CODE ARCHITECTURE

GENERAL EXPLANATION

The application is a Python script that runs a server on a machine and listens for connections from clients.

SERVER

The server listens for connections on port 10000 of the machine's IP address and can handle multiple connections at once using the listen method of the server_socket object.

When a client connects, the server waits for commands from the client and executes them. The commands that the server understands are:

dos: This command is only available on Windows machines and allows the client to execute a command on the server using the command prompt.

powershell: This command is also only available on Windows machines and allows the client to execute a command on the server using PowerShell.

python --version: This command returns the version of Python installed on the server.

ping: This command allows the client to ping an IP address.

os: This command returns the operating system and version of the server.

name: This command returns the hostname of the server.

ip: This command returns the IP address of the server.

cpu: This command returns the percentage of CPU usage on the server.

ram: This command returns information about the memory usage on the server.

kill: This command ends the connection between the client and the server.

reset: This command closes all connections and causes the server to start listening for new connections again. (haven't done this because of difficulties)

disconnect: This command ends the connection between the client and the server.

The server also uses the psutil library to get information about the server's CPU and memory usage.

CLIENT

Meanwhile on the client side in which we also have the GUI we have a Python script that uses PyQt5 to create a GUI for a client that can connect to a server and send commands to it. The client uses the socket library to set up a connection to the server and send and receive messages.

The Client class is the client and has the following methods:

__init__: This method initializes the client with the host and port of the server, creates a socket object, and initializes an instance variable __thread to None.

connect: This method tries to set up a connection to the server using the host and port provided in the __init__ method. It returns -1 if the connection cannot be set up and 0 if it was successful.

connection: This method sets up a connection to the server using the host and port provided in the __init__ method.

communication: This method sends a message to the server and returns the response from the server.

The MainWindow class stands for the main window of the GUI and has the following attributes:

__host: A label that displays the text "Host".

__port: A label that displays the text "Port".

__newip: A label that displays the text "Add a new IP to the list:".

__addip: A push button that allows the user to add a new IP to a list of IPs.

__entercommand: A label that displays the text "Command:".

__response: A label that displays the text "Server response:".

__newiptext: A line edit that allows the user to enter a new IP to add to the list.

__entertext: A text edit that displays the response from the server.

__iptext: A combo box that displays a list of IPs that the user can choose from.

__porttext: A line edit that allows the user to enter the port to use to connect to the server.

__commandstext: A line edit that allows the user to enter a command to send to the server.

__filetext: A line edit that displays the file path of a file chosen by the user.

__savefile: A line edit that allows the user to enter a file name to use to save the file received from the server.

__addip: A push button that allows the user to add a new IP to the list.

__connectbutton: A push button that allows the user to connect to the server.

__send: A push button that allows the user to send a command to the server.

__filename: A push button that allows the user to choose a file to send to the server.

__filenames: A label that displays the text "Source file:".

The MainWindow class also has the following methods:

__init__: This method initializes the main window of the GUI and creates all the widgets listed above. It also sets up the layout of the widgets using a grid layout.

command: This method is called when the user clicks the __send button and sends the command entered in the __commandstext line edit to the server.

change: This method is called when the user clicks the __filename button and allows the user to choose a file to send to the server. It opens a file dialog and sets the file path chosen by the user to the __filetext line edit.

send: This method is called when the user clicks the __send button and sends a file to the server. It first sends the command "send" to the server, followed by the file name and the contents of the file.

receive: This method is called when the user clicks the __send button and receives a file from the server. It first sends the command "receive" to the server, followed by the file name, and then receives and saves the file received from the server.

__init__IP: This method initializes the list of IPs displayed in the __iptext combo box.

__addIP: This method is called when the user clicks the __addip button and adds a new IP to the list displayed in the __iptext combo box.

__add: This method is called when the user clicks the __addip button and adds a new IP to the list displayed in the __iptext combo box.

There are a lot of things that I couldn't do because I had issues with the logic and fitting everything in because whenever I tried adding the other missing functions the application would crash so I wanted to have something at least but without crashes. I put these documents with a bit of delay because I didn't know we were supposed to put them in our github. Thank you.