COMP 7005

Assignment 2

Design

Aditya Singh Attri

A01276335

Oct 6th, 2024

# Purpose

This program implements a client-server model using TCP sockets. The server listens for client connections, processes files sent by the client, counts the number of alphabetic characters in the file, and sends the result back to the client. The client connects to the server, sends the file, and receives the response.

# Data Types

## Client-Side

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| server\_ip | string | The IP address of the server to connect to |
| server\_port | int | The port number of the server to connect to |
| file\_path | string | The path of the file to send to the server |
| file\_data | binary | Data read from the file to be sent to the server |
| response | string | Response received from the server |

## Server-Side

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| server\_socket | socket | The server’s socket to accept connections |
| client\_socket | socket | The socket for each client connection |
| client\_address | tuple | The IP address and port of the connected client |
| file\_data | binary | Data received from the client |
| response | string | Response sent to the client |

# Pseudocode

## Client-Side

**send\_file\_to\_server**

### Parameters

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| server\_ip | string | IP address of the server |
| server\_port | int | Port number of the server |
| file\_path | string | The file path of the file to be sent |

### Pseudocode

1. Create a TCP socket.
2. Connect to the server using the server IP and port.
3. Open the specified file and read its contents.
4. Send the file data to the server.
5. Receive the response from the server.
6. Print the response received from the server.
7. Close the connection.

## Server-Side

**handle\_client**

### Parameters

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| connection | socket | The socket representing the client connection |
| address | tuple | The IP and the port of the connected client |

### Pseudocode

1. Set a timeout for receiving data.
2. Receive the file data from the client.
3. If no data is received, send a message indicating the file is empty.
4. If data is received, count the number of alphabetic characters in the file.
5. Send the result back to the client.
6. Close the client connection.

**start\_server**

### Parameters

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| port | int | The port number on which the server will listen |

### Pseudocode

1. Create a TCP socket and bind it to the specified port.
2. Listen for incoming client connections.
3. Accept incoming connections and handle each client in a new thread.
4. Continuously wait for new client connections.