

18CSC302J (Computer Networks Lab)

Lab session 3 - Simple TCP /IP Client -Server Communication

Name :- Puneet Sharma

Reg. No. :- RA1911003010331

Class :-CSE F1

TCP SERVER: SERVER CODE:

```
#include <stdio.h>
#include <netdb.h>
#include <netinet/in.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/socket.h>
#include <sys/types.h>
#define MAX 80
#define PORT 8084
#define SA struct sockaddr
void func(int sockfd)
{
    char buff[MAX];
    int n;
    for (;;) {
        bzero(buff, MAX);
        read(sockfd, buff, sizeof(buff));
        printf("From client: %s\t To client : ", buff);
        bzero(buff, MAX);
        n = 0;
        while ((buff[n++] = getchar()) != '\n');
        write(sockfd, buff, sizeof(buff));
        if (strcmp("exit", buff, 4) == 0) {
            printf("Server Exit...\n");
            break;
        }
    }
}
int main()
{
    int sockfd, connfd, len;
    struct sockaddr_in servaddr, cli;
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd == -1) {
        printf("socket creation failed...\n");
        exit(0);
    }
    else
        printf("Socket successfully created..\n");
```

```

bzero(&servaddr, sizeof(servaddr));
servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
servaddr.sin_port = htons(PORT);
if ((bind(sockfd, (SA*)&servaddr, sizeof(servaddr))) != 0) {
printf("socket bind failed...\n");
exit(0);
}
else
printf("Socket successfully binded..\n");
if ((listen(sockfd, 5)) != 0) {
printf("Listen failed...\n");
exit(0);
}
else
printf("Server listening..\n");
len = sizeof(cli);
connfd = accept(sockfd, (SA*)&cli, &len);
if (connfd < 0) {
printf("server acccept failed...\n");
exit(0);
}
else
printf("server acccept the client...\n");
func(connfd);
close(sockfd);
}

```

The screenshot shows a Visual Studio Code editor window with a project named '18CSC302J Batch 1'. The file explorer on the left shows a directory structure with files for '331' (tcp_clnt_331.c, tcp_server_331.c, udp_clnt_331.c, udp_server_331.c). The main editor displays the source code for 'tcp_server_331.c'. The code includes standard headers, defines a port of 8084, and implements a server that listens for connections, reads data from clients, and prints it. The terminal at the bottom shows the execution of the server, which successfully binds, listens, and receives a connection from 'Puneet'.

```

1 #include <stdio.h>
2 #include <unistd.h>
3 #include <netinet/in.h>
4 #include <stdlib.h>
5 #include <string.h>
6 #include <unistd.h>
7 #include <sys/socket.h>
8 #include <sys/types.h>
9 #define MAX 80
10 #define PORT 8084
11 #define SA struct sockaddr
12 void func(int sockfd)
13 {
14     char buff[MAX];
15     int n;
16     for (;;) {
17         bzero(buff, MAX);
18         read(sockfd, buff, sizeof(buff));
19         printf("From client: %s\n", buff);
20         bzero(buff, MAX);
21         n = 0;
22         while ((buff[n++] = getchar()) != '\n');
23         write(sockfd, buff, sizeof(buff));
24         if (strcmp("exit", buff) == 0) {
25             printf("Server Exit...\n");
26             break;
27         }
28     }
29 }
30 int main()
31 {
32     int sockfd, connfd, len;
33     struct sockaddr_in cli;

```

Terminal Output:

```

331/tcp_server_331.c - Run
Socket successfully created..
Socket successfully binded..
Server listening..
server acccept the client...
From client: Puneet
To client: Puneet

```

TCP CLIENT: CLIENT CODE

```
#include <netdb.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/socket.h>
#define MAX 80
#define PORT 8084
#define SA struct sockaddr
void func(int sockfd)
{
    char buff[MAX];
    int n;
    for (;;) {
        bzero(buff, sizeof(buff));
        printf("Enter the string : ");
        n = 0;
        while ((buff[n++] = getchar()) != '\n')
            ;
        write(sockfd, buff, sizeof(buff));
        bzero(buff, sizeof(buff));
        read(sockfd, buff, sizeof(buff));
        printf("From Server : %s", buff);
        if ((strcmp(buff, "exit", 4)) == 0) {
            printf("Client Exit...\n");
            break;
        }
    }
}

int main()
{
    int sockfd, connfd;
    struct sockaddr_in servaddr, cli;
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd == -1) {
        printf("socket creation failed...\n");
        exit(0);
    }
    else
        printf("Socket successfully created..\n");
    bzero(&servaddr, sizeof(servaddr));
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    servaddr.sin_port = htons(PORT);
    if (connect(sockfd, (SA*)&servaddr, sizeof(servaddr)) != 0) {
        printf("connection with the server failed...\n");
        exit(0);
    }
    else
        printf("connected to the server..\n");
    func(sockfd);
    close(sockfd);
}
```

