**Client.cpp**

#include <iostream>

#include <fstream>

#include <cstring>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <unistd.h>

#include "Logger.h"

// Constants

const int PORT = 8080;

const int BUFFER\_SIZE = 1024;

// Function to Send a File

void sendFile(int socket, const std::string& filePath, Logger& logger) {

std::ifstream file(filePath, std::ios::binary);

if (!file) {

logger.log("Error opening file: " + filePath);

return;

}

char buffer[BUFFER\_SIZE];

logger.log("Starting file transfer: " + filePath);

send(socket, "FILE\_START", strlen("FILE\_START"), 0);

while (file.read(buffer, BUFFER\_SIZE)) {

send(socket, buffer, file.gcount(), 0);

}

send(socket, buffer, file.gcount(), 0); // Send remaining bytes

send(socket, "FILE\_END", strlen("FILE\_END"), 0);

logger.log("File sent successfully: " + filePath);

}

// Function to Send a Message

void sendMessage(int socket, const std::string& message, Logger& logger) {

send(socket, message.c\_str(), message.size(), 0);

logger.log("Message sent: " + message);

}

// Main Function

int main() {

Logger logger("log.txt");

int socketFd;

struct sockaddr\_in serverAddr;

std::string input;

// Create socket

socketFd = socket(AF\_INET, SOCK\_STREAM, 0);

if (socketFd < 0) {

logger.log("Socket creation error.");

return -1;

}

logger.log("Socket created successfully.");

// Connect to server

serverAddr.sin\_family = AF\_INET;

serverAddr.sin\_port = htons(PORT);

if (inet\_pton(AF\_INET, "172.20.0.45", &serverAddr.sin\_addr) <= 0) {

logger.log("Invalid address/Address not supported.");

return -1;

}

if (connect(socketFd, (struct sockaddr\*)&serverAddr, sizeof(serverAddr)) < 0) {

logger.log("Connection error.");

return -1;

}

logger.log("Connected to server.");

// Loop for sending messages or files

while (true) {

std::cout << "Enter 'file' to send a file, 'message' to send a message, or 'exit' to quit: ";

std::cin >> input;

std::cin.ignore(); // Ignore the newline character after input

if (input == "file") {

std::string filePath;

std::cout << "Enter file path: ";

std::getline(std::cin, filePath);

logger.log("User chose to send a file.");

sendFile(socketFd, filePath, logger);

} else if (input == "message") {

std::string message;

std::cout << "Enter message: ";

std::getline(std::cin, message);

logger.log("User chose to send a message.");

sendMessage(socketFd, message, logger);

} else if (input == "exit") {

logger.log("User chose to exit.");

std::cout << "Exiting..." << std::endl;

break;

} else {

logger.log("Invalid option chosen.");

std::cerr << "Invalid option." << std::endl;

}

}

close(socketFd);

logger.log("Socket closed, client exited.");

    return 0;

}