|  |
| --- |
| Trường Đại Học Sư Phạm Kỹ Thuật  Khoa Đào Tạo Chất Lượng Cao  Ngành Công Nghệ Thông Tin    BÁO CÁO ĐỒ ÁN 1  Đề tài: Xây dựng ứng dụng chat  sử dụng giao thức TCP  **Giảng viên hướng dẫn:**  **Nhóm sinh viên thực hiện:**   * Lê Bảo Châu - 15110017 * Nguyễn Hữu Khang - 15110062   **Tp. Hồ Chí Minh, tháng 11 năm 2018** |

**Nhận xét của giáo viên hướng dẫn**

**………………………………………………………………………………………………………………………………………………………………………..………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………**

**Nhận xét của giáo viên hướng dẫn**

**………………………………………………………………………………………………………………………………………………………………………..………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………**

Lý do chọn đề tài

Bước qua thế kỉ 21, công nghệ thông tin ngày càng phát triễn một cách nhanh chống và mạnh mẽ, nó đã góp một phần rất lớn trong sự phát triển của nhân loại, nâng cao chất lượng sống của con người. Một trong những phát triển mang tính then chốt nhất của công nghệ thông tin đó chính là việc giúp cho con người dễ dàng giao tiếp, trao đổi thông tin, tài liệu với nhau hơn thay vì sử dụng các phương tiện giao tiếp đã trở nên quá lỗi thời, bất tiện và tốn nhiều thời gian như trước đây.

Ứng dụng chat chính là một trong những phương tiện giao tiếp đã trở nên quá quen thuộc và là một phần không thể thiếu của cuộc sống hiện nay, vậy nên chúng em chọn đề tài “TCP CHAT APPLICATION” để tìm hiểu và cùng thực hiện với mong muốn được vận dụng và củng cố lại những kiến thức đã học, tiếp thu thêm những điều mới và sáng tạo thêm những ứng dụng có ích cho xã hội.

**Mục lục**

1. Giới thiệu đề tài:
2. Tên đề tài:

Xây dựng ứng dụng chat sử dụng giao thức TCP.

1. **Ngữ cảnh sử dụng phần mềm:**

Ứng dụng dùng để kết nối tới một server máy chủ do người dùng tùy chỉnh, từ đó tạo nên một môi trường giúp cho người dùng có thể trò chuyện hoặc trao đổi tài liệu với một hay nhiều người trong cùng server.

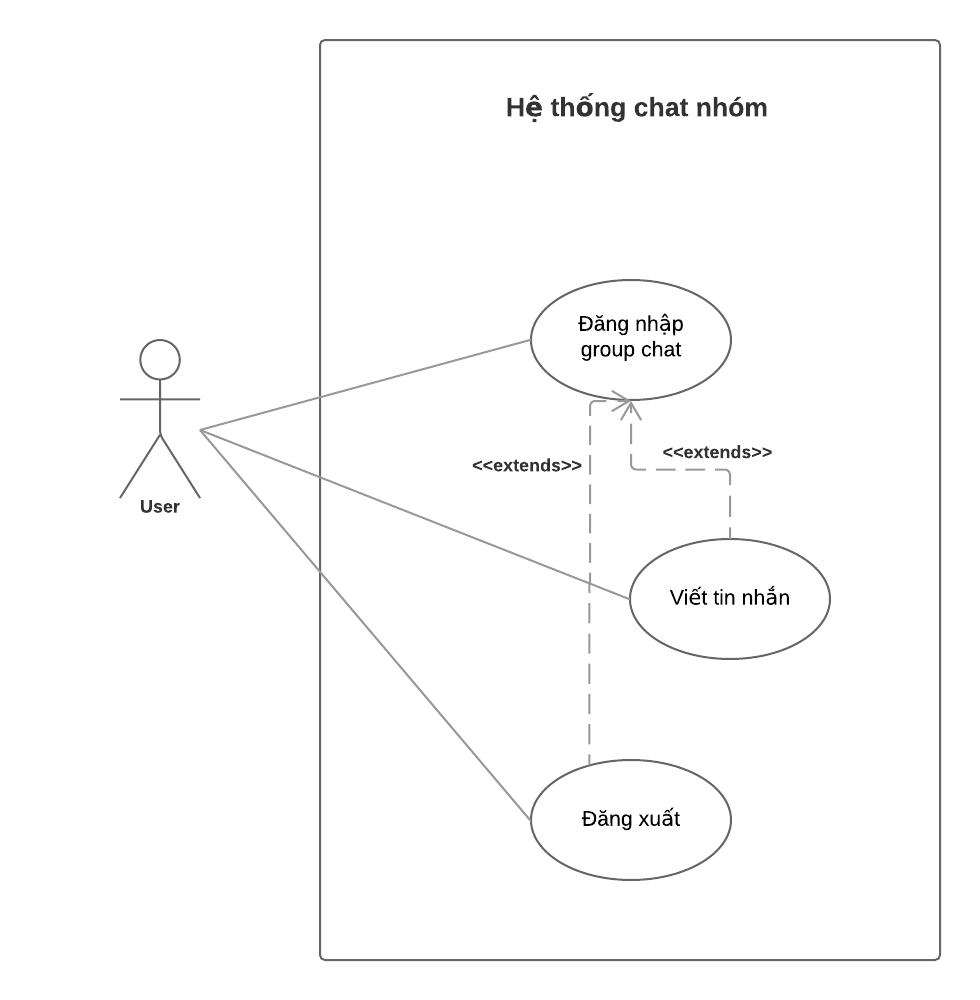
1. Mô tả ứng dụng:

* Tên ứng dụng: TCP Chat
* Ngôn ngữ lập trình: Java
* Thư viện sử dụng cho giao diện: Swing
* Môi trường lập trình: Eclipse
* Hướng lưu trữ dữ liệu: File định dạng .xml
* Môi trường thực thi ứng dụng: Hệ điều hành Windows

1. Đặc điểm, tính năng:

* Đăng nhập.
* Cho phép nhiều user đăng nhập cùng một lúc.
* Hỗ trợ cho cả tin nhắn riêng tư hoặc tin nhắn nhóm.

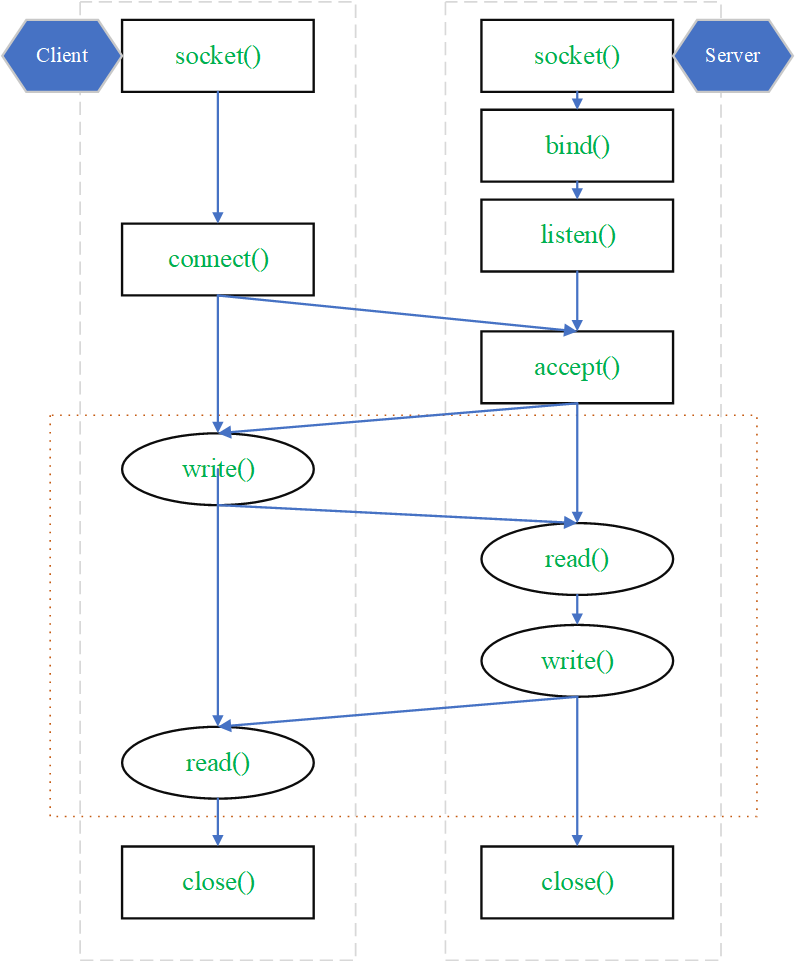
1. **Lược đồ Use Case:**



1. Cấu trúc project:
2. Công nghệ Socket

Socket là phương tiện để xây dựng các ứng dụng theo kiến trúc Client-Server.

Đây là mô hình Client-Server sử dụng Socket ở chế độ kết nối TCP.



Ta chia ra 4 giai đoạn:

* Giai đoạn 1: Server tạo Socket, chọn cổng để lắng nghe yêu cầu kết nối. Server sẽ phục vụ Client socket với các dịch vụ như:

- bind(): Server yêu cầu gắn port cho Socket.

- listen():Server lắng nghe các yêu cầu kết nối từ client trên port đã gắn

* Giai đoạn 2: Client tạo Socket và yêu cầu kết nối đến Server với địa chỉ IP và Port nhất định. Khi Server chập nhận kết nối của Client, khi đó một kênh giao tiếp ảo được hình thành, Client và Server có thể trao đổi thông qua kênh ảo này.
* Giai đoạn 3: Trao đổi thông tin giữa Client và Server. Sauk hi chấp nhận yêu cầu kết nối, thông thường Server thực hiện read() và nghẽn cho đến khi Client gửi yêu cầu. Server sẽ phân tích và thực thi yêu cầu. Kết quả sẽ được gởi về client bằng lệnh write() của Server, còn Client sẽ nhận kết qua bằng lệnh read().
* Giai đoạn 4: Kết thúc phiên làm việc. Kênh ảo sẽ bị xóa khi Server và Client đóng socket bằng lệnh close().

2. Hệ thống Server:

Mục đích: Hệ thống Server có nhiệm vụ thiết lập một phòng chat cho tất cả các Client kết nối đến có thể trò truyện được với nhau.

Cơ chế:

* Hệ thống Server bao gồm một Socket và một Vector Clietns để toàn bộ lưu địa chỉ các client đang vào trong chat nhóm.
* Khi khởi động, Server sẽ mở kênh ảo ở port 5555 và bắt đầu nhận yêu cầu kết nối của Client.
* Khi Client gửi yêu cầu kết nối. Server sẽ lưu lại địa chỉ của Client để phục vụ viết gửi tin sau này.
* Khi Client gửi tin nhắn. Server sẽ lấy toàn bộ địa chỉ các Client hiện tại trong kênh tin nhắn và gửi tin cho tất cả.
* Khi Client thoát thì Server sẽ xóa địa chỉ Client trên mảng Vector Clients.

3. Hệ thống Client:

Mục đích: Hệ thống Client có nhiệm vụ kết nối với server và tạo một tài khoản cho người dùng và sau đó đưa client mới tạo vào kênh trò truyện để bắt đầu chat nhóm.

Cơ chế: Hệ thống chia ra 2 Form

* Form đăng nhập: Nhiệm vụ là tạo một Form cho người dùng nhập tên của Client vào. Sau khi người dùng đăng nhập, thì khời tạo Form chat.
* Form chat: nhận tên nhập của người dùng. Form chat sẽ tạo ra một Socket cho Client để kết nối Socket của Server. Form cho một bảng hiện thị lịch sử tin nhắn chat và input nhập tin nhắn để gửi cho Server. Sau khi người dùng thoát thì Form Chat gửi thông tin đến cho Server để loại Client ra khỏi kênh chat.

1. Quá trình thực hiện
2. Thiết kế giao lớp cho Server
   1. Các thư viện được sử dụng trong lớp Server

|  |  |
| --- | --- |
| Thư viện | Ý nghĩa |
| import java.io.\* | Thư viện cung cấp các hàm xử lý vào/ra trên các thiết bị chuẩn và thiết bị ngoại vi |
| import java.net.\* | Thư viện để lập trình mạng để xử lý Socket TCP/IP |
| import java.util.\* | cung cấp các hàm tiện ích xử lý liên quan đến kiểu dữ liệu có cấu trúc như Date, Stack, Vector. |

* 1. . Khai báo class Server

|  |  |  |
| --- | --- | --- |
| Dòng | Code | Giải thích |
| 5 | class Server implements Runnable { | Khai báo class Server có dùng interface Runnable để chạy bất đồng bộ |
| 6 | Socket connectionSocket; | Khai báo biến connectionSocket thuộc kiểu Socket để tạo socket chat |
| 7 | public static Vector clients = new Vector(); | Khai báo mảng Vector clients để lưu những người dùng kết nối đến Server chat. |

* 1. . Mô tả các phương thức trong lớp Server

1.3.1. Hàm khởi tạo

|  |  |  |
| --- | --- | --- |
| Dòng | Code | Giải thích |
| 8 | public Server (Socket s) { | Khai báo hàm khởi tạo lớp Server với tham số truyền vào là một biến Socket s |
| 9, 10, 11 | Try{  System.out.println(“Server started”;  connectionSocket = s;  } | Viết trong hàm try catch tránh lỗi. Ta sẽ in ra màn hình server đã khởi động và gán biến Socket s vào biến connectionSocket. |

1.3.2. Hàm run()

|  |  |  |
| --- | --- | --- |
| Dòng | Code | Giải thích |
| 16 | BufferedReader reader = new BuffererReader(new InputStreamReader(  connectionSocket.gettInputStream()));) | Tạo biến reader lấy từ connectionSocket. Nhiệm vụ của reader dùng để nhận tin nhắn từ Clients |
| 17 | BufferWriter writer = new BufferWriter(new OutputStreamWriter(  connectionSocket.getOutputStream())); | Tạo biến biến writer khi có một client tham gia vào group chat. Đây chính là biến writer của client để có thể viết tin nhắn đến client đó. |
| 18 | clients.add(writer); | Đưa biến writer vào mảng clients. Khi có tin nhắn nào gửi đến tao dùng các writer này để gửi tin cho những người dùng khác. |
| 19 | While (true) { | Đây là vòng lặp vô tận. |
| 23 | String data1 = reader.readLine().trim(); | Biến data1 nhận tin nhắn khi có thông báo đến qua biến reader. |
| 24 | System.out.println(“Received: “+data1); | In thử ra ra tin nhắn |
| 25 | for (int i=0; i<clients.size(); i++){ | Chạy vòng lặp với số lượng client đang có trong group chat |
| 27,28 | try{  BufferedWriter bw = (BufferedWriter) clients.get(i); | Đưa vào trong try catch. Lấy BufferedWriter của client ở vị trí thứ i trong mảng |
| 29,30 | bw.writer(data1);  bw.write(“\r\n”); | Gửi tin nhắn và xuống dòng để hiện thị trong bảng chát |
| 31 | bw.flush() | Gọi phương thức flush để gửi tin nhắn đi. |

1.3.3. Hàm main()

|  |  |  |
| --- | --- | --- |
| Dòng | Code | Giải thích |
| 38 | System.out.println(“Threaded Chat Server is Running “); | In chương trình đang khởi động |
| 39 | ServerSocket mysocket = null; | Khai báo một biến socket thuộc đối tượng ServerSocket |
| 40,41 | try {  mysocket = new ServerSocket(5555); | Khởi tạo biến mysocket với cổng 5555. Đây là cổng để user kết nối vào group chat |
| 42 | while (true) { | Vòng lặp vô hạn |
| 43 | Socket sock = mysocket.accept() | mysocket sẽ lắng nghe xem có kết nối server không rồi chấp nhận. Sau đó trả về một socket |
| 44 | Server server = new Server(sock); | Tạo một server từ socket đã nhận |
| 45 | Thread serverThread = new Thread(server); | Tạo một tiến trình cho server để chạy bất đồng bộ |
| 46 | serverThread.start(); | Gọi hàm start để chạy server |
| 48-53 | catch(Exception e){}  finally {  if (mysocket != null) {  mysocket.close();  }  } | Sau khi chạy xong thì tắt server Socket. |

1. Thiết kế giao lớp cho Client
   1. . Các thư viện được sử dụng trong lớp Client

|  |  |
| --- | --- |
| Thư viện | Ý nghĩa |
| import javax.swing.\*; | Thư viện thiết kế giao diện cho Java |
| import javax.swing.border.TitledBorderr | Thư viện tạo đường viền quanh giao diện |
| import java.awt.\*; | Thư viện thiết kế giao diện cho Java |
| import java.awt.event.ActionEvent; | Event bắt sự kiện nhấn button |
| import java.awt.event.ActionListener; | Hàm thực hiện các sự kiện |
| import java.awt.event.KeyAdapter; | Bắt xự kiện nhấn bàn phím |
| Import java.awt.event.KeyEvent | Thư viện lưu các tên trên bàn phím |

* 1. . Khai báo class Client

|  |  |  |
| --- | --- | --- |
| Dòng | Code | Giải thích |
| 10 | JFRAME main; | Khai báo biến main thuộc JFRAME để thiết kế giao diện ứng dụng |
| 11 | JTextField txtUser; | Khai báo biến txtUser thuộc JTextField để tạo nhập tài khoản user |
| 12 | JButton btnDangNhap, btnThoat; | Khai báo 2 biến btnDangNhap, btnThoat thuộc JButton để tạo 2 button đăng nhập và thoát |
| 13 | mychat c = null | Khai báo đối tượng c thuộc class mychat |

2.3. Hàm Client()

|  |  |  |
| --- | --- | --- |
| Dòng | Code | Giải thích |
| 15 | Dimension sreenSize = Toolkit.getDefaultToolkit().  getScreenSize(); | Lấy kích thước màn hình của máy tính hiện tại |
| 16 | double width = screenSize.getWidth(); | Lấy chiều ngang màn hình |
| 17 | double height = screenSize.getHeight(); | Lấy chiều dọc màn hình |
| 19 | main = new JFrame(“Đăng nhập”); | Tạo khung màn hình ứng dụng có tiêu để phía trên là “Đăng nhập” |
| 20 | main.setBounds((int) width/2-250, (int) height/2-200, 500, 400); | Đặt tọa độ cho khung main có chiều ngang là 500 và chiều dọc 400. Tọa độ nằm ở vị trí trung tâm màn hình khi chia chiều cao và chiều ngang màn hình ra làm đôi rồi trừ đi một nửa chiều cao và chiều ngang của chính nó. |
| 21 | main.setResizable(fasle); | Không cho Frame main có thể resize |
| 22 | main.setLayout(null); | Không tạo layout cho Frame main |
| 23 | main.setVisible(true); | Cho phép JFrame main hiển thị |
| 24 | main.setDefaultCloseOperation  (JFrame.EXIT\_ON\_CLOSE); | Thiết lập khi tắt Frame main thì chương trình sẽ tắt |
| 26 | JLabel label = new JLabel(“Chat nhóm”); | Tạo một label có tiêu đề là “Chat nhóm” |
| 27 | Label.setFont(new Font(“Courier New”, Font.BOLD, 25)); | Xét Font cho Label với kiểu “Courier New”, in đậm và có size là 25 |
| 28 | Label.setBounds(170, 20, 400, 50); | Đặt tọa độ (170,20) trên Frame main và có chiều dài 400, chiều cao 50 |
| 29 | main.add(label); | Đưa label vào Frame main |
| 31 | JPanel panel = new Jpanel(); | Tạo một Panel (1 khung nhỏ trên Frame) |
| 32 | panel.setBorder(new TitledBorder(null, “Viết tên đăng nhập”)); | Đặt khung viền quanh panel với tiêu đề phía góc là “Viết tên đăng nhập” |
| 33 | panel.setBounds(50,100, 400,150); | Đặt panel ở tọa độ (50,100) với chiều ngang 100, chiều cao 150 |
| 34 | panel.setLayout(null); | Không đặt layout cho panel |
| 35 | JLabel label1 = new JLabel(“Tên người dùng”); | Tạo label có tiêu đề “Tên người dùng” |
| 36 | label1.setBounds(10, 40, 150, 20); | Cho label1 ở tọa độ (10,40) có chiều ngang 150, chiều cao 20 |
| 37 | panel.add(label1); | Đưa label1 vào panel |
| 38 | txtUser = new JTextField(); | Khởi tạo txtUser là nơi nhập tên user |
| 39 | txtUser.setBounds(170, 40, 200,20); | Cho txtUser ở tọa độ (170,40) có chiều ngang 200, chiều cao 20 |
| 40 | panel.add(txtUser); | Đưa txtUser vào panel |
| 43 | btnDangNhap = new JButton(“Đăng nhập”); | Khởi tạo btnDangNhap là button Đăng nhập |
| 44 | btnDangNhap.setBounds(80, 260, 120, 60); | Cho bntDangNhap ở tọa độ (80,260) có chiều ngang 120, chiều cao 60 |
| 45 | btnThoat = new JButton(“Thóat”); | Khởi tạo btnThoat là button thoát chương trình |
| 46 | btnThoat.setBounds(280, 260, 120, 60); | Cho btnThoat ở tọa độ (280, 260) có chiều ngang là 120, chiều cao là 60 |
| 47 | main.add(btnDangNhap); | Đưa btnDangNhap vào main |
| 48 | main.add(btnThoat); | Đưa btnThoat vào main |
| 50 | txtUser.addKeyListener(new KeyAdapter() {  @Override  pulibc void keyPressed(KeyEvnet e) {  if (e.getKeyCode()==KeyEvent  .VK\_ENTER){  DangNhap();  }  }  }); | Bắt sự kiện nhấn phim ở txtUser nếu nhấn phim Enter thì gọi hàm DangNhap |
| 58 | bntDangNhap.addActionListener(  new ActionListerner() {  @Override  public void actionPerformed(ActionEvent e){  DangNhap();  }  }); | Bắt sự kiện nhấn nút DangNhap thì gọi hàm DangNhap |
| 64 | btnThoat.addActionListener(new ActionListener() {  @Override  pubic void  actionPerformed(ActionEvent e){  System.exit(0);  }  }); | Bắt sự kiện nhấn nút Thoát thì sẽ tắt chương trình |

2.4. Khai báo hàm DangNhap()

|  |  |  |
| --- | --- | --- |
| Dòng | Code | Giải thích |
| 73 | c = new mychat(txtUser.getText()); | Khởi tạo một đối tượng mychat c với tham số truyền vào là tên user người dùng nhập vào |
| 74 | Thread t1 = new Thread(c); | Khởi tạo một tiến trình để chạy đối tượng c |
| 75 | t1.start(); | Bắt đầu chạy tiến trình |
| 76 | main.dispose(); | Tắt giao diện JFrame main để bắt đầu mở giao diện chat trong đối tượng c |

2.5. Khai báo hàm main()

|  |  |  |
| --- | --- | --- |
| Dòng | Code | Giải thích |
| 80 | EventQueue.invokeLater(new Runnable() {  @Override  public void run() {  Client form = new Client();  }  }); | Tạo một tiến trình đưa vào EventQueue để đảm bảo an toàn. Trong tiến trình sẽ khởi tạo đối tượng form thuộc Client |

1. Thiết kế giao lớp cho mychat
   1. . Các thư viện được sử dụng trong lớp Client

|  |  |
| --- | --- |
| Thư viện | Ý nghĩa |
| Import javax.swing.\*; | Thư viện thiết kế giao diện cho Java |
| import java.awt.\*; | Thư viện thiết kế giao diện cho Java |
| import java.awt.event.\*; | Thư viện xử lý sự kiện của giao diện awt cho Java |
| import java.io.\* | Thư viện cung cấp các hàm xử lý vào/ra trên các thiết bị chuẩn và thiết bị ngoại vi |
| import java.net.\* | Thư viện để lập trình mạng để xử lý Socket TCP/IP |

3.2. Khai báo class Client

|  |  |  |
| --- | --- | --- |
| Dòng | Code | Giải thích |
| 6 | public class mychat implements Runnable { | Class mychat gọi interface Runnable để xử lý bất đồng bộ |
| 7 | pulic JTextField tx; | Biến tx thuộc JTextField dùng để ghi nội dung tin nhắn |
| 8 | public JTextField ta; | Biến ta thuộc JTextField dùng để hiện thị lịch sử tin nhắn |
| 9 | public String login; | Biến login dùng để lưu tên người dùng hiện tại |
| 19 | BufferedWriter writer; | Biến writer dùng để gửi tin nhắn đến server |
| 20 | BufferdReader reader; | Biến reader dùng để đọc tin nhắn khi server gửi đến |
| 21 | Socket socketClient=null; | Biến socketClient dùng để kết nối đến server |

3.3. Hàm mychat(String l)

|  |  |  |
| --- | --- | --- |
| Dòng | Code | Giải thích |
| 15 | login = l; | Gắn biến login vào tham số l (lưu tên người dùng hiện tại) |
| 17 | JFrame f = new JFrame(login); | Tạo khung màn hình ứng dụng có tiêu để phía trên là tên người dùng |
| 18 | f.setSize(400, 400); | Đặt kích thước khung có chiều ngang là 400, chiều cao là 400 |
| 20 | JPanel p1 = new JPanel(); | Tạo một Panel p1 (1 khung nhỏ trên Frame) |
| 21 | p1.setLayout(new BorderLayout()); | Cho p1 có layout thuộc dạng BorderLayout (layout thường các máy tính hiện tại chia khung phần mềm ra 5 phần Bắc-Trung-Nam-Đông-Tây) |
| 22 | JPanel p2 = new JPanel(); | Tạo một Panel p2 (1 khung nhỏ trên Frame) |
| 23 | p2.setLayout(new BorderLayout()); | Cho p2 có layout thuộc dạng BorderLayout (layout thường các máy tính hiện tại chia khung phần mềm ra 5 phần Bắc-Trung-Nam-Đông-Tây) |
| 26 | tx = new JTextField(); | Khởi tạo biến tx |
| 27 | p1.add(tx, BorderLayout.CENTER); | Đưa tx vào phần phía Trung trong p1 |
| 29 | JButton b1 = new JButton(“Send”); | Tạo một button b1 có nội dung là “Send” |
| 30 | p1.add(b1, BorderLayout.EAST); | Cho b1 vào phần phía Đông trong p1 |
| 32 | ta = new JTextArea(); | Khởi tạo biến ta |
| 33 | p2.add(ta, BorderLayout.CENTER); | Đưa ta vào phần trung tâm của p2 |
| 34 | p2.add(p1, BorderLayout.SOUTH); | Đưa p1 vào phần phía nam của p2 |
| 36 | f.setContentPane(p2); | Đưa p2 vào JFrame f |
| 37 | f.setDefaultCloseOperation(  JFrame.EXIT\_ON\_CLOSE); | Khi tắt giao diện thì đồng thời tắt chương trình |
| 38 | f.addWindowListener(new WindowAdapter() {  @Override  public void WindowClosing( windowEvent e) {  close();  }  }); | Khi người dùng tắt phần mềm thì chạy hàm close() rồi mới tắt |
| 46 | socketClient = new Socket(“localhost”, 5555); | Khởi tạo socketClient kết nối đến localhost với cổng 5555 |
| 47 | writer = new BufferedWriter(new OutputStreamWriter( socketClient.getOutputStream())); | Lấy Writer từ socketClient để gửi tin nhắn sau này |
| 48 | reader = new BufferedReader(new InputStreamReader(  socketClient.getInputSteam(), “UTF8”)); | Lấy Reader từ socketClient để nhận tin nhắn từ Server |
| 53 | tx.addKeyListener(neww KeyAdapter() {  @Override  public void keyPressed(KeyEvent e){  if (e.getKeyCode()  ==KeyEvent.VK\_ENTER) {  send();  }  }  }); | Bắt sự kiện nhấn phím từ tx, nếu nhấn phím Enter thì gọi hàm send |
| 62 | b1.addActionListener(new ActionListerner() {  Public void actionPerformed(ActionEvent ev){  send();  }  }  }); | Bắt sự kiện button b1, nếu nhấn thì gọi hàm send |
| 69 | f.setVisible(true); | Cho phép JFrame f hiển thị |

3.4. Hàm send()

|  |  |  |
| --- | --- | --- |
| Dòng | Code | Giải thích |
| 74 | String s = login+ “ : “+tx.getText(); | s sẽ lưu đoạn thông tin để gửi bao gồm tên người dùng ghép với tin nhắn |
| 75 | tx.setText(“”); | Xóa tin nhắn trên thanh ghi tx |
| 77 | writer.write(s); | writer sẽ viết tin nhắn sẽ gửi |
| 75 | writer.write(“\r\n”); | writer sẽ thêm xuống dòng để kết thúc tin nhắn |
| 79 | writer.flush(); | writer sẽ gửi tin nhắn đến server |

3.5. Hàm run()

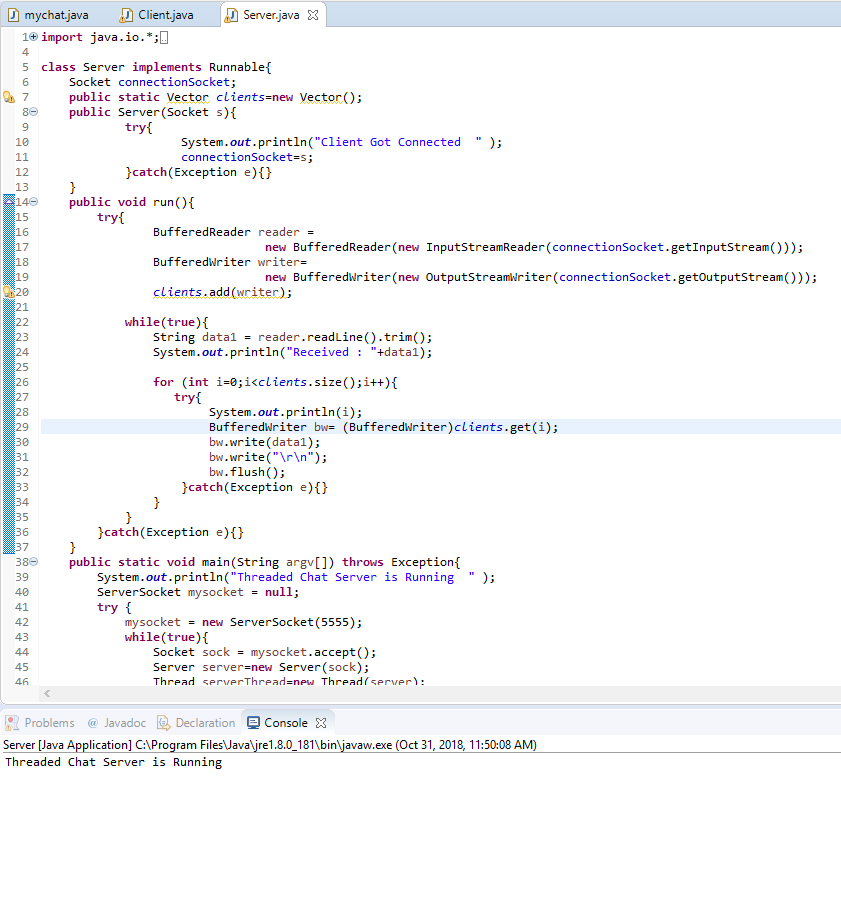
|  |  |  |
| --- | --- | --- |
| Dòng | Code | Giải thích |
| 84 | String serverMsg=””; | Khai báo biến serverMsg để nhận tin nhắn gửi từ Server |
| 85 | while((serverMsg = reader.readline()) != null) { | Kiếm tra xem Server có gửi tin nhắn gì đến Client chưa |
| 86 | System.out.println(“from server: “ + serverMsg); | Thử in ra tin nhắn trên console |
| 87 | ta.append(serverMsg+”\n”); | In ra trên của sổ lịch sử tin nhắn kèm xuống dòng |

3.6. Hàm close()

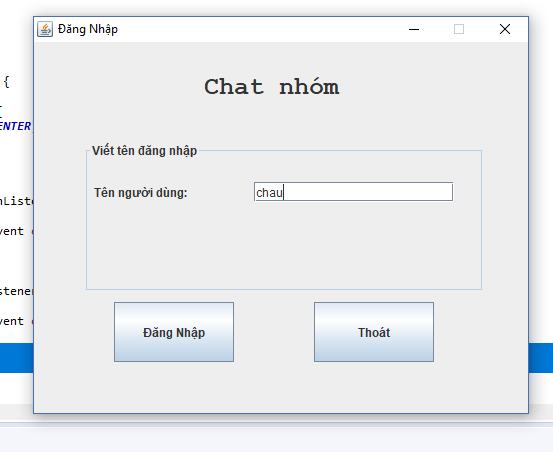
|  |  |  |
| --- | --- | --- |
| Dòng | Code | Giải thích |
| 93 | String s = login + “ had left group”; | Tạo đoạn nhắn user đang rời khỏi nhóm chat |
| 94..96 | writer.write(s);  writer.write(“\r\n”);  writer.flush(); | Gửi thông báo đến Server |
| 97 | socketClient.close(); | Đóng kết nối đến Server chat |

1. Demo chương trình:

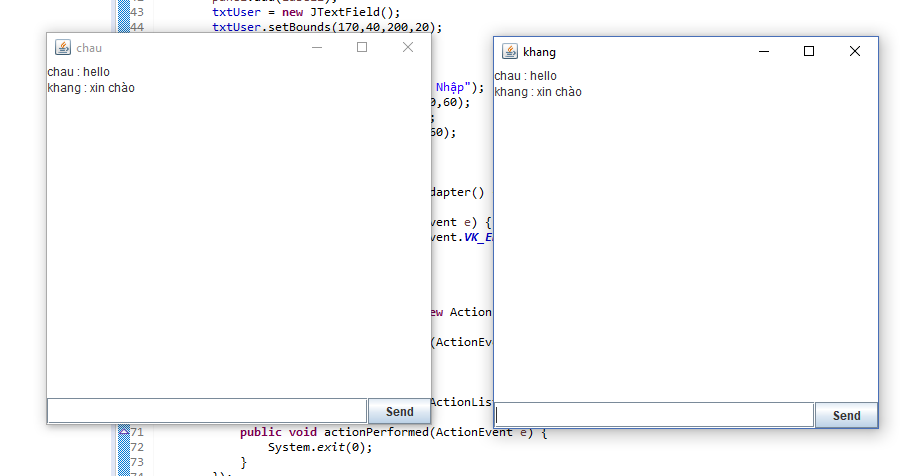
Bước 1: Khởi động Server (Server sẽ chạy ngầm trong hệ thống)



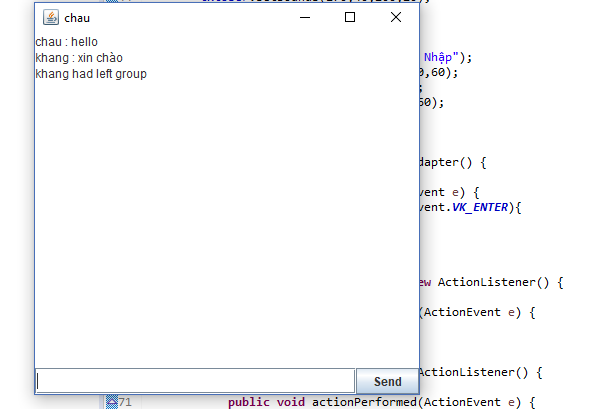
Bước 2: Mở Form đăng nhập và nhập tên user



Bước 3: Chuyển sang Form chat. Ta đồng thời tạo thêm 1 Form Client để có 2 Client chat với nhau



Bước 4: Thoát chương trình



1. Phân công công việc:

|  |  |  |
| --- | --- | --- |
| Tên SV | Phần trăm đóng góp | Nội dung đóng góp |
| Lê Bảo Châu | 60% | Tìm hiểu và phân tích đề tài, vẽ lược đồ Use case và chịu trách nhiệm code chính. |
| Nguyễn Hữu Khang | 40% | Tìm hiểu và phân tích đề tài, code một vài phần đơn giản, viết báo cáo. |

*TP.HCM, Ngày…. tháng…. năm …*

TRƯỞNG KHOA CNTT GIẢNG VIÊN HƯỚNG DẪN

(Ký và ghi rõ họ tên) (Ký và ghi rõ họ tên)