Communication Protocol Code

Server.py

The server is written in Python, and implements multiple client thread communication.
It's able to show 2 different types of debugging, human like (- Rocco is happy(1) with Eva) or dev mode (- Rocco says: 1A21), decided when first running the code.
The server is open on port 8090 by default, and when run the client should put the IP address of the computer on which the server is running, no need to edit the IP in the server.

There are 3 useful arrays, CHARACTER, IPs, EMOTIONS, each of them contains the related data useful to display a human language log to the users.

- CHARACTERS = ["All","Rocco","Eva","Lele","Carlotta","Peppe","Bianca","Cosimo"]
- IPs = [HOST,"","","","","","","",""]
- EMOTIONS = {"A": "idle", "B":"happy","C": "angry","D": "shocked","E": "sad","F": "relaxed", "G": "afraid","H": "cautious","I": "surprised","J": "annoyed","K": "embarrassed","L": "anxious"}

Inhibit variable, in order to decide whether a message should be forwarded immediately or after a delay of 5 seconds.

Then there is an array that will contain all the clients connected.
Row 29 and 30 are present to choose the debug mode, chosen from the user input.

Next is handle_client, a thread responsible to save and manage any client connected. For our purpose when a client sends a reaction, this is forwarded after 5 seconds, allowing each robot to show the emotion the same time the console shows the log.
When a GOD message (GX - X is for 1-E in hex) is sent, the robot, after performing the message audio, will send a GG (works like an ACK), and this is immediately forwarded to all other clients.
In each message sent is appended a \n to allow ESP32 to differentiate each message.
When a connection is lost, the client is removed from the list and connection is closed.

The other thread is console_input that will handle any input and forward them to each client. If the user enters quit, the program is terminated and all connections will be closed. When a GF is sent, the inhibit variable is set to FALSEin order to allow code to send messages without delay. Otherwise is set to TRUE

Last, is a while loop that will accept any incoming connection and start another thread of handle_client. Adding it to the list of connections.

Client.py

Along with the server a client is available to test and debug the server.
It just connects to the server (IP of the computer on which server running should be inserted) and then sends and receives messages to and from the server.