Multimedia Group Chat

Socket Programming Project in Python

Overview

Overview

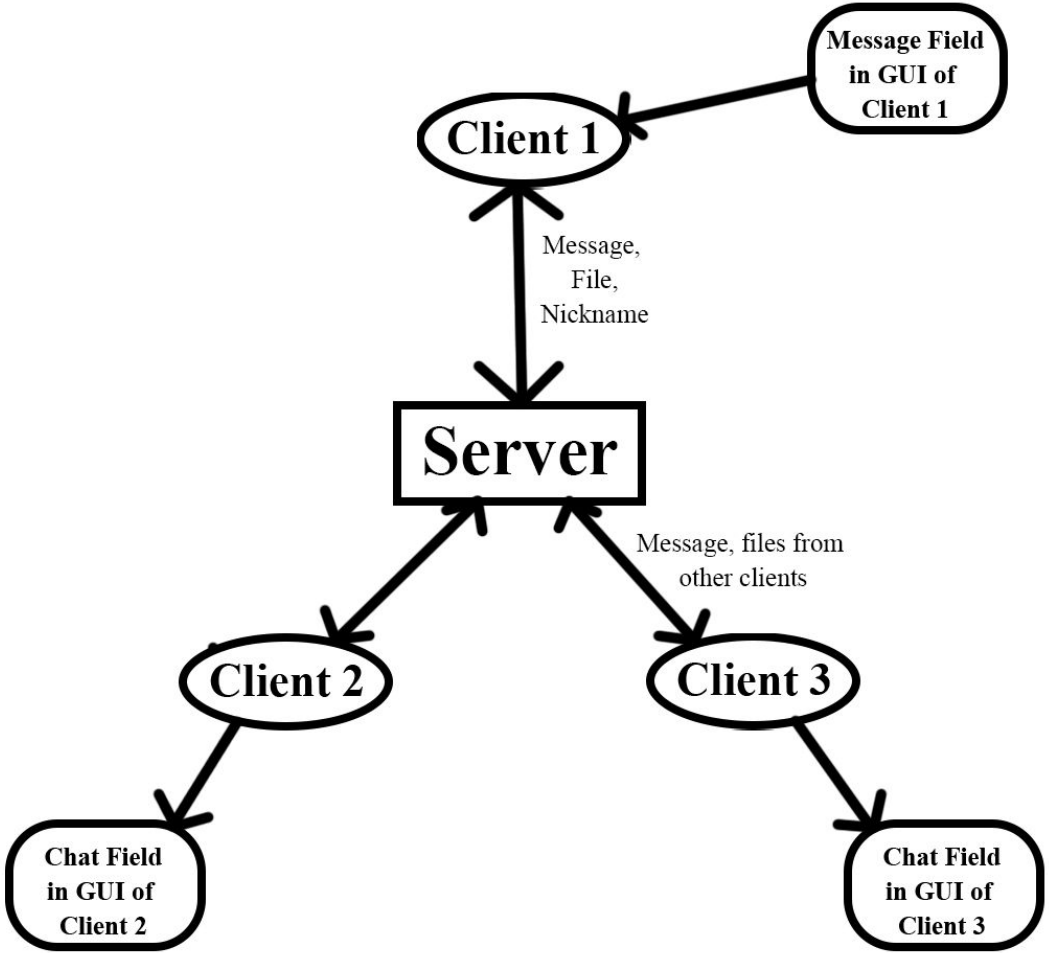
1. Socket programming is a way of connecting two nodes on a network to communicate with each other
2. One socket listens on a particular port of an IP, while the other socket reaches out to form connections.
3. The server is the listener socket and the client reaches out to connect to the server.
4. The Programming language that we have used for this project is Python.
5. We have used tkinter package to build our Graphical User Interface for the chat room.

Features

Features

1. Any system with a client.py file can connect to the server if the server is online and enter the chat room and they will be prompted to enter a nickname.
2. Text messages are supported, and we have used utf-8 encoding to encode the messages that are sent and to decode the messages that are received.
3. Any client can send a file in the same folder as the client.py file by typing in !FILE <filename> into the chat window.
4. Any client can check the information about the current chat room by typing in !INFO into the chat window.
5. We have used multiple threads in both the server - main thread, one thread for handling each client and client - one thread for handling the GUI window, and one thread for communicating with the server.

Flowchart


Message Field in GUI of Client 1
Client 1
Message, File, Nickname
Server
Message, files from other clients
Client 2
Client 3
Chat Field in GUI of Client 2
Chat Field in GUI of Client 3

Flowchart

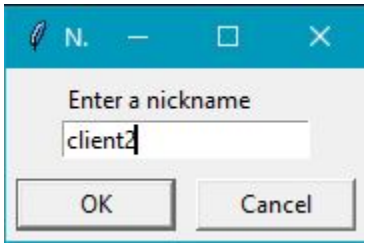
1. First the server.py file is ran in a system and the server starts listening.
2. Then the file client.py is ran from each of the required clients.
3. The client tries to connect to the server.
4. When a message is typed in the message field of the GUI of the client and the send button is pressed, the message is sent to the server encoded in utf-8 format.
5. The server receives the message, and broadcasts the message to all the connected clients.
6. Once a client receives the message, the message is displayed in the message field of the GUI of the client.
7. If a client tries to send a file, the file is received by the server and broadcasted to all the clients and the file can be viewed in the same folder as the file client.py

Result and Screenshots

Screenshots

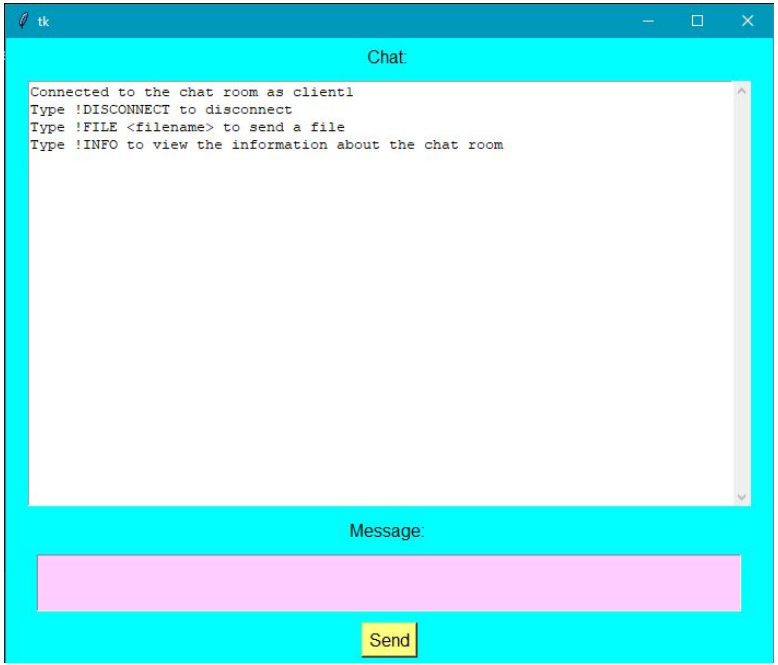
```
Server is starting...
Server is listening on 192.168.1.203.
Connected with ('192.168.1.203', 60703)
Nickname of the client is client1
Connected with ('192.168.1.203', 60733)
Nickname of the client is client2
client1: hello

client2: hello world!
```

Server starts,
Clients connects with the server
I.e. enter the chat room with nicknames client1 and client2 and some messages from them

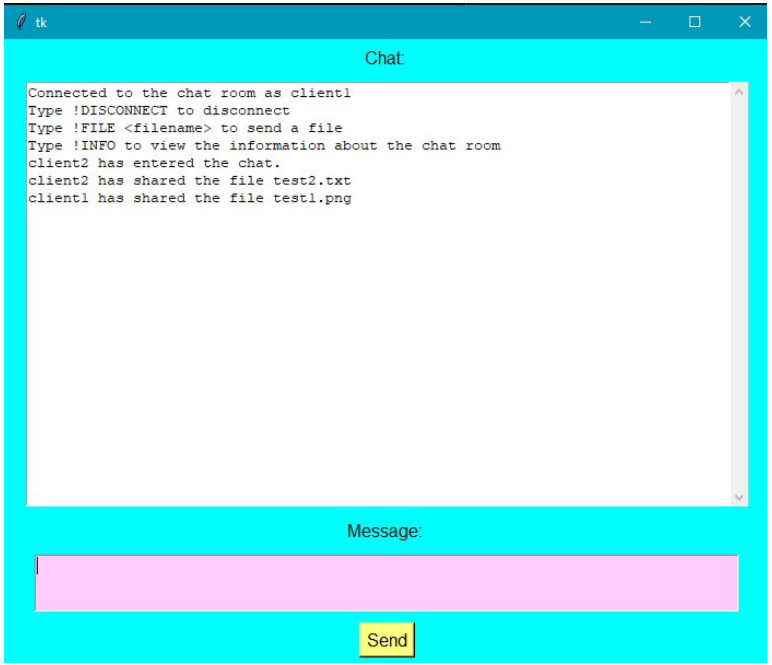


Window that prompts the user to enter a nickname to enter the chat room

Screenshots

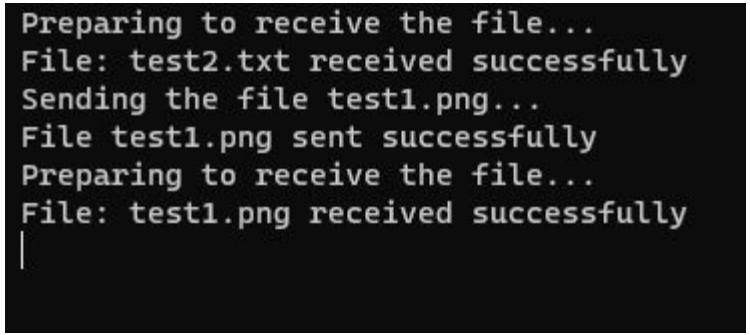


Initial chat window, as soon as connected to the server

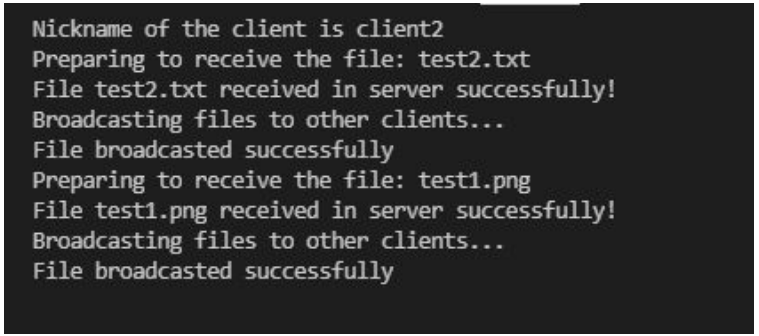


When a new client connects, and file sharing happens using the !FILE <filename> command

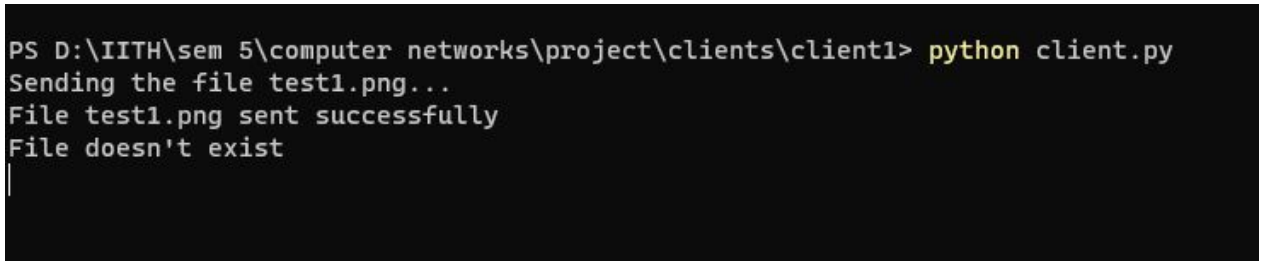
Screenshots



Client side terminal, when trying to send/receive files

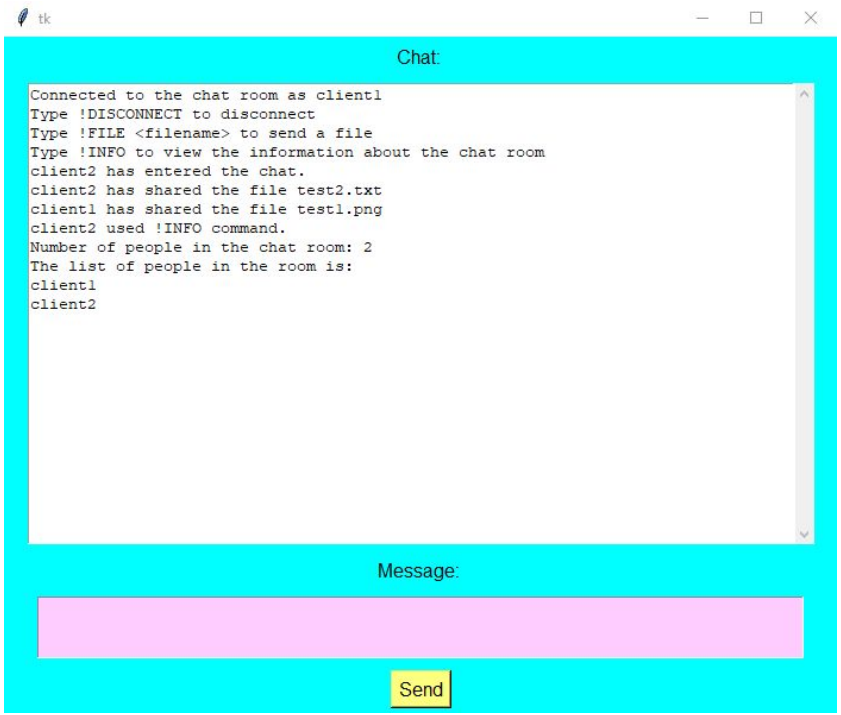


Server side terminal, when trying to receive file from client, and broadcast it to all clients

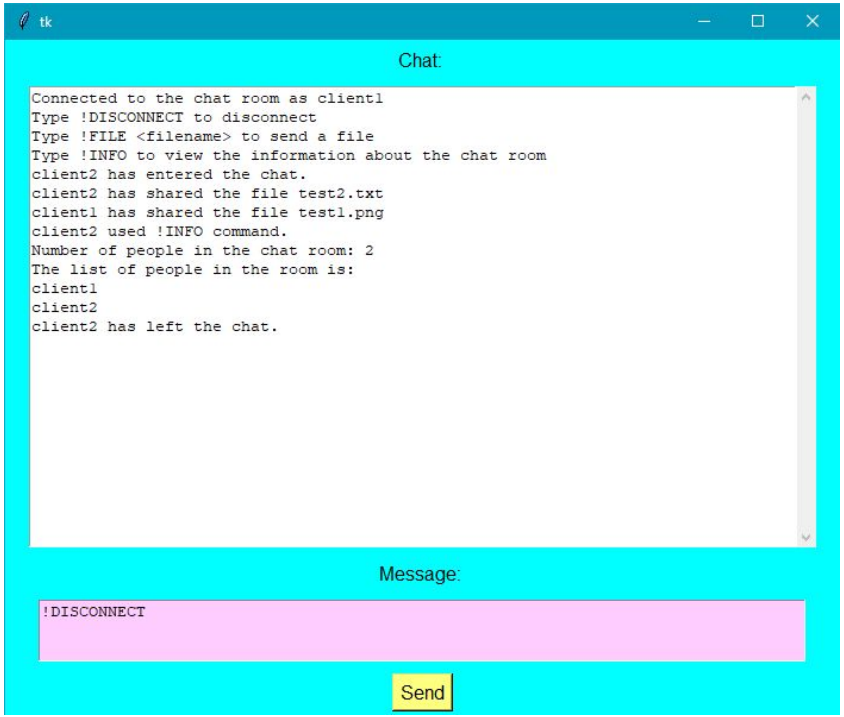


When trying to send a file that doesn't exist in the client side.

Screenshots



When someone uses the !INFO command



When client 2 uses !DISCONNECT and leaves the chat room

Future Improvements

Future Improvements

1. When the server broadcasts a file/message, it is sent to every client including the one that sent the file/message, that needs to be avoided.
2. Handling server crash has to be implemented.
3. Thought of adding a Tic-Tac-Toe game within the chat room, by using pygame, which a user can play Tic-Tac-Toe with someone else by entering “!TTT <nickname of the 2nd person>.
4. File preview can be made available in the GUI of the client, at least for image files.
5. Features like voice call, video call can be added.
6. GUI can be made better. More colorful, more interactive, etc.

References

References

1. https://docs.python.org/3/library/tkinter.html
2. https://www.geeksforgeeks.org/socket-programming-python/
3. https://realpython.com/python-sockets/
4. https://www.thepythoncode.com/article/send-receive-files-using-sockets-python
5. https://www.tutorialspoint.com/python/python_gui_programming.htm
6. https://www.youtube.com/watch?v=3QiPPX-KeSc

Conclusion

We learnt a lot of things from this project including but not limited to how socket programming works, how to implement socket programming, learnt a bit about tkinter GUI. Ran into some problems, but found a way out of it by referring online.