BTN415 Lab TCP/IP Single Client

Introduction to Sockets – TCP/IP Communication

In this lab, you will learn the step-by-step procedures and syntax required to implement an application that uses a socket based program.

LEARNING OUTCOMES

Upon successful completion of this lab, you will have demonstrated the ability to:

- Create a client application
- Create a server application
- Implement a reliable data communication using the TCP/IP standards and protocols

IN CLASS

In this lab we will enhance the capabilities of two applications: a client application, and a server application. The objective is to show how these two applications interact with one another under different scenarios. Note that you should use the object oriented socket programming source code that was discussed at the end of our last class as a starting point and all messages should be written to the ***print*** function so that they are displayed on both the std::cout as well as the std::ostream object.

Download/Clone the Visual Studio solution *"TCP IP Sockets – Single Client"* folder from the class Github and perform the following:

STEP #1

Update your code so that your client can keep sending messages until it sends the word "quit" in a message. The server should display each received message. After the "quit" message, both the client as well as the server applications should close.

STEP #2

Update your code so that your server will send the string "Received Message" to the client, confirming that it has received the user defined message. The client should output this received message on its own screen. Following, the client should be ready to send more messages until it sends a "quit" message. After this message, both the client as well as the server applications will close.

STEP #3

Update your code so that your server will not close after receiving a "quit" message, instead it will just start accepting new connections. In this way, you should be able to start and finish client applications without having to restart the server application.

Take Home

Change your client application so that it will ask for the file name of a simple text file. Then, it should send the contents of the file, character by character, to the server. At the server, your code should change so that it creates a file called temp.txt which should be filled with the contents coming from your client. The Server should continue to send "Received Message" after every packet it receives.

SUBMISSION REQUIREMENTS

Once you have completed your lab create and upload the following files:

- All your source files (*.h and *.cpp)
- The client_output.txt and server_output.txt files generated by the lab
- The input.txt and the temp.txt file used/created in the take home part of the lab.