây dựng chương trình Chat hoạt động trong mạng LAN với chức năng cơ bản gửi tin nhắn.

* 1. **Đối tượng và phạm vi nghiên cứu:**
     1. **Đối tượng nghiên cứu**

Tìm hiểu được cơ chế hoạt động của Socket và Thread trong .NET Framwork từ đó viết ứng dụng Chat trong mạng LAN.

* + 1. **Phạm vi nghiên cứu**

Chương trình Chat được xây dựng với khả năng gởi các được văn bản qua lại giữa các user thông qua sự điều khiển của một Server trong mạng LAN.

**Chương 2: KIẾN THỨC ỨNG DỤNG**

* 1. **Lập trình Socket**
     1. **Địa chỉ và Cổng (Address & Port)**

Mỗi ứng dụng sẽ được gán một số cổng (Port) từ 1 – 65535. Khi một ứng dụng khác muốn gửi tin đến thì điền thêm số cổng vào gói tin. Ứng dụng đích sẽ kiếm tra giá trị cổng trên gói tin và xử lý nếu trùng với số cổng của nó.

* + 1. **Lớp IPAddress**

Trên Internet mỗi một trạm (máy tính, thiết bị,..) đều có một định danh duy nhất được gọi là Address. Address trên Internet là một tập hợp gồm 4 con số có giá trị từ 0 – 255 và cách nhau bởi dấu chấm.

Trong .NET, IPAddress là một lớp dùng để mô tả địa chỉ này. Đây là lớp rất cơ bản được sử dụng khi chúng ta thao tác (truyền) vào các lớp như IPEndpoint, UDP, TCP, Socket …

Hàm IPAdress.Any() cung cấp một địa chỉ IP để chỉ ra rằng Server phải lắng nghe các hoạt động của Client trên tất cả các mạng.

* + 1. **Lớp IPEndPoint**

Trên Internet, để hai trạm có thể trao đổi thông tin được với nhau thì chúng cần phải biết được địa chỉ (IP) của nhau và số cổng (Port). Lớp IPEndPoint là một lớp chứa đựng cả IP Address và Port Number.

Constructor IPEndPoint(IPAdress, int) tạo một đối tượng IPEndPoint mới có tham số truyền vào là một địa chỉ IP và số cổng (Port number).

* 1. **Socket**

Sau khi Client đã Connect và Server đã Accept thì hai bên có thể trao đổi với nhau thông qua Socket. Socket trong C#, thông qua namespace System.Net.Socket, là một API (Application Programming Interface) cung cấp các phương thức cơ bản để giao tiếp qua mạng. Mọi dữ liệu gửi đi sẽ được lưu trữ vào Socket qua phương thức socket.Send(data), sau đó với phương thức socket.Receive(data), phía còn lại sẽ nhận dữ liệu về.

* 1. **Lập trình đa luồng (Thread)**

Một Thread là một chuỗi liên tiếp những công việc trong chương trình. Trong C#, Namespace System.Threading sẽ cung cấp cho ta lớp Thread. Lớp này tạo một vỏ bọc hướng đối tượng bao quanh một App nào đó trong suốt quá trình thực thi chương trình. Lớp Thread còn định nghĩa một số hàm cho phép ta tạo mới, Start, Stop, Kill một luồng nào đó.

**Chương 3: PHÂN TÍCH THIẾT KẾ CHƯƠNG TRÌNH**

**3.1 Yêu cầu đặt ra**

Yêu cầu đặt ra là xây dựng chương trình Chat hoạt động trong mạng Lan sử dụng Socket và Multithreading đòi hỏi các chức năng nghiệp vụ sau:

* Chat riêng tư giữa hai người với nhau.
* Chat chung giữa tất cả mọi người đang online.

**3.2 Các thành phần xử lý**

**3.2.1 Xử lý đăng nhập**

Khi một User tham gia vào Chat sẽ chọn một tên riêng dùng để phân biệt với mọi users khác, nếu tên người này chọn chưa có trong hệ thống thì Server sẽ lưu thông tin User (name) này vào hệ thống và thông báo đến các users khác là User này mới online.

**3.2.2 Xứ lý đăng xuất**

Khi một User thoát khỏi Chat thì Server sẽ kiểm tra xem User này có tồn tại không, nếu có sẽ xóa thông tin User này khỏi hệ thống và thông báo đến các users khác là User đã offline.

**3.2.3 Xử lý gửi tin**

**a/ Giữa tất cả mọi người**

Với button Public Chat được chọn, khi người dùng nhập tin nhắn và nhấn gửi (phím Enter hoặc button Send), Client sẽ thêm tin nhắn đó vào màn hình Chat, đồng thời gửi tin nhắn và tên người gửi lên cho Server. Server sau đó sẽ gửi tin nhắn lại cho tất cả các user đang online (trừ người gửi). Khi nhận được tin nhắn, tất cả các Client sẽ hiển thị tin nhắn lên màn hình Chat.

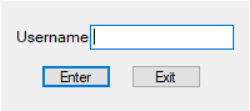
**b/ Giữa hai người với nhau**

Khi muốn nhắn tin riêng với một người nào đó, người dùng sẽ chọn tên người đó trong List users online và nhấn nút Private Chat. Sau khi button Private chat được chọn, màn hình sẽ thông báo cho cả 2 Users là họ đang chat riêng tư với người còn lại. Người dùng chỉ cần nhập tin nhắn và nhấn gửi, Client sẽ hiển thị tin nhắn đó vào màn hình Chat với từ khóa (Private) ở trước, và gửi tin nhắn đó cho Server. Server sau đó sẽ tìm trong list user tên người còn lại và gửi tin lại cho người đó. Khi nhận được tin nhắn, Client đó cũng sẽ hiển thị tin nhắn lên màn hình với từ khóa (Private).

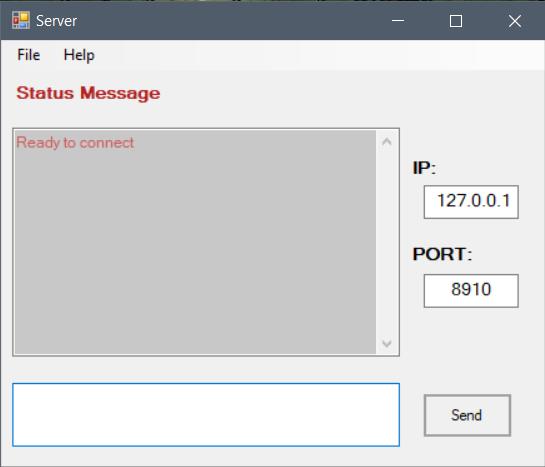
Nếu một trong hai người muốn kết thúc cuộc trò chuyện riêng tư, họ chỉ việc nhấn chọn button Public Chat, màn hình sẽ thông báo là họ đã trờ lại Chat Public.

**3.3 Thiết kế giao diện**

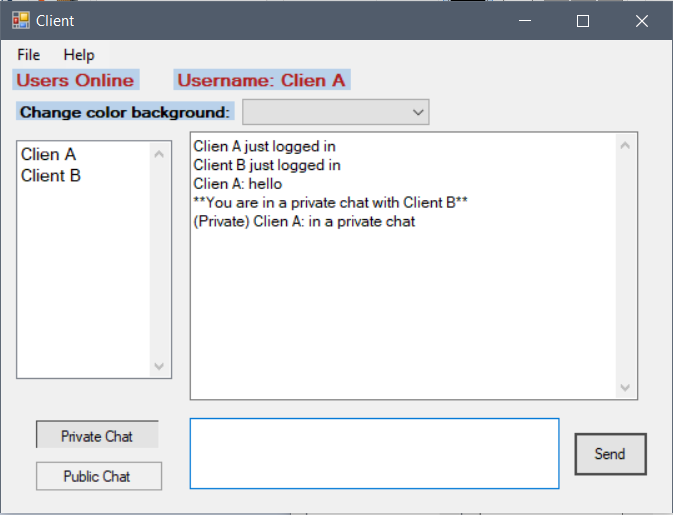
**3.3.1 Màn hình login**

****

**3.3.2 Màn hình Server**

****

**3.3.3 Màn hình Client**

****

**Chương 4: KẾT LUẬN**

**4.1 Kết quả đạt được**

* Tìm hiểu được về Socket, IPEndPoint và Thread trong môi trường .NET
* Tìm hiểu được cách thức hoạt động của một chương trình chat LAN đơn giản
* Tuy nhiên vẫn còn tồn tại các nhược điểm:
  + Chưa có database chứa thông tin (Username và Password) của người dùng
  + Chưa hỗ trợ chức năng gửi hình ảnh và File
  + Màn hình chat riêng tư chưa được tối ưu hóa

**4.2 Hướng phát triển**

Bên cạnh các chức năng cơ bản đã hoàn thành, nếu có điều kiện, đồ án sẽ cố gắng nghiên cứu thêm các chức năng sau:

* Tạo màn hình pop up mới khi Chat riêng tư với ai đó
* Tạo database lưu trữ thông tin người dùng
* Hỗ trợ chức năng gửi File và hình ảnh