**BÁO CÁO BÀI TẬP LỚN MÔN LẬP TRÌNH MẠNG**



**Nhóm Network THT gồm các thành viên:**

|  |  |
| --- | --- |
| **Nguyễn Anh Tuấn** | **18021376** |
| **Nguyễn Văn Trường** | **18021337** |
| **Trương Việt Hoàng** | **17020776** |

# **Mục lục**

[**Mục lục**](#_rzi06bba5kh9) **2**

[**Ghi nhận thay đổi**](#_mpvbabnrw1jm) **3**

[**Giới thiệu**](#_i2fwbn85l349) **5**

[Tổng quan](#_16gol1jsmt2m) 5

[Công nghệ sử dụng](#_64nfg7lha0n1) 5

[**Thiết kế**](#_2qkt42raq0vd) **5**

[Thiết kế giao thức](#_cyak1etlbq1v) 5

[Thiết kế giao diện](#_4n2c6pha6pwr) 5

[**Hoạt động của ứng dụng**](#_k0h94lv84fn7) **7**

[**Phân tích code**](#_ruyrptcdimvj) **8**

[**Mục tiêu phát triển**](#_uddo3eda293a) **18**

# Ghi nhận thay đổi

**\*A - Added (bổ sung) M - Modified (sửa đổi) D - Deleted (xóa)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ngày** | **Thay đổi** | **A, M, D** | **Mô tả** | **Người thêm** |
| **01/12/2020** | **Thảo luận về ứng dụng** | **A** | **Thảo luận về các chức năng sẽ có của ứng dụng để thiết lập giao thức và phác thảo giao diện.** | **Cả nhóm** |
| **02/12/2020** | **Khung chương trình cơ bản** | **A** | **Khi kết nối với Server phải thêm định danh, xử lý giao tiếp giữa các client với nhau qua màn hình console.** | **Nguyễn Anh Tuấn** |
| **03/12/2020** | **Thông báo người dùng đang hoạt động** | **A** | **Khi vừa kết nối với Server sẽ thêm thông báo về các user khác đã kết nối.** | **Nguyễn Anh Tuấn** |
| **05/12/2020** | **Thiết kế giao thức** | **A** | **Tạo lập cách thức giao tiếp giữa Client với Server và giữa các Client với nhau.** | **Nguyễn Anh Tuấn** |
| **07/12/2020** | **Thiết kế giao diện cho Server** | **A** | **Phác thảo giao diện cho Server với các chức năng đã được quy ước.** | **Trương Việt Hoàng** |
| **09/12/2020** | **Thiết kế giao diện cho Client** | **A** | **Phác thảo giao diện cho Client với các chức năng đã được quy ước.** | **Nguyễn Văn Trường** |
| **12/12/2020** | **Thảo luận để thống nhất thiết kế** | **M** | **Thảo luận để đưa ra những quyết định chung về giao thức, giao diện, các chức năng.** | **Cả nhóm** |
| **13/12/2020** | **Định dạng lại code** | **M** | **Thay đổi lớp Server và Client thành các hàm riêng biệt theo từng chức năng để tiện cho thêm giao diện** | **Nguyễn Anh Tuấn** |
| **13/12/2020** | **Giao diện cho Server** | **A** | **Chuyển bản thiết kế phác thảo giao diện cho Server thành code. Có thể bật tắt Server từ giao diện.** | **Nguyễn Anh Tuấn** |
| **13/12/2020** | **Giao diện cho Client** | **A** | **Chuyển bản thiết kế phác thảo giao diện cho Client thành code. Thay đổi từ việc giao tiếp từ console sang giao tiếp bằng giao diện.** | **Nguyễn Anh Tuấn** |
| **13/12/2020** | **Chức năng upload file** | **A** | **Chuyển đổi giao thức bằng viết dòng lệnh thành các chức năng trên giao diện. Có thể upload file từ local.** | **Nguyễn Anh Tuấn** |
| **13/12/2020** | **Chức năng download file** | **A** | **Chuyển đổi giao thức bằng viết dòng lệnh thành các chức năng trên giao diện. Có thể download file về local.** | **Nguyễn Anh Tuấn** |
| **15/12/2020** | **Sửa một số lỗi** | **M** | * **Lỗi Server khi tắt không thể bật lại được.** * **Lỗi hiển thị các user đang hoạt động.** * **Lỗi phần chọn đối tượng tương tác.** | **Nguyễn Anh Tuấn** |
| **16/12/2020** | **Sửa file giao thức trên console** | **M** | **Giao thức chạy trên console đã được chuyển thành chức năng trên phần giao diện.** | **Nguyễn Anh Tuấn** |
| **16/12/2020** | **Báo cáo chi tiết** | **A** | **Báo cáo về các hoạt động đã làm được của nhóm.** | **Cả nhóm** |
| **16/12/2020** | **Thảo luận** | **A** | **Test chương trình, kiểm tra lỗi.** | **Cả nhóm** |
| **16/12/2020** | **Thêm folder storage** | **A** | **Folder được thêm để lưu trữ dữ liệu gửi lên server và dữ liệu client tải về.** | **Nguyễn Anh Tuấn** |
| **17/12/2020** | **Sửa lỗi** | **M** | **Sửa lỗi đọc được file trong tên có dấu space.** | **Nguyễn Anh Tuấn** |
| **17/12/2020** | **Sửa lỗi, Thêm log file** | **M, A** | * **Lỗi hiển thị target khi có user ngắt kết nối.** * **Thêm file log ghi lại dữ liệu về cuộc hội thoại.** | **Nguyễn Anh Tuấn** |
| **17/12/2020** | **Sửa lỗi** | **M** | **Sửa lỗi hiển thị danh sách user đang online.** | **Nguyễn Anh Tuấn** |
| **17/12/2020** | **Sửa lỗi** | **M** | **Sửa lỗi nhập sai username (có chứa ký tự ‘@’) vẫn kết nối được với Server.** | **Nguyễn Anh Tuấn** |

Dự án được quản lý trên github tại [đây](https://github.com/NAT1804/chat-application).[[1]](#footnote-0)

# Giới thiệu

Một ứng dụng chat đơn giản cho phép mọi người chat với nhau hay chuyển file cho nhau. Với các công nghệ được sử dụng:

* Ngôn ngữ sử dụng: Java (Java Swing để viết code cho phần giao diện).
* IDE: Intelliji, VS Code.
* VCS: Git.
* Tool Design: Figma.

# Thiết kế

## Thiết kế giao thức

Giao thức được thiết kế trên console, khi thêm giao diện các dòng lệnh sẽ chuyển thành các chức năng trên giao diện

Client1 connect với Server

Client2 connect với Server

Client1: msg

Client2: [Client1] msg

Client1: @Client2 msg

Client2: [Client1 to you] msg

Client1: SEND filename (chỉ file .txt, .png, .jpg, .jpge)

Client2: [Client1] SEND filename

Client2: DOWNLOAD filename

Nếu filename tồn tại thì Server gửi tới Client2 : Download file filename successfully

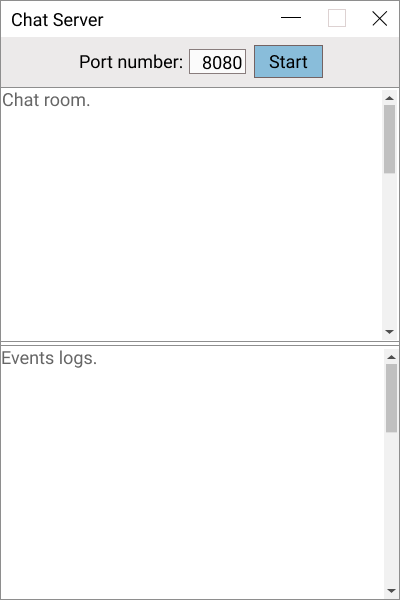
Nếu filename không tồn tại thì Server gửi tới Client2: filename not exist

## Thiết kế giao diện

Nhóm đã thảo luận và đưa đến quyết định dùng công cụ thiết kế [figma](https://www.figma.com/file/yuYtzwbppjrGccPmh4X07e/GUI?node-id=0%3A1)[[2]](#footnote-1) để thiết kế giao diện. Công cụ figma cho phép 1 nhóm nhiều người cùng thực hiện thiết kế giúp dễ dàng trao đổi và chỉnh sửa phù hợp.

Phần Server có các chức năng :

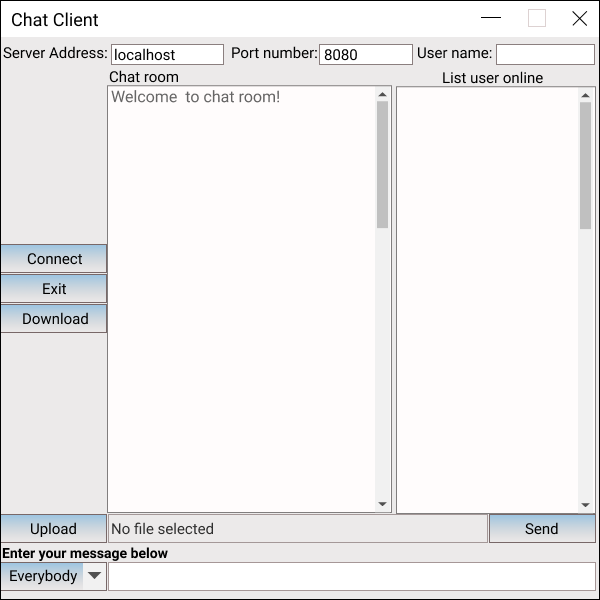
* Khởi động / ngắt Server Socket với số hiệu cổng kết nối có thể thay đổi
* Phần ghi lại các log sự kiện server đóng/ngắt kết nối hay có người dùng kết nối tới/ngắt kết nối.
* phần ghi lại log các hoạt động chat hay trao đổi file của người dùng. Nên giao diện cũng được chia thành 3 phần chính.



[*Hình 1: Giao diện Server trên Figma*](https://user-images.githubusercontent.com/51477539/102202884-70ca8a00-3efa-11eb-9cb9-32721c9a93c6.png)

Phần Client gồm các chức năng:

* Điền thông tin host address, port muốn kết nối
* Điền tên người dùng khi kết nối với Server
* Nút kết nối với Server
* Nút thoát kết nối
* Nút download file
* Nút upload file để chọn file muốn gửi
* Nút gửi file
* Nút chọn đối tượng muốn tương tác
* Thanh chat để gửi tin nhắn
* Phần ghi lại các tin nhắn từ Server gửi đến hay từ các người dùng khác
* Phần thông báo người dùng khác đang online.

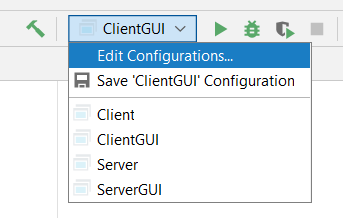


[*Hình 2: Giao diện của Client trên Figma*](https://user-images.githubusercontent.com/51477539/102202891-72944d80-3efa-11eb-989f-bf745a9e601a.png)

# Hoạt động của ứng dụng

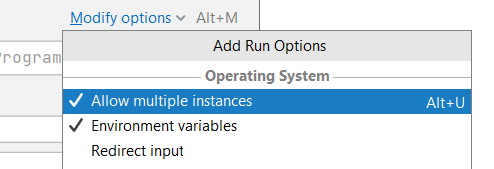
Ta có thể chạy ứng dụng bằng cách chạy trên terminal các file ServerGUI, ClientGUI. Nếu chạy trên Intellij ta phải cho phép các file ServerGUI và ClientGUI chạy đa luồng bằng cách sau:

Chọn vào phần Edit Configurations...



[Hình 18: Hình ảnh thực tế minh họa](https://user-images.githubusercontent.com/51477539/102248240-8c9e5200-3f33-11eb-81b3-bf3c7343b993.png)

Một cửa sổ hiện lên chọn vào Modify options rồi chọn Allow multiple instance

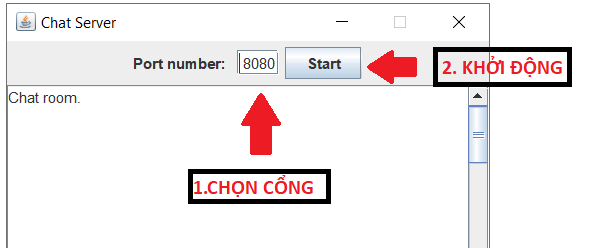


[*Hình 19: Hình ảnh thực tế minh họa*](https://user-images.githubusercontent.com/51477539/102248367-afc90180-3f33-11eb-8d44-a0955b615057.png)

## Cách thức hoạt động

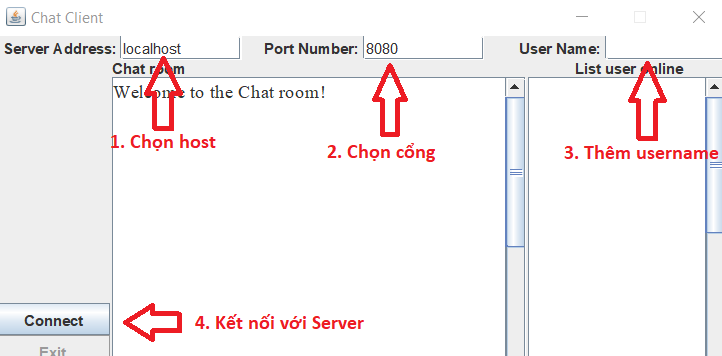
* Chạy class ServerGUI.

Để khởi động Server ta thực hiện chọn cổng và bấm nút Start như hình minh họa dưới đây.



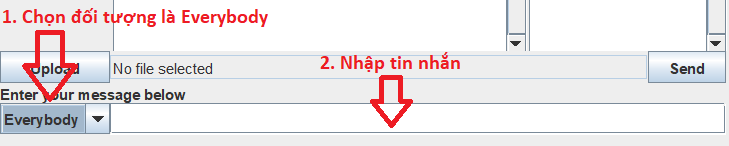
* Chạy file ClientGUI

Để kết nối với Server ta chọn host (mặc định vẫn là localhost), chọn cổng cùng với cổng của Server vừa chạy, username điền tùy ý (không được phép chứa ký tự @) rồi bấm nút Connect để kết nối như hình minh họa bên dưới.



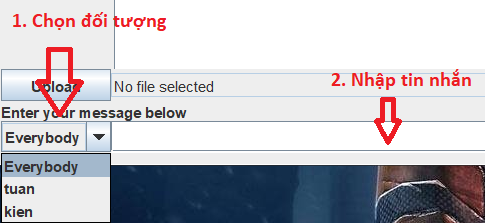
* Để gửi tin nhắn cho mọi người trong nhóm (cùng cổng)

Chọn đối tượng tin nhắn đến là Everybody nhập tin nhắn và Enter để gửi



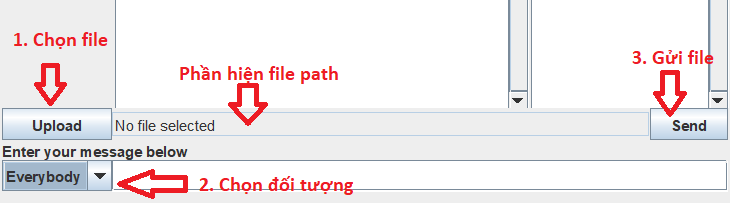
* Để gửi tin nhắn riêng tư

Chọn đối tượng cần gửi rồi nhập tin nhắn và Enter để gửi.



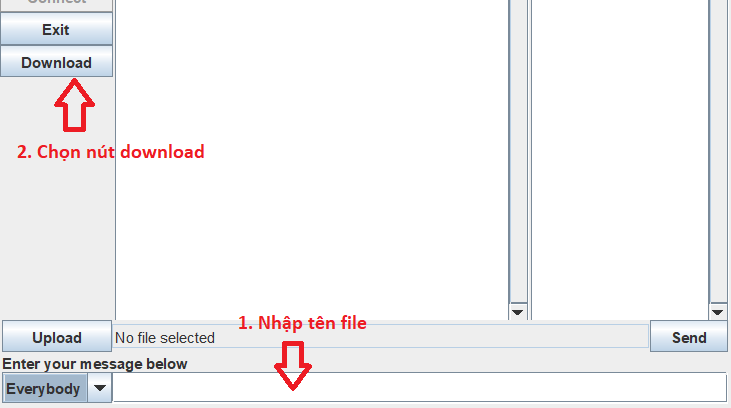
* Để gửi upload file

Bấm nút upload để chọn file muốn gửi rồi chọn đối tượng muốn gửi và nhấn nút Send để gửi file. (Lưu ý: Chỉ upload được các file có đuôi mở rộng .txt, .png, .jpg, .jpeg)



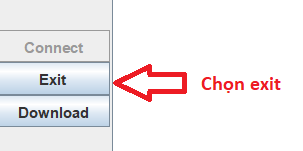
* Để download file

Để download file ta nhập tên file rồi chọn nút Download để download file về.



* Để ngắt kết nối với Server

Chọn nút Exit để ngắt kết nối với Server.

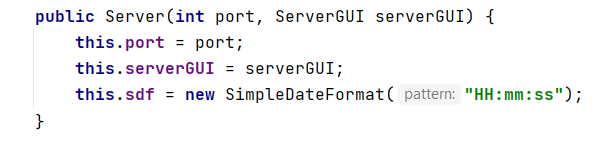


# Phân tích code

Source code của dự án được lưu trữ trên [Github](https://github.com/NAT1804/chat-application).

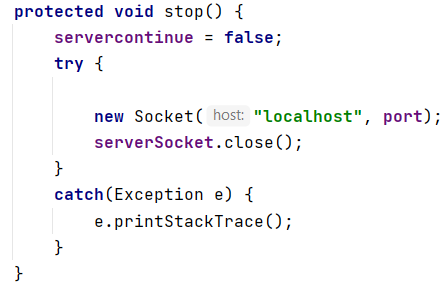
Đầu tiên trình bày về class Server:

* Hàm **constructor** để khởi tạo đối tượng Server với 2 tham số là số hiệu cổng và đối tượng ServerGUI (là đối tượng giao diện của Server).



[*Hình 3: Hàm constructor của class Server*](https://user-images.githubusercontent.com/51477539/102203598-5cd35800-3efb-11eb-940c-e04aefb69b3f.png)

* Hàm **init()** để bắt đầu thực hiện chờ kết nối của các Client.
* Hàm **stop()** để ngắt server socket không chờ kết nối của các Client nữa.

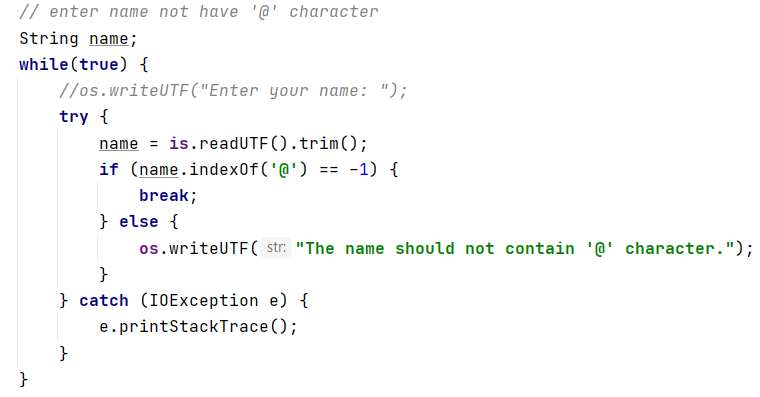


[*Hình 4: Hàm init của class Server*](https://user-images.githubusercontent.com/51477539/102243887-42ff3880-3f2e-11eb-8f16-8bf330509607.png)

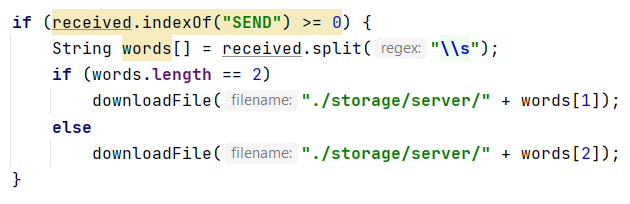
* Hàm **displayEvent()** để hiển thị các sự kiện kết nối của Server hay của các Client khác.

Tiếp theo trình bày về class ClientHandler, lớp này kế thừa lớp Thread là một luồng Client kết nối với Server, giúp quản lý các hoạt động của Client khi gửi tới Server:

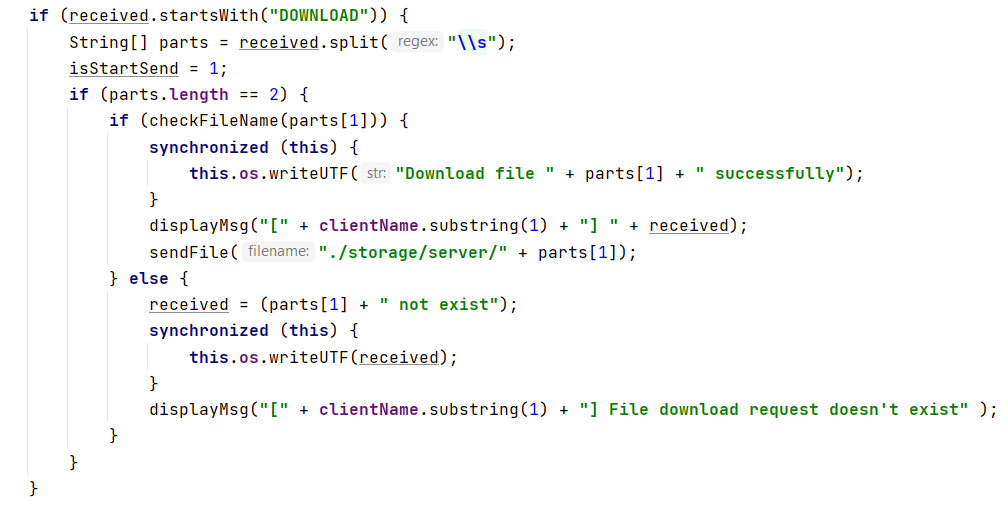
* Hàm **constructor** để khởi tạo luồng ClientHandler với tham số thứ nhất là định danh của Socket kết nối tới, tham số thứ hai là màng các luồng Client, tham số thứ ba là Server quản lý luồng này.
* Hàm **run()** override của lớp Thread.



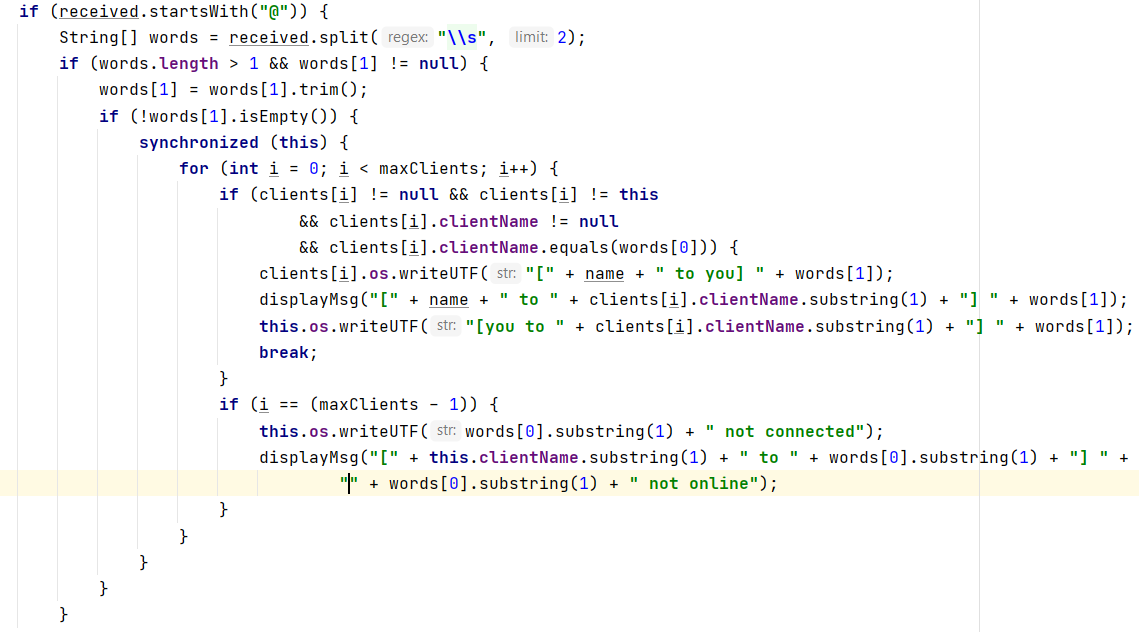
[*Hình 5: Phần xử lý username không được chứa ký tự @ trong hàm run*](https://user-images.githubusercontent.com/51477539/102244186-91acd280-3f2e-11eb-991b-129e6fc1d884.png)



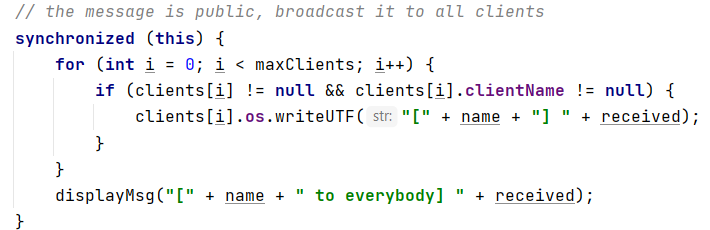
[*Hình 6: Phần xử lý nhận file từ Client gửi tới trong hàm run*](https://user-images.githubusercontent.com/51477539/102244461-eb150180-3f2e-11eb-8333-8bfe62cc87ff.png)



[*Hình 7: Phần xử lý gửi file cho Client trong hàm run*](https://user-images.githubusercontent.com/51477539/102244662-2adbe900-3f2f-11eb-9c8c-d42b388ae75b.png)



[*Hình 8: Phần xử lý gửi tin nhắn riêng tư*](https://user-images.githubusercontent.com/51477539/102244815-5fe83b80-3f2f-11eb-8220-8a4d6473bf81.png)



[*Hình 9: Phần xử lý gửi tin nhắn cho tất cả user trong cùng port*](https://user-images.githubusercontent.com/51477539/102244879-77272900-3f2f-11eb-81a5-5d984008369e.png)

* Hàm **displayMsg()** để hiển thị các tin nhắn chat của các Client.
* Hàm **checkFileName()** để kiểm tra xem file mà Client muốn download về có tồn tại hay không.
* Hàm **sendFile()** để gửi file cho Client muốn download file về.
* Hàm **downloadFile()** để nhận file của Client gửi lên.
* Hàm **close()** để ngắt kết nối luồng.

Tiếp theo là lớp ServerGUI:

* Hàm **constructor**
* Hàm **buildGUI()** để tạo ra giao diện cho Server.



[*Hinh 10: Phần xử lý sự kiện bấm nút bật tắt Server trong class ServerGUI*](https://user-images.githubusercontent.com/51477539/102245015-9cb43280-3f2f-11eb-9ced-af0b3e55d850.png)

* Hàm **appendEvent()** để thêm tin nhắn vào log sự kiện.
* Hàm **appendRoom()** để thêm tin nhắn vào log tin nhắn.
* Hàm **actionPerformed()** override từ ActionListener để bắt sự kiện bấm nút Start/Stop
* Hàm **windowClosing()** override từ WindowListener để bắt sự kiện tắt cửa sổ Server.

Tiếp theo là lớp Client:

* Hàm **constructor**
* Hàm **init()**

****

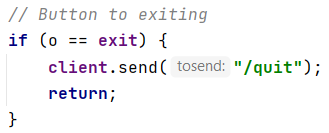
[*Hình 11: Phần xử lý luồng nhận tin nhắn trả về từ Server trong class Client*](https://user-images.githubusercontent.com/51477539/102245187-d4bb7580-3f2f-11eb-8bf3-7cd61481834a.png)

* Hàm **display()** để hiển thị tin nhắn trên phần Chat room.
* Hàm **displayUser()** để hiển thị các người dùng đang online.
* Hàm **removeUser()** để xóa người dùng khi người đó ngắt kết nối.
* Hàm **send()** để gửi tin nhắn cho Server xử lý
* Hàm **close()** để ngắt kết nối tới Server
* Hàm **sendFile()** để gửi file cho Server
* Hàm **downloadFile()** để download file từ Server về local.

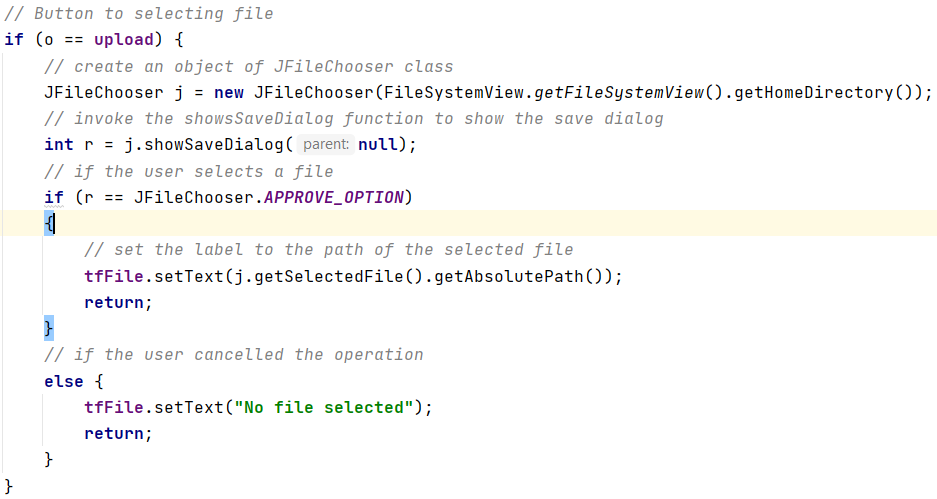
Cuối cùng là lớp ClientGUI:

* Hàm **constructor**
* Hàm **buildGUI()** để tạo giao diện cho Client.
* Hàm **appendRoom()** để hiển thị tin nhắn.
* Hàm **appendUser()** để hiển thị người dùng đang online.
* Hàm **delUser()** để xóa người dùng ngắt kết nối.
* Hàm **connectionFailed()** thay đổi trạng thái của các nút trên giao diện Client khi người dùng ngắt kết nối
* Hàm **actionPerformed()** override từ ActionListener

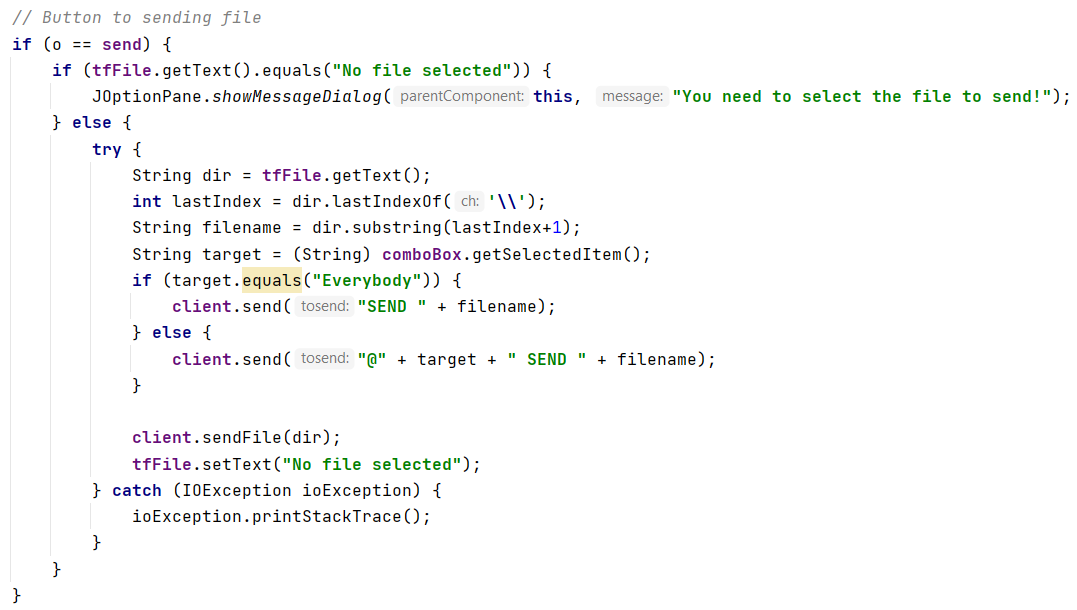
Một số phần quan trọng



[*Hình 12: Phần xử lý sự kiện bấm nút exit ngắt kết nối với Server trong file ClientGUI*](https://user-images.githubusercontent.com/51477539/102245357-059baa80-3f30-11eb-93a2-786312238876.png)



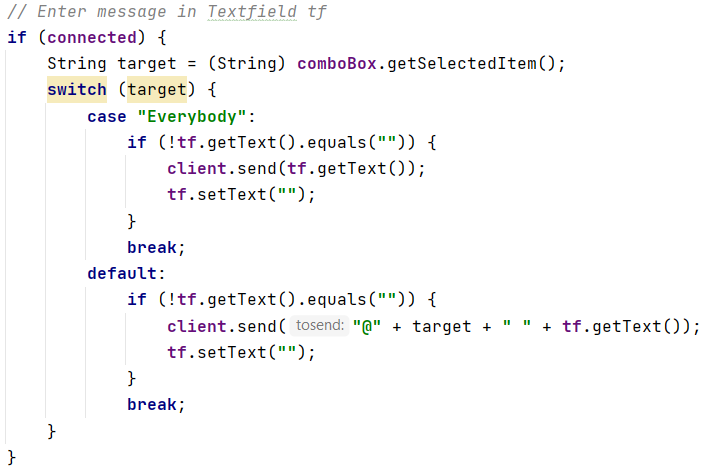
[*Hình 13: Phần xử lý upload file trong file ClientGUI*](https://user-images.githubusercontent.com/51477539/102245436-26640000-3f30-11eb-82e1-b2cb2abeaac3.png)



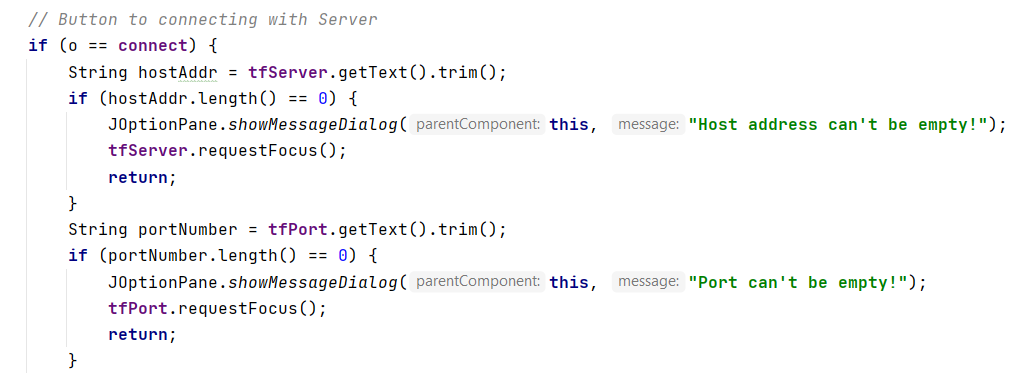
[*Hình 14: Phần xử lý nút bấm gửi file trong file ClientGUI*](https://user-images.githubusercontent.com/51477539/102245554-485d8280-3f30-11eb-91e9-97b90a25ab8d.png)



[*Hình 15: Phần xử lý download file trong file ClientGUI*](https://user-images.githubusercontent.com/51477539/102245636-60cd9d00-3f30-11eb-8401-c837f864f834.png)



[*Hình 16: Phần xử lý tin nhắn gửi đi trong file ClientGUI*](https://user-images.githubusercontent.com/51477539/102245703-75aa3080-3f30-11eb-8ad8-b855d951c356.png)



[*Hình 17: Phần xử lý nút bấm để kết nối với Server*](https://user-images.githubusercontent.com/51477539/102245806-92466880-3f30-11eb-9527-88e6b8dfe569.png)

* Hàm **windowClosing()** override từ WindowListener để bắt sự kiện tắt cửa sổ Client sẽ đồng thời ngắt kết nối với Server.

# Mục tiêu phát triển

Ứng dụng còn nhiều thiếu sót cần phát triển thêm.

1. Link github quản lý dự án: https://github.com/NAT1804/chat-application [↑](#footnote-ref-0)
2. Link giao diện sản phẩm trên Figma: https://www.figma.com/file/yuYtzwbppjrGccPmh4X07e/GUI?node-id=0%3A1 [↑](#footnote-ref-1)