# CSCI 367 - Computer Networks I

# Assignment 07 – TCP Transport Layer

# Client-Server –Load Balancing Relay Server Message Recorder

# Program Specification

Create a TCP client-server application that uses TCP/IP and the Sockets API.

* The load balancing relay server listens for client connections.
* The client connects to the load balancing relay server.
* The load balancing relay server sends the client the following message:
  + Would you like to send a message (y/n)?"
  + The client responds, y or n.
* The load balancing relay server should then acknowledge the client’s response.
* If the client responded with y, the client sends a message to the load balancing relay server.
* The load balancing relay server receives the client message and forwards the message to one of two data store servers. The load balancing relay server should toggle between the two data store servers.
* The data store server receives the message and saves the message to an ASCII text file.
  + The message written to the ASCII text file should be prepended with the following two fields:
    - Date-Time Stamp
    - Relay Server’s IP address
* After recording the message information, the data store server sends back a status message represented as an ASCII text string to the load balancing relay server:
  + SUCCESS
  + FAIL
* The load balancing relay server forwards the status message to the client application:
* The client should then acknowledge the load balancing relay server’s status message.
* The load balancing relay server should prompt the user to see if he/she wants to send another message:
  + "Would you like to send a message (y/n)?"
* Since two data store servers are running
  + The data store server’s port number must be a command-line parameter.
  + The load balancing relay server needs two command-line parameters for each of the two data store server port numbers.

# Program Execution

Compile your source code files.

* If your source code has no syntax errors, an executable file is produced.
* Execute the program and check the accuracy of the program outputs.
* Below is a sample run showing the server and client output, along with the ASCII text file content.

Client

Graphical user interface, text, application, email

Description automatically generated

Relay Server

Graphical user interface, text, application, email

Description automatically generated

Data Store Server

Graphical user interface, text, application, email

Description automatically generated

ASCII Text File

Graphical user interface, application, Word

Description automatically generated

# File Header

At the top of the source code files, add the following file header:

 /\*

# Name:

# Description:

# Date:

# Specification:

\*/

# Submission

1. Upload your source code and header ﬁles to Canvas.
2. For this lab, Canvas has been conﬁgured to permit only files that end with the .c and .h file extensions.