The client code

**#include** <string.h>

**#include** <stdio.h>

**#include** <arpa/inet.h>

**#include** <sys/socket.h>

**#include** <unistd.h>

int **main**(int argc, char **const**\* argv**[]**)

{

**const** int port = 3400;

**const** char \*host\_service = "127.0.0.1";

int socket\_sock\_cn = 0, value\_read, client\_instance;

**struct** sockaddr\_in serv\_addr;

char\* hello = "This is the message from the client";

char buffer[1024] = { 0 };

**if** ((socket\_sock\_cn = **socket**(AF\_INET, SOCK\_STREAM, 0)) **<** 0) {

**printf**("\n Could not create the socket connection\n");

**return** -1;

}

serv\_addr**.**sin\_family = AF\_INET;

serv\_addr**.**sin\_port = **htons**(port);

// *Convert IPv4 and IPv6 addresses from text to binary*

**if** (**inet\_pton**(AF\_INET, host\_service, &serv\_addr**.**sin\_addr)

**<=** 0) {

**printf**(

"\n The address format passed is not supported \n");

**return** -1;

}

**if** ((client\_instance

= **connect**(socket\_sock\_cn, (**struct** sockaddr\*)&serv\_addr,

**sizeof**(serv\_addr)))

**<** 0) {

**printf**("\nConnection Failed \n");

**return** -1;

}

**send**(socket\_sock\_cn , hello, **strlen**(hello), 0);

**printf**("Message from the client has been sent to the server\n");

value\_read = **read**(socket\_sock\_cn, buffer, 1024);

**printf**("%s\n", buffer);

// *closing the connected socket*

**close**(client\_instance);

// *return an integer value as the function is of type integer*

**return** 5;

}

The server code

**#include** <stdio.h>

**#include** <stdlib.h>

**#include** <string.h>

**#include** <netinet/in.h>

**#include** <sys/socket.h>

**#include** <unistd.h>

int **main**(int argc, char **const** \*argv**[]**)

{

**const** int port = 3400;

int server\_intance, new\_socket, value\_read;

**struct** sockaddr\_in address;

int opt = 1;

int addrlen = **sizeof**(address);

char buffer[1024] = {0};

char \*hello = "Hello from the server :)";

// *Creating socket file*

**if** ((server\_intance = **socket**(AF\_INET, SOCK\_STREAM, 0)) **<** 0)

{

**perror**("socket failed");

**exit**(EXIT\_FAILURE);

}

// *Forcefully attaching socket to the port 3400*

**if** (**setsockopt**(server\_intance, SOL\_SOCKET,

SO\_REUSEADDR | SO\_REUSEPORT, &opt,

**sizeof**(opt)))

{

**perror**("setsockopt");

**exit**(EXIT\_FAILURE);

}

address**.**sin\_family = AF\_INET;

address**.**sin\_addr**.**s\_addr = INADDR\_ANY;

address**.**sin\_port = **htons**(port);

// *Forcefully attaching socket to the port 3400*

**if** (**bind**(server\_intance, (**struct** sockaddr \*)&address,

**sizeof**(address)) **<** 0)

{

**perror**("bind failed");

**exit**(EXIT\_FAILURE);

}

**if** (**listen**(server\_intance, 3) **<** 0)

{

**perror**("listen");

**exit**(EXIT\_FAILURE);

}

**if** ((new\_socket = **accept**(server\_intance, (**struct** sockaddr \*)&address,

(socklen\_t \*)&addrlen)) **<** 0)

{

**perror**("accept");

**exit**(EXIT\_FAILURE);

}

value\_read = **read**(new\_socket, buffer, 1024);

**printf**("%s\n", buffer);

**send**(new\_socket, hello, **strlen**(hello), 0);

**printf**("Hello message sent\n");

// *closing the connected socket*

**close**(new\_socket);

// *closing the listening socket*

**shutdown**(server\_intance, SHUT\_RDWR);

// *return an integer value as the function is of type integer*

**return** 5;

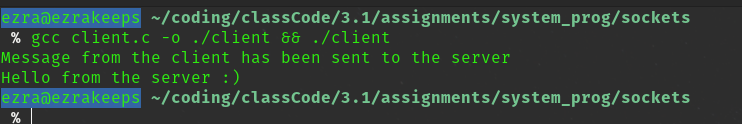
}

Code Results

Starting the server



Send data from the client



Response from server

