Simple Chat Application

Version 1.0

Môn: Mạng máy tính

GVHD: Bùi Xuân Giang

Nhóm: 1999

TP Hồ Chí Minh, tháng 10 năm 2019

Danh sách thành viên

Họ và tên	MSSV
Nguyễn Quang Vương	1714037
Nguyễn Khải Vy	1714050
Đỗ Đức Trung	1713691

Contents

1. Giới thiệu chung

Ứng dụng chat cho phép người dùng sử dụng app bằng cách login vào hệ thống khi biết được địa chỉ IP của server và có thể chat trực tiếp, đồng thời với những người bạn của mình trong các private session chat. Bên cạnh đó, app còn hỗ trợ việc truyền gửi file cho nhau. Và cụ thể, app phát triển theo các chức năng chính sau: Login, Chat riêng tư và truyền gửi file.

a. Login

Sau khi Login vào hệ thống với ràng buộc rằng, UserName không có ký tự đặc biệt và không trùng với UserName của người đã đăng ký trước đó. Người dùng phải biết Port và địa chỉ IP của server để tiến hành Login.

b. Chat riêng tư

Sau khi vào hộp thoại Form MainGui sẽ xuất hiện sau khi Login. Tại đây hiển thị UserName khác đang online, truy cập vào hệ thống. Ta có thể chọn bất kỳ đối tượng nào để mở một hộp thoại chat riêng tư. Bên cạnh đó một User có thể nhắn tin riêng tư trực tiếp với nhiều User khác trong cùng một thời điểm.

c. Truyền gửi file

Người dùng có thể gửi và nhận file với định dạng và kích thước được quy định.

2. Định nghĩa giao thức cho từng chức năng

Dưới đây là ý nghĩa, nôi dụng và một số ghi chú cho các giao thức được định nghĩa để hiện thực các chức năng của App

|<SESSION_REQ>clientID</SESSION_REQ> |
Yêu cầu tạo một chat session từ client, clientID là id của người dung, sử dụng 1 lần cho mỗi chat session.
|<PEER_NAME>userName</PEER_NAME>|
Tạo user khi người dùng đăng nhập vào server, userName là tên người dùng|
|<PORT>numPort</PORT>|
Tạo port cho user, numPort là số hiệu port.
|<SESSION_KEEP_ALIVE></SESSION_KEEP_ALIVE>|
Nội dung request do user gửi lên server, User request 10s/lần bao gồm tên user và trạng thái.
|<STATUS>statusUser</STATUS>|
Thông báo trạng thái user, statusUser là trạng thái user.
|<SESSION_DENY />|
Từ chối người dùng kết nối tới server, từ chối sau khi người dùng login.

|<SESSION_ACCEPT></SESSION_ACCEPT>|
Chấp nhận người dùng kết nối tới server, chấp nhận sau khi người dùng login.
|<CHAT_REQ>userName</CHAT_REQ>|
Yêu cầu tạo chat riêng tư từ một user với user khác | userName là tên user tạo yêu cầu chat riêng tư.
|<IP>stringIP</IP>|
Định danh IP cho user| stringIP là IP của user.
|<CHAT_DENY />|
Từ chối tạo chat session, từ chối sau khi có yêu cầu tạo chat riêng tư.
|<CHAT_ACCEPT />|
Chấp nhận tạo chat session, chấp nhận sau khi có yêu cầu tạo chat riêng tư.
|<CHAT_MSG> Message</CHAT_MSG>|
Gửi nội dung chat, message được gửi khi user đồng ý tạo chat riêng tư
|<PEER> peerData </PEER>|
Xác định peer của user do server trả về, peerData là peer của user gồm name, port, IP.
|<FILE_REQ> fileName</FILE_REQ>|
Yêu cầu gửi file, fileName là tên file muốn gửi.
|<FILE_REQ_NOACK />|
Từ chối yêu cầu gửi file, sử dụng sau khi có yêu cầu gửi file.
|<FILE_REQ_ACK></FILE_REQ_ACK>|
Chấp nhận yêu cầu gửi file, sử dụng sau khi có yêu cầu gửi file.
|<FILE_DATA_BEGIN />|
Bắt đầu gửi file, quá trình gửi file sau khi người gửi và nhận đồng ý.
|<FILE_DATA>fileData</FILE_DATA>|
Nội dung file cần chuyển.
|<FILE_DATA_END />|
Kết thúc quá trình gửi file| Yêu cầu bởi người gửi file.
|<CHAT_CLOSE />|
Kết thúc chat riêng tư| Sử dụng sau khi tạo chat riêng tư.
|MAX_MSG_SIZE| 102400 |
Kích thước tối đa của một message(file) trong chat.
|SERVER_ONLINE| "RUNNING"|
User đang online.
|SERVER_OFFLINE| "STOP"|
User thoát khỏi chat session.

3. Thiết kế ứng dụng

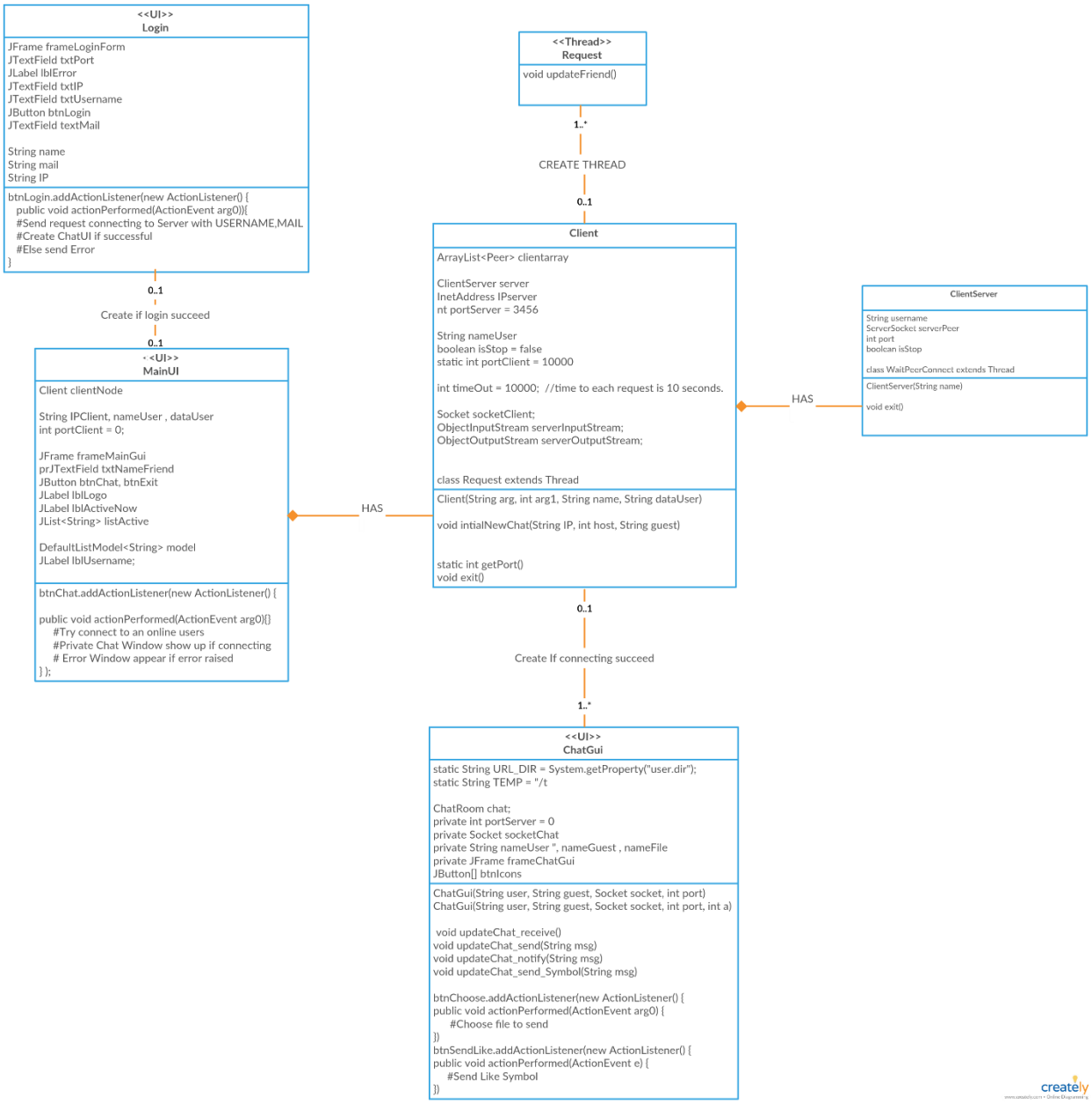
a. Kiến trúc ứng dụng

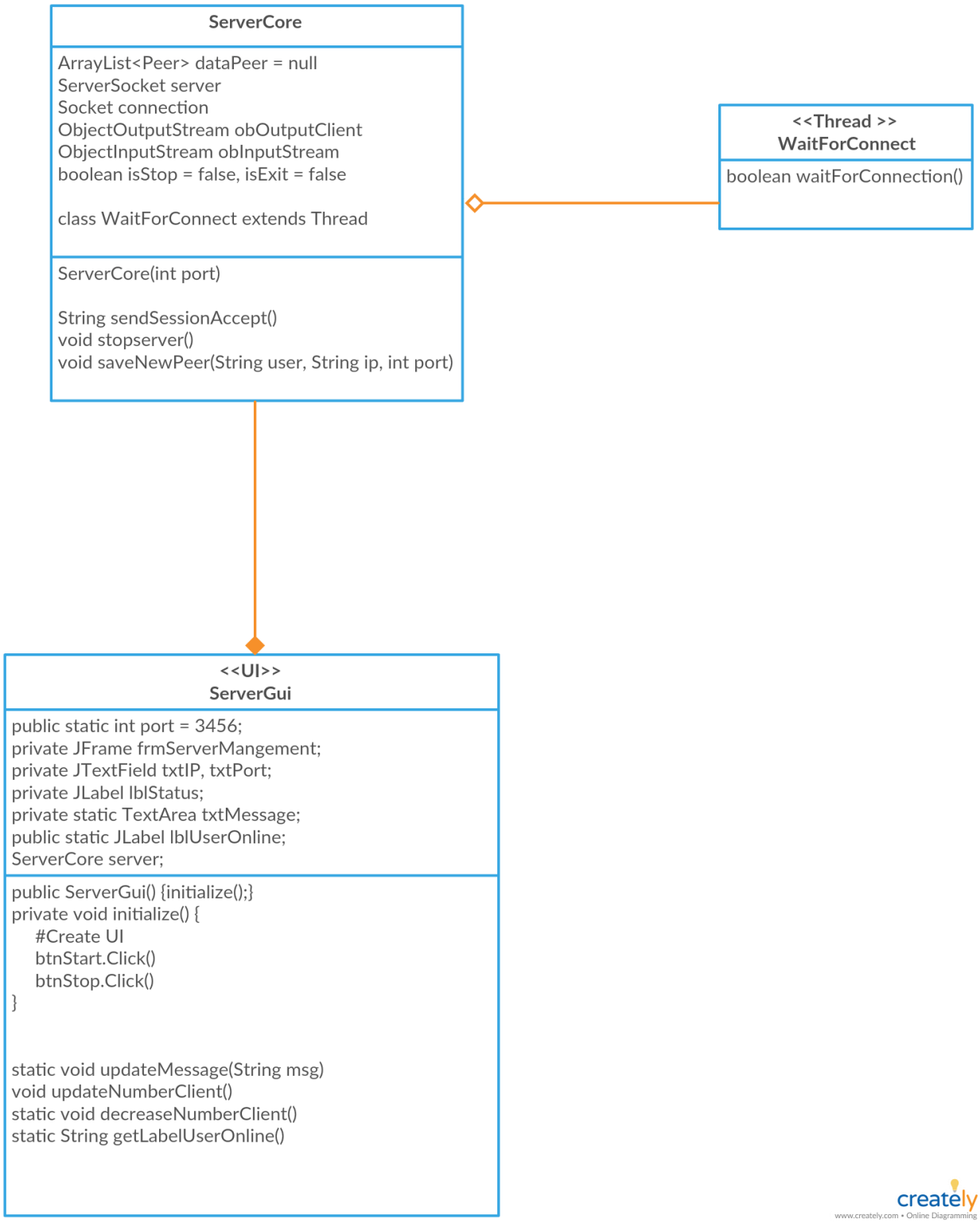
Ứng dụng gồm bốn form chính:

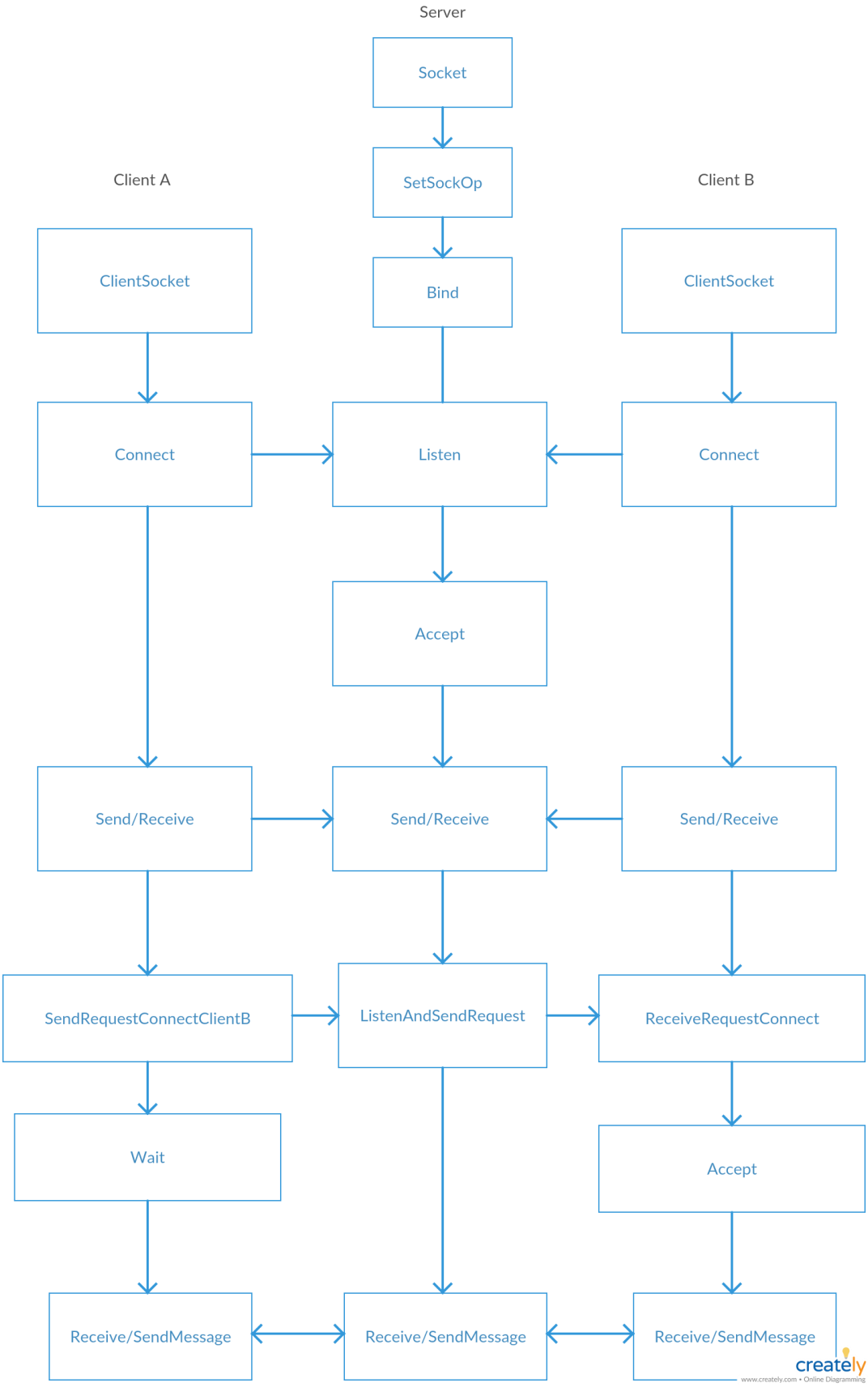
- **Server form**: Form quản lí server, hiển thị thông tin server, các request user gửi lên.
- **Login form**: Form đăng nhập vào server. Để kết nối được, user phải cung cấp username chưa có ai đăng kí trước đó với server.
- **Main form**: Form để quản lí danh sách những người dùng khác đang online cùng với user hiện tại. User có thể chọn bất kì ai đang online để bắt đầu một cuộc trò chuyện.
- **Chat form**: Khi hai user đã chấp nhận trò chuyện với nhau thì form này sẽ xuất hiện. Hiển thị thông tin cuộc trò chuyện giữa hai người.

b. Xây dựng class

Class Diagram của ứng dụng






Server
Socket
SetSockOp
Bind
Client A
Client B
ClientSocket
ClientSocket
Connect
Listen
Connect
Accept
Send/Receive
Send/Receive
Send/Receive
SendRequestConnectClientB
ListenAndSendRequest
ReceiveRequestConnect
Wait
Accept
Receive/SendMessage
Receive/SendMessage
Receive/SendMessage
creately
www.creately.com • Online Diagramming

- **Server Class**
  - ServerGUI.java: sử dụng để thao tác với server như bật, tắt server. Ngoài ra còn hiển thị các thông tin server như: IP, Port, số lượng người kết nối, các protocol mà client gửi lên.
  - ServerCore.java: Đây là class để quản lí server, gồm đầy đủ thông tin, chức năng chính của server.
    - Tạo một SocketServer.
    - Quản lí danh sách các user đang online
    - Tạo một thread là WaitForConnect để xử lý các request kết nối từ user. Có thể là yêu cầu dăng nhập, thoát ứng dụng, lấy thông tin user đang online. Nếu đăng nhập thành công, trả về cho client danh sách user đang online, nếu không sẽ trả về một protocol từ chối đăng nhập.

- **Client Class**
  - MainGUI.java: GUI class để hiển thị danh sách user khác đang online, bắt sự kiện gửi yêu cầu chat của user.
  - ChatGUI.java: Xử lí việc chat, gửi nhận File giữa hai user. Hiển thị nội dung chat giữa hai user.
  - ClientServer.java: Tạo và quản lí ServerSocket của mỗi user. Xử lí request chat gửi đến từ user khác.
  - Client.java: Quản lí thông tin hiện tại của user. Nó sẽ tạo một thread để liên tục gửi request tới server để cập nhật danh sách user. Ngoài ra nó còn xử lí request chat tới user khác.

- **Other Class**
  - DataFile.java: Tạo một đối tượng File để gửi đi.
  - Peer.java: Tạo một đối tượng Peer có các thuộc tính: IP, port, name. * Tags.java: Định nghĩa các protocol được sử dụng trong ứng dụng. * Encode.java: Định nghĩa các phương thức user dùng để gửi request lên server.
  - Dedode.java: Giải mã các phương thức để lấy thông tin user, port hay ip...

c. Công cụ sử dụng

- Ngôn ngữ lập trình: Java
- Trình IDE hỗ trợ lập trình: Eclipse
- Kỹ thuật lập trình socket: Kỹ thuật này hỗ trợ lập trình các ứng dụng giao tiếp qua mạng. TCP socket sử dụng Stream để thực hiện quá trình truyền dữ liệu giữa hai máy tính khác nhau đã thiết lập.
- WindowBuilder Editor của Java để xây dựng các form (User Interface)

4. Kết quả đạt được

- Ứng dụng được xây dựng trên mô hình lai client- server với P2P. Trong đó Server quản lý thông tin đăng nhập của người dùng còn P2P cho việc trò chuyện trực tiếp giữa 2 User với nhau.
- Bên cạnh việc chát thuần đơn giản, ứng dụng còn import thêm một số Icon để tăng tính đa dạng trong cuộc hội thoại.
- Chức năng gửi và nhận file giữa 2 User cũng là một vấn đề đã được nhóm giải quyết và hiện thực.

Hạn chế:

- Nhóm vẫn đang cố gắng phát triển thêm chức năng tạo phòng chat Group để các User khác tham gia vào.
- Mã nguồn code còn thiếu sạch sẽ và mạch lạc cũng như tính tối ưu của nó. Chức năng call và video call cũng được nhóm quan tâm trong tương lai.