**­­­­­­­­**

**HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY**

**FINAL REPORT**

**NETWORK PROGRAMMING**

**Application for storing files**

**Group 10:**

**Nguyễn Trọng Khang - 20184275**

**Hoàng Thọ Tùng - 20184325**

**Nguyễn Thị Thu Thảo - 20184308**

**Instructor: Ph.D Tran Nguyen Ngoc**

**Hanoi, 08/2022**

Table of Contents

[Task allocation 3](#_Toc111013864)

[Introduction 4](#_Toc111013865)

[I. Application design 5](#_Toc111013866)

[1. Requirement 5](#_Toc111013867)

[2. Use case diagram 5](#_Toc111013868)

[3. Flow chart 7](#_Toc111013869)

[4. Protocol messages 7](#_Toc111013870)

[5. Activity diagram 9](#_Toc111013871)

[6. UI design 11](#_Toc111013872)

[II. Technologies 15](#_Toc111013873)

[1. Client 15](#_Toc111013874)

[1.1. Java and SpringBoot 15](#_Toc111013875)

[1.2. JavaFx 15](#_Toc111013876)

[2. Server 16](#_Toc111013877)

[3. Socket programming 16](#_Toc111013878)

[3.1. Socket programming 16](#_Toc111013879)

[3.2. TCP 17](#_Toc111013880)

[III. Implementation 19](#_Toc111013881)

[1. Step 1 - Import needed libraries, packages 20](#_Toc111013882)

[2. Step 2 - Set up socket 20](#_Toc111013883)

[3. Step 3 - Handle message 21](#_Toc111013884)

[3.1. Create folder 21](#_Toc111013885)

[3.2. List files 23](#_Toc111013886)

[3.3. Upload 24](#_Toc111013887)

[3.4. Download 25](#_Toc111013888)

[3.5. Delete 26](#_Toc111013889)

[IV. Drawback/Improvement 27](#_Toc111013890)

# Task allocation

* Tung: GUI + Client, make plan + design, write report
* Thao: design message + server (message create folder, listfile)
* Khang: design message + server (message download, upload)

# Introduction

Nowadays, Internet is very popular. People can communicate and send documents, information from anywhere in the world. Therefore, we have a lot of applications for storing files like Google Drive, OneDrive… In this Network Programming course, we have an idea to build an application to store files. When people use our application, each of them is provided an account and they can login, upload files to our server and download files to their local. The architechture of our application is Client-Server and they connect by TCP protocol. Technology that we use including TCP, multi-client, socket, Java, C…

# Application design

## Requirement

Server should have following functions: Allocates a folder on server for each registered user; When an user logins, server will checks user\_name whether it already exists or not in the root folder in server; Only allows each user to operate on his allocated folder.

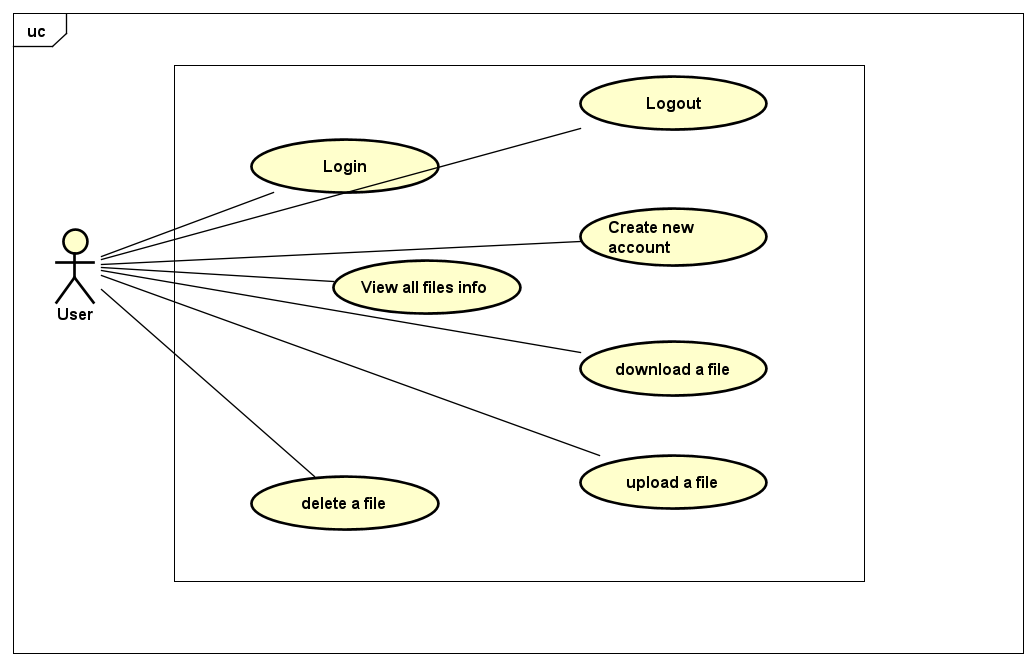
Client should have following functions: Users can upload/download files to/from their folder; Delete files in their folder; When uploading, client need to check whether file size is less than 10 Mb or not, if not, don’t allow the user upload this file.

In file tranferring, file is divided into blocks of data. Client and server will send/receive file block by block. If there is any error with any block, file transferring process will stop and start again from the beginning.

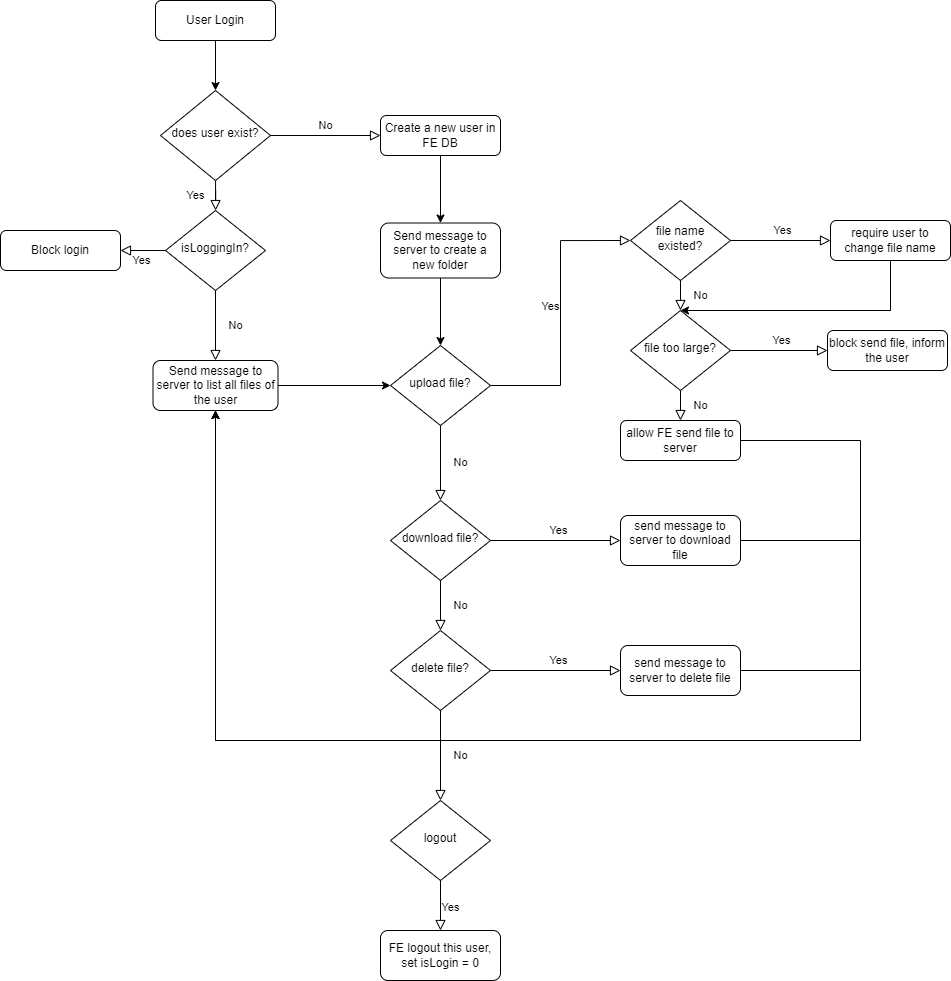
## Use case diagram

The system has only one actor: user.

General use case diagram:



## Flow chart



## Protocol messages

We design a set of protocol messages between client and server, specifically for our system. They are shown in the below table:

|  |  |  |
| --- | --- | --- |
| Opcode | Client request | Server respone |
| 0: Client requests server to create new folder for new user | 0+<user\_name> | “Create success!” |
| 1: Client requests server to receive information of user’s files  2: Server responds information of user’s files | 1+<user\_name> | 2+<number\_files>(2 bytes)+<file\_name\_1>+<file\_size\_1>(2 bytes)+<file\_name\_2>+<file\_size\_2>+… |
| 3: Client requests to upload file. Server sends confirm message 60. Then client begins to send data blocks to server | 3+<file\_name>+  <number\_blocks> | 6+0 |
| 4: Client requests to download file and server send data blocks | 4+<file\_name>  Client sends back 6 | 6+<number\_of\_block>  Server begins to send file |
| 5: Send data blocks | 5+<block\_no>(2 bytes)+<block\_data>(0-1024) bytes | |
| 6: Confirm msg | 6 | |
| 7: Client requests to delete a file | 7+<file\_name> | Server deletes the file  Send back “delete success” |
| 8: Error | 8 | |
| 9: Client requests to exit | 9 | Server closes connection |

## Activity diagram

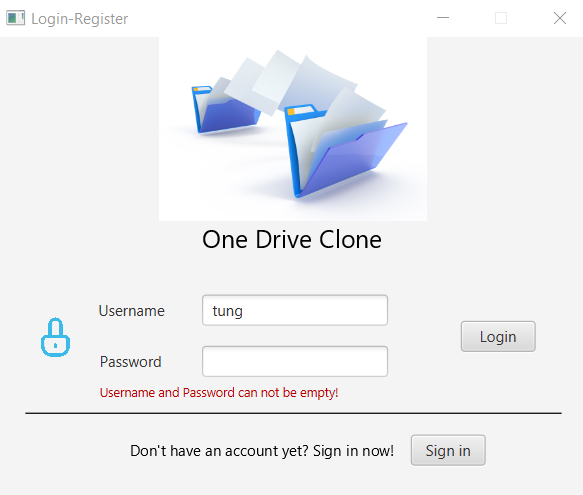
Below is the activity diagram of our system, it shows how client and server communicate with each other:

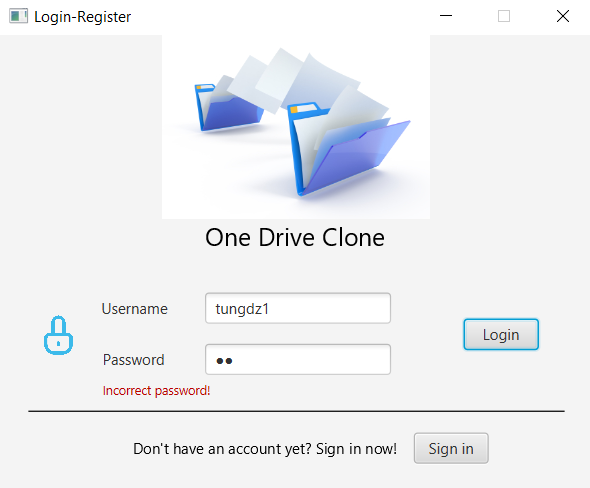
Diagram

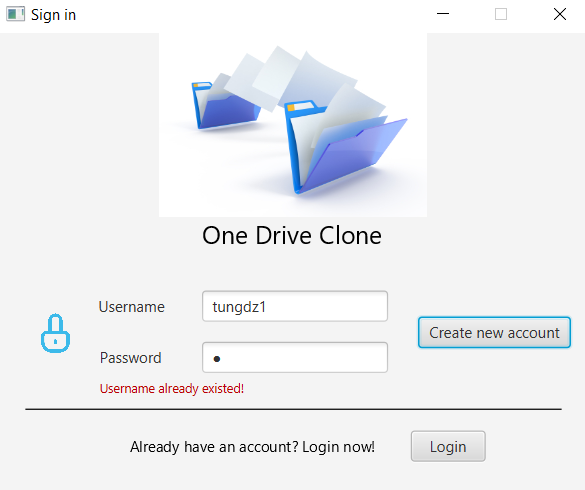
Description automatically generated

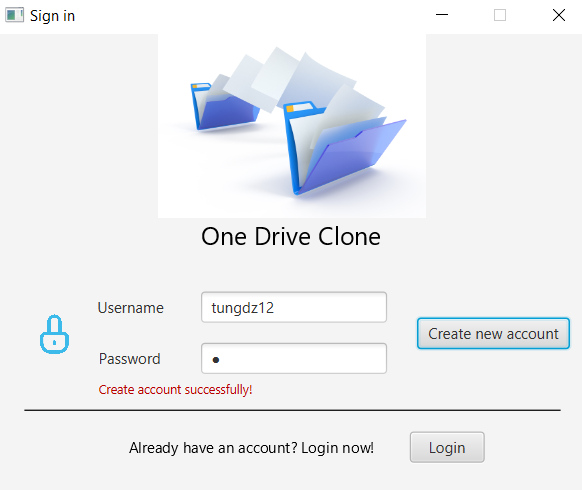
## UI design

Login-Register interface:

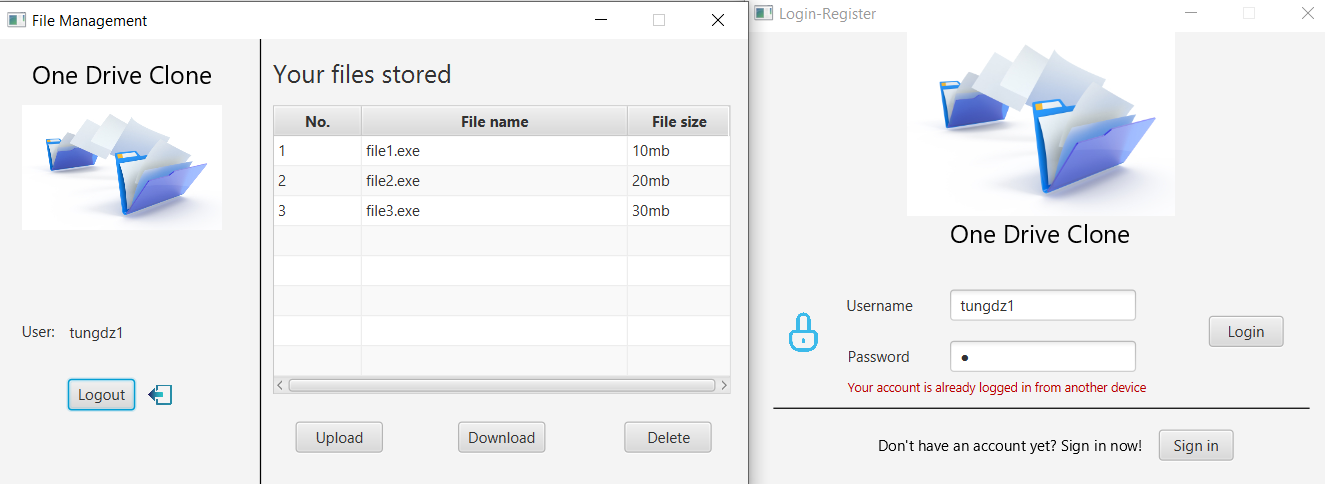




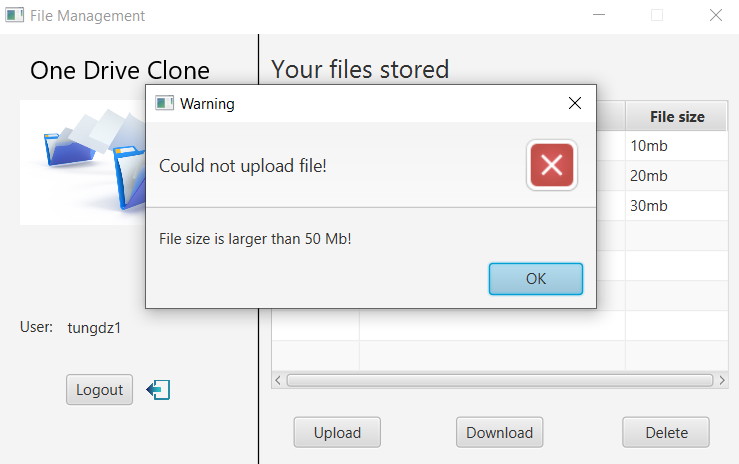




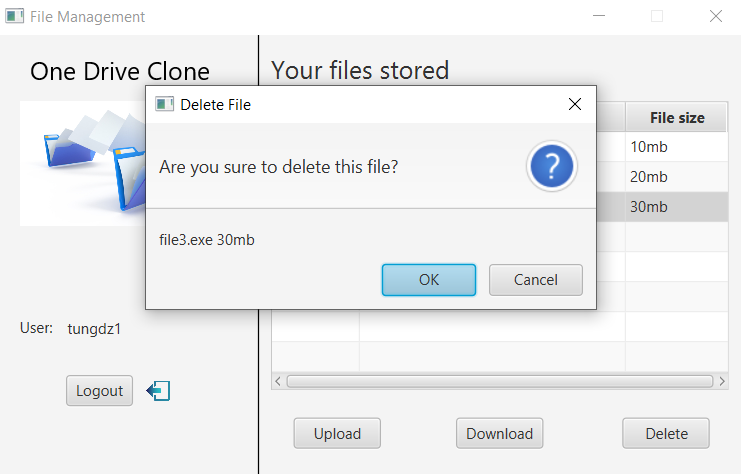
Main page with multi clients:



Check if upload file has size larger than 10 Mb:



Delete a file:



# Technologies

With this project, we choose these technologies:

* Client: Java, JavaFX (for UI), Spring Boot, MySQL, Java WebSocket
* Server: C, TCP server manages many clients using fork()
* Operating system: Server only runs on Ubuntu. Client can run on Ubuntu, Windows, MacOS, …

## Client

### Java and SpringBoot

Java is an object-oriented programming language (OOP). Object-oriented programming, with the ability make anything be classes, objects, attributes, methods, so that they interact with each other like real-life things. The object-oriented programming system has been very developed, with SOLID principles, many types of design partners that make OOP easy to build, maintain, and extend. OOP code may seem long and cumbersome, but it is easy to read and understand. Java is a very standard object-oriented language, it inherits all the above advantages, in addition, Java's performance is very good and stable. It is for that reason that Java is preferred to build large and complex systems such as banks, e-government. Also, in networking, Java strongly supports socket libraries, which work on any operating system because of platform independence of Java. In this topic, we choose Java as the client-side language.

Spring Boot is a project developed by Java in the Spring framework ecosystem. It helps us programmers simplify the process of programming an application with Spring, focusing only on developing tasks for the application. Using the Maven/Gradle library/package managers, along with the built-in Tomcat Server, makes this framework the best choice for building web-based applications.

### JavaFx

JavaFX is a Java library used to develop Desktop applications as well as Rich Internet Applications (RIA). The applications built in JavaFX, can run on multiple platforms including Web, Mobile and Desktops.

JavaFX is intended to replace swing in Java applications as a GUI framework. However, It provides more functionalities than swing. Like Swing, JavaFX also provides its own components and doesn't depend upon the operating system. It is lightweight and hardware accelerated. It supports various operating systems including Windows, Linux and Mac OS.

## Server

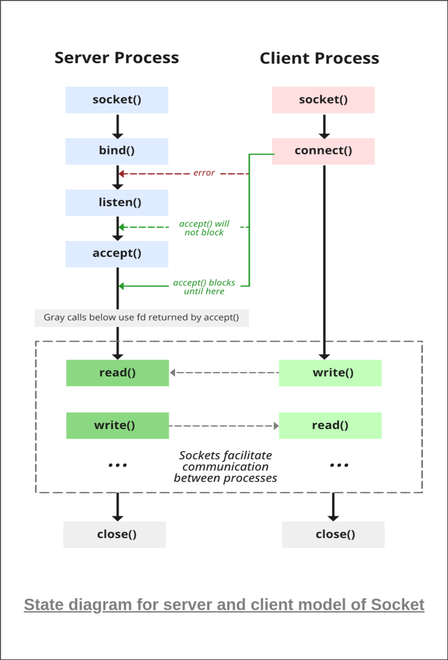
Network programming enables processes to communicate with each other over a computer network, but it is a complex task that requires programming with multiple libraries and protocols. With its support for third-party libraries and structured documentation, C programming language is an ideal language to write network programs. Besides, C is closed to system level, and performance of C is very good. So, we choose C as the programming language to develope our server, which will handle many requests from multi clients.

## Socket programming

### Socket programming

Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket(node) listens on a particular port at an IP, while the other socket reaches out to the other to form a connection. The server forms the listener socket while the client reaches out to the server.

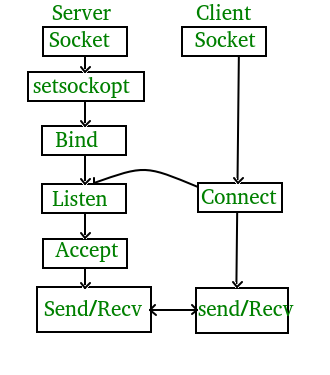
State diagram for server and client model:



### TCP

If we are creating a connection between client and server using TCP then it has few functionality like, TCP is suited for applications that require high reliability, and transmission time is relatively less critical. It is used by other protocols like HTTP, HTTPs, FTP, SMTP, Telnet. TCP rearranges data packets in the order specified. There is absolute guarantee that the data transferred remains intact and arrives in the same order in which it was sent. TCP does Flow Control and requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control. It also does error checking and error recovery. Erroneous packets are retransmitted from the source to the destination.

The entire process can be broken down into following steps:



The entire process can be broken down into following steps:

* TCP Server:
  + using create(), Create TCP socket.
  + using bind(), Bind the socket to server address.
  + using listen(), put the server socket in a passive mode, where it waits for the client to approach the server to make a connection
  + using accept(), At this point, connection is established between client and server, and they are ready to transfer data.
  + Go back to Step 3.
* TCP Client:
  + Create TCP socket.
  + connect newly created client socket to server.

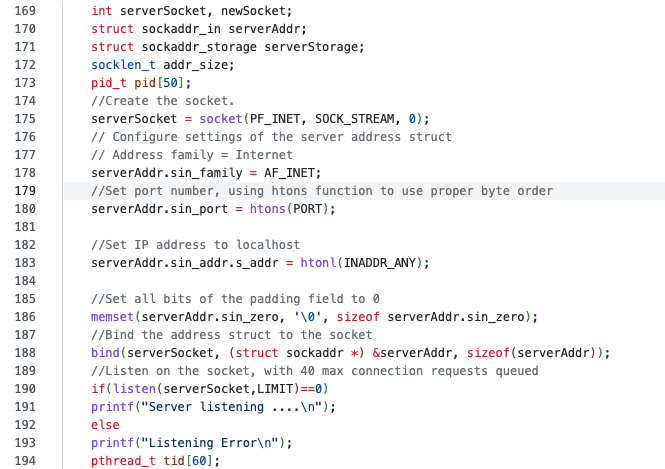
# Implementation

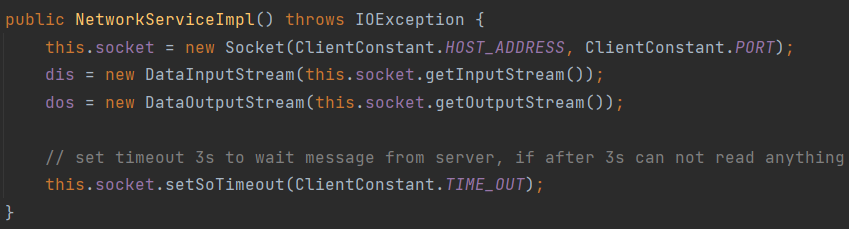
The implementation is divided into 3 main steps:

* Step 1: Import needed libraries, packages.
* Step 2: Set up socket.
* Step 3: Handle message.
  + Create folder
  + List files
  + Upload
  + Download
  + Delete
  + Exit

## Step 1 - Import needed libraries, packages

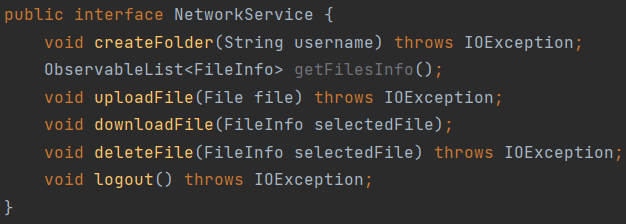
## Step 2 - Set up socket

****



## Step 3 - Handle message

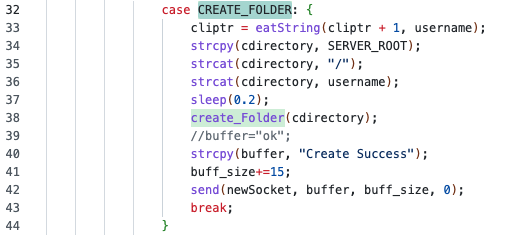
Client will have these functions to send messages to the server, and wait responses from the server:



### Create folder

Server checks if opcode == ‘0’ -> CREATE\_FOLDER

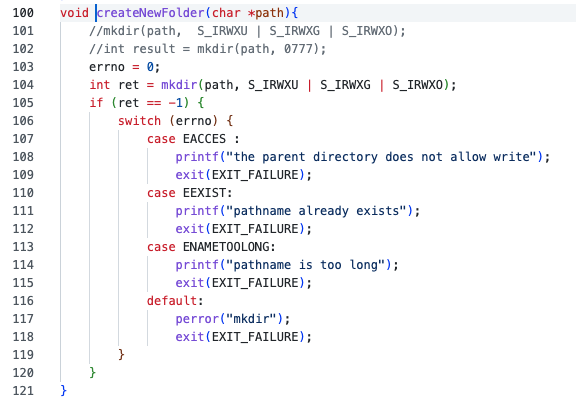
Read username from message. Get directory in format root/username:



Check if the folder exists:



If not, create a new folder:



### List files



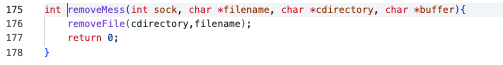
### Upload

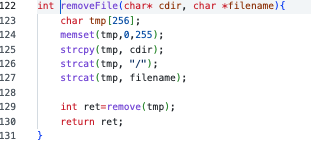


### **Download**



### Delete





# Drawback/Improvement

Our application has a friendly UI and users can download, upload, and delete files quite smoothly. However, it still has some drawbacks.

Firstly, we authenticate clients by using a local database to check the account. This is a big problem with our application. Users who really understand our system can use our application, but it is not convenient for normal users. In the future, we will check the authentication on the server. Also, we will add a function that allows users to create and work with subfolders. We did create a new folder and could work with the subfolder, but we were not sure about the authentication, because the user can access the parent folder and maybe it is the root folder. It is not secured. Therefore, we are going to develop this task in the future.