Green University of Bangladesh
Department of Computer Science and Engineering(CSE)
Faculty of Sciences and Engineering
Semester: (Fall,Year:2024), B.Sc. in CSE (Day)

LAB REPORT NO #03
Course Title: Computer Networking Lab
Course Code: CSE-312 Section: 221_D16

Lab Experiment Name: Implementation of socket programming using threading.

Student Details

Name		ID
	Md. Moshiur Rahman	**221902324**

Lab Date : 19 / 10/ 2023
Submission Date : 11 / 11 / 2023
Course Teacher's Name : Maisha Muntaha

[For Teachers use only: Don't Write Anything inside this box]

Lab Report Status

Marks: …… **Signature:.....**
Comments:..... **Date:.....**

1. TITLE OF THE LAB EXPERIMENT

Socket programming is a fundamental aspect of network communication, enabling processes on different devices to communicate over a network. Threading is often used in socket programming to handle multiple client connections concurrently, improving the efficiency of the server.

In this lab, we implemented a basic client-server application where the server could handle up to five clients concurrently. The clients sent integer values and mathematical operators to the server, which then performed the requested operation and sent the result back to the clients.

2. OBJECTIVES/AIM

The objective of this lab was to implement a simple client-server application using socket programming in Java. Threading was employed to handle multiple client connections simultaneously, allowing the server to serve multiple clients concurrently. The server performed basic mathematical operations based on client requests, and the communication between the server and clients was facilitated through TCP sockets

3. PROCEDURE / ANALYSIS / DESIGN

3.1 Server :

The server was implemented in Java using the ServerSocket class to listen for incoming client connections. For each connected client, a new thread (ClientHandler) was created to handle the communication with that specific client. Threading was crucial to allow multiple clients to connect simultaneously without blocking the server.

3.2 Client :

The client was implemented as a separate Java application. It used the Socket class to connect to the server. The client could send integer values and mathematical operators to the server and receive the results. The client could continue sending requests until it decided to end the connection by sending the "ENDS" command.

4. IMPLEMENTATION

Server Implementation:

1. The server utilized a ServerSocket to listen for incoming connections.
2. For each client connection, a new thread (ClientHandler) was spawned to handle the communication with that client.
3. The ClientHandler class processed client requests, performed the specified mathematical operation, and sent the result back to the client.
4. The server could handle a maximum of 5 clients concurrently, and the server terminated when this limit was reached.

Client Implementation:

1. The client used a Socket to establish a connection with the server.
2. User input (integer values and mathematical operator) was sent to the server.
3. The client displayed the result received from the server.
4. The client could send multiple requests until the user entered "ENDS" to terminate the connection.

5. TEST RESULT / OUTPUT

Fig-1.1: Client getting response from Server Side.

Fig-1.2: Multiple-Client getting response from Server Side.

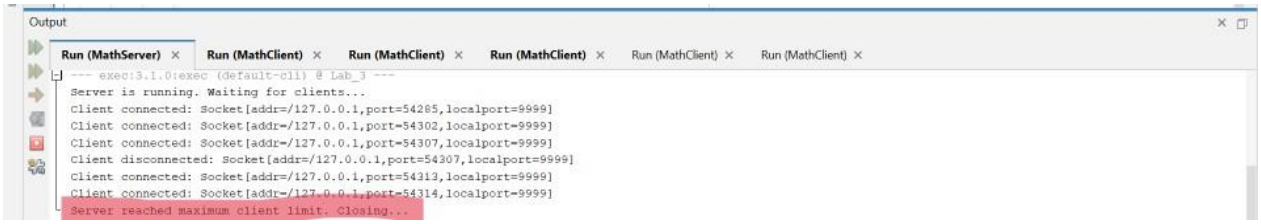

Fig-1.3: Server reached maximum client connection. [Max 5 Clients]

6. ANALYSIS AND DISCUSSION

The use of threading in socket programming significantly improved the efficiency of the server by allowing it to handle multiple clients concurrently. Without threading, the server would be limited to serving one client at a time, resulting in potential bottlenecks and decreased performance.

The implementation demonstrated the basic principles of socket programming, including socket creation, client-server communication, and the use of threading for concurrent client handling. However, this implementation is simplistic and lacks robust error handling and security features that would be necessary in a real-world scenario.

7. Conclusion :

The lab successfully implemented socket programming with threading to create a basic client-server application. The use of threading allowed the server to handle multiple clients concurrently, enhancing the overall efficiency of the system. This lab provided valuable insights into the practical application of socket programming and threading for network communication.