Lab 12 TCP client and server in Go

Exercise 1: Building the TCP Echo Server

Goal: Create a server that listens for a connection, receives a message, and sends the same message back to the client

1. Project Setup:
-Create a new folder for this lab (e.g., go_tcp_echo)
-Inside the folder, create a new file named server.go

2. Code the Server:

-In server.go , write a program that uses the net.Listen function to listen for incoming TCP connections on port 8080

-Create an infinite for loop to continuously accept new connections using listener.Accept()

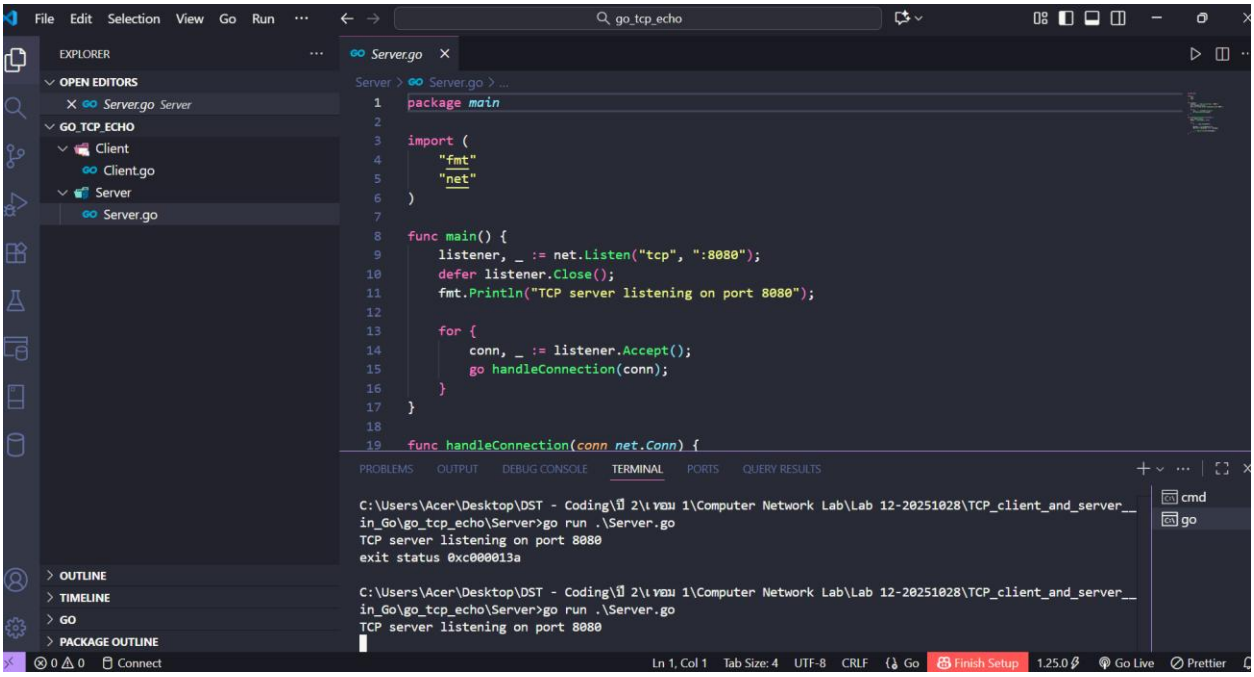
-For each new connection, launch a goroutine to handle it

-The goroutine should read data from the connection and then write the exact same data back to the connection. Remember to use defer conn.Close() in the goroutine

3. Run the Server:

-Open your terminal, navigate to the project folder, and run the server with: go run server.go

-The program should be running and waiting for a connection



Exercise 2: Building the TCP Echo Client

Goal: Create a client that connects to the server, sends a message, and prints the server's response

1. Project Setup:

-In the same folder, create a new file named client.go

2. Code the Client:

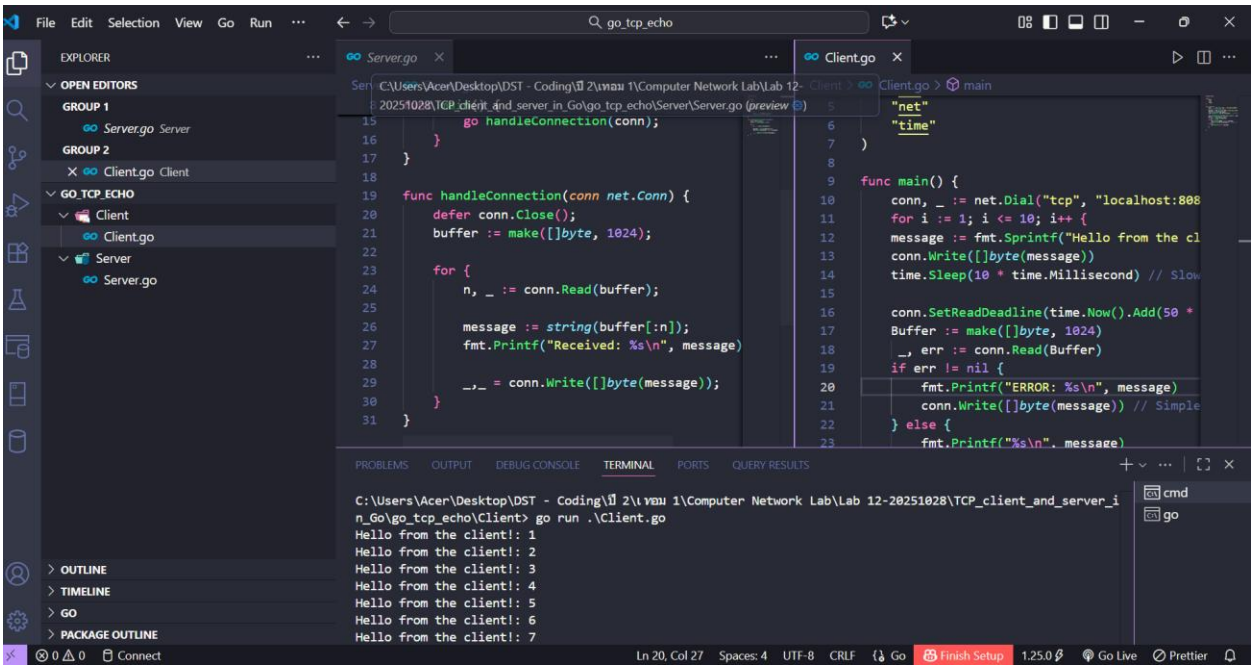
-In client.go client.go , write a program that uses net.Dial to connect to localhost:8080

-Once connected, write a message (e.g., "Hello from the client!") to the connection.
-Read the response from the server into a buffer.

-Print the server's response to the console. Remember to use defer conn.Close()

3. Run the Client:

-Open a second terminal window and run the client with: go run client.go



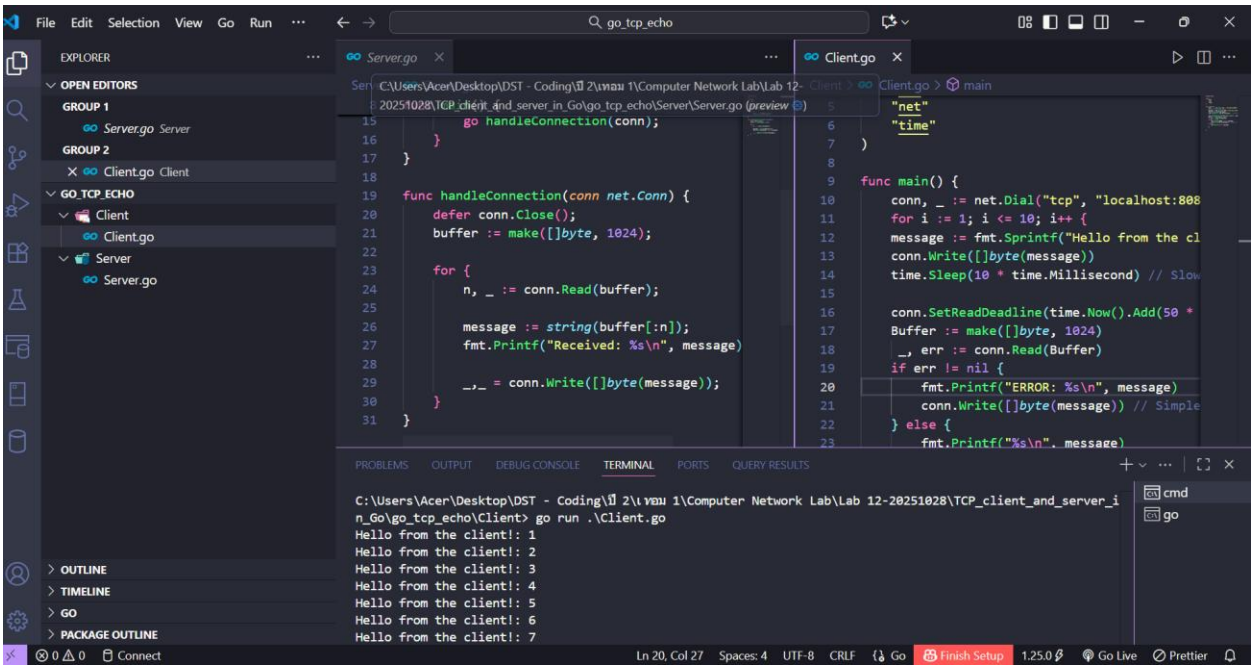
Exercise 3: Verification and Discussion

Goal: Verify that your client and server are working correctly and discuss the results

1. Check Output:

   What message did the client print in its terminal?
   Hello from the client!

นายศักดิ์สิทธิ์ จิตตะโสภี 6787077 Section 2



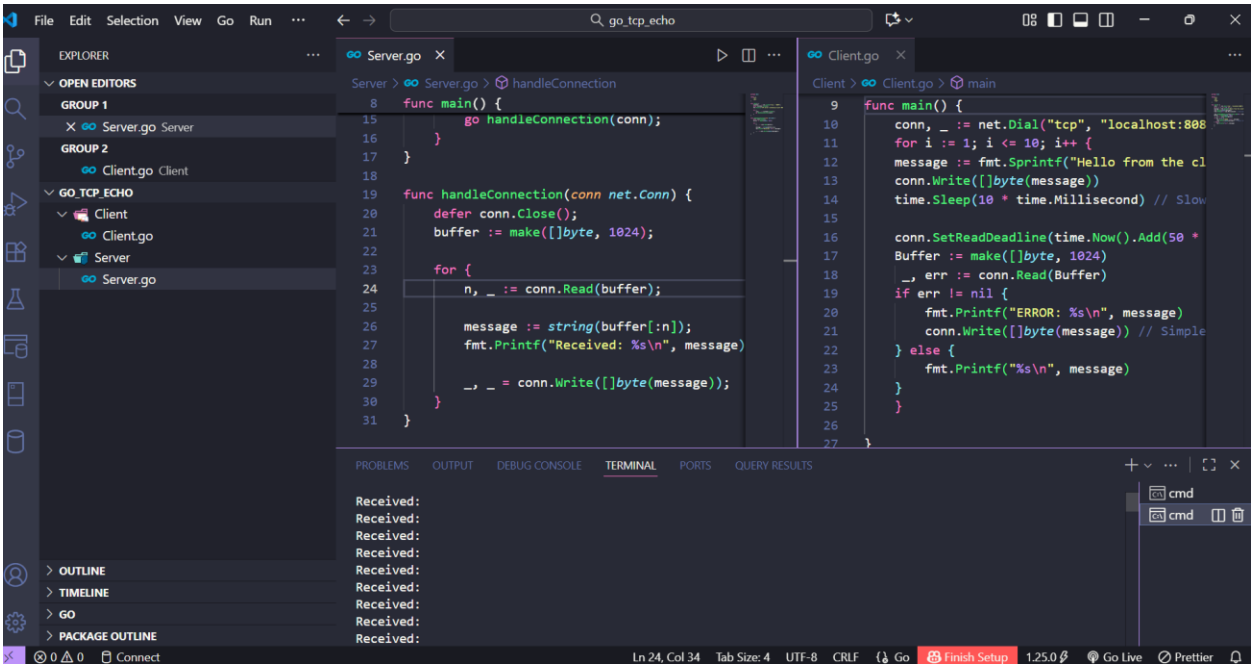
Did the server print anything? (Hint: You can add fmt.Println to the server's goroutine see what it receives.)

TCP server listening on port 8080

Received:

Received:

//infinite loop



2. Troubleshooting: If the client fails to connect, what might be the problem?
   (Hint: Is the server running? Is it on the correct port?)
   ถ้า client ไม่สามารถเชื่อมต่อ server ได้ อาจเกิดจากการเชื่อมต่อผิด port หรือ localhost ผิด หรือเกิดจากการที่ client run ก่อนที่ server จะ run.