

DISTRIBUTED SYSTEMS - ASSIGNMENT - 5

Implement given extensions to the Client Server Programming.

1. Extend your echo Client Server message passing application to chat application.

- Client and Server are able to send the message to each other until one of them quits or terminates.

CODE=>

SERVER SIDE

```
//U19CS009
//Brijesh Rohit

#include<stdio.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<unistd.h>
#include<netinet/in.h>
#include<string.h>

int main()
{
    //creating a socket for server
    int server_socket, client_socket;
    server_socket = socket(AF_INET, SOCK_STREAM, 0);
    if (server_socket < 0)
    {
        perror("--> Server error !!!\n");
        exit(1);
    }
    //define server_address and client_address
    struct sockaddr_in server_address, client_address;
    server_address.sin_family = AF_INET;
    server_address.sin_port = htons(9002); //passing port number 9002
    server_address.sin_addr.s_addr = INADDR_ANY; //specifying local machine
    address
    //binding socket with specific IP and port number
    bind(server_socket, (struct sockaddr *)&server_address,
    sizeof(server_address));
    printf("Bind to the port number : 9002");
    //listening to connection
    listen(server_socket, 4);
    printf("\nListening.....\n");
    //accept a connection
    int size_client_addr = sizeof(client_address);
    client_socket = accept(server_socket, (struct sockaddr *)&client_address,
    &size_client_addr);
```

```

    if (client_socket < 0)
    {
        printf("Error in accepting request!!\n");
        exit(1);
    }
    printf(".....Client is connected.....\n");
    //recive data from client
    while (1)
    {
        char response_client[256];
        recv(client_socket, &response_client, sizeof(response_client), 0);
        //last parameter is optional so putting 0
        printf("Client : ");
        printf("%s", response_client);
        printf("\n");
        printf("Server : ");
        char send_server[256];
        scanf("%s", send_server);
        send(client_socket, send_server, sizeof(send_server), 0);
        if (strncmp(send_server, "bye", 3) == 0)
            break;
    }
    //close the socket
    close(client_socket);
    close(server_socket);
    printf("\n Client disconnected.....\n");
    return 0;
}

```

CLIENT SIDE

```

//U19CS009
//Brijesh Rohit

#include<stdio.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<unistd.h>
#include<netinet/in.h>
#include<string.h>

int main()
{
    //Creating a socket for client
    int no_socket;
    no_socket = socket(AF_INET, SOCK_STREAM, 0);
    //specifying address for client socket
    struct sockaddr_in client_addr;
    client_addr.sin_family = AF_INET;
    client_addr.sin_port = htons(9002); //passing port number 9002
    client_addr.sin_addr.s_addr = INADDR_ANY; //specifying local machine address
    (equivalent to 0.0.0.0)
}

```

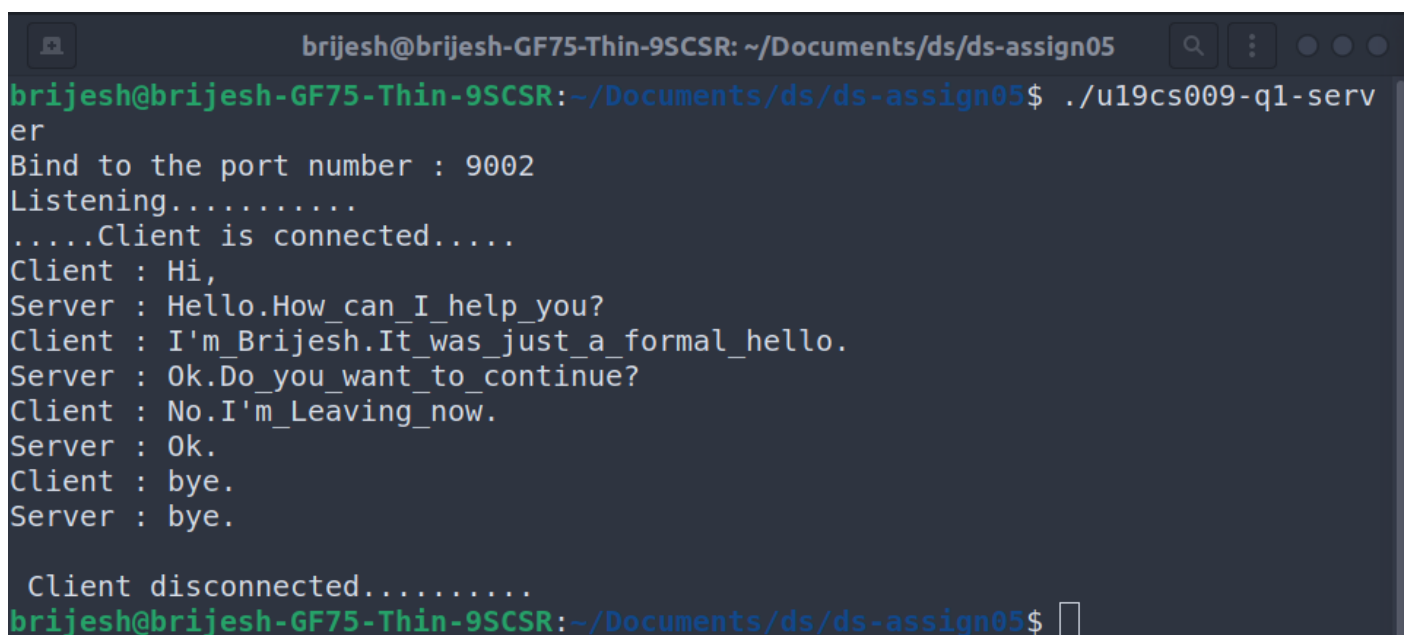
```

    int connection_status = connect(no_socket, (struct sockaddr *)&client_addr,
sizeof(client_addr)); // 0 OK -1 error
//Checking whether there is an error in connection
    if (connection_status < 0)
    {
        perror("--->There was an error making connection with the remote
socket\n\n");
        exit(1);
    }
    printf("Connected to the server.....\n");
//send data to server
    while (1)
    {
        char send_client[256];
        printf("Client : ");
        scanf("%s", send_client);
        send(no_socket, send_client, sizeof(send_client), 0);
        char recv_client[256];
        recv(no_socket, &recv_client, sizeof(recv_client), 0);
        printf("Server : %s", recv_client);
        printf("\n");
        if (strncmp(send_client, "bye", 3) == 0)
            break;
    }
//close the connection
    close(no_socket);
    printf("\n Disconnected from server...\n");
    return 0;
}

```

OUTPUT=>

SERVER SIDE



A terminal window titled 'brijesh@brijesh-GF75-Thin-9SCSR: ~/Documents/ds/ds-assign05' showing the server's output. The prompt is 'brijesh@brijesh-GF75-Thin-9SCSR:~/Documents/ds/ds-assign05\$' and the command executed is './u19cs009-q1-serv er'. The output shows the server binding to port 9002, listening, and then receiving a connection from a client. The client sends 'Hi', the server responds 'Hello.How can I help you?'. The client sends 'I'm Brijesh.It was just a formal_hello.', the server responds 'Ok.Do you want to continue?'. The client sends 'No.I'm Leaving now.', the server responds 'Ok.'. The client sends 'bye.', the server responds 'bye.'. Finally, the client disconnects, and the server outputs 'Client disconnected.....'.

```

brijesh@brijesh-GF75-Thin-9SCSR: ~/Documents/ds/ds-assign05
brijesh@brijesh-GF75-Thin-9SCSR:~/Documents/ds/ds-assign05$ ./u19cs009-q1-serv
er
Bind to the port number : 9002
Listening.....
.....Client is connected.....
Client : Hi,
Server : Hello.How can I help you?
Client : I'm Brijesh.It was just a formal_hello.
Server : Ok.Do you want to continue?
Client : No.I'm Leaving now.
Server : Ok.
Client : bye.
Server : bye.

Client disconnected.....
brijesh@brijesh-GF75-Thin-9SCSR:~/Documents/ds/ds-assign05$ 

```

CLIENT SIDE

```
brijesh@brijesh-GF75-Thin-9SCSR: ~/Documents/ds/ds-assign05
brijesh@brijesh-GF75-Thin-9SCSR:~/Documents/ds/ds-assign05$ ./u19cs009-q1-client
Connected to the server.....
Client : Hi,
Server : Hello.How can I help you?
Client : I'm Brijesh.It was just a formal hello.
Server : Ok.Do you want to continue?
Client : No.I'm Leaving now.
Server : Ok.
Client : bye.
Server : bye.

Disconnected from server...
brijesh@brijesh-GF75-Thin-9SCSR:~/Documents/ds/ds-assign05$
```

2. Using the Client-Server communication mechanism get the load status of other nodes in your network (identify the states of other nodes in the system – Overload, Moderate, Lightly).

- Implement the Client-Server model. Run the client and server instance on same machine and pass the message from client to server or server to client
- Get the CPU load of the client or server and state that either it is under loaded or overloaded.

The client server communication mechanism has the limitation that it only handles one connection at a time and then terminates. A real-world server should run indefinitely and should have the capability of handling a number of simultaneous connections, each in its own process.

CODE=>

SERVER SIDE

```
//U19CS009
//Brijesh Rohit

#include<stdio.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<unistd.h>
#include<netinet/in.h>
#include<string.h>

int main() {
//creating a socket for server
int server_socket, client_socket;
server_socket = socket(AF_INET, SOCK_STREAM, 0);
if (server_socket < 0)
{
perror("--> Server error !!!\n");
exit(1);
}
//define server_address and client_address
```

```

    struct sockaddr_in server_address, client_address;
    server_address.sin_family = AF_INET;
    server_address.sin_port = htons(9002); //passing port number 9002
    server_address.sin_addr.s_addr = INADDR_ANY; //specifying local machine
address
//binding socket with specific IP and port number
    bind(server_socket, (struct sockaddr *)&server_address,
sizeof(server_address));
    printf("Bind to the port number : 9002");
//listening to connection
    listen(server_socket, 4);
    printf("\nListening.....\n");
//accept a connection
    int size_client_addr = sizeof(client_address);
    client_socket = accept(server_socket, (struct sockaddr *)&client_address,
&size_client_addr);
    if (client_socket < 0)
    {
        printf("Error in accepting request!!\n");
        exit(1);
    }
    printf(".....Client is connected.....\n");
    char response_client[256];
    recv(client_socket, &response_client, sizeof(response_client), 0);
//last parameter is optional so putting 0
    printf("Client message : ");
    printf("%s", response_client);
    printf("\n");
//close the socket
    close(client_socket);
    close(server_socket);
    printf("\n Client disconnected.....\n");
    return 0;
}

```

CLIENT SIDE

```

//U19CS009
//Brijesh Rohit

#include<stdio.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<unistd.h>
#include<netinet/in.h>
#include<string.h>

int main()
{
//Creating a socket for client
    int no_socket;
    no_socket = socket(AF_INET, SOCK_STREAM, 0);

```

```

//specifying address for client socket
struct sockaddr_in client_addr;
client_addr.sin_family = AF_INET;
client_addr.sin_port = htons(9002); //passing port number 9002
client_addr.sin_addr.s_addr = INADDR_ANY; //specifying local machine address
(equivalent to 0.0.0.0)
int connection_status = connect(no_socket, (struct sockaddr *)&client_addr,
sizeof(client_addr)); // 0 OK -1 error
//Checking whether there is an error in connection
if (connection_status < 0)
{
    perror("--->There was an error making connection with the remote
socket\n\n");
    exit(1);
}
printf("Connected to the server.....\n");
printf("Client information : ");
FILE *client;
char send_client[256];
client = popen("./u19cs009-ds-assign05-q2.sh", "r");
if (client != NULL)
{
    while (1)
    {
        char *line;
        line = fgets(send_client, sizeof(send_client), client);
        if (line == NULL)
            break;
        printf("%s", line); /* line includes '\n' */
    }
    pclose(client);
}
//send data to server
send(no_socket, send_client, sizeof(send_client), 0);
//close the connection
close(no_socket);
printf("\n Disconnected from server...\n");
return 0;
}

```

OUTPUT=>

SERVER SIDE

```
brijesh@brijesh-GF75-Thin-9SCSR: ~/Documents/ds/ds-assign05
brijesh@brijesh-GF75-Thin-9SCSR:~/Documents/ds/ds-assign05$ ./u19cs009-q2-server
Bind to the port number : 9002
Listening.....
.....Client is connected.....
Client message :

Client disconnected.....
brijesh@brijesh-GF75-Thin-9SCSR:~/Documents/ds/ds-assign05$ ./u19cs009-q2-server
Bind to the port number : 9002
Listening.....
.....Client is connected.....
Client message : Lightly Loaded

Client disconnected.....
brijesh@brijesh-GF75-Thin-9SCSR:~/Documents/ds/ds-assign05$
```

CLIENT SIDE

```
brijesh@brijesh-GF75-Thin-9SCSR: ~/Documents/ds/ds-assign05
brijesh@brijesh-GF75-Thin-9SCSR:~/Documents/ds/ds-assign05$ ls -l | tail -5
-rw-rw-r-- 1 brijesh brijesh 366 Feb 15 22:29 u19cs009-ds-assign05-q2.sh
-rwxrwxr-x 1 brijesh brijesh 16544 Feb 15 22:43 u19cs009-q1-client
-rwxrwxr-x 1 brijesh brijesh 16632 Feb 15 22:43 u19cs009-q1-server
-rwxrwxr-x 1 brijesh brijesh 16480 Feb 15 22:40 u19cs009-q2-client
-rwxrwxr-x 1 brijesh brijesh 16496 Feb 15 22:40 u19cs009-q2-server
brijesh@brijesh-GF75-Thin-9SCSR:~/Documents/ds/ds-assign05$ ./u19cs009-q2-client
Connected to the server.....
sh: 1: ./u19cs009-ds-assign05-q2.sh: Permission denied
Client information :
Disconnected from server...
brijesh@brijesh-GF75-Thin-9SCSR:~/Documents/ds/ds-assign05$ chmod +x u19cs009-ds-assign05-q2.sh
brijesh@brijesh-GF75-Thin-9SCSR:~/Documents/ds/ds-assign05$ ./u19cs009-q2-client
Connected to the server.....
Client information : CPU Usage: 2.1%
Lightly Loaded

Disconnected from server...
brijesh@brijesh-GF75-Thin-9SCSR:~/Documents/ds/ds-assign05$
```