



BÀI TẬP LỚN

HỌC PHẦN: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG
(Mã học phần: IT3100)

Đề tài
XÂY DỰNG APP NHẮN TIN

Sinh viên thực hiện:

Hồ Xuân Thái

20225393

Lê Việt Hoàng

20225321

Lớp: 149455

Giảng viên hướng dẫn: TS.Ngô Thành Trung

Hà Nội, tháng 5 năm 2023

PHÂN CÔNG THÀNH VIÊN TRONG NHÓM

STT	Họ và tên	MSSV	Email	Công việc	Mức độ hoàn thành	Ghi chú
1	Hồ Xuân Thái	20225393	Thai.hx225393@sis.hust.edu.vn	Trưởng nhóm , code chính . Xây dựng nền móng và những phần quan trọng nhất của app	Tốt	
2	Lê Việt Hoàng	20225321	Hoang.lv225321@sis.hust.edu.vn	Hỗ trợ , code phụ , viết báo cáo , làm slide	Tốt	

LỜI NÓI ĐẦU

Mặc dù đã cố gắng hoàn thiện sản phẩm nhưng không thể tránh khỏi những thiếu hụt về kiến thức và sai sót trong kiểm thử. Chúng em mong muốn nhận được những nhận xét thẳng thắn, chi tiết đến từ cô và các bạn để tiếp tục hoàn thiện hơn nữa. Cuối cùng, nhóm chúng em xin được gửi lời cảm ơn đến thầy Ngô Thành Trung đã hướng dẫn bọn em trong suốt quá trình hoàn thiện bài tập lớn. Xin trân trọng cảm ơn cô.

CHƯƠNG 1. KHẢO SÁT, ĐẶC TẢ YÊU CẦU BÀI TOÁN

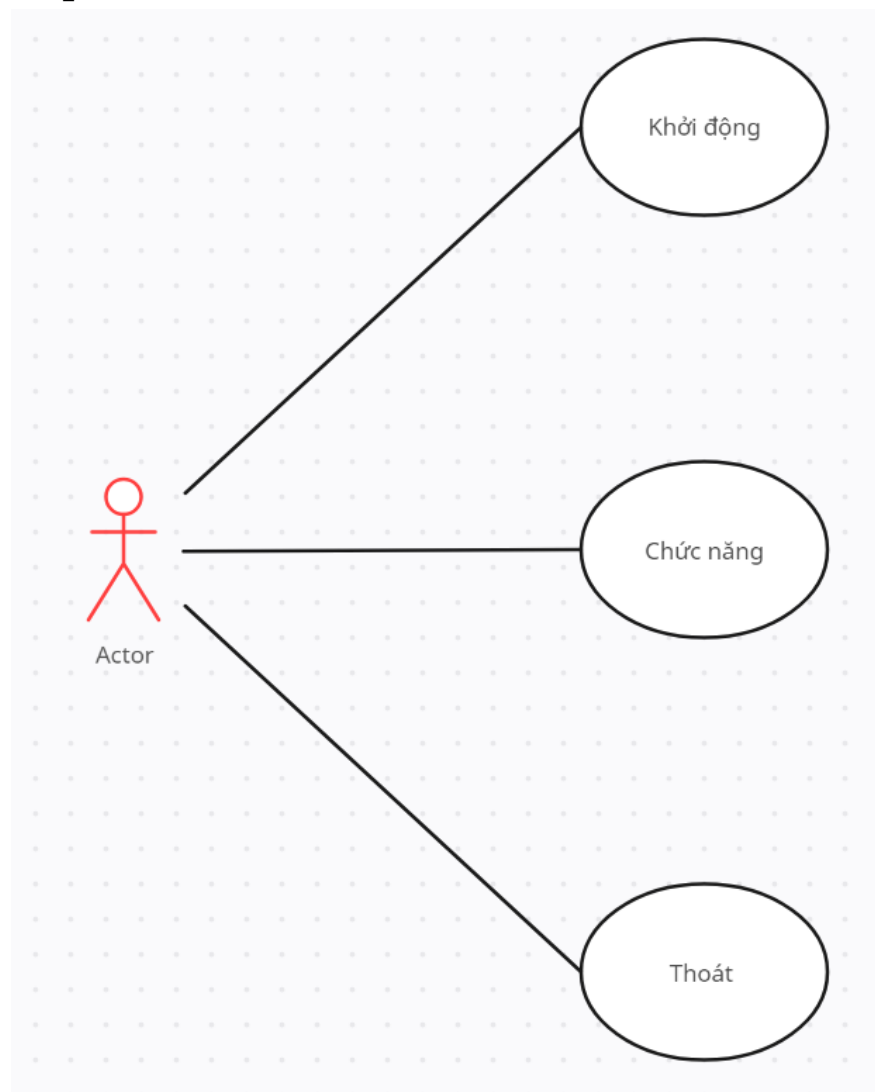
1.1. Mô tả yêu cầu bài toán

Chúng em đã xây dựng chương trình chuyển tin nhắn, ảnh giữa 2 máy tính có giao diện đồ họa. Bao gồm các chức năng sau:

- Đăng ký tài khoản
- Đăng nhập
- Gửi tin nhắn text, gửi emoji, gửi ảnh, gửi file.

1.2. Biểu đồ Use Case

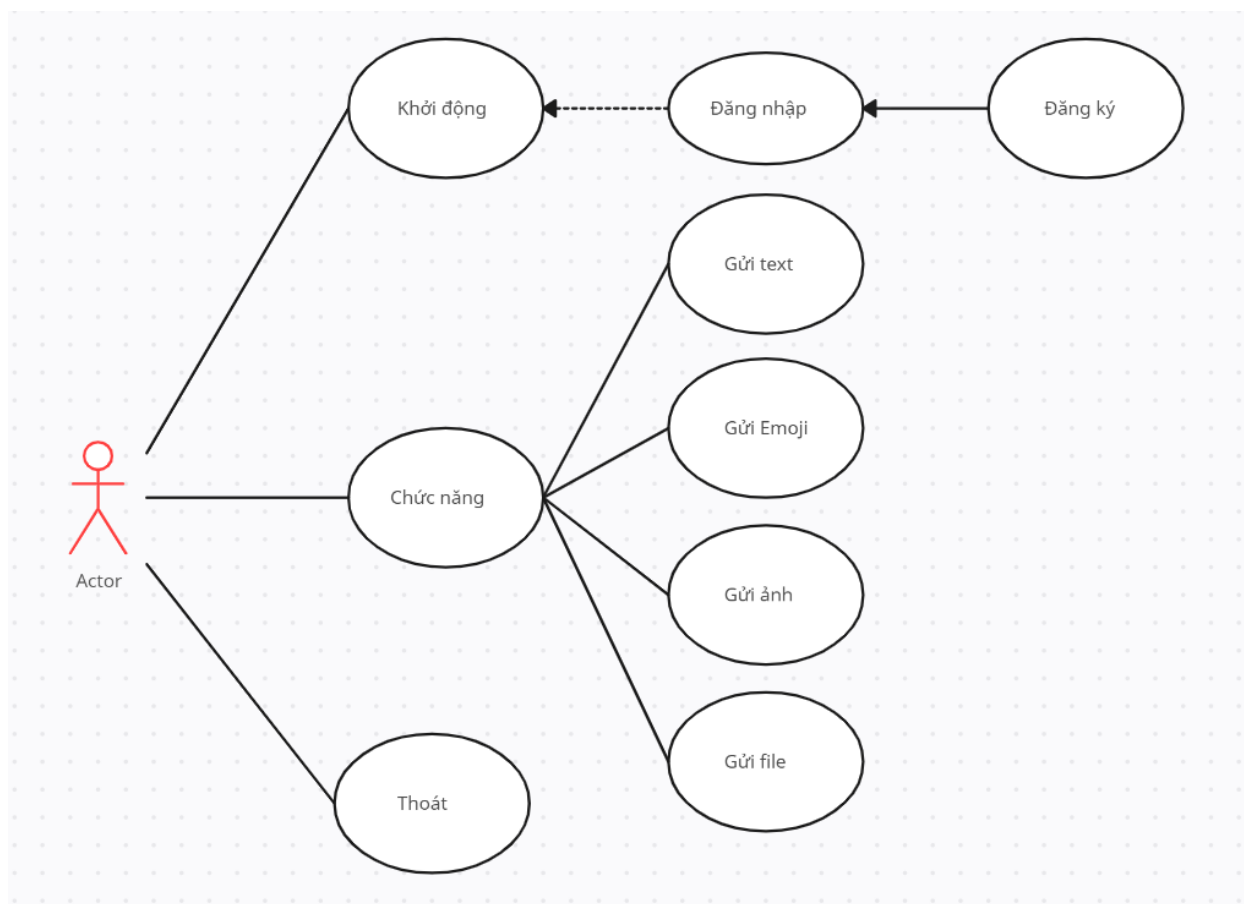
1.2.1. Biểu đồ phân rã mức 1



Chương trình của nhóm có 3 use case chính là :

1. Khởi động ứng dụng
2. Chức năng
3. Thoát ứng dụng

1.2.2. Biểu đồ phân rã mức 2



1.2.3. Đặc tả Use Case

1.2.3.1. Danh sách Actor

Tên Actor	Ý Nghĩa
Actor	Người dùng

1.2.3.2. Danh sách Use Case

Tên Use Case	Ý nghĩa
Khởi động ứng dụng	Đăng ký/ Đăng nhập, nạp danh sách người dùng từ database
Chức năng (Options)	Đưa người dùng các chức năng
Thoát ứng dụng	Đưa người dùng thoát khỏi ứng dụng

1.2.3.3. Đặc tả Use Case

1. Use case Khởi động ứng dụng

Tên Use case : Khởi động ứng dụng

Tên tác nhân : User

STT	Thực Hiện	Hành động
1	System	Hiển thị màn hình đăng ký/ đăng nhập
2	System	Hiển thị: Nhập username, password, button login, button register
3	User	Nhập vào username, password (đối với login), username, password, confirm password (đối với register)
4	User	Nhấp login để đăng nhập, register để đăng ký

2. Use case Chức năng (Options)

Tên use case: Chức năng (Options)

Tên tác nhân: User

STT	Thực hiện	Hành động
1	System	Hiển thị menu left chứa các user, chat body hiển thị phần tin nhắn, chat bottom là nơi nhập tin nhắn, chat title là nơi hiển thị tên người dùng đích và trạng thái hoạt động của họ
2	User	<p>Người dùng chọn người muốn nhắn tin trong menu left.</p> <p>Nhập tin nhắn vào phần “Enter message here”, nhấn biểu tượng gửi để gửi hoặc nhấn tổ hợp phím Shift-Enter.</p> <p>Nhấn vào biểu tượng More để mở panel more chứa các emoji hoặc gửi ảnh.</p> <p>Nhấn vào biểu tượng emoji để gửi emoji.</p> <p>Nhấn vào biểu tượng picture để chọn file ảnh và gửi.</p>

3. Use case Thoát ứng dụng

Tên Use case: Thoát ứng dụng

Tên tác nhân: User

STT	Thực hiện	Hành động
1	User	Nhấn vào biểu tượng thoát chương trình
2	System	Thoát khỏi chương trình ứng dụng

CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ BÀI TOÁN

1.1. Thiết kế Cơ sở dữ liệu hoặc Cấu trúc tệp dữ liệu

Cơ sở dữ liệu mà chúng em sử dụng là MySQL 8.3. Cơ sở dữ liệu bao gồm 3 table là user_account, user, files :

Tables_in_chatapp
files
user
user_account

Table user chứa danh sách người dùng, bao gồm UserID, UserName và Password, trong đó UserID là Primary key. Primary Key vì mỗi user là duy nhất và tự động tăng khi nhập user mới vào.

UserID	UserName	Password
16	user1	1
17	user2	22
18	user3	333
19	user4	4444
20	user5	55555
21	user5.1	515151
22	user6	666666
23	user0	0000

Table user_account chứa UserID, UserName, Gender, Image, ImageString, Status, trong đó UserID là Primary key. Phần này chúng em mới chỉ dùng tới Status nhưng vẫn tạo đủ các trường để tiện cho sau này nâng cấp.

UserID	UserName	Gender	Image	ImageString	Status
16	user1		NULL		1
17	user2		NULL		1
18	user3		NULL		1
19	user4		NULL		1
20	user5		NULL		1
21	user5.1		NULL		1
22	user6		NULL		1
23	user0		NULL		1

Table user và user_account liên kết với nhau qua UserID.

Table files chứa FileID, FileExtension, BlurHash, Status, trong đó FileID là Primary key. Khi gửi file thì chương trình sẽ thêm các dữ liệu về file vào trong bảng này bao gồm FileID và FileExtension. BlurHash là phần đã mã hóa của ảnh để tạo thành 1 ảnh mờ. Khi gửi một ảnh thay vì gửi thẳng 1 ảnh đi thì chúng em sẽ gửi trước một ảnh mờ (chỉ khoảng vài byte) rồi sau đó mới gửi ảnh.

FileID	FileExtension	BlurHash	Status
52	.png	L4SY{q9F~q%M?bofWBj[-;WBxuof	1
53	.jpg	L7ABneE.8xyC~ptmNKRjIC?^IURP	1
54	.jpg	L7ABneE.8xyC~ptmNKRjIC?^IURP	1
55	.jpg	LC9j} M_4T%MITWAjYWB00xv?wM	1
56	.jpg	L5Aw@:tmEn-.~okCxtxsE.ng-gIV	1
57	.png	L6S?DVHtIEAV?wWqRjnkNgbbaer@	1
58	.jpg	LC9j} M_4T%MITWAjYWB00xv?wM	1
59	.jpg	LC9j} M_4T%MITWAjYWB00xv?wM	1
60	.jpg	L5Aw@:tmEn-.~okCxtxsE.ng-gIV	1

1.2. Hệ thống các package

Đối với Client

1. Package Component: Bao gồm các thành phần thiết kế của ứng dụng như Chat_Body, Chat_Bottom, Chat_Image, Chat_Item, Chat_Left, Chat_Right, Chat_Title, Image_Item, Item_People, Panel_More và các lớp Menu_Button, Option_Button có các phương thức hành động của Button_Option và Button_Menu.
2. Package emoji: Gồm 2 lớp Model_Emoji và Emoji. Lớp Model_Emoji là lớp mẫu của một emoji bao gồm hình ảnh và tên của emoji phục vụ cho quá trình truy xuất hình ảnh của lớp Emoji. Lớp Emoji có tác dụng trả về một Model_Emoji từ những hình ảnh được lấy theo đường dẫn.
3. Package emoji_icon: Thư mục chứa hình ảnh emoji để chuyển thành Model_Emoji.
4. Package event: Bao gồm các interface EventChat, EventFileReceiver, EventFileSender, EventImageView, EventLogin, EventMain, EventMenuLeft, EventMessage và lớp PublicEvent. Các interface định nghĩa các phương thức tương ứng của từng hoạt động, lớp PublicEvent trả về từng interface event.
5. Package form: Bao gồm các lớp thiết kế sử dụng java swing để thiết kế các thành phần: Chat, Home, Loading, Login, Menu_Left, P_Login, P_Register, View_Image. Phần source của Chat, Login, Menu_Left, P_Register triển khai các interface ở Package event.
6. Package icon: Bao gồm các hình ảnh để đặt cho các button, icon app.
7. Package main: Bao gồm lớp Main triển khai interface ở package event.
8. Package model: Chứa các lớp định nghĩa các lớp dữ liệu được sử dụng trong ứng dụng. Các lớp này đóng vai trò như những mô hình dữ liệu để truyền tải thông

tin giữa các thành phần khác nhau của ứng dụng, đảm bảo tính nhất quán và dễ dàng quản lý dữ liệu.

9. Package service: Nhiệm vụ chính của package service là cung cấp các dịch vụ liên quan đến việc kết nối và giao tiếp với máy chủ, xử lý các sự kiện và thông báo từ máy chủ, và quản lý việc gửi và nhận file trong ứng dụng. Package này đóng vai trò trung gian, điều phối các hoạt động liên quan đến kết nối mạng và truyền thông tin giữa các thành phần của ứng dụng.

10. Package swing: Nhiệm vụ chính của package swing là cung cấp các thành phần giao diện người dùng tùy chỉnh và quản lý các tương tác với chúng, giúp tạo ra các giao diện người dùng thân thiện và hiện đại hơn so với các thành phần mặc định của Swing.

11. Package BlurHash: Nhiệm vụ chính của package blurhash là xử lý hình ảnh khi gửi, mã hóa chúng thành các hình ảnh mờ và hiện lên trước khi hình ảnh được gửi hoàn tất

Đối với Server

1 Package connection: Kết nối tới cơ sở dữ liệu MySQL

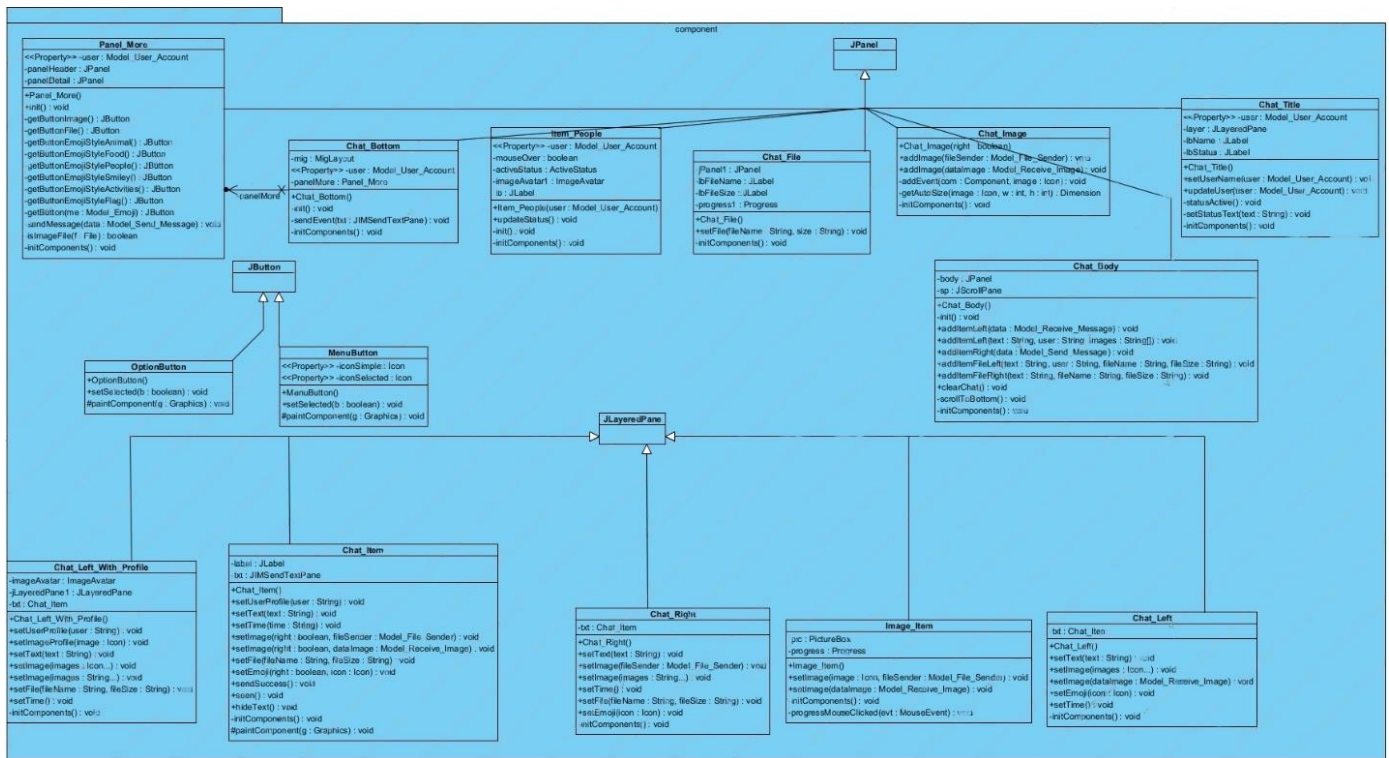
2 Package main: Giao diện của server, hiển thị các kết nối từ client tới và báo lỗi nếu có

3 Package model: Chứa các lớp định nghĩa các lớp dữ liệu được sử dụng trong ứng dụng. Các lớp này đóng vai trò như những mô hình dữ liệu để truyền tải thông tin giữa các thành phần khác nhau của ứng dụng, đảm bảo tính nhất quán và dễ dàng quản lý dữ liệu.

4 Package service: Nhiệm vụ chính của package service là cung cấp các dịch vụ liên quan đến việc kết nối và giao tiếp với Client, xử lý các sự kiện và thông báo từ Client, và quản lý việc gửi và nhận file trong ứng dụng. Package này đóng vai trò trung gian, điều phối các hoạt động liên quan đến kết nối mạng và truyền thông tin giữa các thành phần của ứng dụng.

5 Package BlurHash: Nhiệm vụ chính của package blurhash là xử lý hình ảnh khi gửi, mã hóa chúng thành các hình ảnh mờ và hiện lên trước khi hình ảnh được gửi hoàn tất

1.3.1. Package Component



1. Chat_Body : Đóng vai trò là thân chính của giao diện chat, nơi để hiển thị tin nhắn đến và tin nhắn gửi. Các phương thức bao gồm addItemLeft(), addItemRight(). Phương thức addItemLeft() thêm một item Chat_Left vào trong Chat_Body và set các giá trị cho item đó (text, image, time, file). Phương thức addItemRight() cũng hoạt động tương tự như addItemLeft().
2. Chat_Title : Hiển thị tên người dùng đích, trạng thái hoạt động của họ. Có các phương thức setUsername(), updateUser(), statusActive(), setStatusText(). Phương thức setUsername() đặt tên người dùng và trạng thái hoạt động của người đó. Phương thức updateUser() hoạt động khi người dùng nhấn vào một user trong menu_left thì phần chat_title cần update lại tên người dùng và trạng thái hoạt động. Phương thức statusActive() đặt phần text ở chat_title là Active now với màu chữ màu xanh. Phương thức setStatusText() sẽ thay đổi phần text trong chat_title và đổi màu của text đó.
3. Chat_Image : Xử lý và hiển thị hình ảnh trong cuộc trò chuyện. Gồm các phương thức addImage(), addEvent(). Phương thức addImage() sẽ tạo một ImageIcon từ đường dẫn của ảnh nhân được, set các kích thước cho ảnh, hiển hình

ảnh đó lên khung chat. Phương thức `addEvent()` sẽ thay đổi con trỏ chuột khi di chuột vào hình ảnh và khi nhấn vào ảnh sẽ hiện lên chức năng xem ảnh.

4. `Chat_File` : Xử lý và hiển thị các file được gửi trong cuộc trò chuyện. Có phương thức để hiển thị lên khung chat `fileID` gửi và nhận.

5. `Item_People` : Hiển thị tên user và trạng thái hoạt động, là một phần của `menu_left`. Gồm các phương thức `updateStatus()`, và các sự kiện tương tác với chuột. Phương thức `updateStatus()` sẽ update trạng thái người dùng là `active now` khi được gọi tới. Các sự kiện tương tác với chuột gồm: di chuột vào sẽ làm `background` của `item_people` thay đổi để dễ nhận biết đang chọn người nào, di chuột ra khỏi phạm vi `item_people` đó sẽ set là `background` ban đầu, nhấn chuột vào `item_people` đó sẽ update `chat_title`.

6. `Chat_Bottom` : Phần dưới của giao diện chat, có thể chứa các nút và trường nhập liệu để gửi tin nhắn. Các phương thức bao gồm `init()` và `sendEvent()`. Phương thức `init()` chứa các thành phần thiết kế của `chat_bottom` và các sự kiện khi nhấn vào các button như `sendButton` và `moreButton`. Khi nhấn vào `sendButton` thì gọi phương thức `sendEvent` để gửi tin nhắn. Khi nhấn `moreButton` sẽ hiện lên `panel_more` để gửi emoji, ảnh, file. Trong phương thức `init()` cũng có bộ lắng nghe sự kiện bàn phím, khi gõ phím thì `chat_bottom` sẽ cập nhật lại bố cục liên tục và khi nhấn tổ hợp phím `shift + enter` thì sẽ gọi phương thức `sendEvent`. Phương thức `sendEvent()` sẽ lấy phần text trong `JIMSendTextPanel` và loại bỏ các khoảng trắng hai đầu và gửi đi tới server.

7. `Panel_More` : Một bảng điều khiển bổ sung, có thể chứa các nút. Chứa các button liên quan đến gửi ảnh, file, emoji. Khi nhấn vào button gửi ảnh hoặc gửi file sẽ hiện lên cửa sổ chọn file trong máy, khi chọn file hoặc ảnh xong sẽ gửi đi. Nếu nhấn button file thì cửa sổ hiện lên chỉ lọc thư mục hoặc file ảnh. Khi nhấn vào các button emoji thì phần `panelDetail` bên dưới của `panelMore` sẽ hiển thị danh sách các emoji cho người dùng lựa chọn, nhấn vào emoji muốn gửi để gửi.

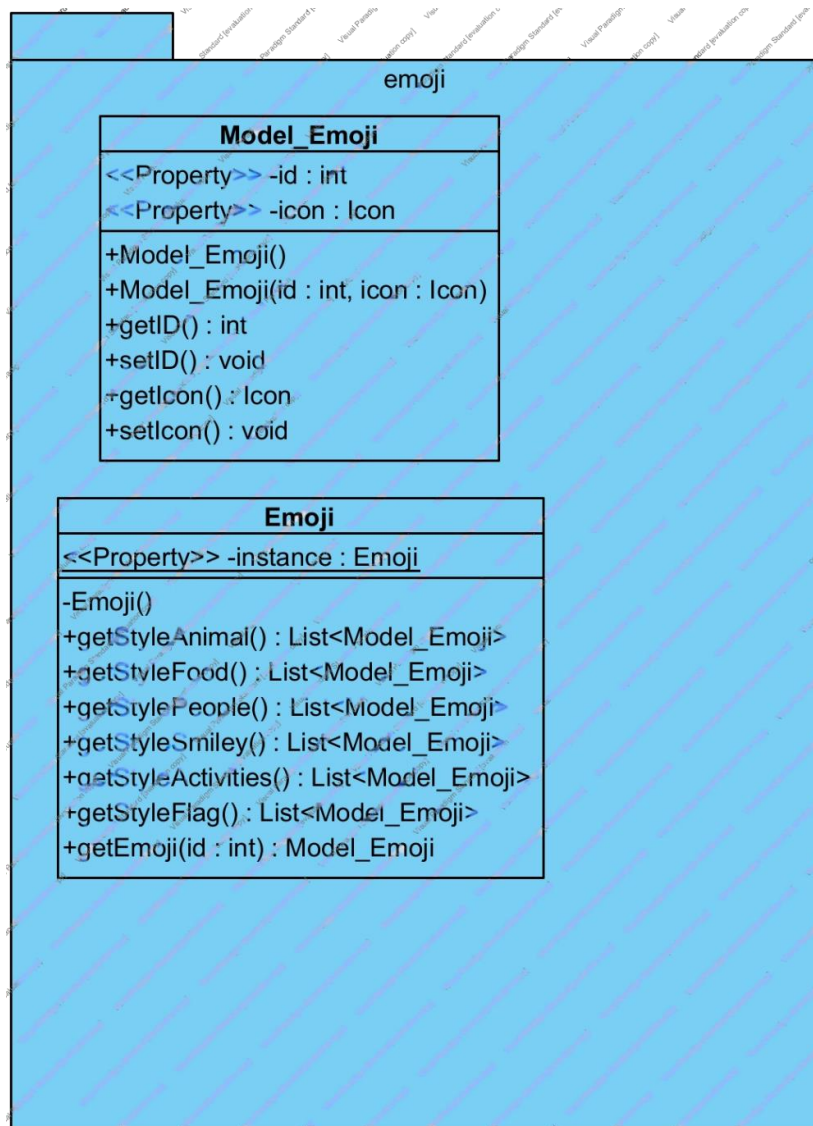
8. `Option_Button` : Nút tùy chọn, có thể mở các menu hoặc thực hiện các hành động khi được nhấn. Chính là các button file, ảnh, emoji trong `panelHeader` của `panel_more`.

9. `Menu_Button` : Một menu các button ở trong phần `menu_left`. Trong chương trình của chúng em chỉ mới có một button dùng được, còn 2 button còn lại được thêm vào sau này phát triển.

10. `Chat_Left_With_Profile` : Một phần của giao diện chat, hiển thị tin nhắn từ người gửi cùng với hồ sơ hoặc ảnh đại diện của họ. Phần này chúng em cũng chưa sử dụng tới, phần này được tạo ra với mục đích chat group nhưng hiện tại chúng em chưa làm được.

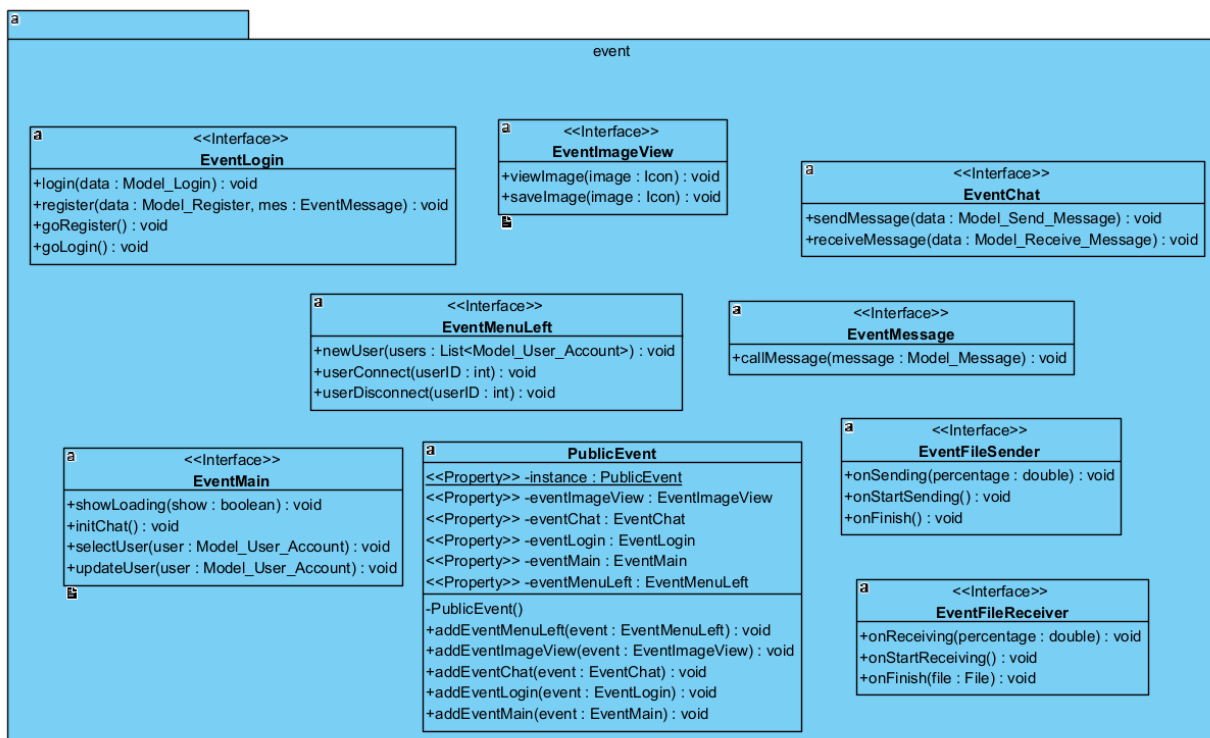
11. Chat_Item : Một mục chat đại diện cho một tin nhắn. Có các phương thức setText(), setTime(), setImage(), setFile(), setEmoji().
12. Chat_Right : Hiển thị tin nhắn từ người gửi. Chứa một Chat_Item bên trong. Trường boolean right = true bên trong xác định đây là một tin nhắn bên phải.
13. Image_Item : Hiển thị một hình ảnh. Gồm phương thức setImage() của file gửi và file nhận. Khi gửi và nhận ảnh, phần progress circle sẽ hiển thị tiến trình gửi, khi gửi/ nhận xong ảnh sẽ ẩn phần progress circle đi.
14. Chat_Left : Hiển thị tin nhắn từ người nhận. Tương tự như Chat_Right.

1.3.2. Package Emoji



1. Model_Emoji : Lớp đại diện riêng được tạo cho Emoji. Một Model_Emoji được xác định bởi ID của Emoji đấy và hình ảnh của nó.
2. Emoji : Bao gồm những phương thức để trả về list emoji theo chủ đề mà người dùng tương tác với nút bấm. Phương thức getStyleEmoji() trả về một List<Model_Emoji> từ những id của các emoji chung một nhóm. Phương thức getEmoji() trả về một model_emoji từ id của emoji đó và hình ảnh tạo từ đường dẫn của emoji đó.

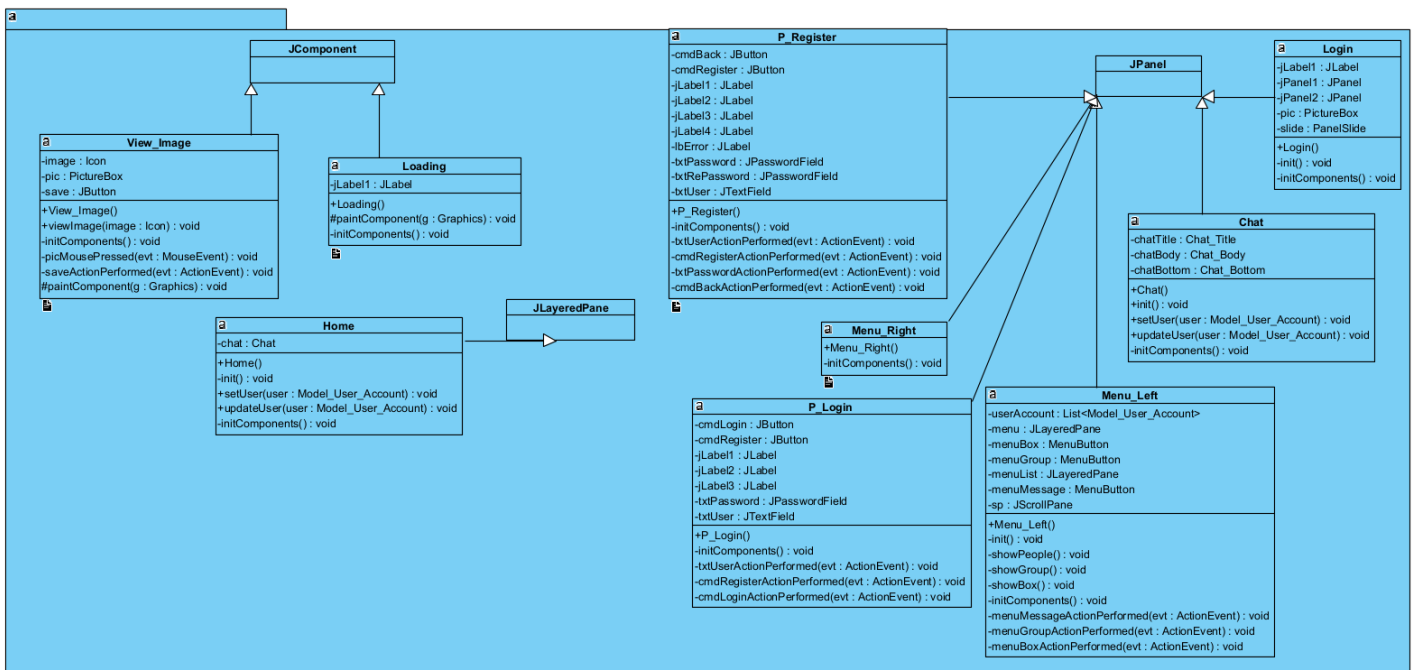
1.3.3 Package Event



1. PublicEvent : lớp chính quản lý và điều phối các sự kiện trong ứng dụng.
2. EventMain : Interface này có các phương thức liên quan đến các sự kiện chính trong ứng dụng khởi tạo giao diện chat, chọn người dùng và cập nhật thông tin người dùng.
3. EventLogin : Interface này định nghĩa các phương thức liên quan đến sự kiện đăng nhập, đăng ký và điều hướng tới giao diện đăng nhập.
4. EventMenuLeft : Interface này định nghĩa các phương thức quản lý người dùng trong menu bên trái.
5. EventChat : Interface này định nghĩa các phương thức gửi và nhận tin nhắn trong giao diện chat.

6. **EventMessage** : Interface này định nghĩa phương thức để gọi sự kiện liên quan đến tin nhắn.
7. **EventFileSender** : Interface này định nghĩa các phương thức liên quan đến việc gửi tệp tin.
8. **EventFileReceiver** : Interface này định nghĩa các phương thức liên quan đến việc nhận tệp tin.
9. **EventImageView** : Interface này định nghĩa các phương thức để xử lý hình ảnh.

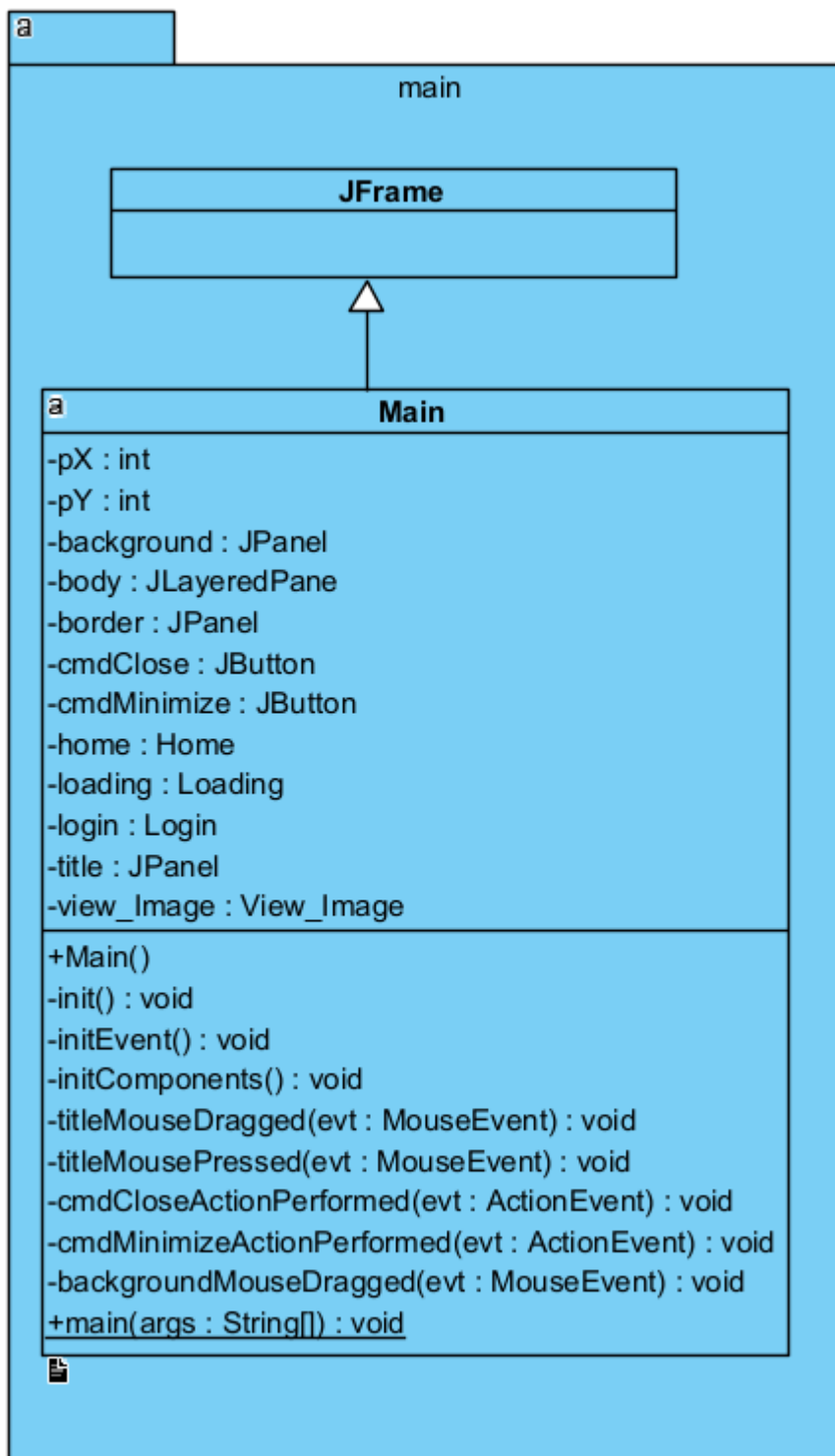
1.3.3. Package Form



1. **P_Register** : Giao diện đăng ký cho người dùng mới. Khi nhấn nút Register thì sẽ thực hiện lấy tên người dùng, mật khẩu, xác nhận mật khẩu từ các TextField tương ứng. Nếu text_field nào chưa nhập thì trở vào ô đấy, nếu thỏa mãn hết thì thực hiện đăng ký tại khoản. Nếu đăng ký thành công sẽ chuyển tới giao diện chat. Khi nhấn vào button back login thì gọi tới eventLogin().goLogin() để quay về giao diện login vì P_Register và P_Login thuộc một Panel_Slide.
2. **P_Login** : Giao diện đăng nhập cho người dùng hiện có. Khi nhấn nút Login thì sẽ thực hiện lấy tên người dùng, mật khẩu từ các TextField tương ứng. Nếu text_field nào chưa nhập thì trở vào ô đấy, nếu thỏa mãn hết thì thực hiện đăng nhập tài khoản. Nếu đăng nhập thành công sẽ chuyển tới giao diện chat. Khi nhấn vào button Register sẽ chuyển sang giao diện register.
3. **Login** : Màn hình chính của ứng dụng khi khởi động. Gồm một Panel_Slide, trong Panel_Slide có 2 Panel là P_Register và P_Login.

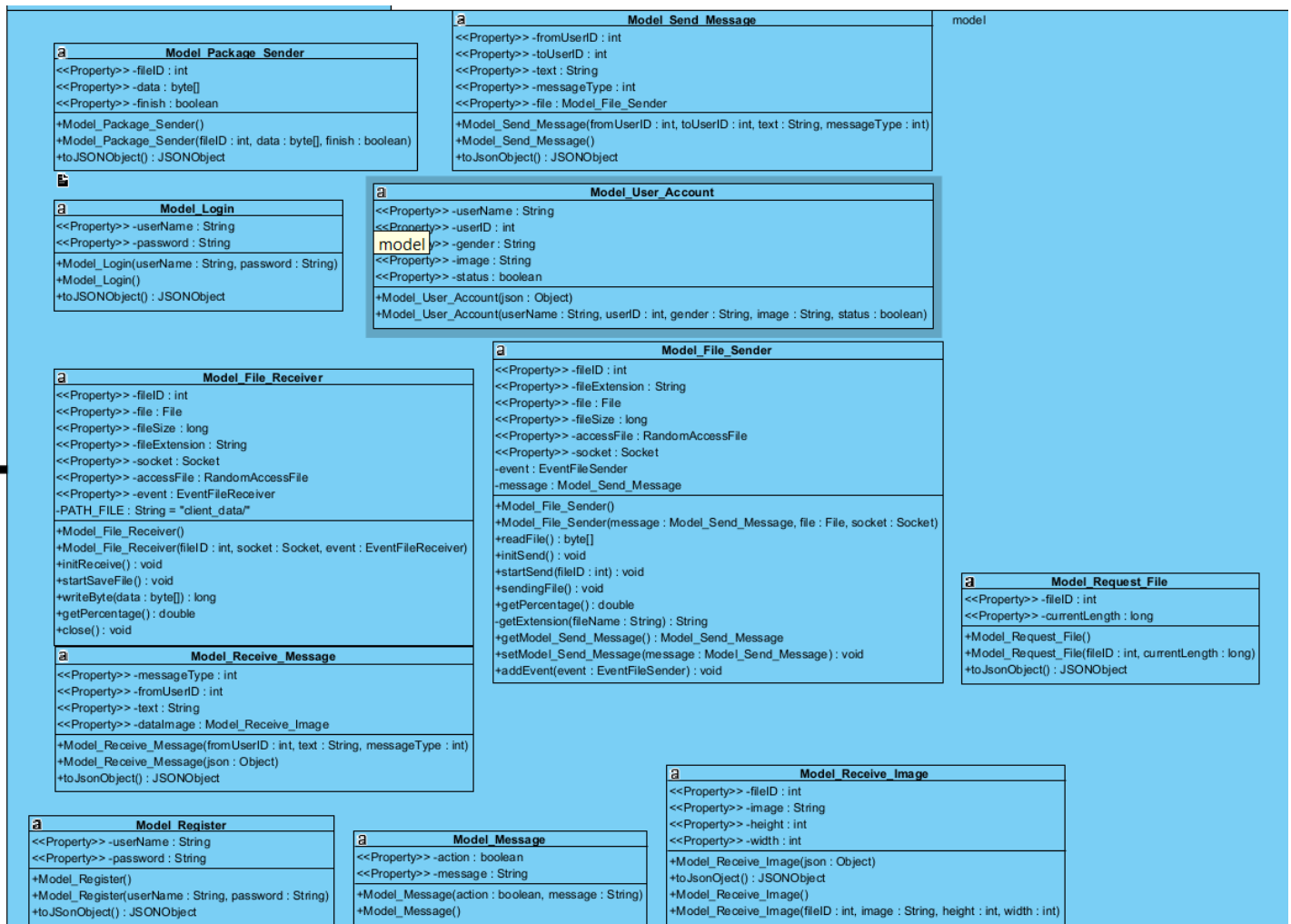
4. Menu_Right : Hiển thị menu điều hướng ở phía bên phải của màn hình. Hiện tại chúng em chưa sử dụng tới phần này, mục tiêu ban đầu là Menu_Right sẽ hiển thị các thông tin của người dùng đích.
5. Menu_Left : Hiển thị menu điều hướng ở phía bên trái của màn hình. Hiển thị các người dùng có trong hệ thống, hiển thị cả trạng thái hoạt động của họ, nhấn vào người dùng đó nếu muốn Chat với họ.
6. Chat : Giao diện trò chuyện cho người dùng. Chứa chat_title, chat_body, chat_bottom. Trong phương thức init() còn Override lại Interface EventChat tại Package Event.
7. Home : Màn hình chính sau khi người dùng đăng nhập. Bao gồm Menu_Left và Chat. Ban đầu Home sẽ được setVisible là false để hiển thị phần Loading.
8. View_Image : Giao diện hiển thị hình ảnh. Khi nhấn vào hình ảnh đã gửi / nhận trong đoạn chat sẽ hiển thị giao diện xem hình ảnh đó. Khi click chuột ra ngoài sẽ ẩn đi.
9. Loading : Giao diện hiển thị khi ứng dụng Client đã đăng nhập/ đăng ký xong và đang chờ kết nối tới Server.

1.3.4. Package Main



1. **Main** : Khởi động và khởi tạo giao diện và những chức năng , nút bấm người dùng có thể tương tác . Các button phóng to, thu nhỏ, tắt ứng dụng. Phương thức `init()` cài đặt các thành phần giao diện. Phương thức `initEvent()` Override lại các Interface `EventMain()`, `EventImageView()` trong Package `Event`.

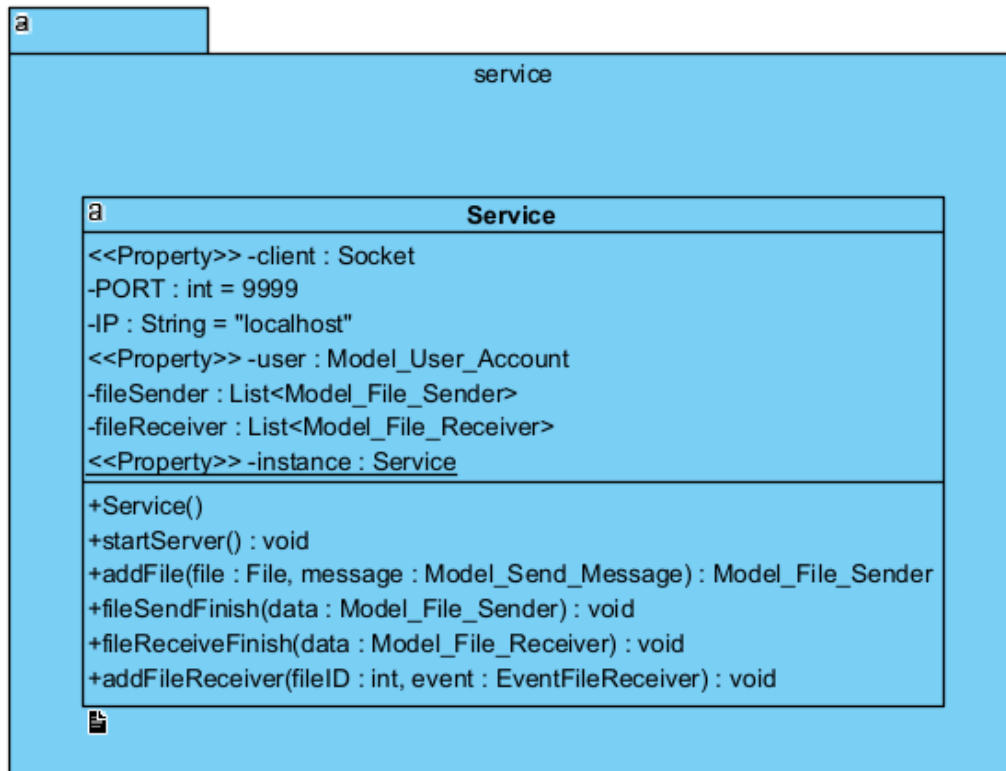
1.3.5. Package Model



1. **Model_Package_Sender** : Lớp đại diện được tạo ra để đóng gói thành kiểu dữ liệu có thể truyền dữ liệu file đi qua bên phía server để server xử lý. Phương thức `toJsonObject()` chuyển dữ liệu của gói tin sang kiểu dữ liệu Json để có thể trao đổi giữa client và server.
2. **Model_Send_Message** : Lớp đại diện được tạo ra để tạo dữ liệu tin nhắn có thể truyền qua bên phía server. Phương thức `toJsonObject()` chuyển dữ liệu của gói tin sang kiểu dữ liệu Json để có thể trao đổi giữa client và server. Gồm các thuộc tính: ID người gửi, ID người nhận, Text, kiểu tin nhắn.
3. **Model_Login** : Lớp đại diện tạo ra để tạo dữ liệu có thể đẩy sang server và đối chiếu với bên cơ sở dữ liệu của server. Gồm các thuộc tính `userName`, `password`. Phương thức `toJsonObject()` chuyển dữ liệu sang kiểu Json để trao đổi giữa client và server.

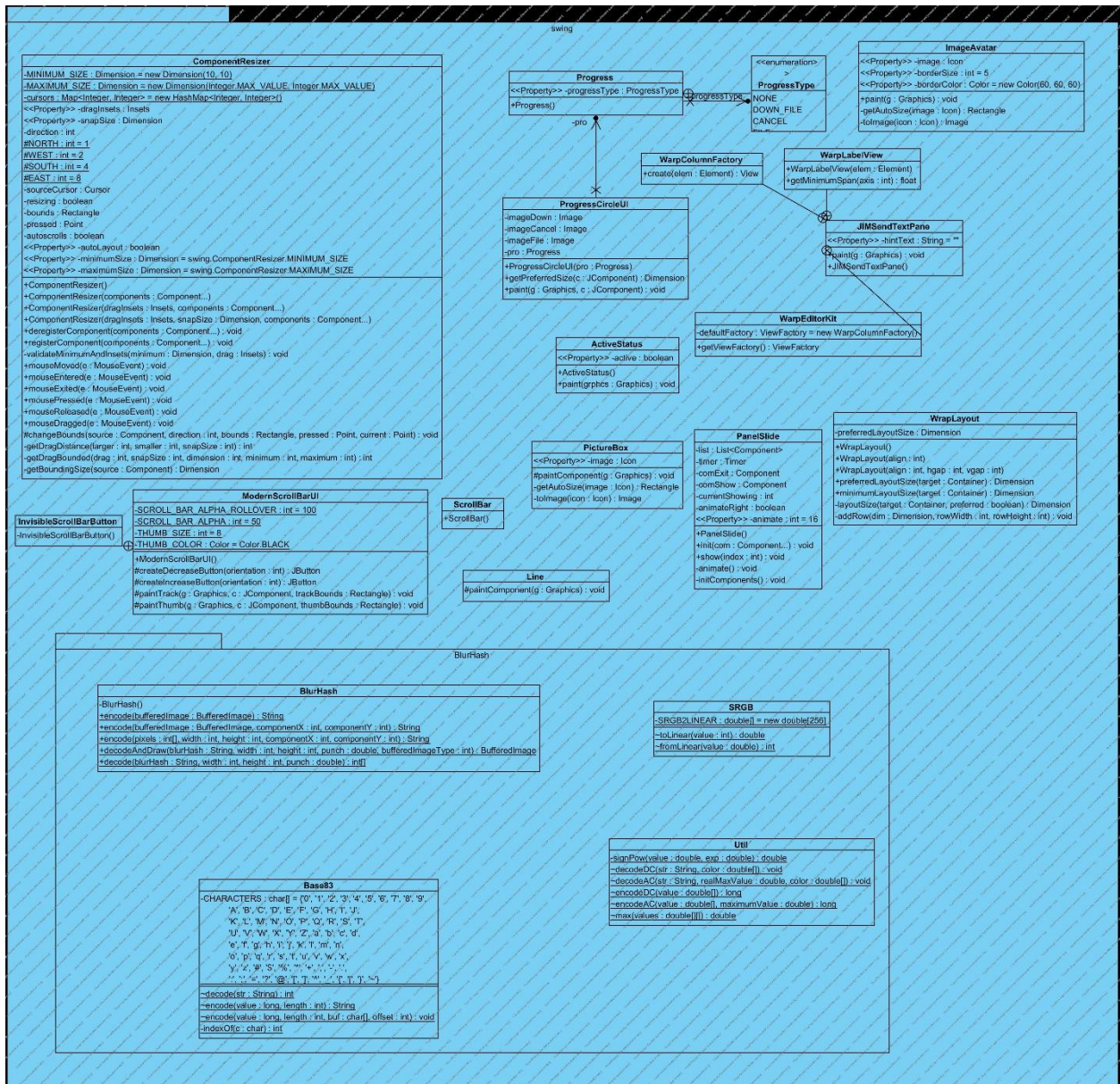
4. Model_User_Account : Lớp đại diện chứa các thuộc tính của một user account.
5. Model_File_Sender : Lớp đại diện cho một tập tin gửi đi. Phương thức readFile() đọc dữ liệu từ trong file ra mảng byte[] và trả về mảng byte[] đã đọc. Phương thức initSendImage() phát đi sự kiện “send_to_user” và xử lý dữ liệu nhận về. Phương thức initSendFile() tương tự phương thức initSendImage(). Phương thức sendingImage() phát đi sự kiện “send_image” và xử lý dữ liệu nhận từ Server. Phương thức sendingFile() tương tự phương thức sendingImage(). Phương thức getExtension() sẽ lấy phần mở rộng của fileName.
6. Model_File_Receiver : Lớp đại diện cho một tập tin nhận được. Phương thức writeFile() ghi dữ liệu từ một mảng byte[] ra một RandomAccessFile. Phương thức getPercentage() trả về phần trăm tiến trình nhận file. Phương thức initReceive() phát đi một sự kiện “get_file” và xử lý dữ liệu nhận về từ Server. Phương thức startSaveFile() phát đi sự kiện “request_file” và xử lý dữ liệu nhận về từ Server.
7. Model_Request_File : Lớp đại diện cho việc yêu cầu gửi file. Bao gồm 2 thuộc tính : fileID và curentLenght. fileID là ID của file đang yêu cầu, currentLenght là vị trí mà đang đọc tới từ lần yêu cầu trước. Phương thức toJsonObject() chuyển dữ liệu sang kiểu Json để trao đổi giữa client và server.
8. Model_Receive_Message : Lớp đại diện cho một tin nhắn nhận được. Bao gồm các thuộc tính : kiểu tin nhắn, ID người gửi, text, dữ liệu của ảnh nhận được (nếu có), dữ liệu của file nhận được (nếu có). Phương thức toJsonObject() chuyển dữ liệu sang kiểu Json để trao đổi giữa client và server.
9. Model_Register : Lớp đại diện được tạo ra để tạo dữ liệu và đẩy sang bên cơ sở dữ liệu lưu trữ của server. Gồm các thuộc tính userName, password. Phương thức toJsonObject() chuyển dữ liệu sang kiểu Json để trao đổi giữa client và server.
10. Model_Message : Lớp đại diện cho tin nhắn.
11. Model_Receive_Image : Lớp đại diện cho một tin nhắn ảnh nhận được. Bao gồm các thuộc tính : fileID, dữ liệu ảnh lưu ở dạng string, kích thước ảnh. Phương thức toJsonObject() chuyển dữ liệu sang kiểu Json để trao đổi giữa client và server.

1.3.6. Package Service



1. **Service** : Lớp có chức năng thực hiện kết nối với bên server và giao tiếp chính với bên server. Ứng dụng sử dụng PORT 9999 và IP là localhost. Phương thức `startServer()` tạo đối tượng Socket tại IP và PORT trên và kết nối tới Server, client cũng lắng nghe các sự kiện “list_user”, “user_status”, “receive_ms” và xử lý các dữ liệu nhận về. Phương thức `addFileImage` sẽ thêm một hình ảnh vào danh sách các hình ảnh cần gửi và gửi từng ảnh đi. Phương thức `addFileFile()` thêm một file vào danh sách các file cần gửi và gửi từng file đi. Phương thức `fileSendFinish()` xóa file đã gửi xong và gửi cái tiếp theo trong danh sách. Phương thức `fileReceiveFinish()` xóa file đã nhận cái tiếp theo. Phương thức `addFileReceiver()` thêm file cần nhận vào danh sách các file và nhận từng cái.

1.3.7. Package Swing

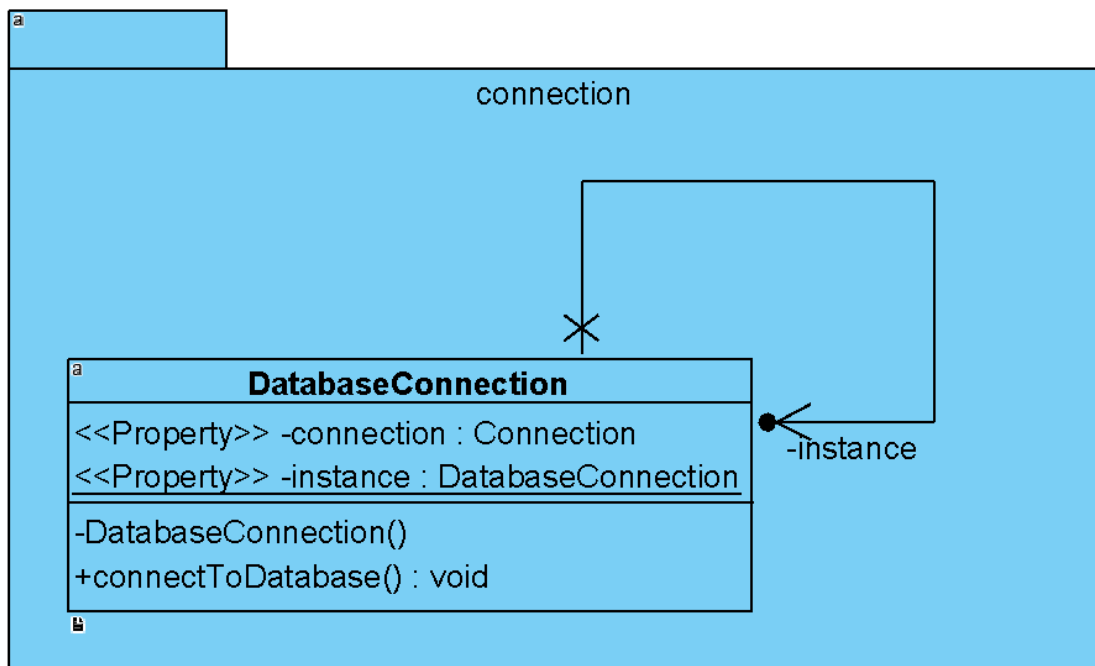


1. **ComponentResizer** : cho phép người dùng thay đổi kích thước các thành phần, chẳng hạn như cửa sổ hoặc bảng, trong ứng dụng Swing
2. **InvisibleScrollbarButton** : Một nút thanh cuộn không hiển thị, hữu ích để tạo giao diện thanh cuộn tùy chỉnh khi không cần hoặc muốn ẩn các nút mặc định.
3. **ModernScrollbarUI** : Giao diện tùy chỉnh cho các thanh cuộn, cung cấp một giao diện hiện đại, khác với giao diện thanh cuộn mặc định của Swing
4. **ScrollBar** : Triển khai tùy chỉnh của thanh cuộn, có thể có tính năng hoặc giao diện mở rộng so với thanh cuộn mặc định
5. **Line** : Một lớp để vẽ hoặc quản lý các đường kẻ trong giao diện đồ họa.
6. **PictureBox** : Khung chứa hình ảnh, có thể hiển thị hình ảnh trong ứng dụng Swing.
7. **PanelSlide** : Bảng trượt, có thể được sử dụng để tạo hiệu ứng chuyển động trượt giữa các bảng hoặc thành phần

8. **WrapLayout** : Bố cục dòng, tự động sắp xếp các thành phần thành nhiều dòng nếu không đủ không gian trong một dòng duy nhất
9. **ActiveStatus** : Có thể là một lớp để quản lý và hiển thị trạng thái hoạt động của một thành phần hoặc người dùng
10. **WrapEditorKit** : Bộ công cụ chỉnh sửa dòng, có thể cung cấp khả năng chỉnh sửa văn bản với tính năng dòng tự động ngắt
11. **ProgressCircleUI** : Giao diện người dùng cho vòng tròn tiến trình, có thể hiển thị tiến trình dưới dạng vòng tròn thay vì thanh tiến trình truyền thống
12. **JIMSendTextPane** : Một bảng văn bản tùy chỉnh, có thể được thiết kế để gửi tin nhắn hoặc văn bản, có thể liên quan đến các ứng dụng nhắn tin hoặc chat
13. **WrapLabelView** : một lớp để hiển thị nhãn (label) với tính năng tự động ngắt dòng.
14. **WrapColumnFactory** : một lớp để tạo ra các cột với tính năng tự động ngắt dòng , tự động chia dòng
15. **Progress** : Một lớp để quản lý hoặc hiển thị tiến trình. Có thể là một thành phần giao diện người dùng để biểu thị mức độ hoàn thành của một tác vụ nào đó
16. **ProgressType** : Một enum hoặc lớp để định nghĩa các loại tiến trình khác nhau, ví dụ như tiến trình tuyến tính, vòng tròn, hoặc kiểu hiển thị khác.
17. **ImageAvatar** : Một lớp để hiển thị hình ảnh đại diện (avatar), thường được sử dụng trong các ứng dụng mạng xã hội hoặc hệ thống người dùng để hiển thị hình ảnh cá nhân.
18. **BlurHash** :
 - ii. **BlurHash** : Thuật toán hoặc triển khai cụ thể của BlurHash để mã hóa và giải mã hình ảnh thành chuỗi mờ
 - iii. **Base83** : Một mã hóa đặc biệt được sử dụng bởi BlurHash để chuyển đổi dữ liệu thành chuỗi ký tự sử dụng 83 ký tự khác nhau.
 - iv. **SRGB** : Không gian màu sRGB được sử dụng để quản lý màu sắc
 - v. **Util** : Các tiện ích hỗ trợ, có thể là một tập hợp các hàm hoặc phương thức tiện ích để hỗ trợ việc thực hiện các tác vụ liên quan đến các lớp hoặc chức năng trên

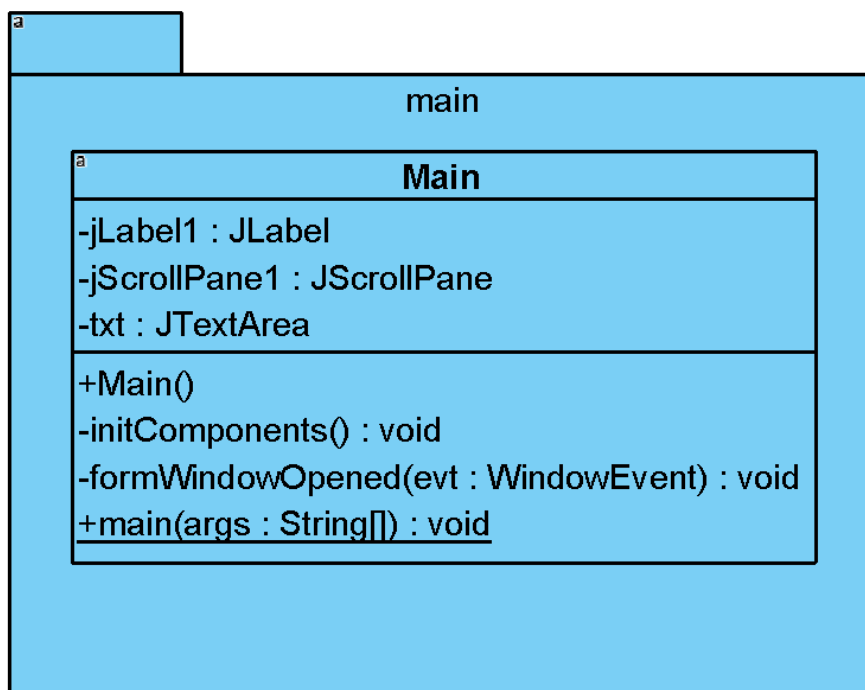
1.4. Thiết kế chi tiết lớp phía Server :

1.4.1. Package Connection



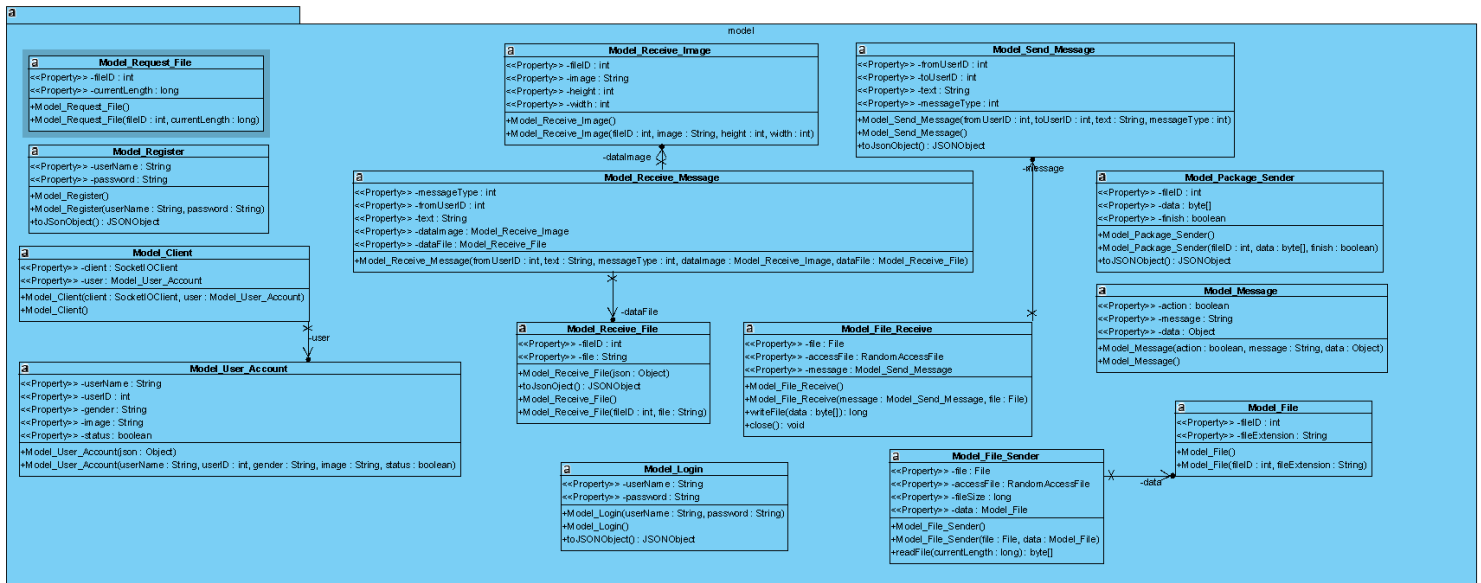
1. Database Connection : Lớp thực hiện chức năng kết nối với cơ sở dữ liệu của Server. Phương thức `connectToDatabase()` sẽ kết nối với MySQL.

1.4.2. Package Main



1. Main : Cung cấp giao diện khi mở server. Hiện thị các client đã kết nối và in ra lỗi nếu như có lỗi kết nối Database.

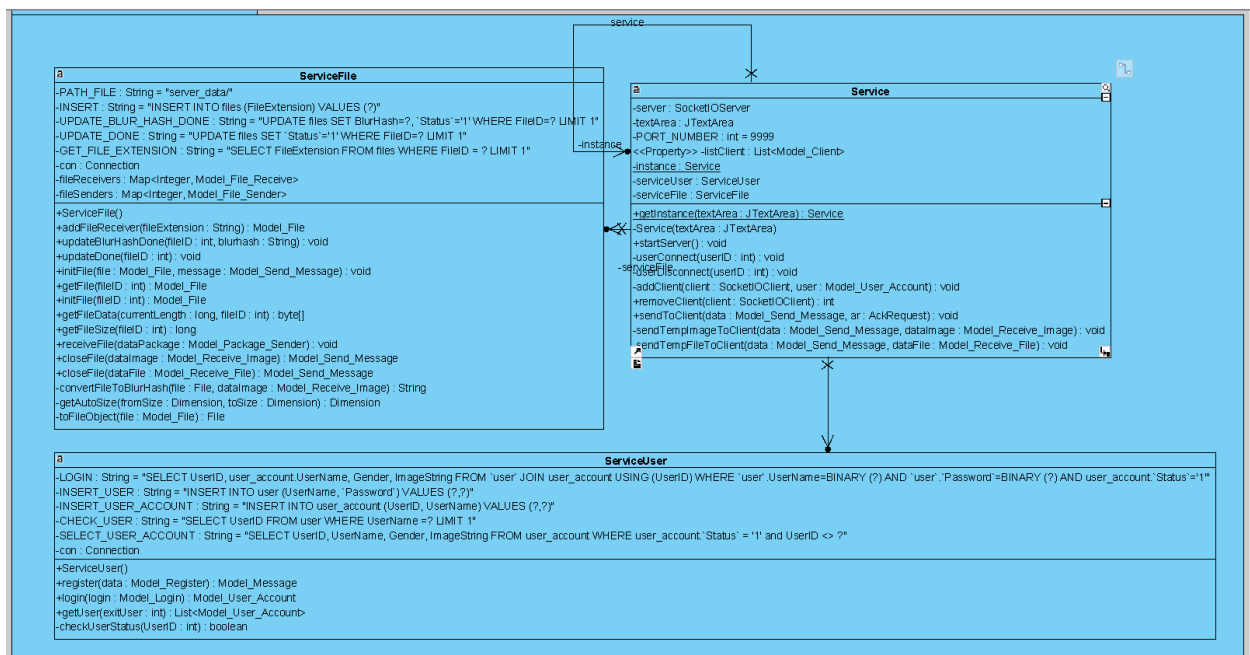
1.4.3. Package Model



1. Model_Request_File : Lớp đại diện cho việc yêu cầu gửi file. Bao gồm 2 thuộc tính : fileID và currentLength. fileID là ID của file đang yêu cầu, currentLength là vị trí mà đang đọc tới từ lần yêu cầu trước.
2. Model_Register : Lớp đại diện được tạo ra để tạo dữ liệu và đẩy sang bên cơ sở dữ liệu lưu trữ của server. Gồm các thuộc tính userName, password. Phương thức toJsonObject() chuyển dữ liệu sang kiểu Json để trao đổi giữa client và server.
3. Model_Client : Lớp đại diện cho 1 client kết nối với server.
4. Model_User_Account : Lớp đại diện chứa các thuộc tính của một user account.
5. Model_Receive_Image : Lớp đại diện cho một tin nhắn ảnh nhận được. Bao gồm các thuộc tính : fileID, dữ liệu ảnh lưu ở dạng string, kích thước ảnh.
6. Model_Receive_Message : Lớp đại diện chứa thông tin về tin nhắn .
7. Model_Receive_File : Lớp đại diện chứa thông tin về file. Phương thức toJsonObject() chuyển dữ liệu sang kiểu Json để trao đổi giữa client và server.
8. Model_Login : Lớp đại diện tạo ra để tạo dữ liệu có thể trao đổi giữa client và server từ các dữ liệu userName và password. Gồm các thuộc tính userName, password. Phương thức toJsonObject() chuyển dữ liệu sang kiểu Json để trao đổi giữa client và server.

9. **Model_File_Receive** : Lớp đại diện cho một tập tin nhận được. Phương thức `writeFile()` ghi dữ liệu từ một mảng `byte[]` ra một `RandomAccessFile`.
10. **Model_Send_Message** : Lớp đại diện được tạo ra để tạo dữ liệu tin nhắn có thể truyền qua bên phía server. Phương thức `toJsonObject()` chuyển dữ liệu của gói tin sang kiểu dữ liệu `Json` để có thể trao đổi giữa client và server. Gồm các thuộc tính: ID người gửi, ID người nhận, Text, kiểu tin nhắn.
11. **Model_Package_Sender** : Lớp đại diện được tạo ra để đóng gói thành kiểu dữ liệu có thể truyền dữ liệu file đi qua bên phía server để server xử lý. Phương thức `toJsonObject()` chuyển dữ liệu của gói tin sang kiểu dữ liệu `Json` để có thể trao đổi giữa client và server.
12. **Model_Message** : Lớp đại diện cho 1 tin nhắn.
13. **Model_File** : Chứa thông tin file gồm `fileID` và `fileExtension`.
14. **Model_File_Sender** : Lớp đại diện cho một tập tin gửi đi. Phương thức `readFile()` đọc dữ liệu từ trong file ra mảng `byte[]` và trả về mảng `byte[]` đã đọc.

1.4.4. Package Service

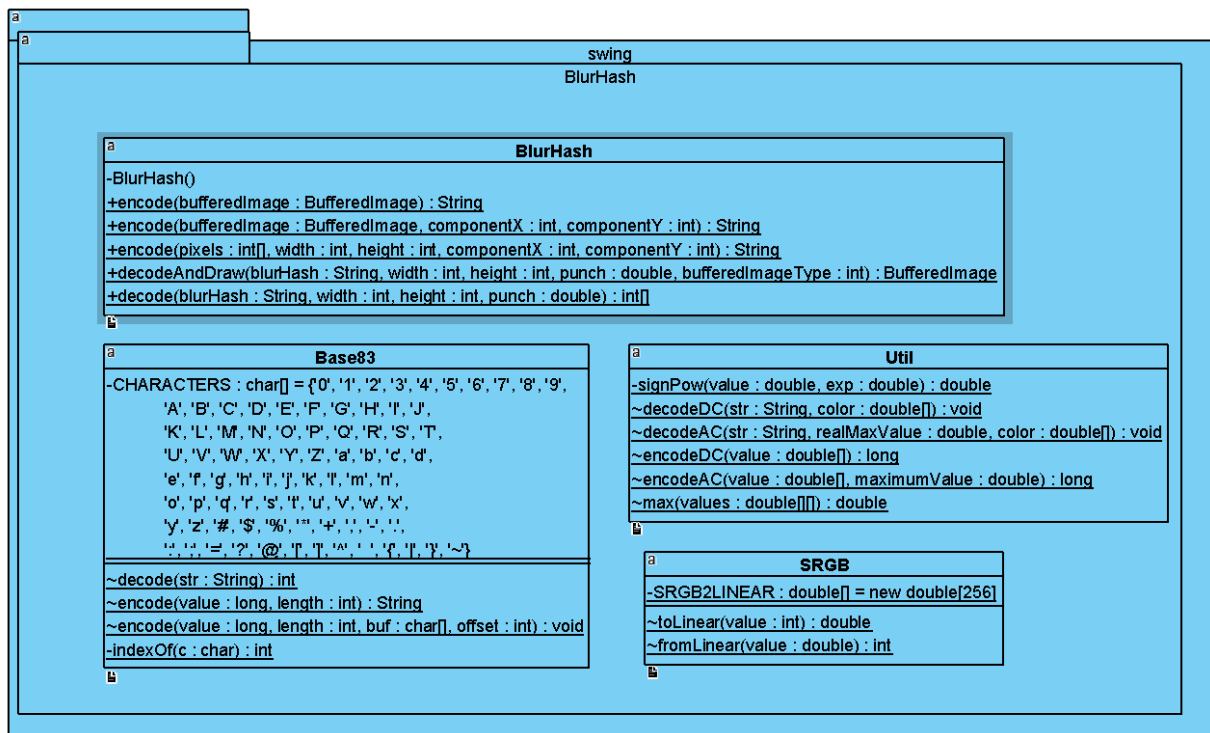


1. **ServiceFile** : Kết nối cơ sở dữ liệu thao tác với cơ sở dữ liệu chứa tập thông qua dãy lệnh được cài sẵn và những phương thức xử lý với những tập . Phương thức `addFileReceiver()` sẽ thêm `fileID` và `fileExtension` vào table `files`. Phương thức

updateBlurHashDone() sẽ cập nhật lại trường BlurHash và trường Status của file trong table files. Phương thức updateDone cập nhật Status tại bản ghi có fileID bằng fileID truyền vào. Phương thức initFile() thêm một cặp khóa giá trị vào HashMap. Phương thức getFile trả về fileID, fileExtension từ fileID truyền vào bằng cách lấy trong database. Phương thức closeFile() sẽ cập nhật lại trường BlurHash và Status và xóa cặp dữ liệu tương ứng ra khỏi hashmap. Phương thức convertFileToBlurHash() sẽ mã hóa ảnh thành ảnh mờ. Phương thức getAutoSize() là phương thức chuyển đổi kích thước của ảnh, chuyển từ kích thước ảnh gốc tới kích thước mới bị giới hạn. Phương thức toFileObject() chuyển từ Model_File sang một File từ fileID và fileExtension của Model_File.

2. Service : Gồm các phương thức khởi tạo server và thực hiện chức năng của 1 server. Phương thức startServer() khởi tạo và bắt đầu server, server lắng nghe nếu có kết nối thì sẽ hiển thị thông báo, server lắng nghe sự kiện "login" và xử lý dữ liệu nhận được và gửi phản hồi, server lắng nghe sự kiện "register" và xử lý dữ liệu nhận được và gửi phản hồi về kết quả đăng ký. server lắng nghe sự kiện "list_user" và xử lý dữ liệu nhận được và gửi danh sách người dùng trở lại client, server đăng ký một DisconnectListener để xử lý khi client ngắt kết nối khi client ngắt kết nối xác định client nào và gửi sự kiện "user_status", server lắng nghe sự kiện "send_to_user" nhận tin nhắn từ client và gửi tới client đích, server lắng nghe sự kiện "send_image" và gửi sự kiện "receive_ms" tới client đích, server lắng nghe sự kiện "send_file" và phát sự kiện "receive_ms" để gửi file tới client đích.
3. ServiceUser : Kết nối cơ sở dữ liệu thao tác với cơ sở dữ liệu User. Phương thức register nhận đối tượng Model_Register chứa thông tin người dùng muốn đăng ký, trả về thông báo kết quả đăng ký, nó kiểm tra xem tên người dùng đã tồn tại hay chưa bằng cách thực hiện câu lệnh SQL CHECK_USER, nếu tên người dùng đã tồn tại, message sẽ được thiết lập với action=false và thông báo "User Already Exit", nếu tên người dùng chưa tồn tại, nó sẽ thực hiện câu lệnh SQL INSERT_USER để thêm người dùng mới vào cơ sở dữ liệu. Phương thức login() lấy tài khoản người dùng từ tài khoản và mật khẩu bằng cách truy cập cơ sở dữ liệu. Phương thức getUser() lấy danh sách tài khoản người dùng, ngoại trừ tài khoản đang đăng nhập. Phương thức checkUserStatus() kiểm tra trạng thái hoạt động của user.

1.4.5. Package Swing.BlurHash



1. BlurHash : Thuật toán hoặc triển khai cụ thể của BlurHash để mã hóa và giải mã hình ảnh thành chuỗi mờ.
2. Base83 : Một mã hóa đặc biệt được sử dụng bởi BlurHash để chuyển đổi dữ liệu thành chuỗi ký tự sử dụng 83 ký tự khác nhau.
3. Util : Các tiện ích hỗ trợ, có thể là một tập hợp các hàm hoặc phương thức tiện ích để hỗ trợ việc thực hiện các tác vụ liên quan đến các lớp hoặc chức năng trên.
4. SRGB : Không gian màu sRGB được sử dụng để quản lý màu sắc

CHƯƠNG 3: CÔNG NGHỆ VÀ THUẬT TOÁN SỬ DỤNG

3.2. Thuật toán và các kiến thức sử dụng

Để thực thi một giải pháp cho một ý tưởng thì không thể thiếu các kỹ thuật, cấu trúc dữ liệu và các thuật toán được áp dụng. Về cơ bản một ứng dụng như Messenger trong quá trình lập trình cần sử dụng rất nhiều những kỹ thuật lập trình và nổi bật hơn cả chúng ta có thể đề ý đến là kỹ thuật hướng đối tượng. Nói về hướng đối tượng, để có một giao diện và các tính năng hoàn chỉnh như mong đợi trước hết chúng ta cần phân tích và thiết kế để hoạch định những tính năng cần làm và những vấn đề cần giải quyết. Qua việc học môn Lập trình hướng đối tượng chúng ta có được tư duy phân tích thiết kế cơ bản cho hệ thống phần mềm với việc chia nhỏ các nhóm tính năng, công việc thành các package, class để vừa dễ dàng phân công trong một nhóm nhiều người làm - tăng năng suất làm việc, lại vừa có thể tổ chức kiểm soát, quản lý và bảo trì mã nguồn. Với việc sử dụng một ngôn ngữ thuần hướng đối tượng như Java, việc tổ chức thực thi lập trình cho phần mềm của chúng em giảm bớt được rất nhiều khó khăn, vướng mắc. Nhờ lập trình hướng đối tượng với ngôn ngữ Java chúng ta thể hiện được các đối tượng, các ý tưởng vào mã nguồn và xử lý để đưa chúng ra thành giao diện thực tế một cách dễ dàng. Các package và class cụ thể cũng như việc vận dụng kỹ thuật hướng đối tượng ra sao đã được trình bày ở chương 2, bao gồm chủ yếu là tính gói của 1 class (Encapsulation), tính kế thừa (Inheritance), nạp chồng phương thức (Overloading), ghi đè phương thức (Overriding), thực thi Giao diện (Interface).

Ngoài ra chúng em có vận dụng các kiến thức về thiết kế giao diện, đồ họa người dùng.

Một khía cạnh quan trọng khác không thể không kể đến là cấu trúc dữ liệu và thuật toán. Nhìn chung trong toàn bộ mã nguồn của phần mềm các cấu trúc dữ liệu và thuật toán cơ bản lớn nhỏ mà các lập trình viên đã quen mặt được sử dụng rất nhiều như việc sử dụng các biến, các cấu trúc lập for, foreach, cấu trúc rẽ nhánh if/else,... trong đó nổi bật lên là cấu trúc ArrayList để khởi tạo cho chuỗi các đối tượng được xây dựng. Đó là những cấu trúc dữ liệu mà chúng ta đã được tìm hiểu trong chương Lập trình tổng quát của học phần. Còn về phía giải thuật thì nổi trội lên là thuật toán Tìm kiếm tuần tự được dùng để tìm kiếm cán bộ theo theo tên, đơn vị công tác, hệ số lương.

CHƯƠNG 4: XÂY DỰNG CHƯƠNG TRÌNH MINH HOẠ

4.1. Kết quả của chương trình minh họa

Chương trình được đặt tên là Messenger, logo như hình bên dưới.

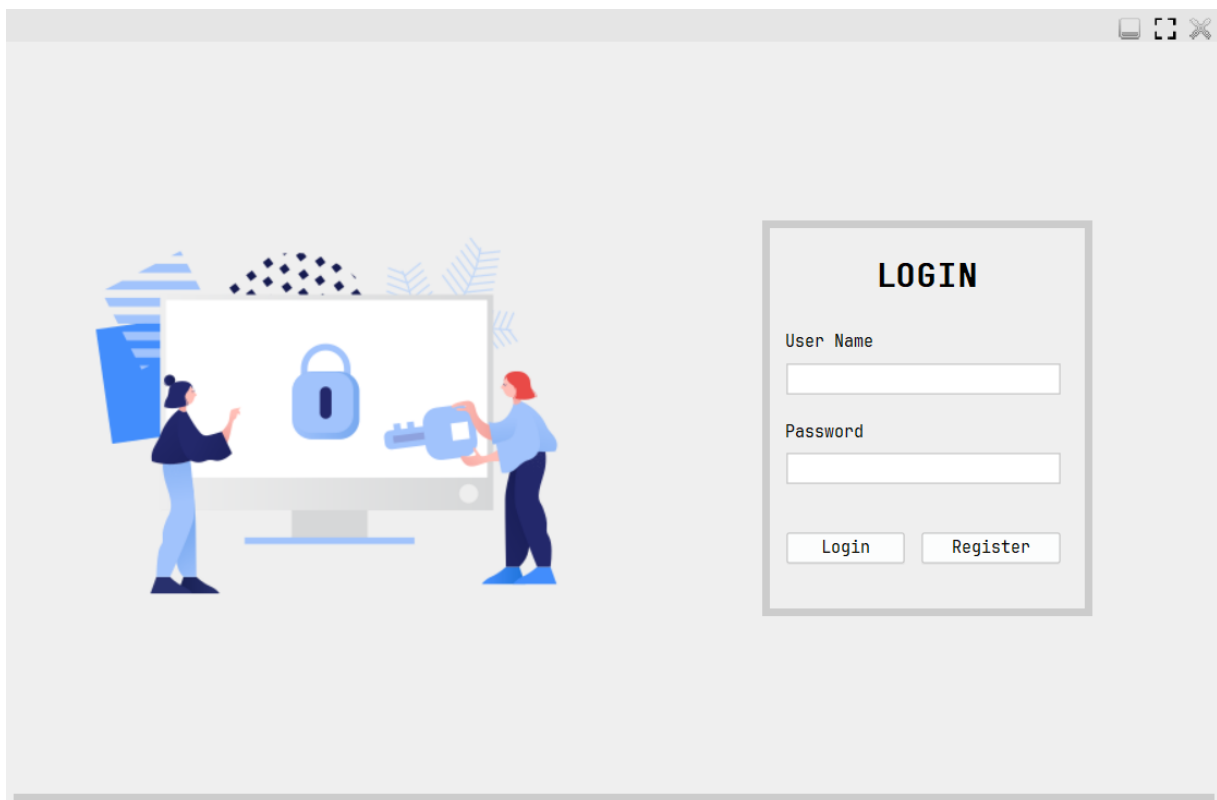


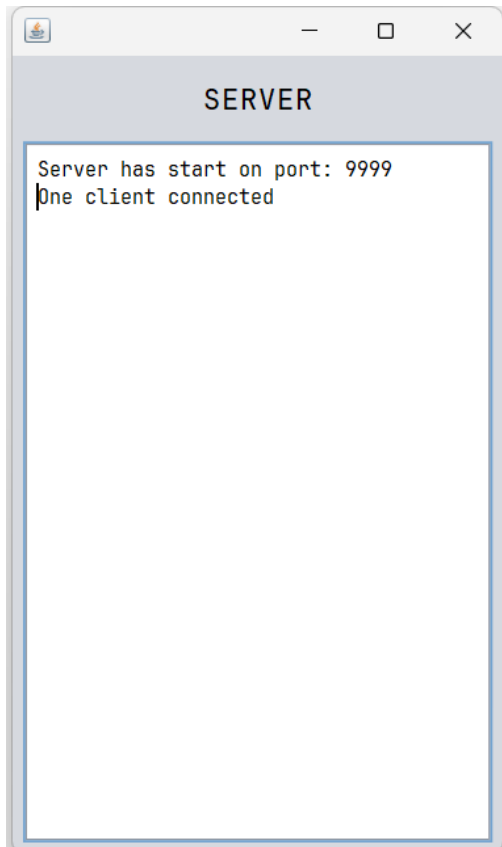
Chương trình được xây dựng đáp ứng đầy đủ tất cả các yêu cầu ban đầu của đề tài về chức năng và giao diện

4.2. Giao diện chương trình

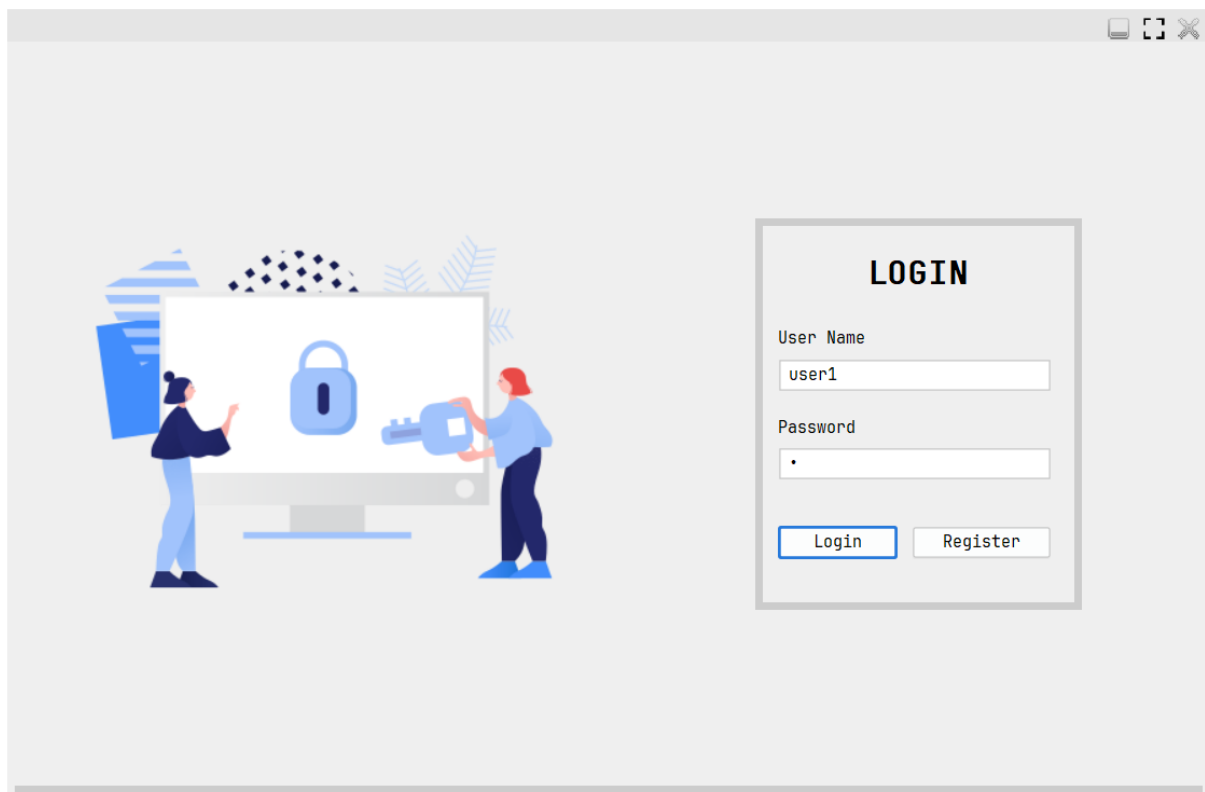
4.2.1. Giao diện khởi động

Bên Client:



Bên Server:**4.2.2. Giao diện đăng nhập và đăng ký**

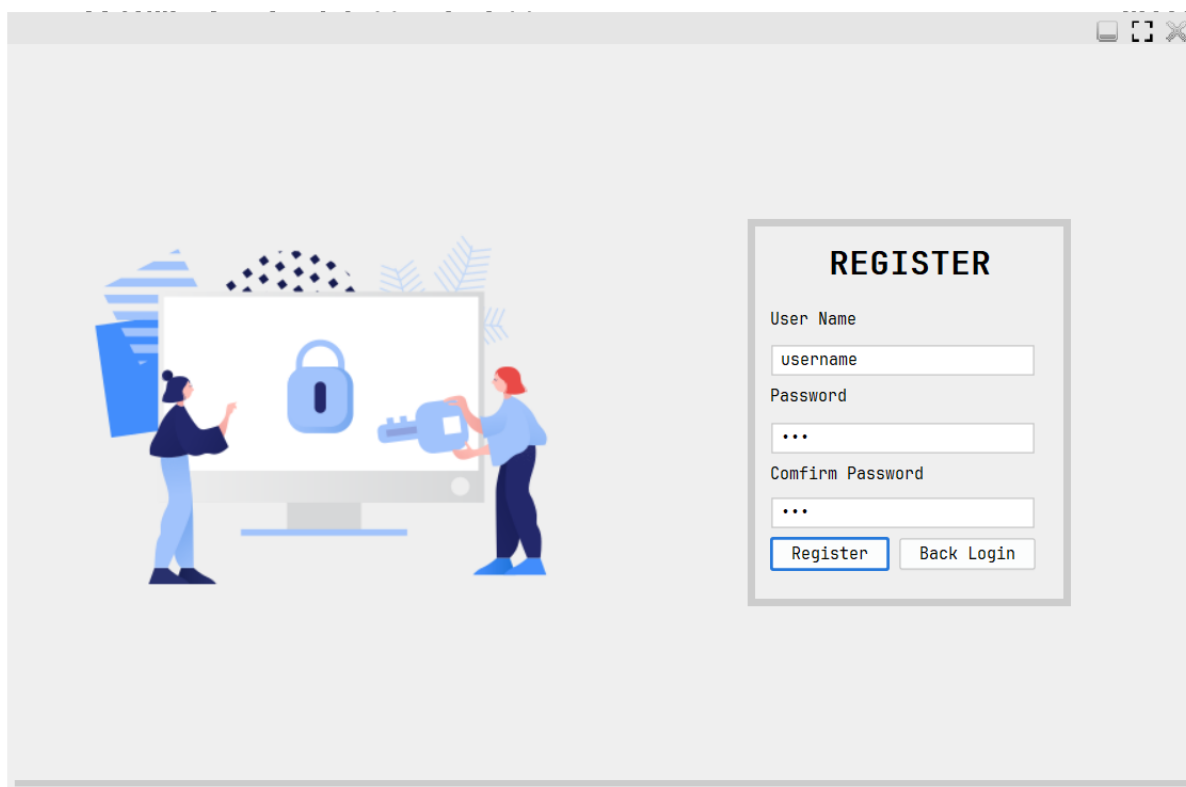
- Giao diện đăng nhập:



Các chức năng tương tác có trong giao diện:

1. Thao tác đăng nhập: Nhập tên người dùng và mật khẩu vào ô tương ứng. Nhấn nút “Login” để đăng nhập.
2. Thao tác chuyển sang giao diện đăng ký: Nhấn vào nút “Register” để chuyển sang giao diện đăng ký.
3. Thao tác phóng to, thu nhỏ, tắt chương trình: Nhấn vào các nút tương ứng ở góc trên của giao diện.

- Giao diện đăng ký:



The screenshot displays a web interface for user registration. On the left, there is an illustration of two people standing in front of a large monitor. The monitor shows a blue padlock icon, and one person is holding a large blue key. On the right, there is a registration form titled "REGISTER". The form contains the following fields and buttons:

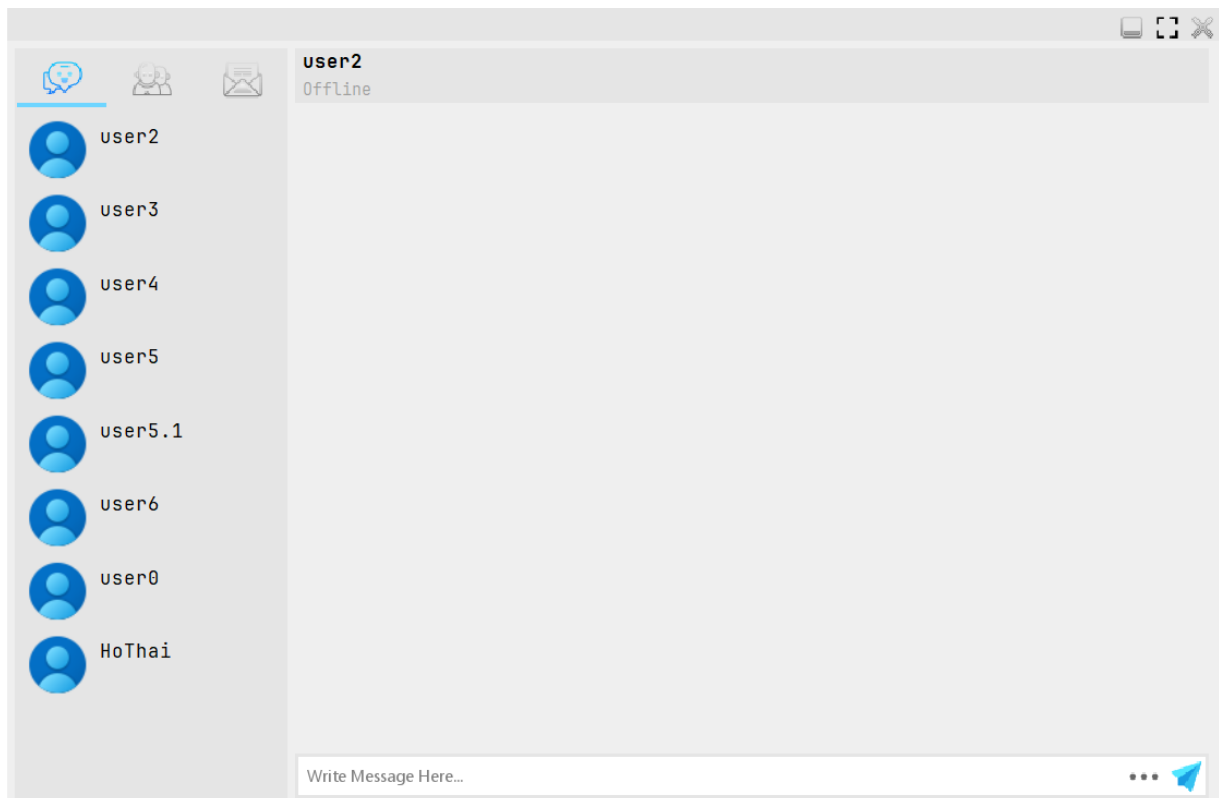
- User Name**: A text input field with the placeholder text "username".
- Password**: A text input field with three dots indicating a password.
- Comfirm Password**: A text input field with three dots indicating a password.
- Buttons**: Two buttons at the bottom, "Register" (highlighted with a blue border) and "Back Login".

Các chức năng tương tác có trong giao diện:

1. Thao tác đăng ký: Nhập tên người dùng, mật khẩu và xác nhận mật khẩu vào các ô tương ứng. Nhấn nút “Register” để đăng ký.
2. Thao tác chuyển về giao diện đăng nhập: nhấn nút “Back Login” để trở về giao diện đăng nhập.
3. Thao tác phóng to, thu nhỏ, tắt chương trình: Nhấn vào các nút tương ứng ở góc trên của giao diện.

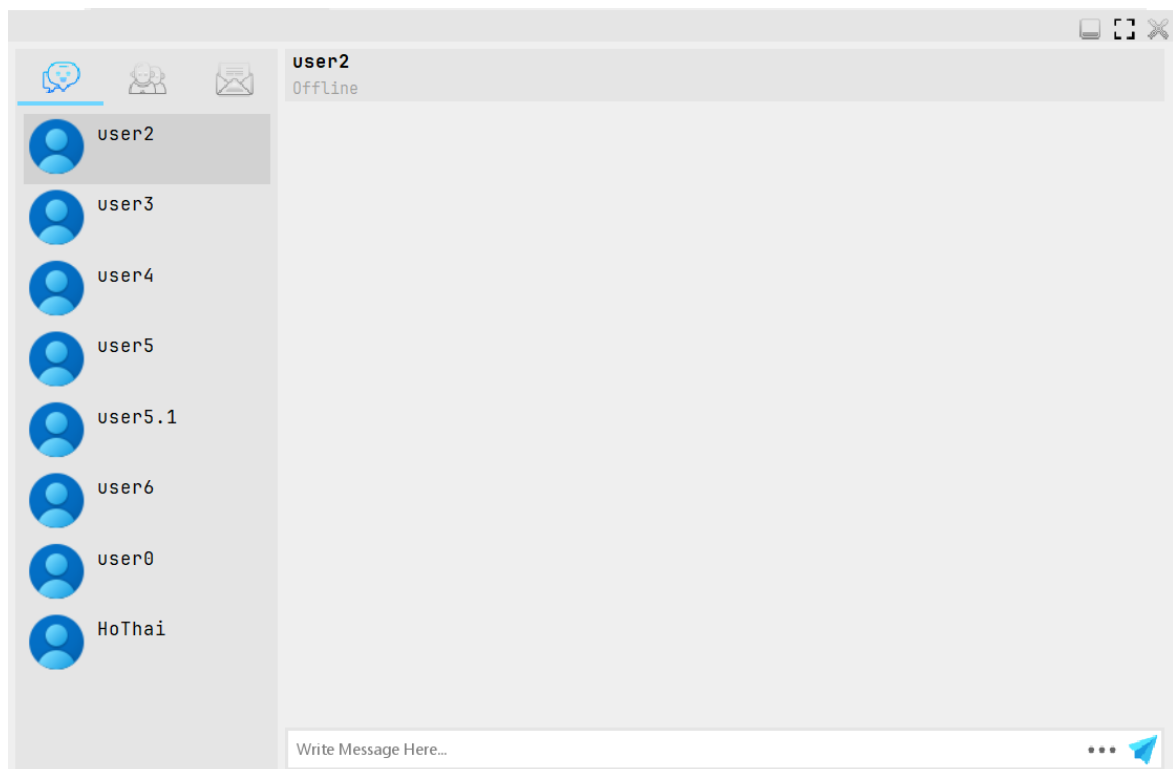
4.2.3. Giao diện nhắn tin chính

Màn hình chính:



Các chức năng tương tác có trong giao diện:

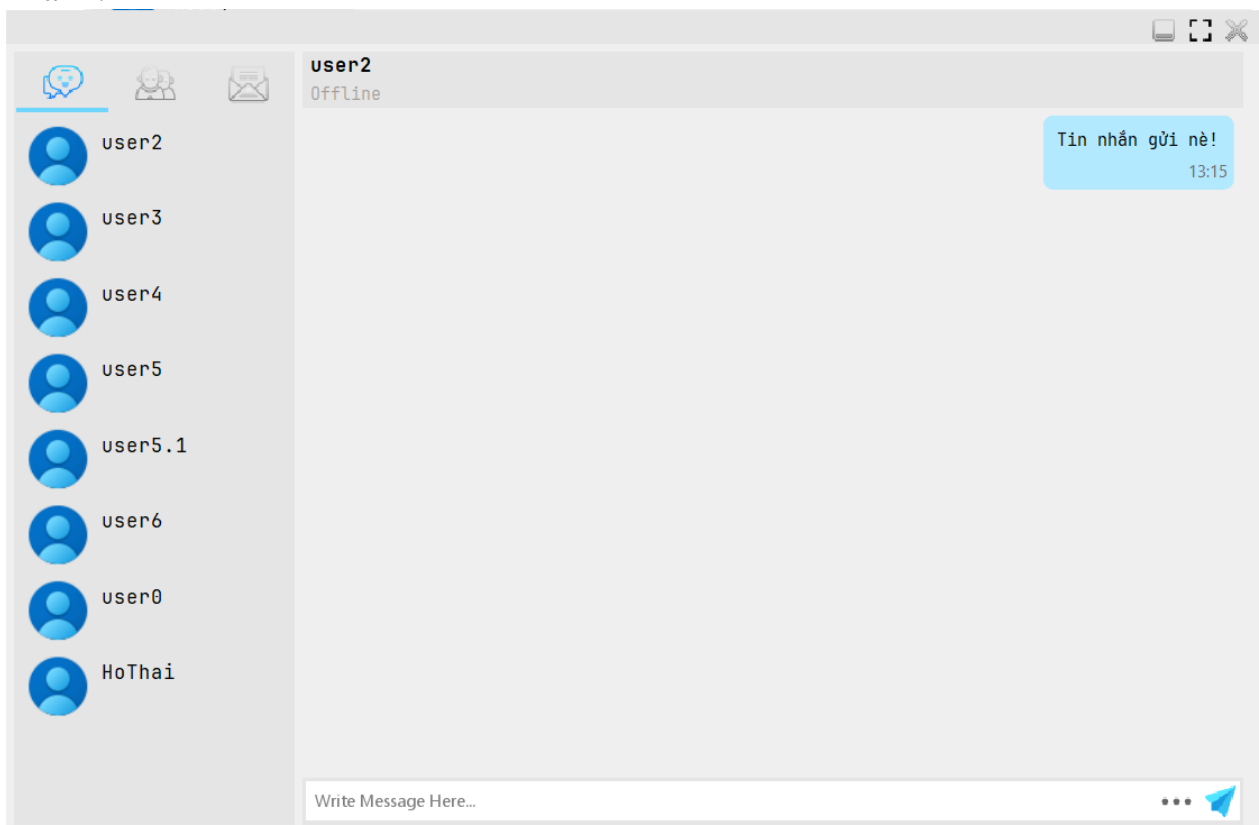
1. Thao tác chọn người dùng đích để gửi tin nhắn: Trong phần menu bên trái, chọn người dùng muốn nhắn tin và nhấp vào. Khi di chuột tới người dùng nào thì phần chọn người dùng đó sẽ được làm nổi bật hơn.



2. Thao tác nhập tin nhắn: Trong phần khung ở phía dưới của giao diện nơi có ghi gợi ý “Write message here...” là nơi để nhập tin nhắn text.

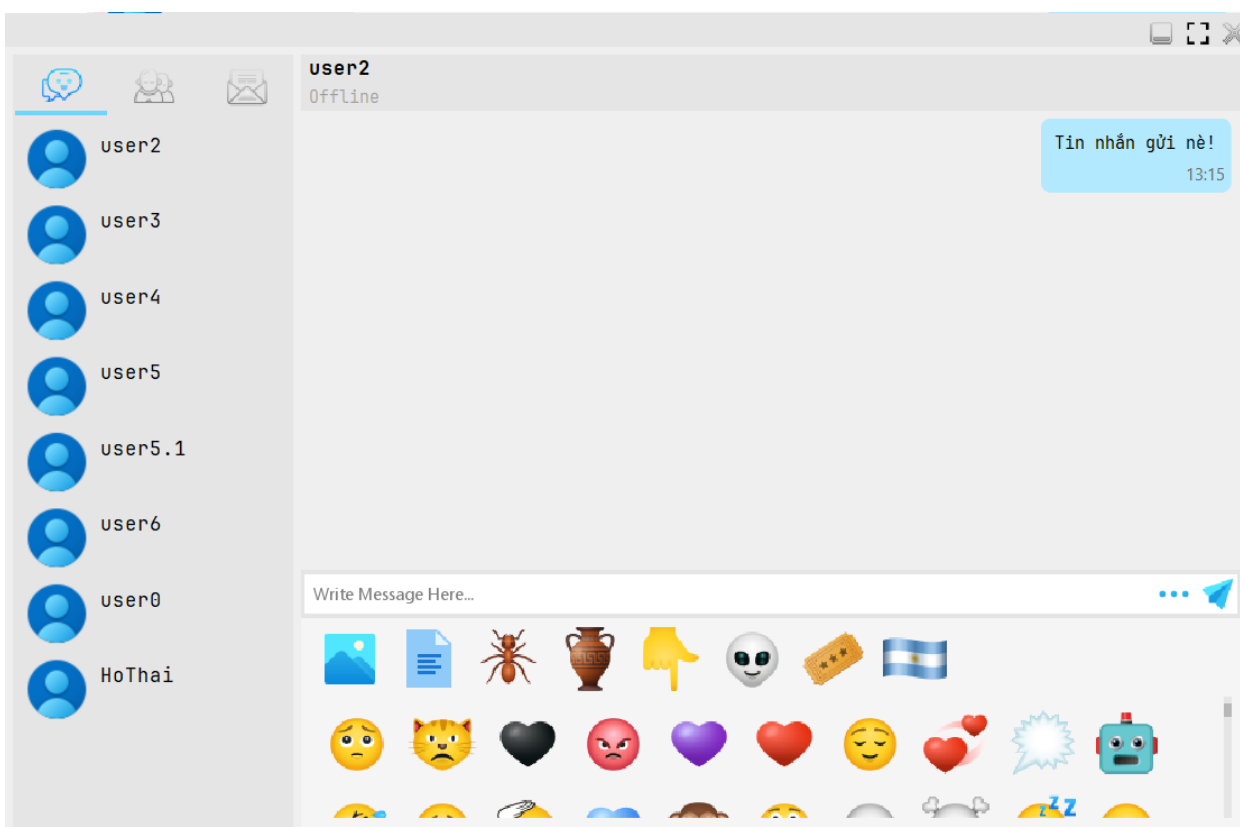
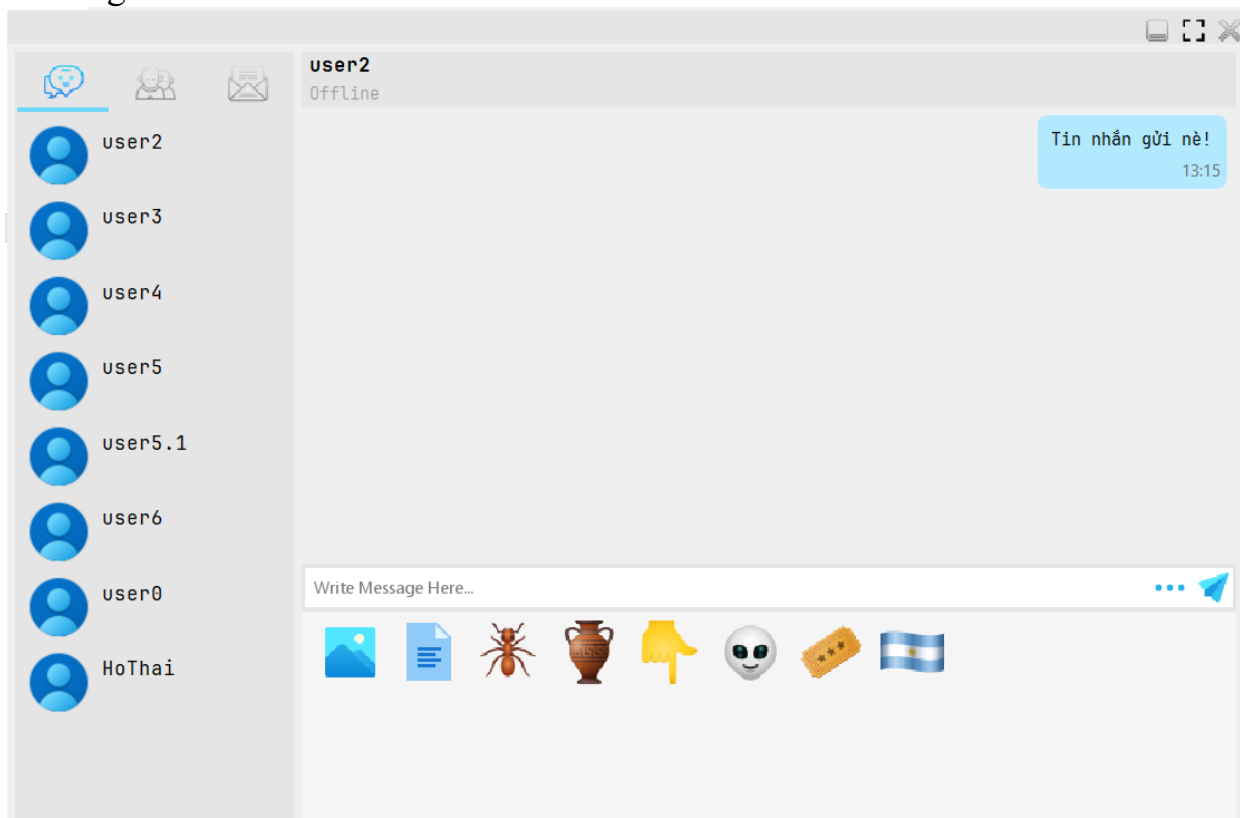


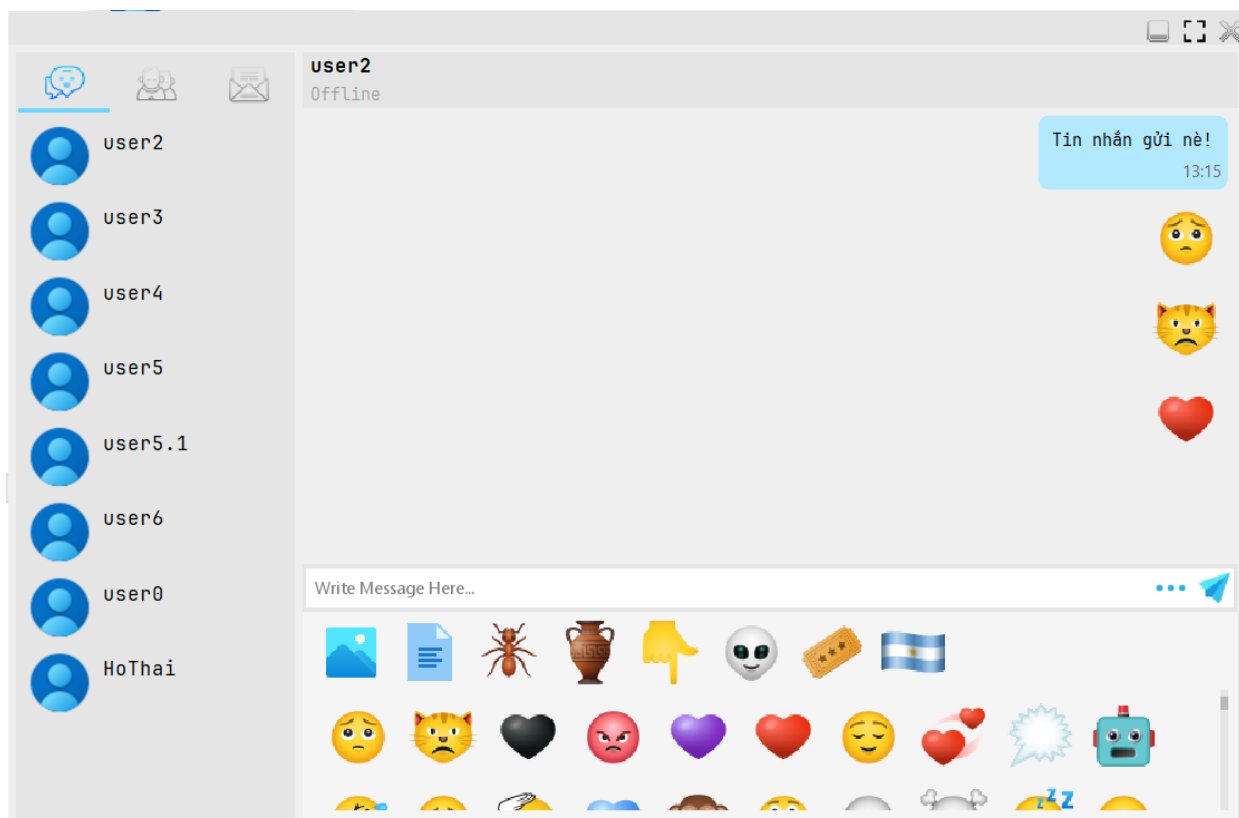
3. Thao tác gửi tin nhắn: Nhấn nút biểu tượng gửi tin nhắn ở góc phải của khung nhập tin nhắn để gửi tin nhắn hoặc nhấn tổ hợp phím “Shift + Enter” để gửi nhanh.



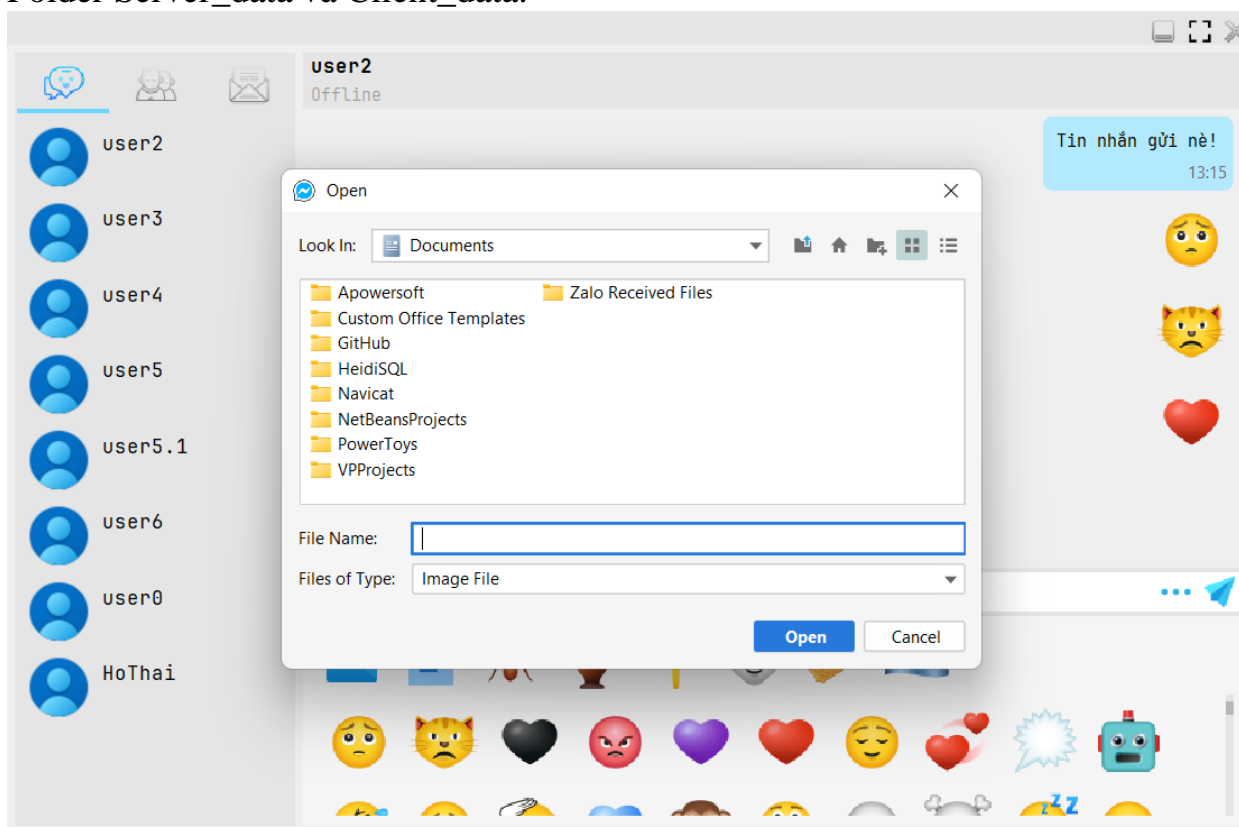
4. Thao tác gửi emoji: Trong phần khung nhập tin nhắn có biểu tượng “...”. Nhấn vào nút này sẽ hiện ra khung mở rộng Panel. Trong phần Menu Button của

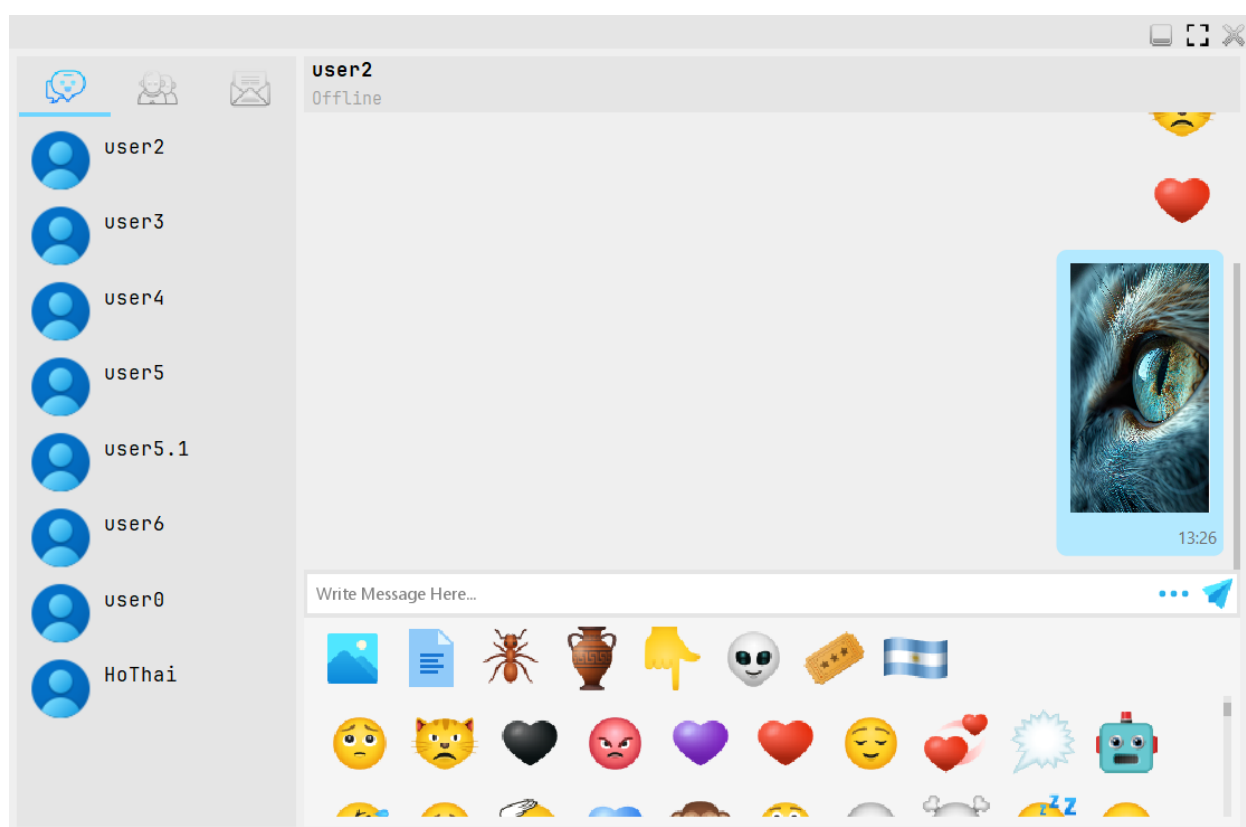
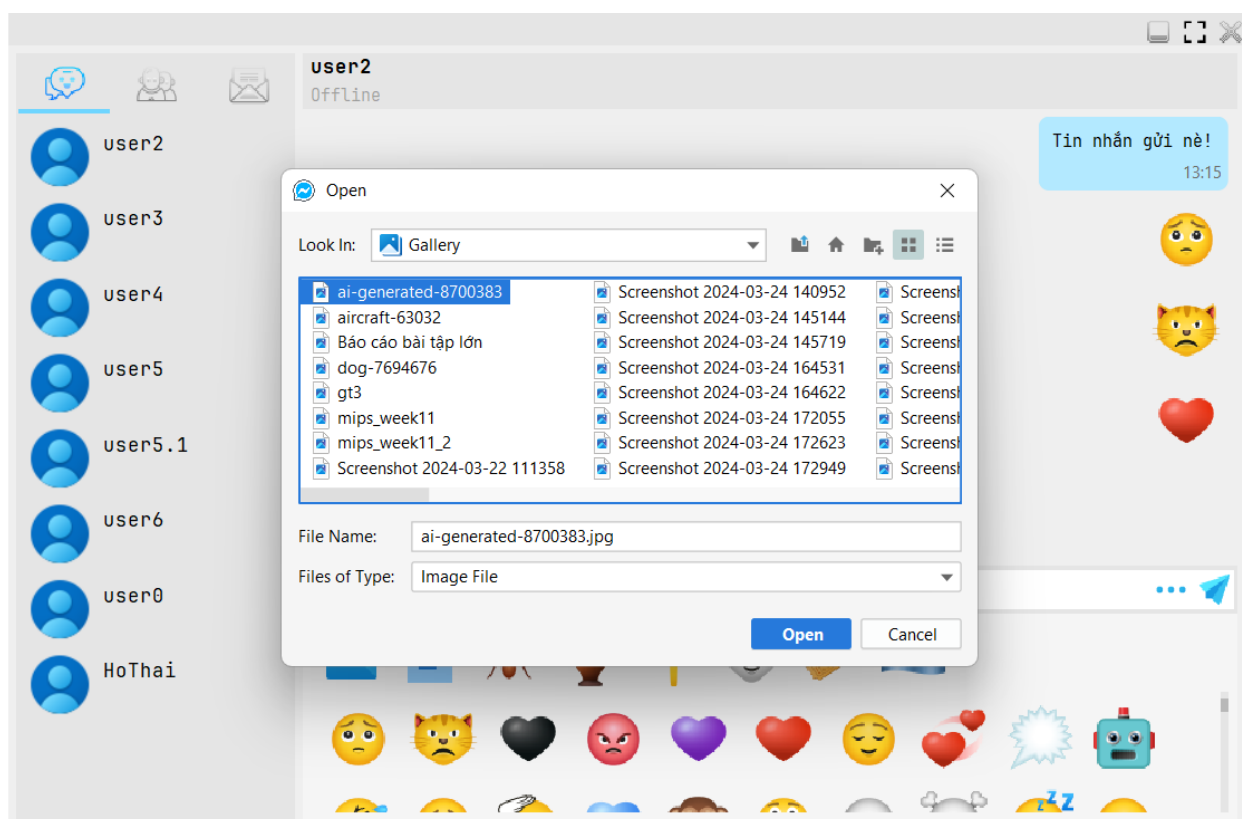
Panel More có các biểu tượng lần lượt là ảnh, file, các emoji. Nhấn chọn vào các biểu tượng emoji để hiển thị thêm nhiều emoji. Nhấn chọn emoji muốn gửi để gửi.

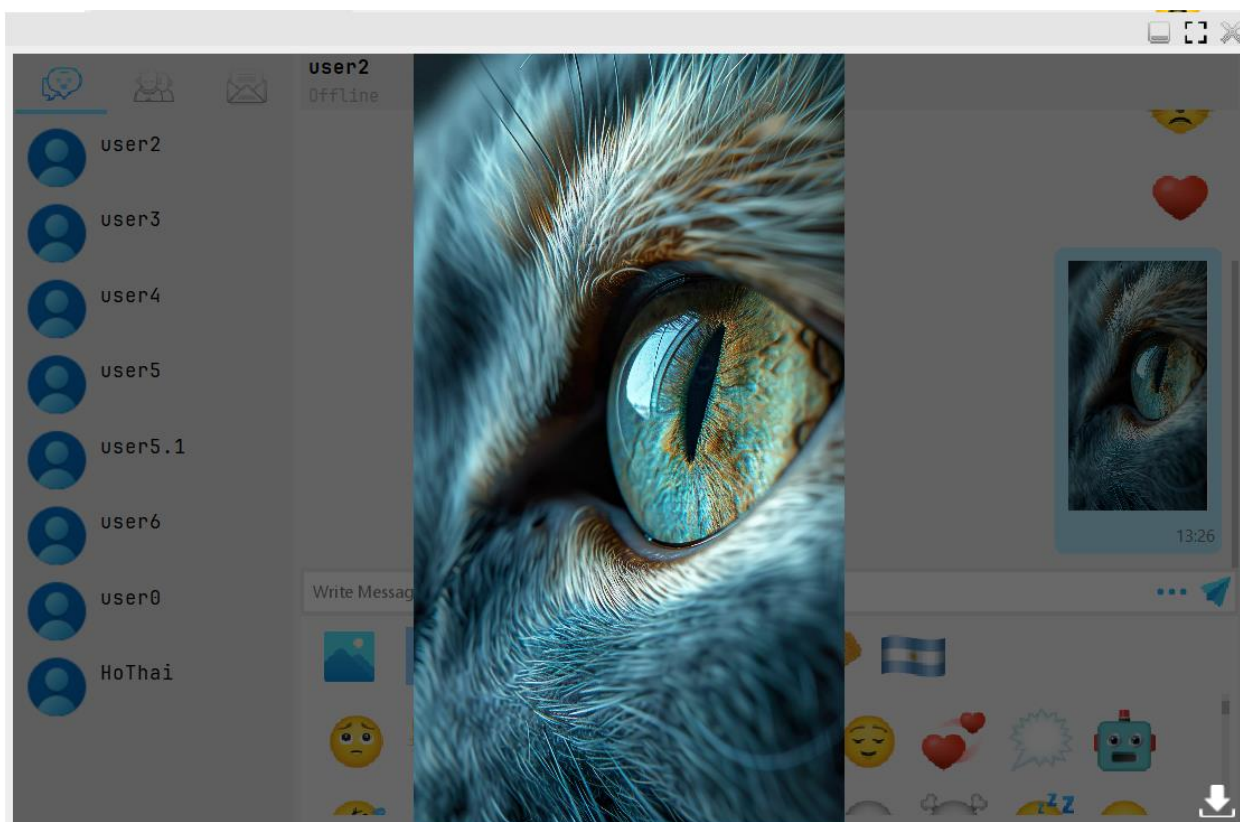




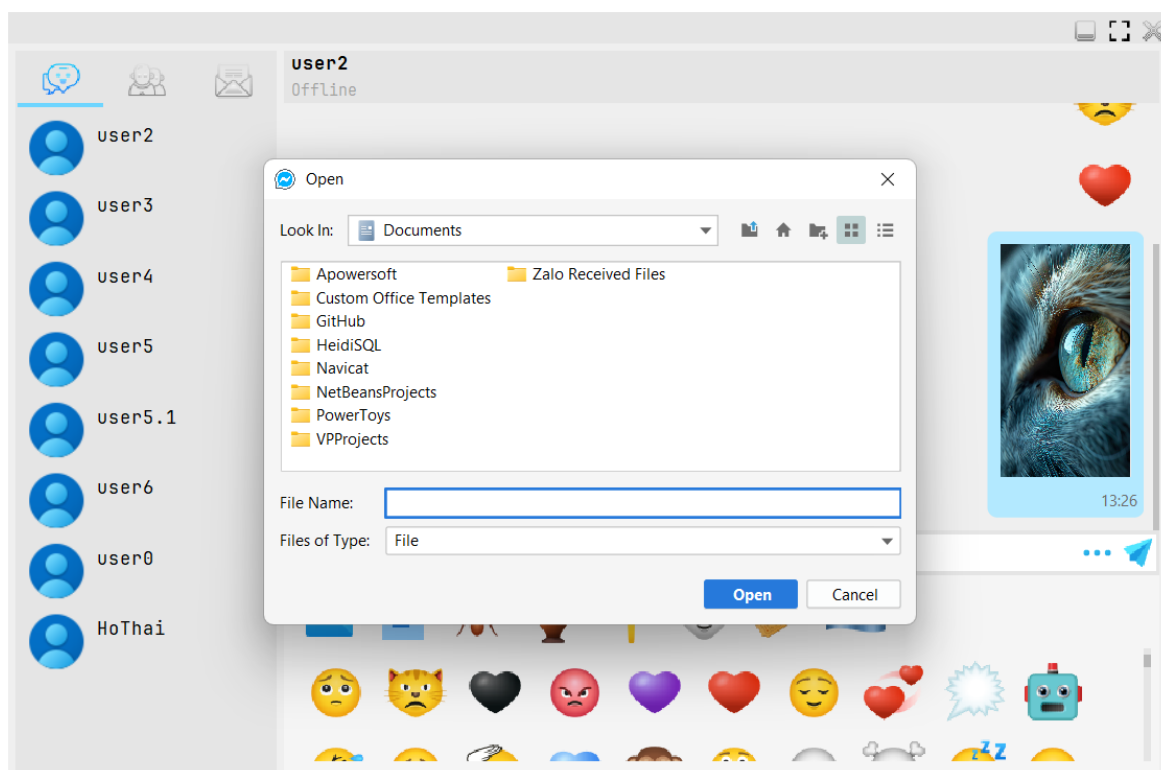
5. Thao tác gửi ảnh: Trong phần Menu Button của Panel More, nhấn vào biểu tượng ảnh để mở cửa sổ chọn file ảnh để gửi. Nhấn vào ảnh để xem ảnh đã gửi, nhấn chuột thêm lần nữa để thoát khỏi chế độ xem ảnh. Ảnh gửi đi sẽ được lưu ở Folder Server_data và Client_data.

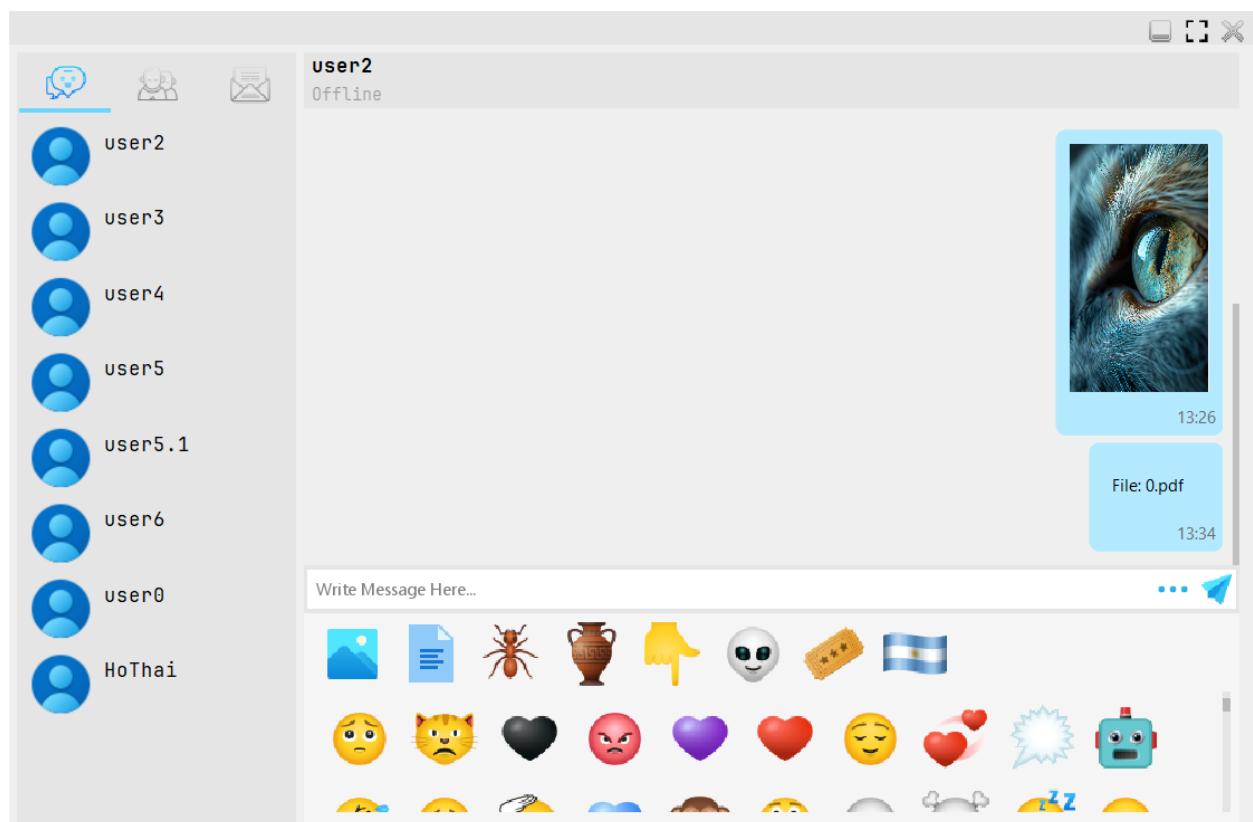
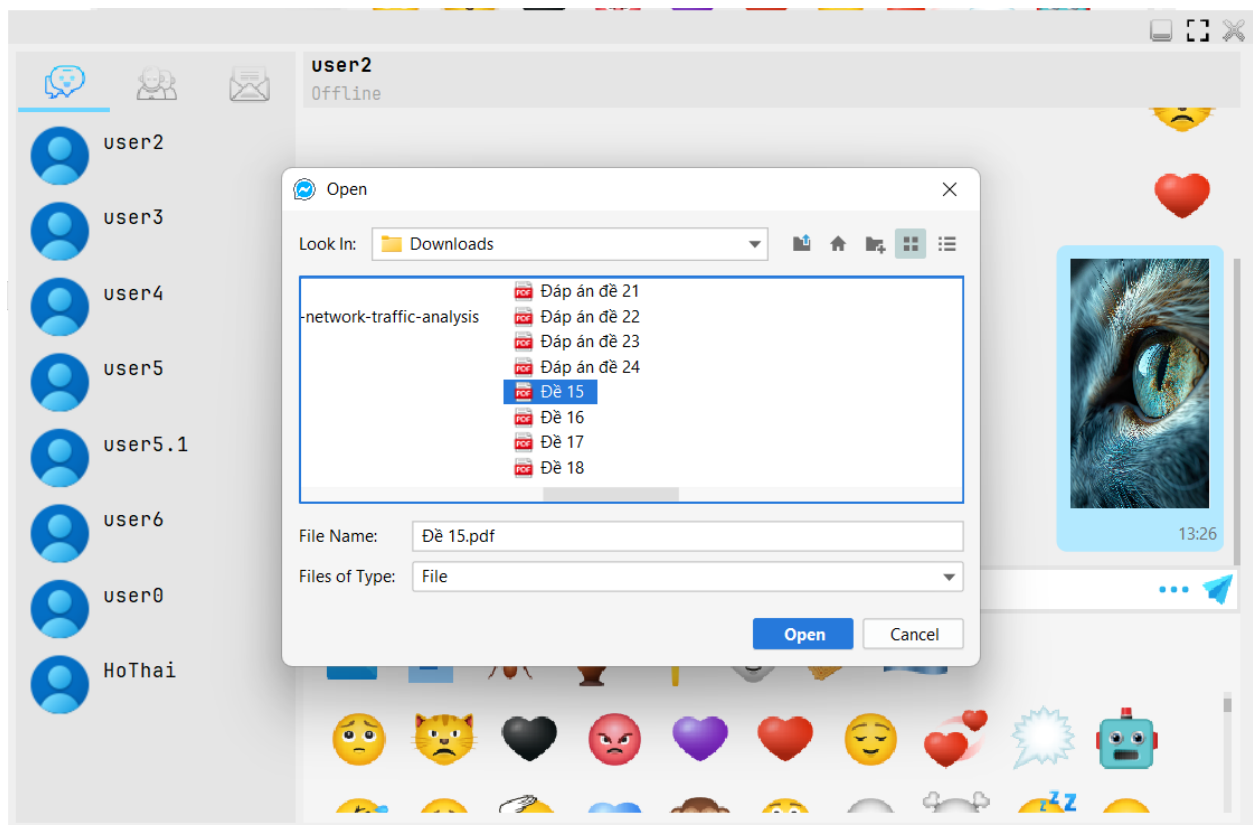




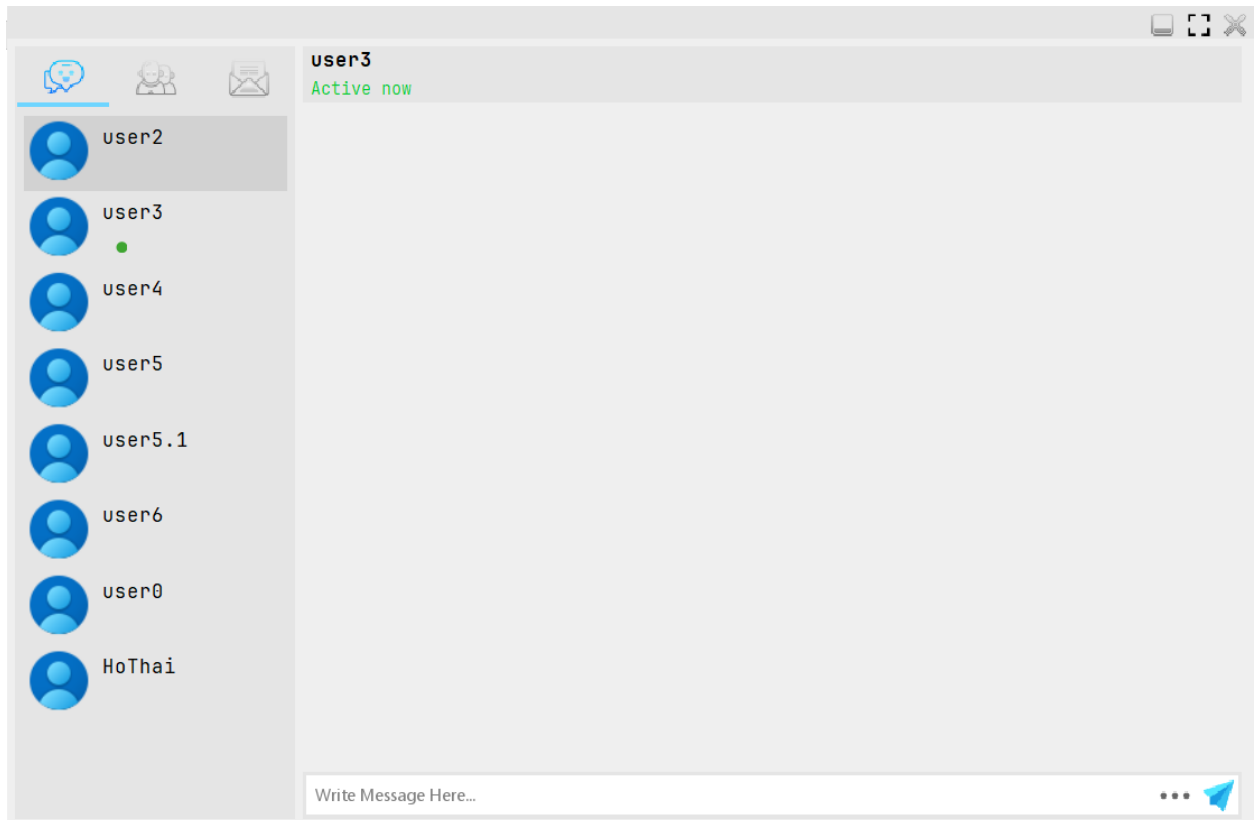


6. Thao tác gửi file: Trong phần Menu Button của Panel More, nhấn vào biểu tượng file để mở cửa sổ chọn file. Trong cửa sổ chọn file, chọn file muốn gửi và nhấn gửi. File gửi và nhận sẽ được lưu lần lượt tại Server_data và Client_data.

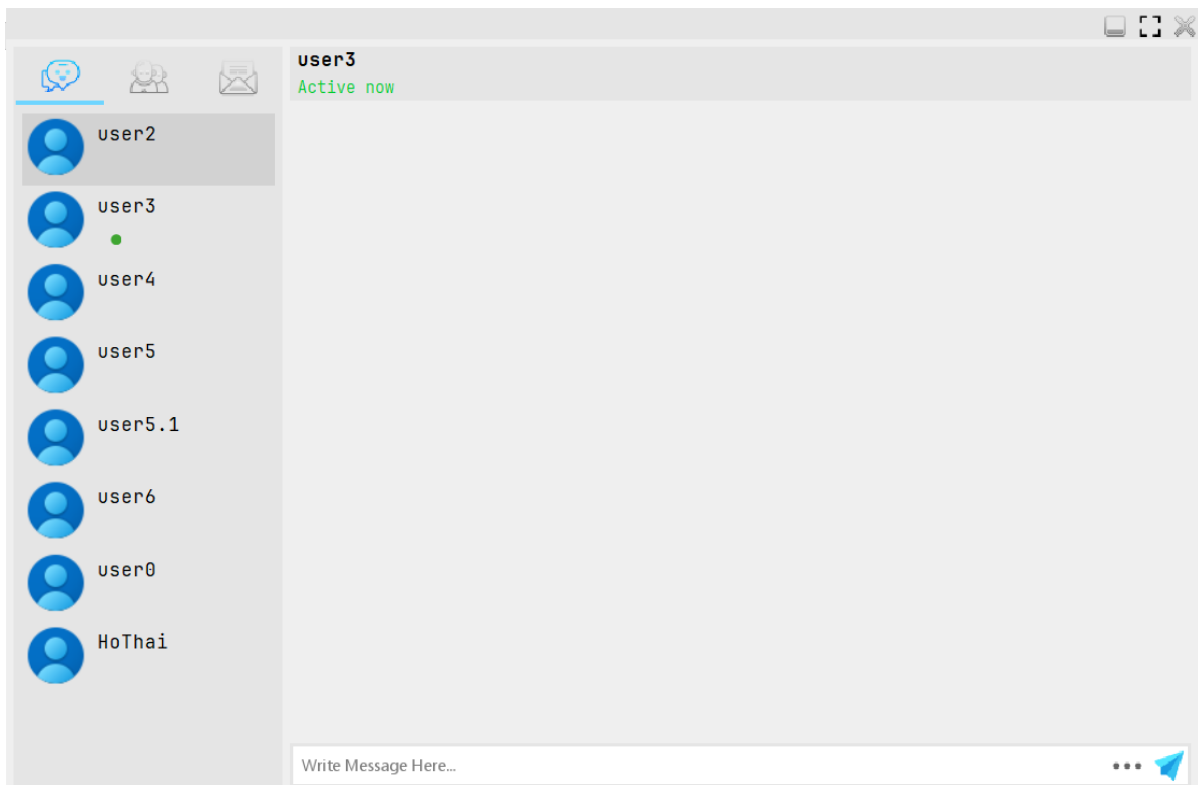




7. Chức năng hiển thị trạng thái hoạt động: Nếu người dùng nào đó hoạt động thì ở phần Chat Title sẽ ghi trạng thái là online. Ở phần Chat Title và Menu Left ứng với người dùng online sẽ hiện nút màu xanh



8. Chức năng reset giao diện Chat khi nhấn chuyển sang người dùng khác: Khi đang nhắn tin với người dùng này mà chọn người dùng khác để nhắn tin thì các tin nhắn với người dùng cũ sẽ được xóa đi.



4.3. Kiểm thử các chức năng đã thực hiện

Chức năng đăng nhập	Hoạt động
Chức năng đăng ký	Hoạt động
Chức năng gửi tin nhắn text	Hoạt động
Chức năng gửi ảnh	Hoạt động
Chức năng gửi file	Hoạt động
Chức năng gửi emoji	Hoạt động
Chức năng hiển thị trạng thái hoạt động	Hoạt động
Chức năng xem ảnh đã gửi	Hoạt động

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Chương trình Message được phát triển dựa trên các kiến thức đã được giảng dạy trên lớp và tự tìm hiểu nên còn sơ sài, tồn tại nhiều thiếu sót và hạn chế, một phần cũng do các thành viên trong nhóm chưa có kinh nghiệm xây dựng và phát triển ứng dụng. Sau quá trình phát triển và thử nghiệm ứng dụng, nhóm chúng em xin đưa ra kết luận và hướng phát triển như sau:

1. Ưu điểm:

- Đáp ứng, hoàn thành đầy đủ các yêu cầu của đề tài, các chức năng cơ bản của 1 ứng dụng nhắn tin (bao gồm đăng ký, đăng nhập, trạng thái hoạt động, gửi tin nhắn text, gửi file, gửi emoji, gửi ảnh).
- Ứng dụng chạy mượt mà và không gặp các lỗi như crash ứng dụng hay lỗi đồ hoạ.

2. Nhược điểm:

- Đồ hoạ chưa thực sự bắt mắt.
- Chưa có chức năng lưu các tin nhắn đã gửi.
- Chưa có chức năng gọi điện.

3. Hướng phát triển:

Với những ưu nhược điểm như trên nhóm có một vài ý tưởng phát triển mới cho ứng dụng đó là:

- Thêm các thao tác cập nhật ảnh, giới tính cho từng user.
- Phát triển thao tác có thể lưu trữ được các tin nhắn vào cơ sở dữ liệu.
- Cải thiện giao diện đồ hoạ thân thiện và dễ dàng tiếp cận hơn, thay đổi từ sử dụng java swing qua javafx.
- Sửa các lỗi sẵn có.

TÀI LIỆU THAM KHẢO

1. Bài giảng học phần Lập trình hướng đối tượng của thầy Ngô Thành Trung.
2. Lập trình hướng đối tượng với Java – Đoàn Văn Ban – NXB Khoa học và kỹ thuật.
3. Các video về lập trình giao diện JavaFX
<https://youtube.com/playlist?list=PL33lvabfss1yRgFCgFXjtYaGAuDJjjH-j&si=DT5k2whxNvvXHPls>.
4. Các video về lập trình mạng:
<https://youtube.com/playlist?list=PLyxSzL3F7485hAzjQl7JrkcoaYmqoPn-5&si=fCLfWwFBg2iDSWXL>
5. Các video về tạo app nhắn tin:
<https://youtube.com/playlist?list=PLyt2v1LVXYS366NRBMaiXYGXv0vI2TiMD&si=dPWd3kSot9lKDC0c>