# Basic Network Scripting Reflection — Socket Programming & Port Scanner

For this assignment, I worked on creating both a Python server and client script, along with a basic port scanner. I started with almost no knowledge of Python socket programming, so I needed help at nearly every step. I used ChatGPT (AI) to guide me through the entire process, from writing the code to fixing errors and understanding the commands.

Step 1: Setting Up the Server and Client  
  
First, I created two Python files: server.py and client.py. ChatGPT explained that the server would wait for connections and the client would send a message once connected. I had trouble knowing where to save these files at first, but with help, I saved them both inside the same folder on my desktop: Network Midterm.  
[Insert Screenshot of the folder showing both files]

Step 2: Running the Server and Client  
  
After saving the files, I opened the Anaconda Prompt (in my specific environment, autosecurity) and navigated to the folder using the cd command. When I ran the server, it displayed:  
Server is listening for connections...  
and when I ran the client, I saw the confirmation that the message was sent and received. This part was fun because I saw the scripts talking to each other!  
[Insert Screenshot of both the server and client running successfully]

Step 3: Error Handling and Troubleshooting  
  
At one point, I ran into a major issue: my terminal wouldn’t recognize python. After a lot of trial and error, I realized I needed to reinstall Python because of a PATH variable problem. ChatGPT walked me through checking the PATH settings and reinstalling Python the correct way.  
[Insert Screenshot of the error and the corrected installation process]

Step 4: Writing and Running the Port Scanner  
  
Next, I created a third file called simple\_port\_scanner.py. ChatGPT gave me the code and explained how it worked — by looping through port numbers and trying to connect to see if they were open. I saved it and tried to run it, but at first, the terminal couldn’t find the file. The issue was that I accidentally typed the file name wrong. Once I corrected that, the script scanned both localhost and scanme.nmap.org and displayed the open and closed ports as expected.  
[Insert Screenshot of port scanner running successfully]

Extra Scan:  
  
I even tried scanning a custom IP address for more practice. ChatGPT explained how to change the target IP in the script, and I added that in and ran the scan successfully.  
[Insert Screenshot of extra scan result]

AI Tool Usage:  
  
Throughout this entire project, I used ChatGPT to help me:  
- Write the original server, client, and port scanner code.  
- Troubleshoot errors like 'No such file or directory' and 'python not found.'  
- Understand basic socket programming and scanning concepts.  
Here’s an example of a prompt I typed:  
"Can you give me step-by-step instructions for creating a server and client in Python using Anaconda?"  
I also asked:  
"Why do I see the error: can't open file — No such file or directory?"

Personal Insights:  
  
This assignment taught me that even when code looks simple, getting it to run can be a challenge if your environment isn’t set up right. I learned the hard way that PATH settings in Windows matter a lot when using Python. It also showed me the power of asking for help, using AI as a learning tool, and sticking with a problem until it’s solved.  
  
I now feel more confident opening terminals, navigating folders, and running Python scripts. I still don’t fully understand every piece of the code, but I can recognize the flow:  
- Server starts and waits.  
- Client connects and sends a message.  
- Port scanner loops through ports to find what’s open.  
  
I also learned the importance of error messages. Before, I would have just given up when I saw something like “No such file,” but now I know how to troubleshoot using directory checks, correct filenames, and Python environment setups.  
  
In the future, I hope to better understand how sockets work under the hood and start experimenting with more advanced features, like handling multiple clients and writing cleaner, more organized code.