ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH

MẠNG MÁY TÍNH

Assignment 1

LoveIs... - Ứng dụng chat qua mạng

GVHD:
Thầy Nguyễn Hồng Nam

Nhóm Fight:
1. Nguyễn Văn Đức - 1410953
2. Vũ Thành Công - 1410413
3. Nguyễn Đức Thọ - 1413817

Ngày 5 tháng 11 năm 2016

Mục lục

1 Chức năng của ứng dụng

- Tạo, đăng nhập tài khoản

  Người dùng có thể nhanh chóng tạo cho mình tài khoản để tham gia chat với bạn bè cùng sử dụng ứng dụng.

- Hiển thị danh sách bạn bè hiện có, đang online, offline

  Người dùng dễ dàng quản lý được bạn bè mình đang online để thực hiện chat giữa hai người, chat nhóm,... trong danh sách bạn bè mình đang có.

- Thêm bạn

  Chỉ cần biết tên tài khoản của ai đó, chỉ với vài bước đơn giản, người dùng có thể chat với người ấy dễ dàng.

- Chat giữa hai người

  Chọn tên người bạn trong danh sách bạn bè của bạn và chat ngay lập tức hoặc có thể gửi tin nhắn, file,... cho người đang offline và người đó sẽ nhận được tin nhắn,file,.. đó khi online.

- Chat đồng thời

  Bạn có thể cùng lúc chat với nhiều người qua nhiều cửa sổ khác nhau.

- Chat nhóm

  Bạn và bạn bè của bạn có thể chat chung với nhau qua một cửa sổ và mọi người đều thấy được tin nhắn của người khác.

- Gửi file qua lại khi chat

  Ngoài tin nhắn văn bản thông thường thì bạn có thể gửi và nhận file bất kì thông qua ứng dụng này. Bạn có thể vừa chờ upload file vừa tiếp tục chat với bạn bè.

2 Thiết kế ứng dụng

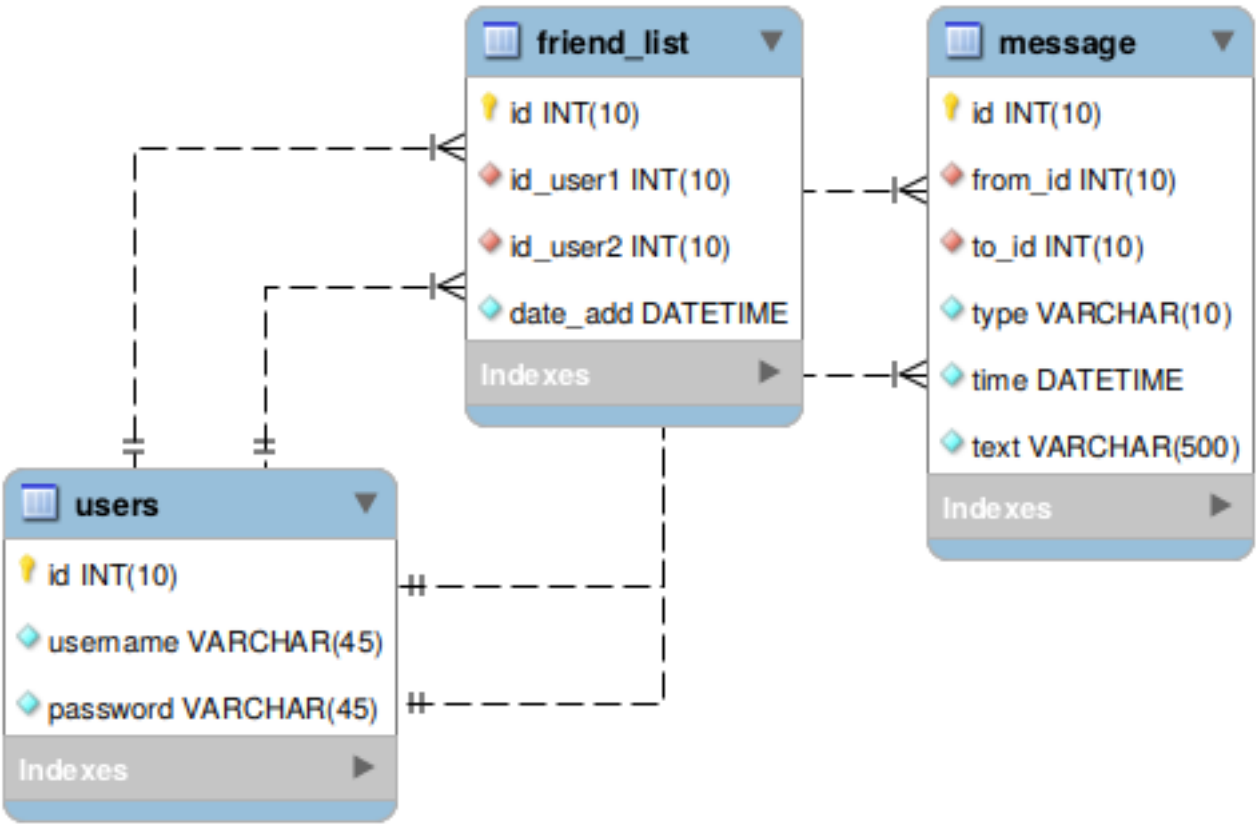
2.1 Mô hình tương tác

Mô hình Client-Server, sử dụng giao thức WebSocket.

- Nhóm chọn mô hình Client-Server. Server có thể quản lý, phân luồng dữ liệu trao đổi giữa các Client. Ngoài ra, Server còn kiểm tra, đảm bảo tính an toàn, toàn vẹn của dữ liệu khi nhận và khi gửi, thực hiện việc lưu dữ liệu vào database. Nhờ vậy, ta có thể quản lý, back up được dữ liệu, đồng thời đảm bảo khả năng dữ liệu được gửi đến Client nhận cao nhất.
- WebSocket là công nghệ hỗ trợ giao tiếp hai chiều giữa Client và Server bằng cách sử dụng một TCP socket để tạo một kết nối hiệu quả, ít tốn kém, có độ trễ thấp và dễ xử lý lỗi. Hỗ trợ giao tiếp hai chiều tức là Client và Server luôn nhận dữ liệu mới theo thời gian thực (2 chiều: server đến client hoặc client đến server).

2.2 Database

Nhóm sử dụng hệ quản trị cơ sở dữ liệu MySQL. MySQL dễ sử dụng, hiệu suất cao, chi phis thấp. Ngoài ra MySQL còn hỗ trợ JSON, một định dạng hoán vị dữ liệu nhanh, trao đổi dữ liệu độc lập, với các built-in function cho phép lưu trữ, tìm kiếm.



2.3 Protocol

Nhóm sử dụng JSON để định dạng dữ liệu gửi đi và nhận về từ client.
Trong các message gửi đi hay nhận về, đều có một trường *type* nhằm xác định loại message để dễ dàng phân loại và xử lý.

userRegReq	Client gửi Request yêu cầu tạo một user account mới
userLoginReq	Client gửi Request yêu cầu kiểm tra thông tin và đăng nhập
userRes	Server trả về Response trả lời cho Request yêu cầu đăng ký hoặc đăng nhập
text	Client/Server gửi text message cho nhau
file	Truyền và nhận file
userUpdate	Server gửi các thông tin về bạn bè của user mỗi khi có thay đổi, realtime
error	Server gửi lỗi về Client

2.3.1 Tạo kết nối

Client phải gửi một WebSocke handshake request đến Server. Server sẽ gửi trả lại WebSocket handshake response. Client sẽ kiểm tra và thực hiện kết nối.

2.3.2 Đăng nhập/đăng kí

Đối với user mới, cần phải đăng ký để sử dụng được ứng dụng. Khi đó với thông tin user nhập vào, Client gửi một message lên Server có dạng như sau:

```
{
        "type": "userRegReq",
        "username": "newuser",
        "password": "newuser"
        "fullname":"New User"
}
```

Đối với user đã đăng ký rồi, chỉ cần nhập username và password để thực hiện đăng nhập.

```
{
        "type": "userLoginReq",
        "username": "ten_dang_nhap",
        "password": "mat_khau"
}
```

Về phía Server khi nhận được dạng message trên, Server chỉ trả về một loại message sau nếu thành công. Nếu có lỗi, Server gửi message dạng *error* về.

```
{
        "type": "userRes",
        "friendList":
        [
                {
                        "friendname":"vtcong",
                        "online": true
                },
                {
                        "friendname":"nvduc",
                        "online": false
                }
        ]
}
```

2.3.3 Chat giữa hai user

Client gửi message sau

```
{
        "type": "text",
        "messType": "peer",
        "time": "yyyy-MM-dd HH:mm:ss",
        "toFriend": "userNhan",
        "text": "noi_dung_chat"
}
```

Server nhận message trên, thực hiện việc lưu vào database đồng thời gửi về cho user nhận.

```
{
        "type": "text",
        "time": "",
        "messType": ""
        "fromFriend": "userGui",
        "text": "noi_dung_chat"
}
```

2.3.4 Truyền và nhận file

Trước khi Client truyền file lên Server cần gửi một message để Server biết và chuẩn bị nhận file.

```
{
        "type": "file",
        "fileName": "fight.txt",
        "fileSize": 1024,
        "toFriend": "userNhan"
}
```

Nhận được message trên, Server gửi message sau cho user nhận

```
{
        "type": "file",
        "fileName": "fight.txt",
        "fileSize": 1024,
        "fromFriend": "userGui"
}
```

nếu user nhận đồng ý nhận file, Client gửi lên Server message sau

```
{
        "type": "file",
        "fileName": "fight.txt",
        "fileSize": 1024,
        "status": "ok"
}
```

nếu không đồng ý, trường *status* có giá trị là "*no*". Server gửi lại message trên về user gửi để đồng ý tiếp tục hoặc hủy việc truyền file. Việc truyền file được thực hiện bằng cách gửi từng byte dữ liệu của file.

2.3.5 Update thông tin về bạn bè của user

Server phát hiện các thay đổi về bạn bè của user(offline, online, thay đổi status,...), gửi về user để cập nhật realtime.

```
{
        "type": "userUpdate",
        "friendname":"",
        "status": "",
        "online": true/false
}
```

2.3.6 Thông báo lỗi

Khi phát hiện lỗi Server trả một message thông báo lỗi cho người dùng.

```
{
        "type": "error",
        "errorCode": 10,
        "errorDescription": "Username/password is incorrect"
}
```

Một số mã lỗi:

10	Registration failed.
11	The username or password for Love Is is incorrect.
12	This account has already been confirmed.

3 Triển khai ứng dụng

3.1 Phía server

Server được hiện thực theo ý tưởng của mô hình 3 lớp, có thay đổi để phù hợp với ứng dụng.

- Presentation Layer: gồm class LoveIsServer, thực hiện việc giao tiếp với client, gửi và nhận các message, kiểm tra message do client gửi lên.
- Business Logic Layer: gồm class MessageDecoder, MessageEncoder, thực hiện việc giải mã, mã hóa dữ liệu theo đúng protocol đã định sẵn, lấy các dữ liệu cần thiết từ database, và gửi lên cho Presentation Layer.
- Data Access Layer: gồm class DataAccessHelper, thực hiện việc giao tiếp với database, lưu trữ, truy xuất dữ liệu (đọc, lưu, cập nhật cơ sở dữ liệu).

3.2 Chi tiết hiện thực

LoveIsServer là phần hiện thực cho giao thức WebSocket. Mỗi khi message được gửi lên Server, class MessageDecoder đưa message thành các đối tượng Message. Và ngược lại, khi Server gửi message về Client, class MessageEncoder chuyển các đối tượng Message cho phù hợp với protocol.

Các đối tượng Message đóng vai trò là Data Access Object trong mô hình 3 lớp.

Về việc lưu lại các user đang đăng nhập, Server có một class User riêng. Mỗi đối tượng User được tạo, tương ứng với một user đang kết nối, và đăng nhập thành công trên Server. Như vậy các đối tượng User hoàn toàn độc lập với nhau. Đối tượng User chứa các behavior để xử lý từng trường hợp message do chính user đó gửi lên cho Server xử lý, ví dụ: *handleLoginReq*, *onlineState*,...

3.3 Phía client

- Wsclient: Là class chứa các hoạt động kết nối Server
- Login: Là class điều khiển cửa sổ login
- Chat: Là class điều khiển cửa sổ chat
- Signup: Là class điều khiển cửa sổ signup

3.4 Các class quan trọng

- LoveIsServer
  Class đóng vai trò mở là Server cho giao thức Websocket, chứa các method có nhiệm vụ nhận mở kết nối (OnOpen), ngắt kết nối (OnClose), nhận message (OnMessage).
- DataAccessHelper
  Class đóng vai trò giao tiếp với database, thực hiện việc lưu, cập nhật dữ liệu.
- User
  Class đóng vai trò lưu lại tất cả các user connect thành công với Server, đồng thời quản lý, xử lý khi nhận message đối với từng user.

4 Đánh giá kết quả

4.1 Kết quả đạt được

Hoàn thành tất cả chức năng cơ bản mà thầy yêu cầu như: hai người ở hai máy khác nhau khi sử dụng ứng dụng thì sẽ chat được với nhau, một người đồng thời có thể chat với nhiều người

khác, cho phép truyền tải file trong quá trình chat, sử dụng protocol mà nhóm đã định nghĩa,...

Giao diện ứng dụng đơn giản, dễ sử dụng.

Tốc độ truyền dữ liệu giữa các máy khi chat khá nhanh.

4.2 Kết quả chưa được

Chưa hiện thực được thêm tính năng nào đặc biệt.

Ứng dụng chưa chạy được trên mobile, mục tiêu mà nhóm hướng tới ban đầu.

ứng dụng chưa chat được với nhau nếu sử dụng mạng khác nhau.

4.3 Khó khăn khi hiện thực ứng dụng

Vì nhóm muốn thử các công nghệ mới nên chọn websocket. Cách hiện thực ứng dụng bằng websocket khá khác so với socket đã học nên nhóm cần khá nhiều thời gian để làm quen.

Tài liệu tham khảo chủ yếu là tiếng anh nên nhiều khi các thành viên khi thực hiện bị hiểu sai cơ chế dẫn đến không đồng bộ cách làm với nhau hoặc phải làm nhiều lần theo nhiều cách hiểu.

Ứng dụng được viết trên nền tảng Java, nền tảng nhóm mới tiếp xúc lần đầu nên khó khăn trong việc xử lý lỗi.

4.4 Hướng dẫn sử dụng ứng dụng

- Trong cửa sổ login, người dùng có thể đăng nhập hệ thống thông qua tài khoản gồm user và password được điền vào khung Text user và password sau đó nhấn login.
- Nếu lần đầu sử dụng hoặc không có tài khoản thì nhấn Sign Up để đăng kí tài khoản mới theo hướng dẫn trong ứng dụng.
- Sau khi đăng nhập, khung listfriend sẽ được hiện ra, muốn chat với người nào thì nháy đúp vào tên người đó, cửa sổ chat sẽ được hiện ngay sau đó.
- Cửa sổ chat có một Text field để bạn điền nội dung chat sau đó nhấn enter và người nhận sẽ nhận được nội dung chat đó. Ta sẽ thấy thông tin các cuộc hội thoại được hiện lên khung hiển thị. Khi nhận một tin nhắn cũng sẽ được hiển thị lên khung này.

5 Tài liệu tham khảo

- Vì sao nên chọn MySQL
- Hiểu hơn về WebSocket
- JSR 356, Java API for WebSocket, by Johan Vos
- Creating Binary WebSocket Connections with Java EE 7 and JavaFX
- Connecting to MySQL Using the JDBC DriverManager Interface
- The WebSocket Protocol