**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**



**Báo cáo lập trình mạng**

***Đề tài: Xây dựng chương trình quản lí danh mục địa chỉ***

Sinh viên thực hiện:

Đỗ Kim Trà 20183999

Nguyễn Anh Tú 20184000

Giáo viên hướng dẫn: ThS. Bùi Trọng Tùng

**Hà Nội – 8/2021**

**Mục lục**

[Chương I: Tổng quan đề tài 3](#_Toc81525719)

[1. Phát biểu bài toán 3](#_Toc81525720)

[2. Các chức năng của chương trình 3](#_Toc81525721)

[3. Kĩ thuật sử dựng 3](#_Toc81525722)

[4. Phân chia công việc 4](#_Toc81525723)

[Chương II: Khuôn dạng thông điệp 4](#_Toc81525724)

[A. Phía client 4](#_Toc81525725)

[1. Cấu trúc chung 4](#_Toc81525726)

[2. Các bản tin request 4](#_Toc81525727)

[B. Phía server 5](#_Toc81525728)

[1. Cấu trúc chung 5](#_Toc81525729)

[2. Error code 5](#_Toc81525730)

[Chương III: Tổ chức lưu trữ dữ liệu 6](#_Toc81525731)

[Chương IV: Kịch bản hoạt động 7](#_Toc81525732)

[1. Đăng kí 7](#_Toc81525733)

[2. Đăng nhập 7](#_Toc81525734)

[3. Thêm địa chỉ 8](#_Toc81525735)

[4. Xóa địa chỉ 8](#_Toc81525736)

[5. Xem địa chỉ 8](#_Toc81525737)

[6. Chia sẻ địa chỉ 9](#_Toc81525738)

[7. Sao lưu danh mục 9](#_Toc81525739)

[8. Khôi phục danh mục 10](#_Toc81525740)

[9. Đăng xuất 10](#_Toc81525741)

[Chương V: Tổ chức chương trình 11](#_Toc81525742)

[A. Phía server 11](#_Toc81525743)

[1. Hoạt động chung 11](#_Toc81525744)

[2. Bảng liệt kê cấu trúc dữ liệu và hàm mô-đun 11](#_Toc81525745)

[2.1. Các cấu trúc dữ liệu chính 12](#_Toc81525746)

[2.2. Các hàm trong file data\_type.h 13](#_Toc81525747)

[2.3. Các hàm trong file response.h 13](#_Toc81525748)

[2.4. Các hàm trong file unbox\_mess.h 17](#_Toc81525749)

[2.5. Các hàm trong file io\_file.h 17](#_Toc81525750)

[B. Phía client 18](#_Toc81525751)

[1. Hoạt động chung 18](#_Toc81525752)

[2. Bảng liệt kê các hàm mô đun 18](#_Toc81525753)

[2.1. Các hàm trong file message.h 19](#_Toc81525754)

[2.2. Các hàm xử lí trong file response.h 21](#_Toc81525755)

# Chương I: Tổng quan đề tài

## 1. Phát biểu bài toán

**Đề tài**

Đề tài 06: Xây dựng chương trình quản lý danh mục địa điểm.

**Các chức năng cơ bản**

+ Cho phép người dùng lưu các địa điểm yêu thích. Gợi ý các loại danh mục: Cửa hàng ăn uống, Cửa hàng cà phê, Rạp phim, Cửa hàng thời trang…

+ Cho phép người dùng sao lưu danh mục trên server và phục hồi khi cần

+ Cho phép người dùng chia sẻ địa điểm với bạn bè: Người dùng tag tên bạn bè vào địa điểm cần chia sẻ. Mỗi khi người dùng đăng nhập, server gửi thông báo có địa điểm mới do bạn bè chia sẻ để người dùng lưu vào danh mục cá nhân. Lưu ý, khi hiển thị danh mục cần có dấu hiệu phân biệt địa điểm nào là do bạn bè chia sẻ.

## 2. Các chức năng của chương trình

* Đăng kí
* Đăng nhập (hiển thị địa chỉ được chia sẻ)
* Thêm địa chỉ (từ bạn bè và thêm mới)
* Xóa địa chỉ
* Hiển thị địa chỉ
* Chia sẻ địa chỉ (từ địa chỉ cá nhân)
* Sao lưu một danh mục
* Khôi phục danh mục
* Đăng xuất (xóa địa chỉ bạn bè đã hiển thị)

## 3. Kĩ thuật sử dựng

Chương trình sử dụng giao diện dòng lệnh để thao tác với người dùng.

Kĩ thuật socket sử dựng overlaped I/O completion port.

Lưu trữ dữ liệu sử dụng thao tác với file.

## 4. Phân chia công việc

|  |  |  |
| --- | --- | --- |
| Thành viên | Đỗ Kim Trà - 20183999 | Nguyễn Anh Tú - 20184000 |
| Chức năng triển khai | * Đăng kí * Đăng nhập * Thêm địa chỉ * Xóa địa chỉ * Đăng xuất | * Hiển thị địa chỉ * Chia sẻ địa chỉ * Sao lưu một danh mục * Khôi phục danh mục |

# Chương II: Khuôn dạng thông điệp

## A. Phía client

### 1. Cấu trúc chung

|  |  |
| --- | --- |
| **Length** | **request** |

**Length** là **chiều dài cả bản tin** với kích thước **2 bytes**

**Request** có cấu trúc **request\_type param \n param**

**Request\_type** có độ dài **2 bytes** chứa mã yêu cầu, đằng sau là danh sách các tham số được phân cách bởi kí tự **‘\n’** và cách **request\_type** bởi dấu cách.

### 2. Các bản tin request

* Đăng kí RG username \n password
* Đăng nhập LI username \n password
* Lấy gợi ý SG
* Lấy danh mục GC
* Lấy địa chỉ GA category
* Thêm địa chỉ AA category \n [address or fa\_number]
* Xóa địa chỉ RA category \n address\_number
* Chia sẻ địa chỉ SF username \n category \n address\_number
* Sao lưu BK category
* Lấy tên sao lưu GB
* Phục hồi RS category
* Đăng xuất LO

## B. Phía server

### 1. Cấu trúc chung

|  |  |  |
| --- | --- | --- |
| **Opcode (1 byte)** | **Length (2 bytes)** | **payload** |

* Nếu opcode = 0, length = 0 thông báo thành công
* Nếu opcode = -1, length = 1 payload chứa mã lỗi
* Nếu opcode > 0 truyền dữ liệu
* 1 - danh mục gợi ý suggest \n
* 2 - địa điểm bạn bè username category \0x5 address \n
* 3 - danh mục cá nhân category \n
* 4 - danh sách địa chỉ +address \n
* 5 - danh mục backup category \n
* Bản tin có length = 0 kết thúc

### 2. Error code

0 🡪 bản tin không xác định: lỗi mã yêu cầu, lỗi tham số

1 🡪 username không hợp lệ (đăng kí)

2 🡪 password không hợp lệ (đăng kí)

3 🡪 tên đăng nhập hoặc mật khẩu sai (đăng nhập)

4 🡪 tài khoản đã đăng nhập

5 🡪 chưa đăng nhập

6 🡪 chưa đăng xuất (đăng nhập)

11 🡪 category không hợp lệ

21 🡪 address number không hợp lệ

# Chương III: Tổ chức lưu trữ dữ liệu

Qua quá trình tìm hiểu các phương pháp lưu trữ khác nhau như sqlite, xml, file. Em nhân thấy phương pháp lưu trữ tổ chức file dễ dàng thao tác và phù hợp với các chức năng của chương trình. Vì vậy nhóm em đã lựa chọn giải pháp tổ chức file để lưu trữ và quản lí dữ liệu cho chương trình.

***Kiểu lưu trữ sử dụng dạng tổ chức file***

***root***

├───server.exe

├───account.txt

├───suggest.txt

└───users

├───username1

│ ├───fa.txt

│ ├───category1.txt

│ ├───category2.txt

│ ├─── \*\*\*

│ └───back\_up

│ ├───categary1.txt

│ └─── \*\*\*

├───username2

└─── \*\*\*

***Tổ chức thư mục lưu trữ dữ liệu của chương trình***

Ở đây chương trình sử dụng đường đẫn tương đối với thư mục gốc (root) là nơi chứa file thực thi của chương trình và đã được khởi tạo trước file ‘account.txt’ và foulder ‘users’ trước để chương trình không gặp lỗi khi thực thi.

Đối với mỗi tài khoản người dùng khi khởi tạo thì sẽ được cấp phát một foulder có tên trùng với tên đăng nhập để lưu trữ thông tin cá nhân. Trong thư mục cá nhân này chứa file “fa.txt” để chứa thông tin địa chỉ bạn bè chia sẻ, và một foulder ‘back\_up’ để chứa các file danh mục sao lưu được copy vào đây.

Các file có tên dạng ‘category.txt’ được tạo ra khi người dùng thêm địa chỉ vào danh mục mới chưa tồn tại trước đó. Trong file này mỗi dòng là một địa chỉ và có prefix ‘+’ để chỉ địa chỉ được chia sẻ bởi bạn bè và ‘-’ để chỉ địa chỉ cá nhân.

# Chương IV: Kịch bản hoạt động

Kịch bản tập trung vào mô tả trình tự và hoạt động của các bên tham gia để hoàn thành một tác vụ. Nếu trong quá trình thực hiện mà gặp lỗi tại bước nào thì tác vụ sẽ bị hủy.

### 1. Đăng kí

User: nhập username và password

Client: gửi bản tin đăng kí

Server: lấy thông tin tài khoản

Server: kiểm tra username, password có hợp lệ

Server: tạo thư mục với đường dẫn “users/username”

Server: tạo file với đường dẫn “users/*username*/fa.txt”

Server: tạo thư mục với đường đẫn “users/username/back\_up”

Server: lưu thông tin tài khoản vào file “account.txt”

Server: Thêm thông tin tài khoản vào biến toàn cục

Server: thông báo thành công đến client

### 2. Đăng nhập

User: nhập thông tin username, password

Client: gửi yêu cầu đăng nhập

Server: lấy thông tin tài khoản

Server: kiểm tra username, password có hợp lệ

Server: kiểm tra tài khoản có tồn tại

Server: đánh dấu tài khoản đã đăng nhập

Server: gửi friend address đánh dấu fa\_bytes để xóa khi thoát

Client: hiển thị danh sách friend address

### 3. Thêm địa chỉ

Client: lấy danh sách gợi ý và danh mục từ server và hiển thị

User: nhập thông tin danh mục

User: nhập địa chỉ hoặc số của địa chỉ bạn bè chia sẻ

Client: gửi bản tin thêm địa chỉ

Server: lấy thông tin về danh mục và địa chỉ

Server: kiểm tra địa chỉ nếu là số thì sẽ lấy địa chỉ trong file “fa.txt”

Server: thêm địa chỉ vào danh mục

Server: báo thành công

### 4. Xóa địa chỉ

Client: lấy danh sách danh mục từ server và hiển thị

User: nhập tên danh mục

Client: lấy thông tin địa chỉ với tên danh mục trên từ server và hiển thị

User: nhập số của địa chỉ cần xóa

Client: gửi bản tin xóa địa chỉ

Server: lấy thông tin danh mục và số của địa chỉ cần xóa

Server: xóa dòng trong file danh mục

Server: báo thành công

### 5. Xem địa chỉ

***Hiển thị danh sách địa chỉ trong một danh mục***

Client: gửi truy vấn các danh mục của người dùng

Server: kiểm tra người dùng đã đăng nhập hay chưa

Server: trả về client các danh mục

User: nhập danh mục

Client: gửi truy vấn về danh mục người dùng đã nhập

Server: kiểm tra người dùng đã đăng nhập hay chưa

Server: kiểm tra tên danh mục tồn tại hay không

Server: trả về client các địa điểm trong danh mục

### 6. Chia sẻ địa chỉ

Client: gửi truy vấn các danh mục của người dùng

Server: kiểm tra người dùng đã đăng nhập hay chưa

Server: trả về client các danh mục

User: nhập danh mục

Client: gửi truy vấn về danh mục người dùng đã nhập

Server: kiểm tra người dùng đã đăng nhập hay chưa

Server: kiểm tra tên danh mục tồn tại hay không

Server: trả về client các địa điểm trong danh mục

User: nhập tên người dùng của bạn bè

Client: gửi yêu cầu server chia sẻ địa điểm với bạn bè

Server: kiểm tra người dùng đã đăng nhập hay chưa

Server: kiểm tra tên danh mục tồn tại hay không

Server: lấy địa chỉ người dùng muốn chia sẻ

Server: kiểm tra username bạn bè tồn tại hay không

Server: thêm địa chỉ được chia sẻ vào file “*users/friend\_username/fa.txt*”

Server: thông báo kết quả tới client

### 7. Sao lưu danh mục

Client: gửi truy vấn các danh mục của người dùng

Server: kiểm tra người dùng đã đăng nhập hay chưa

Server: trả về client các danh mục

User: nhập danh mục

Client: gửi truy vấn về danh mục người dùng đã nhập

Server: kiểm tra người dùng đã đăng nhập hay chưa

Server: kiểm tra tên danh mục tồn tại hay không

Server: sao chép file *“users/username/category\_name.txt”* dán vào thư mục *“users/username/back\_up”*

Server: thông báo kết quả tới client

### 8. Khôi phục danh mục

Client: gửi truy vấn các danh mục của người dùng

Server: kiểm tra người dùng đã đăng nhập hay chưa

Server: trả về client các danh mục

User: nhập danh mục

Client: gửi truy vấn về danh mục người dùng đã nhập

Server: kiểm tra người dùng đã đăng nhập hay chưa

Server: kiểm tra tên danh mục tồn tại hay không

Server: sao chép file *“users/username/back\_up/category\_name.txt”* dán vào thư mục *“users/username”*

Server: thông báo kết quả tới client

### 9. Đăng xuất

Client: gửi bản tin đăng xuất

Server: kiểm tra đăng nhập của client

Server: xóa phần đầu của gói tin “*fa.txt*” với số bytes là **fa\_bytes**

Server: thông báo thành công

# Chương V: Tổ chức chương trình

## A. Phía server

### 1. Hoạt động chung

1. Thiết lập listen socket để chờ và tiếp nhận các kết nối từ client
2. Nhận các yêu cầu từ client
3. Gọi hàm ***getResponses***() để xử lí các bản tin và lưu các bản tin phản hồi vào biến responses trong cấu trúc ***ClientInfo***
4. Gửi lại các bản tin phản hồi cho client
5. Quay lại bước 2

### 2. Bảng liệt kê cấu trúc dữ liệu và hàm mô-đun

|  |  |  |  |
| --- | --- | --- | --- |
| data\_type.h | response.h | unbox\_mess.h | io\_file.h |
| + user  + users  + ClientInfo  + SocketInfo |  |  |  |
| + initSockInfo()  + removeSockInfo()  + accountLocation()  + checkNum()  + toWchar\_t() | + packageHeader()  + getNoticeMess()  + getRG\_response()  + getLI\_response()  + getGC\_response()  + getSG\_response()  + getAA\_response()  + getRA\_response()  + getGA\_response()  + getSF\_response()  + getGB\_response()  + getBK\_response()  + getRS\_response()  + getLO\_response()  + getResponses() | + getLength()  + getMes\_type()  + getAccount()  + getAddress()  + getSharedAddress() | + prepareUserList()  + makeDirectory()  + createFile()  + getFileNames()  + getFileSize()  + readFileBlock()  + deleteFile()  + appendFile()  + deleteFileLine()  + findFileLine()  + getFileLine()  + copyFile() |

### 2.1. Các cấu trúc dữ liệu chính

Các cấu trúc dữ liệu sử dụng trong chương trình được khai báo trong file data\_type.h bao gồm:

**struct user**{

string username;

string password;

bool islogin; //kiểm tra tài khoản đã đăng nhập

}

Dùng để lưu trữ thông tin tài khoản và trạng thái đăng nhập của tài khoản

**vector<user> users**

users là tập hợp kiểu vector chứa các phần tử kiểu user lưu trữ thông tin về tài khoản người dùng

**struct ClientInfo**{

bool islogin; //lưu trạng thái đăng nhập

int login\_place; //vị trí tài khoản đã đăng nhập trong userList

char username[260];

int fa\_bytes; //số bytes đã đọc file fa.txt(lưu friend address) khi đăng nhập

int recvBytes;

char recvBuff[BUFFSIZE];

queue<char\*> responses;

}

Dùng để lưu trữ thông tin của phiên làm việc của client. Khối ClientInfo sẽ được sử dụng trong việc sử lí thông điệp của client và lưu trữ bản tin phản hồi.

**struct SocketInfo**{

OVERLAPPED overlapped;

SOCKET socket;

int operation; // giá trị khởi tạo RECEIVE

WSABUF dataBuff;

int sendBytes;

int sentBytes;

ClientInfo clientInfo;

}

Dùng để lưu trữ thông tin làm việc của socket. Khối dữ liệu được sử dụng để lưu trữ thông tin trang thái mạng và để thực hiện việc trao đổi dư liệu qua mạng.

### 2.2. Các hàm trong file data\_type.h

+ **initSockInfo(), removeSockInfo()** dùng để khởi tạo và hủy bỏ cấu trúc **SocketInfo** của một phần tử trong một mảng.

+ **accountLocation()** dùng để tìm vị trí của tài khoản trong **userList** phục vụ hàm **getLI\_response()**

+ **checkNum()** dùng để kiểm tra địa chỉ có chứa chỉ kí tự số

+ **toWchar\_t()** dùng để chuyển kiểu dữ liệu ***char array*** hoặc ***string*** sang ***wchar array*** phục vụ hàm **makeDirectory()** và **getFileNames()**

### 2.3. Các hàm trong file response.h

+ **packageHeader**() dùng để dán trường opcode, length vào bản tin.

+ **getNoticeMess()** tạo bảo tin thông báo thành công, thất bại hoặc các bản tin báo hoàn tất truyền dữ liệu.

\* Các hàm có tên dạng **getXX\_Response()** nhận vào các bản tin với **mess\_type XX** từ client từ đó xử lí rồi tạo ra các bản tin phản hồi được lưu vào biến **responses** của cấu trúc **ClientInfo**.

**+ getRG\_response()**

* Gọi hàm ***getAccount()*** lấy thông tin tài khoản từ bản tin
  + *0 → bản tin không hợp lệ*
  + *1 → username không hợp lệ*
  + *2 → password không hợp lệ*
* Tạo foulder “users/username”, “users/username/back\_up”, file “users/username/fa.txt” với hàm ***makeDirectory()*** và ***createFile()***
  + *1 → username không hợp lệ*
* Lưu tài khoản vào file “account.txt” dùng hàm ***appendFile()***
* Lưu tài khoản vào biến toàn cục có kiểu ***users***
* Tạo bản tin thành công với hàm ***getNoticeMess()*** và lưu vào biến ***responses*** trong cấu trúc ***ClientInfo***

**+ getLI\_response()**

* Kiểm tra phiên đã đăng nhập
  + *6 → chưa đăng xuất*
* Gọi hàm ***getAccount()*** lấy thông tin tài khoản
  + *0 → bản tin không hợp lệ*
  + *1 → username không hợp lệ*
  + *2 → password không hợp lệ*
* Gọi hàm ***accountLocation()*** để kiểm tra tài khoản tồn tại
  + *3 → tên đăng nhập hoặc mật khẩu sai*
* Đánh dấu tài khoản đã đăng nhập
* Lưu thông tin đăng nhập vào khối ***clientInfo*** gồm: ***islogin, login\_num, username và fa\_bytes*** từ hàm ***getFileSize***(users/username/fa.txt)
* Tạo các bản tin truyền dữ liệu từ file “fa.txt” với hàm ***readFileBlock()*** rồi thêm vào biến ***responses*** trong cấu trúc ***ClientInfo***

**+ getGC\_response()**

* Kiểm tra phiên đã đăng nhập
  + *6 → chưa đăng xuất*
* Gọi hàm ***getFileNames(***users/username***)*** để lấy thông tin tên các file (tên danh mục) lưu vào biến ***string*** ***fileNames***
* Tạo các bản tin truyền dữ liệu từ ***fileNames*** rồi thêm vào biến ***responses*** trong cấu trúc ***ClientInfo***

**+ getSG\_response()**

* Kiểm tra phiên đã đăng nhập
  + *6 → chưa đăng xuất*
* Lấy kích thước của file “suggest.txt”
* Tạo các bản tin truyền dữ liệu từ file “suggest.txt” với hàm ***readFileBlock()*** rồi thêm vào biến ***responses*** trong cấu trúc ***ClientInfo***

**+ getAA\_response()**

* Kiểm tra phiên đã đăng nhập
  + *6 → chưa đăng xuất*
* Gọi hàm ***getAddress()*** lấy thông tin danh mục, địa chỉ rồi lưu vào hai biến ***string category, address***
  + *0 → bản tin không xác định*
  + *11 → category không đúng*
* Gọi hàm ***checkNum()*** kiểm tra ***address*** chỉ chứa toàn số
  + Nếu phải thì **prefix** = ‘-’ và thực hiện tính ***int num = atoi(address)***
  + Gọi làm ***getFileLine***(‘fa.txt’,**num*)*** để nội dung dòng thư **num** rồi lưu vào biến ***string line***
  + Lấy địa chỉ từ biến **line** lưu vào **address**
    - *21 → address number không hợp lệ*
* Gọi hàm ***appendFile***(users/username/category.txt,***prefix****+****address***) để thêm địa chỉ vào ***category***
* Tạo bảo thành công với hàm ***getNoticeMess()*** thêm vào biến ***responses***

**+ getRA\_response()**

* Kiểm tra phiên đã đăng nhập
  + *6 → chưa đăng xuất*
* Gọi hàm ***getAddress()*** lấy thông tin danh mục, địa chỉ rồi lưu vào hai biến ***string category, address***
  + *0 → bản tin không xác định*
* Gọi hàm ***checkNum()*** để kiểm tra ***address*** chỉ chứa toàn số
* Nếu phải thì thực hiện tính ***int num = atoi(address)***
* Gọi hàm ***deleteFileLine***(***category***, ***num***)
  + *11 → category không đúng*
  + *21 → address number không hợp lệ*
* Tạo bảo tin thành công với hàm ***getNoticeMess()*** thêm vào biến ***responses*** trong cấu trúc ***ClientInfo***

**+ getGA\_response()**

* Kiểm tra phiên đã đăng nhập
  + *6 → chưa đăng xuất*
* Kiểm tra cú pháp bản tin, lấy thông tin về danh mục lưu vào biến ***string category***
  + *0 → bản tin không xác định*
* Tính kích thước file int ***file\_size = getFileSize***(‘users/username/category’)
  + *11 → category không đúng*
* Tạo các bản tin truyền dữ liệu file “users/username/category.txt” với hàm ***readFileBlock()*** rồi thêm vào biến ***responses*** trong cấu trúc ***ClientInfo***

**+ getSF\_response()**

* Gọi hàm ***getFileLine()*** lấy tên địa điểm mà người dùng đã nhập số thứ tự (chính là số dòng)
* Sử dụng hàm ***appendFile()*** lưu địa điểm được chia sẻ cùng với tên người dùng vào file “users/friend\_username/fa.txt”
* Tạo bản tin thành công với hàm ***getNoticeMess()*** và lưu vào biến ***responses*** trong cấu trúc ***ClientInfo***

**+ getBK\_response()**

* Kiểm tra phiên đã đăng nhập
  + *6 → chưa đăng xuất*
* Kiểm tra cú pháp bản tin, lấy thông tin về danh mục lưu vào biến ***string category***
  + *0 → bản tin không xác định*
* Gọi hàm ***copyFile()*** sao chép file từ “users/username/category\_name.txt” đến foulder “users/username/back\_up/”
* Tạo bản tin thành công với hàm ***getNoticeMess()*** và lưu vào biến ***responses*** trong cấu trúc ***ClientInfo***

**+ getGB\_response()**

* Kiểm tra phiên đã đăng nhập
  + *6 → chưa đăng xuất*
* Sử dụng hàm ***getFileNames()*** lấy tên các danh mục sao lưu trong thư mục “users/username/back\_up”
* Sử dụng các hàm ***length*\_*param()*** và ***packageHeader()*** để tạo thông điệp gửi tới client
* Tạo bản tin thành công với hàm ***getNoticeMess()*** và lưu vào biến ***responses*** trong cấu trúc ***ClientInfo***

**+ getRS\_response()**

* Kiểm tra phiên đã đăng nhập
  + *6 → chưa đăng xuất*
* Kiểm tra cú pháp và lấy thông tin danh mục lưu vào biến string category
  + *0 → bản tin không xác định*
* Gọi hàm ***copyFile()*** sao chép file từ “users/username/back\_up/category\_name.txt” đến “users/username/”
  + *11 → category không hợp lệ*
* Tạo bản tin thành công với hàm ***getNoticeMess()*** và lưu vào biến ***responses*** trong cấu trúc ***ClientInfo***

**+ getLO\_response**

* Kiểm tra tài khoản đã đăng nhập
  + *6 → chưa đăng xuất*
* Thực hiện xóa ***fa\_bytes*** đầu tiên từ file “fa.txt”
* Thiết lập phiên về chưa đăng nhập
* Tạo bản tin thành công với hàm ***getNoticeMess()*** rồi thêm vào biến ***responses*** trong cấu trúc ***ClientInfo***

\*Nếu client thoát ra mà không đăng xuất thì hệ thống chỉ đăng xuất chỉ đăng xuất mà không xóa file ‘fa.txt’.

### 2.4. Các hàm trong file unbox\_mess.h

+ **getLength()**: lấy thông tin trường length trong bản tin của client

+ **getAccount()**: lấy thông tin username, password trong bản tin đăng kí, đăng nhập của client và trả về mã lỗi nếu có.

+ **getAddress()**: lấy tên danh mục và địa chỉ từ bản tin thêm, xóa địa chỉ của client và trả về mã lỗi nếu có.

+ **getSharedAddress():** dùng để lấy username, category và address từ thông điệp chia sẻ địa chỉ client gửi đến.

### 2.5. Các hàm trong file io\_file.h

File io\_file.h chứa các hàm thực hiện thao tác với file

**+ prepareUserList()**: đọc file lưu trữ thông tin tài khoản và lưu thông tin này vào biến toàn cục có kiểu users

**+ makeDirectory()**: tạo foulder với đường dẫn cho trước

**+ createFile()**: tạo mới file từ đường dẫn cho trước

**+ appendFile()**: ghi thêm dữ liệu vào cuối file

**+ getFileSize()**: trả về kích thước file

**+ findFileLine()**: tìm kiếm vị trí bắt đầu, kết thúc và nội dung của dòng cần tìm từ số của dòng và đường dẫn của file

**+ deleteFileLine()**: xóa dòng ra khỏi file với số dòng và đường đẫn file

**+ getFileLine()**: lấy nội dung của dòng với tham số là số dòng, đường đẫn file

**+ readFileBlock()**: đọc một khối dữ liệu từ vị trí đầu, kích thước tối đa và đường dẫn file cho trước và trả về kích thước đọc được

+ **copyFile()** dùng để sao chép một file và dán vào thư mục khác.

## B. Phía client

### 1. Hoạt động chung

1. Khởi tạo socket kết nối đến server
2. Hiển thị menu và cho người dùng chọn
3. Tạo bản tin yêu cầu có thể yêu cầu nhận liệu dùng hàm trong file message.h
4. Gửi bản tin đến server
5. Nhận bản tin phản hồi từ server và hiển thị sử dụng hàm trong file response.h
6. Đến bước 3 nếu tác vụ chưa kết thúc ngược lại quay về bước 2

### 2. Bảng liệt kê các hàm mô đun

|  |  |
| --- | --- |
| message.h | response.h |
| * pasteLength() * getRG\_message() * getLI\_message() * getAA\_message() * getRA\_message() * getGA\_message() * getSF\_message() * getBK\_message() * getRS\_message() * getLO\_message() | * getLength() * getErrorNotice() * showFriendAddress() * showSuggestCategory() * showCategory() * showAddress() * showBackedupCategory() * receive() |

### 2.1. Các hàm trong file message.h

Các hàm getXX\_message() dùng để tạo các bản tin ứng với chức năng tương ứng. Do một chức năng để hoàn thành thì có thể cần phải trao đổi nhiều bản tin với server vì vậy ta dùng biến toàn cục int seq để dùng cho các trường hợp cần tạo nhiều bản tin để hoàn thành chức năng.

**+ pasteLength**(): thêm trường length vào bản tin gửi đi

**+ getRG\_message**()

Tạo bản tin đăng kí tài khoản

*Xử lí*

* Lấy thông tin username và password từ người dùng
* Tạo bản tin đăng kí

**+ getLI\_message**()

Tạo bản tin đăng nhập

*Xử lí*

* Lấy thông tin username và password từ người dùng
* Tạo bản tin đăng nhập

**+ getAA\_message**()

Tạo các bản tin cho chức năng thêm địa chỉ

*Xử lí*

* Nếu seq == 0
  + Tạo bảo tin lấy danh mục gợi ý
  + Chuyển seq = 1 → kết thúc
* Nếu seq == 1
  + Lấy thông tin tên danh mục và địa chỉ từ người dùng
  + Tạo bản tin thêm địa chỉ
  + Chuyển seq = 0 → kết thúc

**+ getRA\_message**()

Tạo các bản tin cho chức năng xóa địa chỉ

*Xử lí*

* Nếu seq == 0
  + Lấy thông tin tên danh mục từ người dùng
  + Tạo bản lấy thông tin địa chỉ
  + Chuyển seq = 1 → kết thúc
* Nếu seq == 1
  + Lấy số của địa chỉ cần xóa từ người dùng
  + Tạo bản tin xóa địa chỉ
  + Chuyển seq = 0 → kết thúc

**+ getGA\_message**()

Tạo các bản tin cho chức năng xem địa chỉ

*Xử lí*

* Nếu seq == 0
  + Tạo bản tin lấy tên danh mục cá nhân
  + Chuyển seq = 1 → kết thúc
* Nếu seq == 1
  + Lấy thông tin tên danh mục từ người dùng
  + Tạo bản tin lấy thông tin địa chỉ
  + Chuyển seq = 0 → kết thúc

**+ getSF\_response**()

Tạo bản tin cho chức năng chia sẻ địa chỉ

Xử lí

* Nếu seq == 0
  + Tạo truy vấn các danh mục mà người dùng hiện có
  + Chuyển seq = 1
* Nếu seq == 1
  + Yêu cầu người dùng nhập tên danh mục vào
  + Tạo truy vấn các địa điểm trong danh mục mà người dùng vừa nhận
  + Chuyển seq = 0

**+ getBK\_response**()

Tạo bản tin cho chức năng backup danh mục

Xử lí

* Nếu seq == 0
  + Tạo truy vấn các danh mục mà người dùng hiện có
  + Chuyển seq = 1
* Nếu seq == 1
  + Yêu cầu người dùng nhập tên danh mục vào
  + Tạo yêu cầu sao lưu danh mục đó
  + Chuyển seq = 0

**+ getRS\_message**()

Tạo các bản tin cho chức năng restore danh mục

*Xử lí*

* Nếu seq == 0
  + Tạo bản tin lấy tên danh mục đã được backup
  + Chuyển seq = 1
* Nếu seq == 1
  + Lấy thông tin danh mục cần khôi phục
  + Tạo bản tin khôi phục
  + Chuyển seq = 0

**+ getLO\_message**()

Tạo bản tin đăng xuất

### 2.2. Các hàm xử lí trong file response.h

* **getLength**(): lấy trường lengh từ bản tin phản hồi của server
* **getErrorNotice**(): hiển thị thông tin lỗi từ mã lỗi
* **showFriendAddress**(): hiển thị địa chỉ chia sẻ của bạn bè
* **showSuggestCategory**(): hiển thị thông tin danh mục gợi ý
* **showCategory**(): hiển thị thông tin danh mục cá nhân
* **showAddress**(): hiển thị thông tin địa chỉ
* **showBackedupCategory**(): hiển thị thông tin danh mục đã backup
* **receive**(): nhận phản hồi từ server và hiển thị chúng cho người dùng