```c
/*
 * Name: Rusho Binnabi
 * Date: 04/29/2022
 * Course Code and Title: ICSI 333 - System Fundamentals
 * Semester: Spring 2022
 * TA Name: Sourav Dutta
 * Student ID: 001427521
 */

/* The work for this assignment was divided between me and my partner/teammate Rockwon Seo.
 * I will be doing either the client side or server side of the program depending on which side my partner decides to work on.
 * We collaborated on this numerous times over email.
 * We both tried to create our own server and or client first then met and exchanged files via email and we helped each other improve it so to speak.
 * The testing process was a similar process where we tried to test it on our own first and then met and exchanged files via email and we helped each other improve it in a sense.
 */

/*
 * Unfornately, we tried the best we could but we unfortunately couldn't get the client working.
 */

#include <netdb.h> // includes the netdb library for use in this program.
#include <stdio.h> // includes the stdio library for use in this program.
#include <stdlib.h> // includes the stdlib library for use in this program.
#include <string.h> // includes the string library for use in this program.
#include <sys/socket.h> // includes the socket library from sys for use in this program.

#define MAX 80 // defines a constant variable called MAX with a value of 80 for use in this program.
#define PORT 8000 // defines a constant variable called PORT with a value of 8000 for use in this program.
#define SA struct sockaddr // defines a constant structure sockaddr variable called SA for use in this program.

/* this program will act as the client side of the web server. */
/* the client will run in an infinite loop taking requests from the user and will send that request to the server for processing. */

void chatFunction(int sockfd) { /* this chatFuntion() function takes one integer argument called sockfd which is the file descriptor name of the socket that will be sent over to the server for processing. */
                                /* this chatFunction() function will be used to send requests from the user for the contents of a html file to the server and will be used to communicate to the server with. */

    char buffer[MAX]; // creates a char array called buffer which has a size of MAX.
    int number; // creates an integer variable called number and wukk be used in this program.

    for (;;) { // the code inside this for loop will run infinitely because it will be waiting for sending requests from the user to the server for processing.

        bzero(buffer, sizeof(buffer)); // using the bzero() function, it erases the data in bytes of data inside buffer using the size of buffer.
        printf("Enter request: "); // prompts the user for a request which will be sent to the server for processing.
        number = 0; // initializes number to 0.

        while ((buffer[number++] = getchar()) != '\n') // the code inside the while loop will run as long as the value inside buffer
                                                       // which has the value of the incremented value of number using the getchar() function
                                                       // and as long as that value isn't equal to a terminating zero.
            ;

        write(sockfd, buffer, sizeof(buffer)); // using the write() function, it writes the number of bytes of data from the buffer using the size of the buffer into the file that's within the file descriptor sockfd.
        bzero(buffer, sizeof(buffer)); // using the bzero() function, it erases the data in bytes of data from the buffer using the size of buffer.
        read(sockfd, buffer, sizeof(buffer)); // using the read() function, it reads the number of bytes of data from the file that's within the file descriptor sockfd into the buffer using the size of the buffer.

        printf("From Server: %s", buffer); // shows the data inside buffer that was receieved from the server.

        if ((strncmp(buffer, "exit", 4)) == 0) { // using the strncmp() function, if the values of buffer, the string "exit" and 4 are equal to 0, then the code inside the if statement will run.
            printf("Client is exitting...\n"); // tells the user that the client is exitting.
            break; // breaks out of the while loop.
        } // the end of the if statement.
    } // the end of the while loop.
}

int main() { /* this int main() function has the code needed for my program to run. */


    int sockfd, connfd; // creates two integer variables called sockfd and connfd for use in this program.
    struct sockaddr_in servaddr, client; // creates a structure sockaddr_in variable called servaddr and a variable called client of the same type.

    sockfd = socket(AF_INET, SOCK_STREAM, 0); // using the socket() function, sockfd will have the value of AF_INET, SOCK_STREAM, and a 0.

    if (sockfd == -1) { // if the value inside sockfd is equal to -1, then the code inside the if statement will run.
        printf("The socket failed to be created...\n"); // tells the user that the socket failed to be created.
        exit(0); // using the exit() function, it exits the program with a value of 0 which means the program ended succesfully.
    } // the end of the if statement.

    else { // otherwise, the code inside the else statement will be run.
        printf("The socket was successfully created...\n"); // tells the user that the socket was successfully created.
        bzero(&servaddr, sizeof(servaddr)); // using the bzero() function, it erases the data at the address location of servaddr using the size of servaddr.

        servaddr.sin_family = AF_INET; // using sin_family, it assigns the value of AF_INET to servaddr.
        servaddr.sin_addr.s_addr = inet_addr("127.0.0.1"); // using sin_addr.s_addr, and using the inet_addr() function, it assigns an ip address to servaddr.
        servaddr.sin_port = htons(PORT); // using sin_port, and using the htons() function, it assigns the value of PORT to servaddr.
    } // the end of the else statement.

    if (connect(sockfd, (SA*)&servaddr, sizeof(servaddr)) != 0) { // using the connect() function, if the values inside sockfd, the address of servaddr using the size of servaddr is not equal to 0, then the code i
        printf("The connection with the server has failed...\n"); // tells the user that the connection to the server has failed.
        exit(0); // using the exit() function, it exits the program with a value of 0 which means the progeam ended successfully.
    } // the end of the if statamenet.

    else { // otherwise, the code inside the else statement will run.
        printf("The client has connected to the server...\n"); // tells the user that the client was able to connect to the server.
    } // the end of the else statement.

    chatFunction(sockfd); // calls the chatFunction() function using sockfd as it's argument.

    close(sockfd); // using the close() function, it closes sockfd.

    return 0; // returns a 0 which means the program ran succesfully without any errors.
} // the end of the int main() function.
```