ĐẠI HỌC QUỐC GIA
THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

CSC10008 - Mạng máy tính

BÁO CÁO ĐỒ ÁN

Lập trình Socket

Họ tên	MSSV
Bùi Minh Duy	23127040
Nguyễn Thị Khánh Linh	23127082
Nguyễn Lê Hồ Anh Khoa	23127211

Giảng viên hướng dẫn

Lê Hà Minh

Thành phố Hồ Chí Minh, 24/07/2024

Lời cảm ơn

Trước tiên, chúng em xin gửi lời cảm ơn chân thành đến các thầy cô trong Khoa Công nghệ Thông tin, Trường Đại học Khoa Học Tự Nhiên, ĐHQG-HCM đã tạo điều kiện cho chúng em thực hiện đồ án lập trình socket này. Sự hướng dẫn nhiệt tình và kiến thức quý báu từ các thầy cô đã giúp chúng em rất nhiều trong quá trình học tập và nghiên cứu.

Chúng em cũng xin chân thành cảm ơn thầy Lê Hà Minh, người đã trực tiếp hướng dẫn và hỗ trợ chúng em trong suốt quá trình thực hiện đồ án. Thầy đã không quản ngại thời gian và công sức để hướng dẫn và giải đáp những thắc mắc của chúng em trong suốt quá trình thực hiện.

Ngoài ra, chúng em xin cảm ơn các anh chị và các bạn trong lớp, những người đã cùng nhau chia sẻ kinh nghiệm, ý tưởng và những lời động viên trong suốt thời gian thực hiện đồ án. Sự hỗ trợ và động viên của mọi người đã giúp chúng em vượt qua những khó khăn và thử thách trong quá trình nghiên cứu và phát triển.

Một lần nữa, xin chân thành cảm ơn tất cả mọi người!

Trân trọng,
Dại diện trưởng nhóm
Nguyễn Lê Hồ Anh Khoa

Mục lục

1 Thông tin giới thiệu

- Tên học phần: Mạng máy tính
- Giảng viên hướng dẫn: Lê Hà Minh
- Đồ án thực hiện: Lập trình Socket
- Thời gian thực hiện: Từ ngày 18/06/2024 đến ngày 31/07/2024
- Danh sách thành viên:

STT	MSSV	HỌ VÀ TÊN	EMAIL	VAI TRÒ
01	23127040	Bùi Minh Duy	bmduy23@clc.fitus.edu.vn	Thành viên
02	23127082	Nguyễn Thị Khánh Linh	ntklinh23@clc.fitus.edu.vn	Thành viên
03	23127211	Nguyễn Lê Hồ Anh Khoa	nlhakhoa23@clc.fitus.edu.vn	Nhóm trưởng

Bảng 1: Danh sách thành viên

2 Thông tin đồ án

2.1 Thông tin chung

- Tên đồ án: Lập trình Socket
- Môi trường lập trình: Visual Studio Code
- Ngôn ngữ lập trình: Python
- Các framework và module hỗ trợ:
  - Framework: `customtkinter`
  - Module: `socket, threading, os, sys, time, customtkinter`
- Giao thức trao đổi giữa Client và Server: Giao thức TCP
- Giao diện: Có hỗ trợ, thông qua `customtkinter`
- Lưu trữ và quản lý source code: Github

2.2 Phần 1

2.2.1 Cấu trúc tổng thể chương trình

Lưu ý: Dữ liệu của phần 1 được lưu trữ trong thư mục **Section_1**

Cấu trúc thư mục Section_1:

- Thư mục **Server**:
  - Thư mục **Cloud**: Chứa các file có thể tải từ Server.

  - Tệp `files_list.txt`: Lưu danh sách các tệp cho phép client download gồm tên file, và dung lượng.
  - Tệp `Server.py`: Chứa mã nguồn dùng để chạy máy chủ.
- Thư mục **Client**:
  - Thư mục **Output**: Chứa các tệp tải về từ Server.
  - Tệp `input.txt`: Ghi nhận danh sách tên các tệp sẽ được tải.
  - Tệp `Client.py`: Chứa mã nguồn dùng để chạy máy khách.

Cấu trúc các tệp mã nguồn:

- Cấu trúc mã nguồn `Server.py`:
  - Lớp `Server` để chứa các hàm khởi tạo chương trình.
  - Hàm `__init__`: Hàm mặc định của Python, dùng để khởi tạo các biến trong lớp đối tượng.
  - Hàm `log_message`: In các thông báo được gửi ra giao diện GUI.
  - Hàm `gen_file_list`: Tạo danh sách các file có trong thư mục **Server/Cloud**.
  - Hàm `send_file_list`: Đọc danh sách các tập tin có trên Server và gửi cho Client.
  - Hàm `send_file`: Thực hiện gửi file được yêu cầu cho Client.
  - Hàm `run`: Khởi tạo và kiểm soát socket kết nối giữa Server và Client.
  - Hàm `start_server`: Bắt đầu chạy server và gọi hàm `run` để khởi tạo Socket.
  - Hàm `stop_server`: Gửi tín hiệu ngừng Server và thoát khỏi chương trình.
  - Hàm `GUI`: Cấu hình các thành phần có trên giao diện của Server.
- Cấu trúc chương trình Client.py:
  - Lớp `Client` để chứa các hàm khởi tạo chương trình.
  - Hàm `__init__`: Hàm mặc định của Python, dùng để khởi tạo các biến trong lớp đối tượng.
  - Hàm `log_message`: In các thông báo được gửi ra giao diện GUI.
  - Hàm `get_file_list`: Gửi yêu cầu và nhận danh sách các tập tin có trên Server.
  - Hàm `get_new_files`: Kiểm tra file `input.txt` để lấy được danh sách tập tin vừa được thêm mới và thực hiện download từ Server.
  - Hàm `request_file_download`: Gửi yêu cầu tải file và tải file từ Server.
  - Hàm `run`: Khởi tạo và kiểm soát Socket kết nối giữa Server và Client.

– Hàm `start_client`: Bắt đầu chạy Client và gọi hàm `run` để khởi tạo Socket.

– Hàm `stop_client`: Gửi tín hiệu ngừng Client và thoát khỏi chương trình.

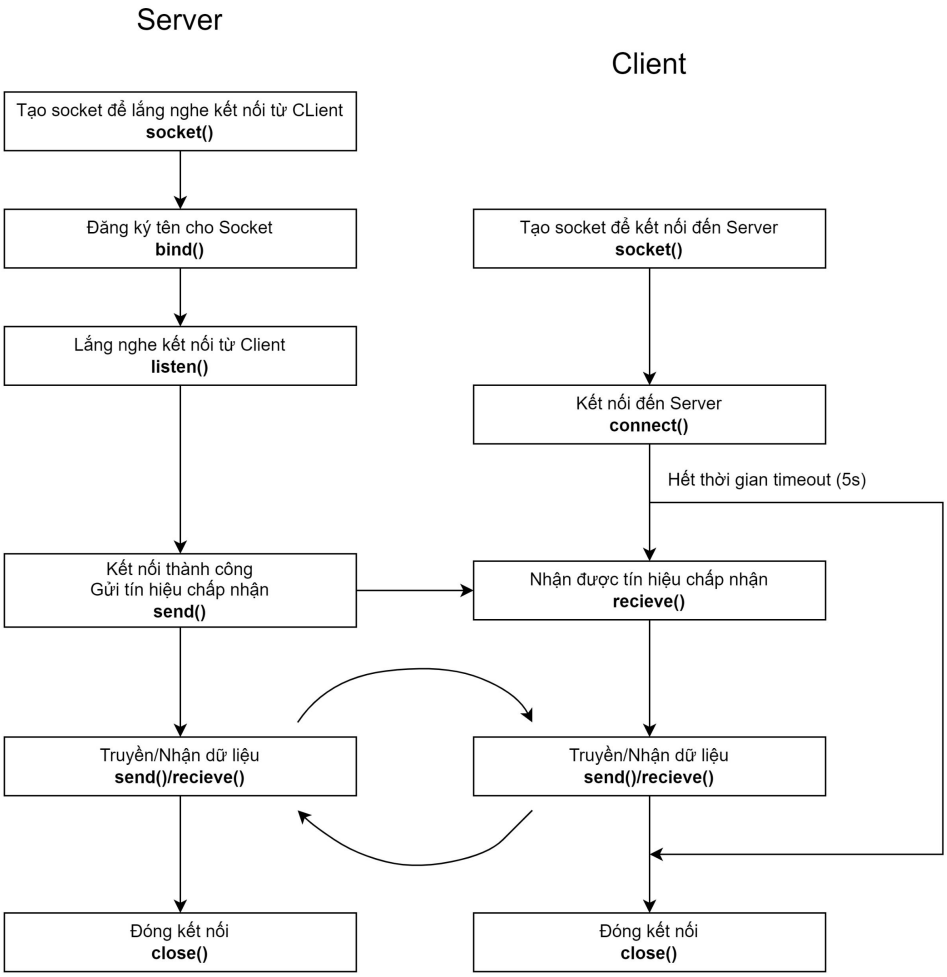
– Hàm `input_file_name`: Thu thập các tập tin cần tải người dùng nhập trên giao diện và ghi vào tập tin `input.txt`.

– Hàm `clear_placeholder`: Xóa đoạn hướng dẫn nhập tệp mới trên giao diện.

– Hàm `addr_placeholder`: Ghi đoạn hướng dẫn nhập tệp mới trên giao diện.

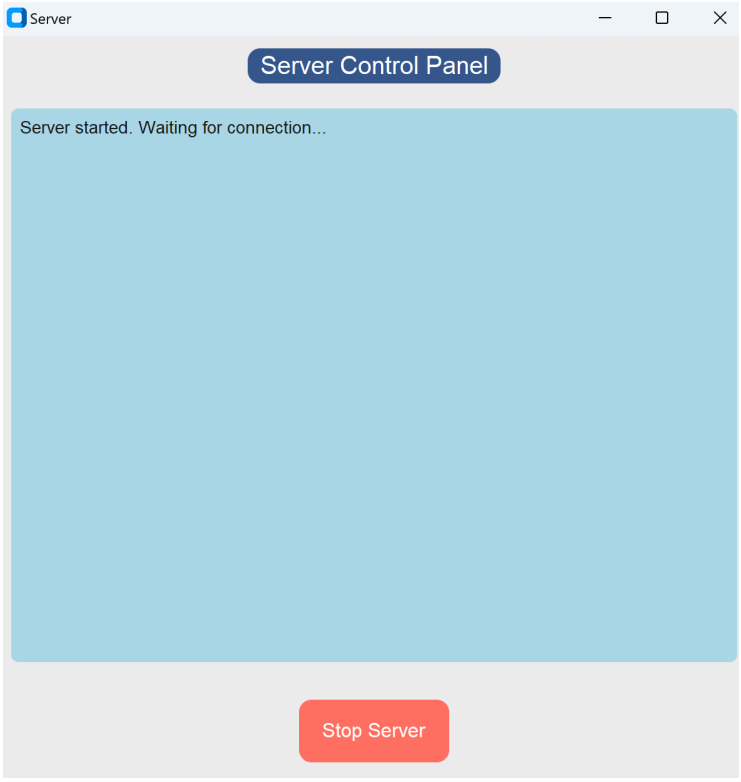
– Hàm `GUI`: Cấu hình các thành phần có trên giao diện của Client.

2.2.2 Kịch bản giao tiếp

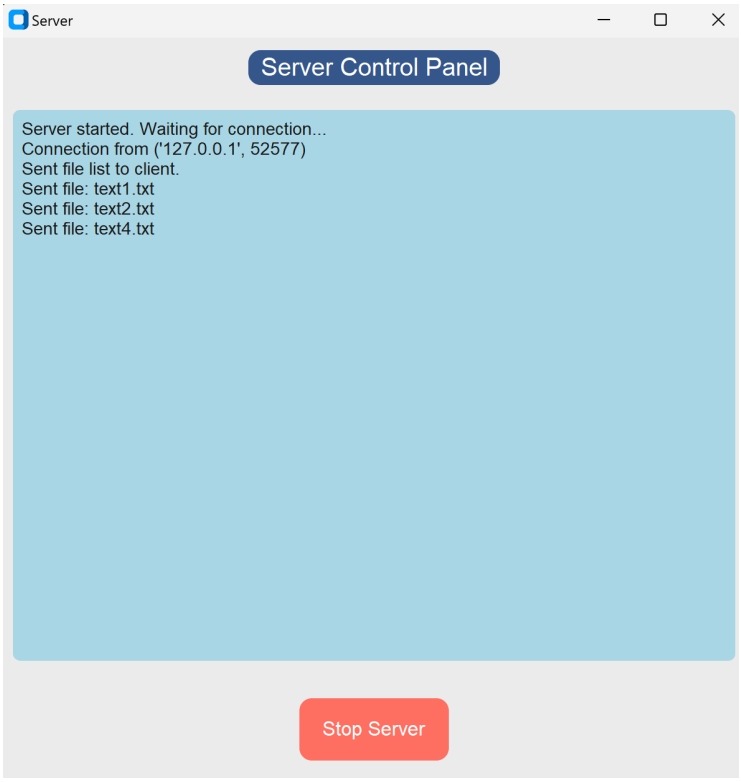


Hình 1: Sơ đồ kịch bản giao tiếp.

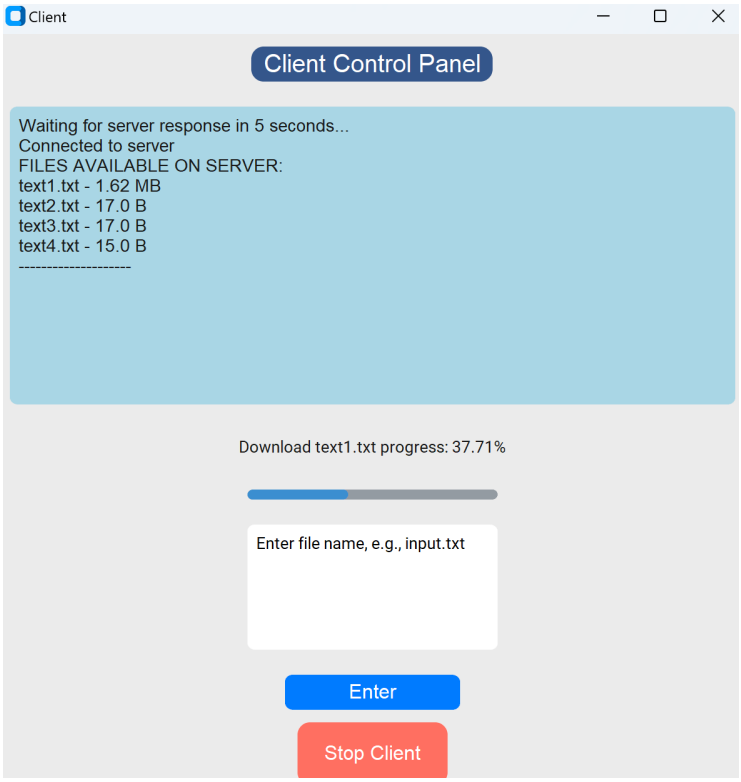
2.2.3 Minh họa quá trình chạy chương trình



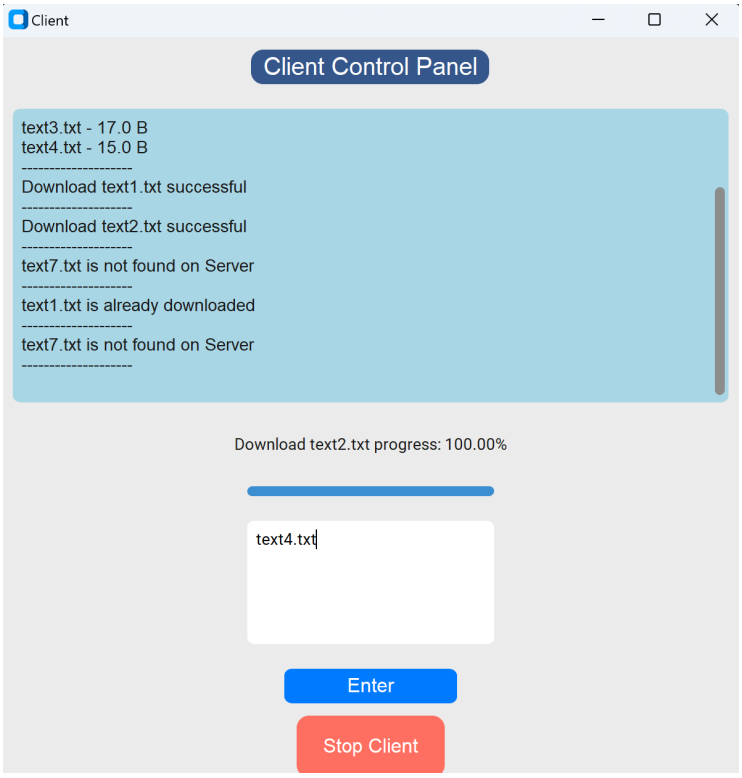
Hình 2: Giao diện Server khi vừa được khởi tạo.



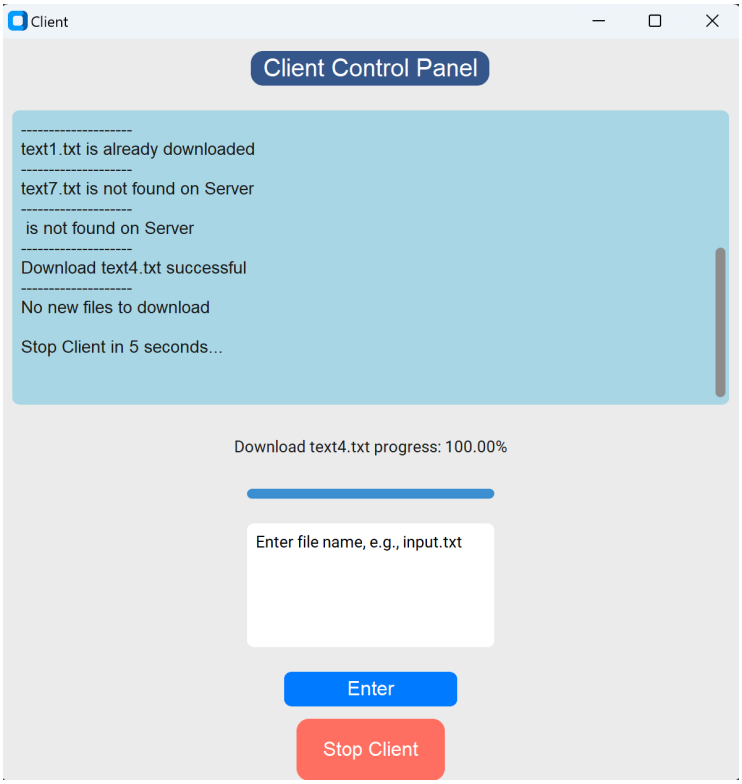
Hình 3: Giao diện Server khi đã kết nối và gửi file cho Client.



Hình 4: Giao diện Client khi được khởi tạo và bắt đầu tải tập tin.



Hình 5: Giao diện Client thông báo tình trạng tải các tập tin. Nhập tên tập tin cần tải vào ô như minh họa.



Hình 6: Giao diện Client thông báo không nhận được thêm yêu cầu tải tập tin, Client được tự động tắt trong 5 giây tới.

2.3 Phần 2

2.3.1 Cấu trúc tổng thể chương trình

Cấu trúc thư mục Section_2:

- Thư mục Server
  - Thư mục Cloud chứa các file có trên Server.
  - File "Files_list.txt"để lưu danh sách các file cho phép client download gồm tên file, và dung lượng.
  - File "Server.py"dùng để chạy máy chủ.
- Thư mục Client
  - Thư mục Output chứa cái file tải về từ Server.
  - File "input.txt"để ghi nhận danh sách các tên file sẽ được download.
  - File "Client.py"dùng để chạy máy khách
- Cấu trúc mã nguồn `Server.py`:
  - Lớp `Server` để chứa các hàm khởi tạo chương trình.
  - Hàm `__init__`: Hàm mặc định của Python, dùng để khởi tạo các biến trong

lớp đối tượng.

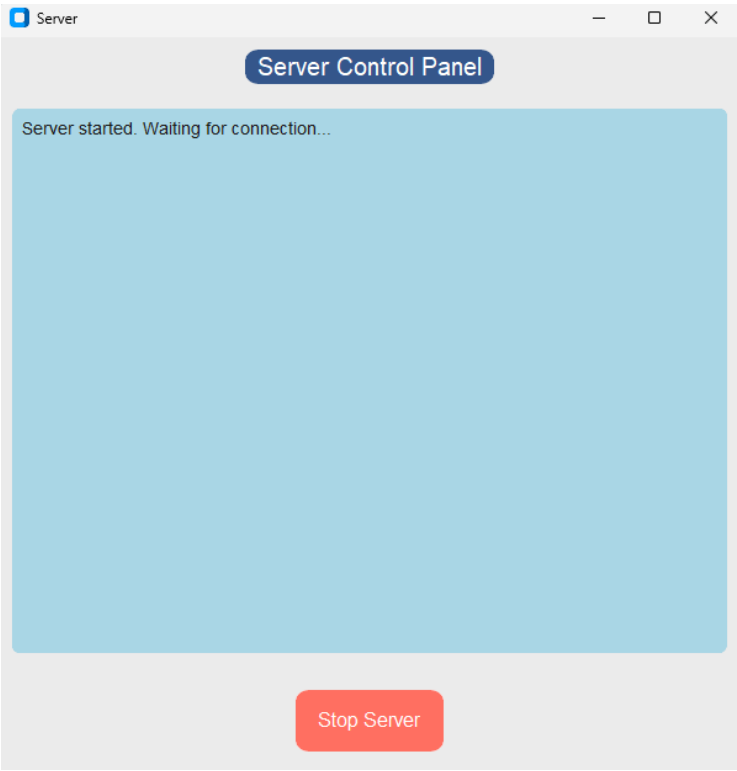
- Hàm `log_message`: In các thông báo được gửi ra giao diện GUI.
- Hàm `gen_file_list`: Tạo danh sách các file có trong thư mục **Server/Cloud**.
- Hàm `send_file_list`: Đọc danh sách các file có trên Server và gửi cho Client.
- Hàm `file_chunk_generator`: Đọc dữ liệu trong các file theo từng chunk và trả về chunk dữ liệu đó.
- Hàm `handle_client`: Xử lý việc giao tiếp và gửi dữ liệu cho Client.
- Hàm `send_data`: Gửi số chunk dữ liệu tương thích với độ ưu tiên của file cho Client.
- Hàm `process`: Tiếp nhận kết nối từ Client và đưa vào thread.
- Hàm `run`: Khởi tạo và kiểm soát socket kết nối giữa Server và Client.
- Hàm `start_server`: Bắt đầu chạy server và gọi hàm `run` để khởi tạo Socket.
- Hàm `stop_server`: Gửi tín hiệu ngừng Server và thoát khỏi chương trình.
- Hàm `GUI`: Cấu hình các thành phần có trên giao diện của Server.

• Cấu trúc chương trình Client.py:

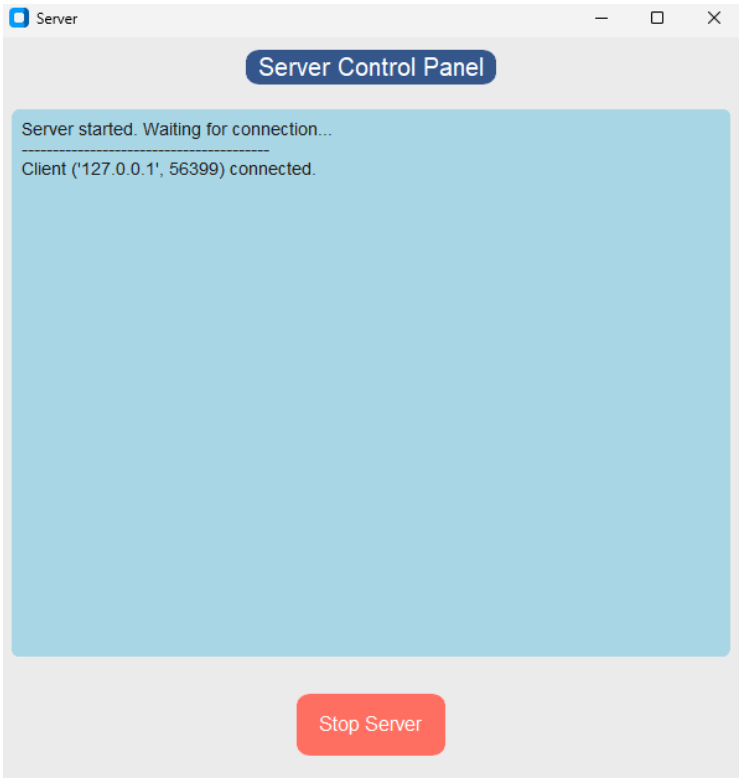
- Lớp `Client` để chứa các hàm khởi tạo chương trình.
- Hàm `__init__`: Hàm mặc định của Python, dùng để khởi tạo các biến trong lớp đối tượng.
- Hàm `get_file_list`: Gửi yêu cầu và nhận danh sách các tập tin có trên Server.
- Hàm `read_input_file`: Đọc file `input.txt` và cập nhật danh sách file cần tải mỗi 2 giây.
- Hàm `get_standard_size`: Chuẩn hóa dung lượng file theo B, KB, MB, GB, TB.
- Hàm `get_priority_size`: Chuyển đổi độ ưu tiên của file từ dạng chữ sang bytes.
- Hàm `write_file`: Ghi dữ liệu tải được từ Server vào thư mục Output mặc định.
- Hàm `is_all_done`: Kiểm tra các file cần tải đã được tải xong hết chưa.
- Hàm `client_request`: Gửi yêu cầu tải file và thực hiện tải file từ Server.
- Hàm `start_client`: Bắt đầu chạy Client và gọi hàm `run` để khởi tạo Socket.
- Hàm `stop_client`: Gửi tín hiệu ngừng Client và thoát khỏi chương trình.
- Hàm `log_message`: In các thông báo được gửi ra giao diện GUI.

- Hàm `create_progress_bar`: Tạo thanh tiến độ tải file.
- Hàm `update_progress_bar`: Cập nhật thanh tiến độ tải file.
- Hàm `add_placeholder`: Ghi đoạn hướng dẫn nhập tệp mới trên giao diện.
- Hàm `clear_placeholder`: Xóa đoạn hướng dẫn nhập tệp mới trên giao diện.
- Hàm `input_file_name`: Thu thập các tập tin cần tải người dùng nhập trên giao diện và ghi vào tập tin `input.txt`.
- Hàm `GUI`: Cấu hình các thành phần có trên giao diện của Client.

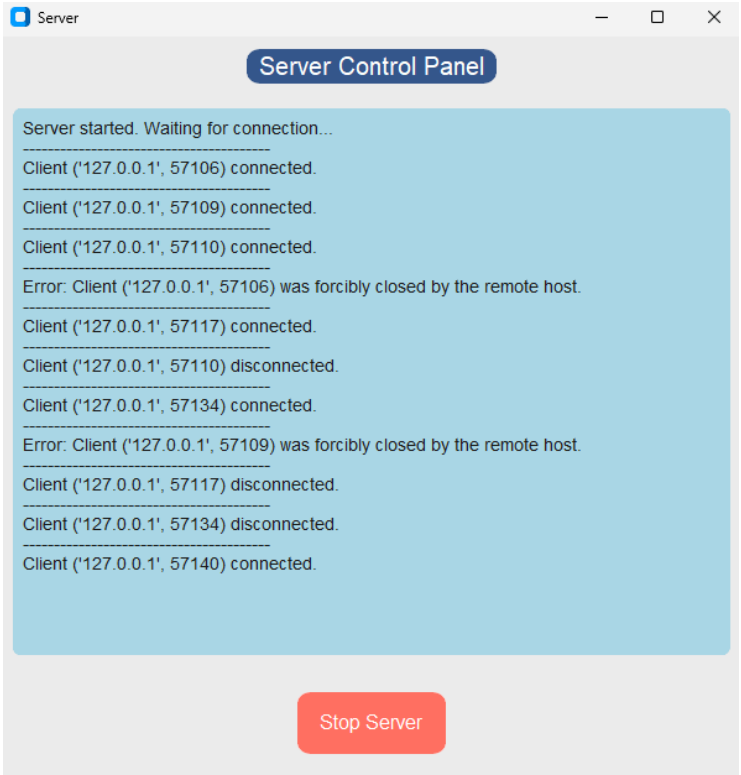
2.3.2 Minh họa quá trình chạy chương trình



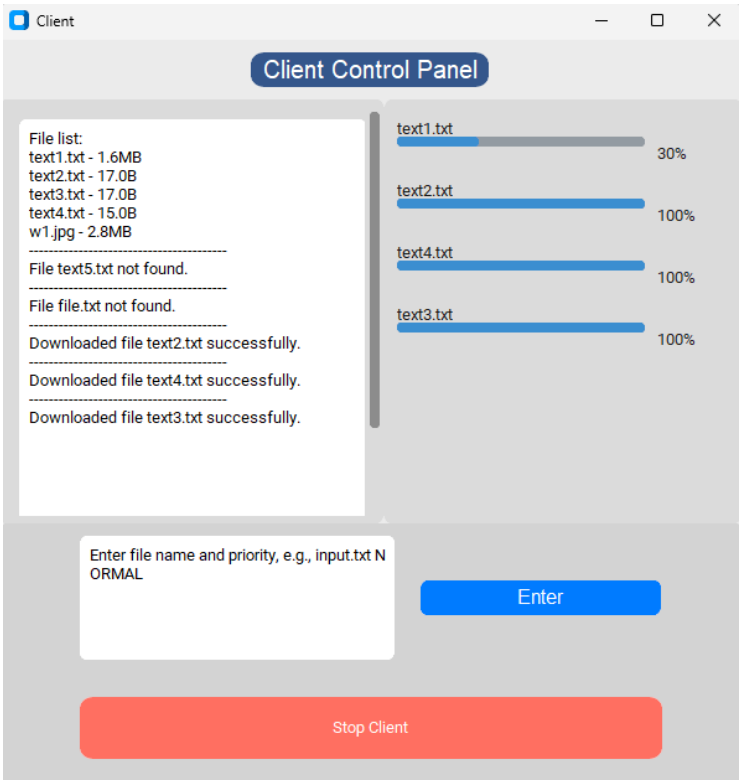
Hình 7: Giao diện Server khi vừa được khởi tạo.



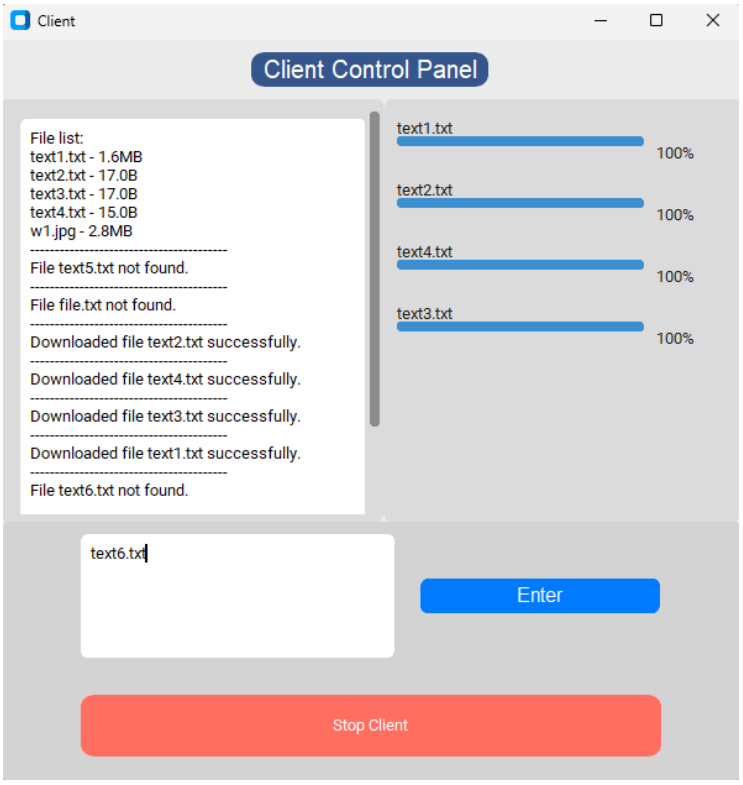
Hình 8: Giao diện Server khi có kết nối từ Client.



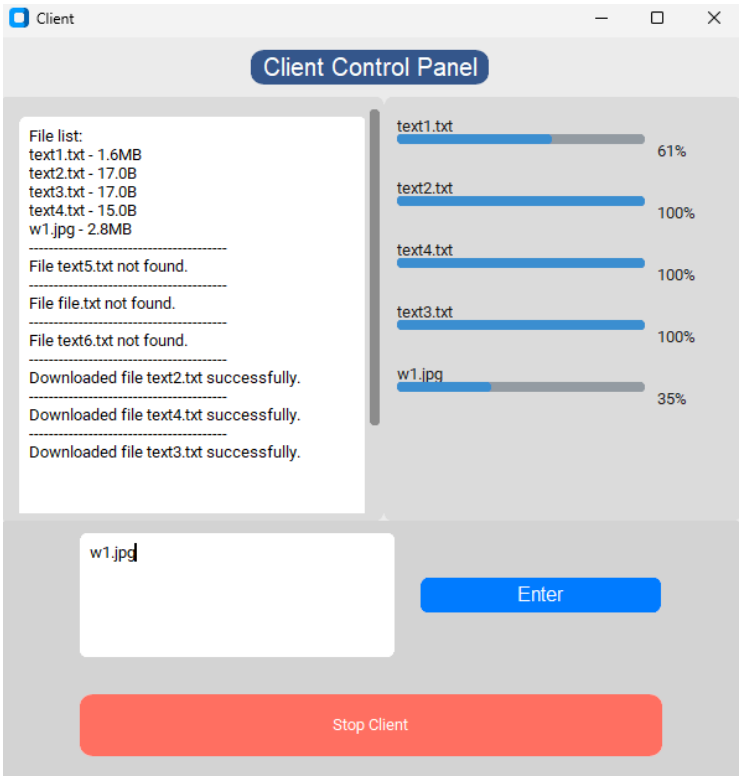
Hình 9: Giao diện Server khi phục vụ đồng thời nhiều Client.



Hình 10: Giao diện Client khi được khởi tạo và bắt đầu tải tập tin.

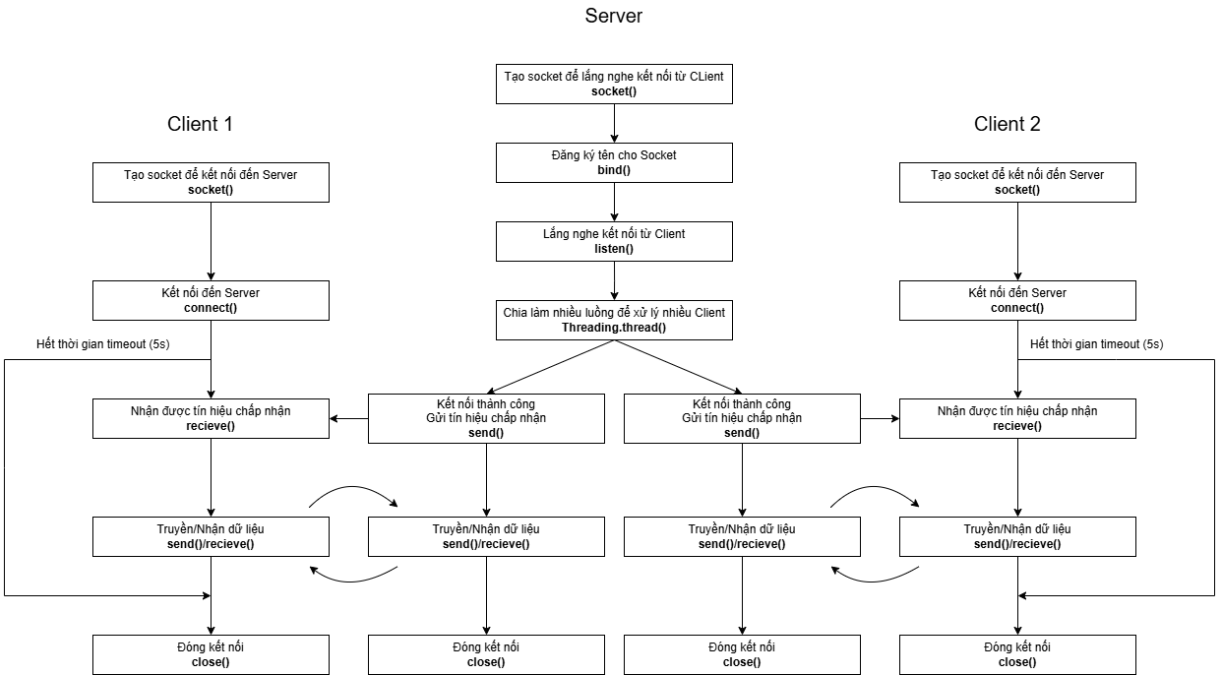


Hình 11: Giao diện Client thông báo tình trạng tải các tập tin. Nhập tên tập tin cần tải vào ô như minh họa.



Hình 12: Giao diện Client thực hiện tải file mới nhập vào.

2.3.3 Kịch bản giao tiếp



Hình 13: Sơ đồ kịch bản giao tiếp

2.4 Hướng dẫn sử dụng chương trình

Lưu ý:

- Cài đặt customtkinter bằng cú pháp sau trên terminal: `pip install customtkinter`

Hướng dẫn chi tiết

- **Bước 1:** Mở file Server trong thư mục của bài cần chạy, chạy file Server.py bằng câu lệnh `py` (hoặc `python`) `Server.py` trên terminal.
- **Bước 2:** Mở file Client trong thư mục của bài cần chạy, chạy file Client.py bằng câu lệnh `py` (hoặc `python`) `Client.py` trên terminal.
- **Bước 3:** Thao tác trên giao diện của Client, có thể thêm tập tin cần tải bằng cách gõ tên tập tin và nhấn nút **Enter**.
- **Bước 4:** Đóng Client bằng cách nhấn vào nút **Stop Client**
- **Bước 5:** Đóng Server bằng cách nhấn vào nút **Stop Server**

3 Đánh giá mức độ hoàn thành

STT	Yêu cầu	Mức độ hoàn thành	Ghi chú
Phần I	Client có thể nhận được danh sách các file từ Server và ctrl-c	100%	
Phần I	Client có thể nhận lần lượt từng file thành công từ Server. Server có thể gửi file thành công tới Client	100%	
Phần I	Hiển thị percent download file và phát hiện những file cần download tiếp theo	100%	
Phần II	Client có thể nhận được danh sách các file từ Server và ctrl-c	100%	
Phần II	2s quét file input.txt 1 lần	100%	
Phần II	Hiển thị percent download files	100%	
Phần II	Client có thể nhận files thành công từ Server. Tập tin sau khi download phải đúng và đủ dung lượng	100%	
Phần II	Độ ưu tiên CRITICAL, HIGH, NORMAL	100%	
	Báo cáo	100%	

Bảng 2: Bảng đánh giá mức độ hoàn thành

4 Bảng phân công công việc

STT	MSSV	HỌ VÀ TÊN	PHÂN CÔNG CÔNG VIỆC
01	23127040	Bùi Minh Duy	Lên ý tưởng thuật toán và thực hiện phần 2 Kiểm tra và chỉnh sửa phần 2 Chỉnh sửa báo cáo phần 2
02	23127082	Nguyễn Thị Khánh Linh	Kiểm tra và chỉnh sửa phần 1 Lên ý tưởng phần 2 Viết báo cáo
03	23127211	Nguyễn Lê Hồ Anh Khoa	Lên ý tưởng và thực hiện phần 1 Làm giao diện Server và Client Chỉnh sửa báo cáo

Bảng 3: Bảng phân công công việc

5 Nguồn tài liệu tham khảo

- **Playlist "[Đồ án mạng máy tính] - Python Socket"** | by duchieuvn
- **Playlist "Socket Programming in Python"** | by Idiot Developer
- **Documentation Introduction | Custom Tkinter**
- **Video "Python GUI Tkinter Download Manager Tutorial Part 18 | How to Create Download Manager in Python"** | by Super Coders