COMPUTER NETWORKS

LAB REPORT # 6

SUBMITTED TO: **Ma'am Sundas Ashraf**

SUBMITTED BY: **M. Wajih Haider**

CE-40

SYNDICATE - B

DATE: 22nd December 2020

DEPARTMENT OF COMPUTER AND SOFTWARE ENGINEERING

TASK # 6.1:

CODE:

Client Code:

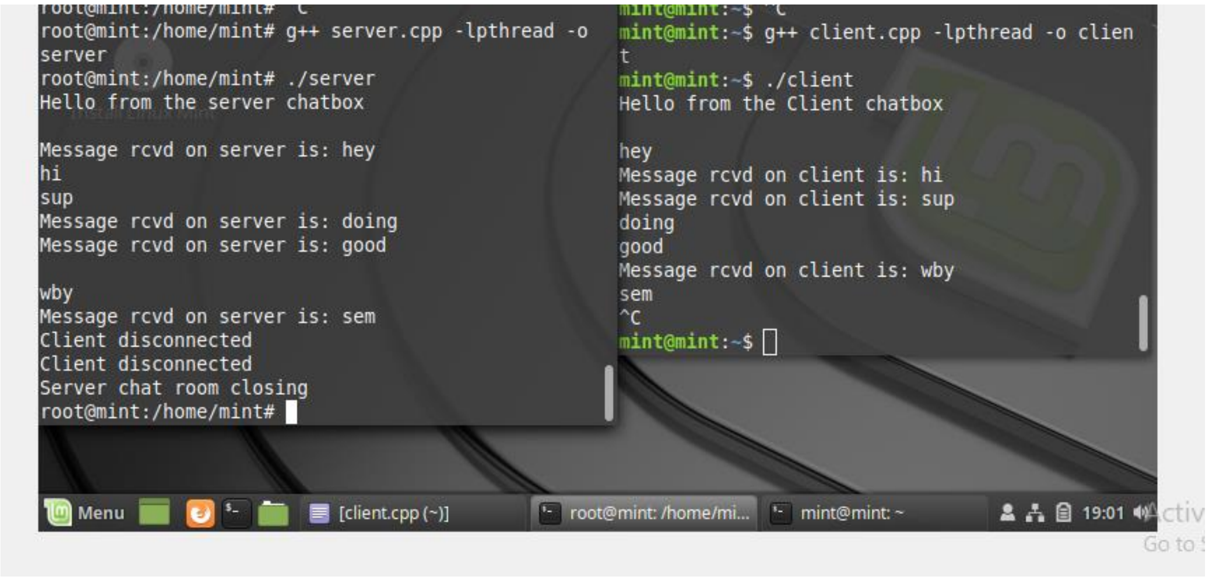
```
#include<sys/types.h>
#include<arpa/inet.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<iostream>
#include<pthread.h> //Pthreads header include file
#include<unistd.h>
#include<cstring>

#define THREAD_COUNT 2 //Controls the number of threads
char buffer[100];
char buf1[100];
void* SendCM(void *rank);
void* RcvCM(void *rank);

using namespace std;
int main()
{
    int fd=socket(AF_INET,SOCK_STREAM,0);
    if(fd==-1)
    {
        perror("socket creation failed");
        exit(-1);
    }
    struct sockaddr_in s_addr;
    s_addr.sin_family=AF_INET;
    s_addr.sin_port=htons(80);
    inet_aton("127.0.0.1",&s_addr.sin_addr);

    connect(fd,(struct sockaddr*)&s_addr,sizeof(s_addr));

pthread_t thread_handles[THREAD_COUNT];

//Create THREAD_COUNT number of Pthreads
for (long thread=0; thread<THREAD_COUNT; thread++) {
pthread_create(&thread_handles[thread], NULL, SendCM, (void*)&fd);
pthread_create(&thread_handles[thread], NULL, RcvCM, (void*)&fd);
}

cout<<"Hello from the Client chatbox\n"<<endl;

//Join all created Pthreads
for (long thread=0; thread<THREAD_COUNT; thread++) {
pthread_join(thread_handles[thread], NULL);
}
//This line executes after completion of joined threads
cout<<"Client chat room closing\n";

    return 0;
}

//Thread function, Check the return type and parameter list
void* SendCM(void *rank) {
long connfds = *((int*) rank);
while (1) {
        fgets(buffer,100,stdin);
        send(connfds,buffer,strlen(buffer),0);
        //sleep(2);
}
return NULL;
}

void* RcvCM(void *rank) {
long connfdr = *((int*) rank);
while (1) {
        int check = recv(connfdr,buf1,100,0);
        if (check < 1){
            cout << "Server Disconnected" << endl;
            break;
        }
        cout << "Message rcvd on client is: " << buf1;
        //sleep(2);
}
return NULL;
}
```

Server Code:

```
#include<sys/types.h>
#include<sys/socket.h>
#include<arpa/inet.h>
#include<netinet/in.h>
#include<string.h>
#include<iostream>
#include<pthread.h> //Pthreads header include file
#include<unistd.h>
#include<stdlib.h>
#include<cstring>
#include<string>

#define THREAD_COUNT 2 //Controls the number of threads
char buffer[100];
char buf[100] = "hello server here";
char buf1[100];
void* SendM(void *rank);
void* RcvM(void *rank);

using namespace std;
int main()
{
    int fd=socket(AF_INET,SOCK_STREAM,0);
    if(fd==-1)
    {
        perror("Socket not made");
        exit(-1);
    }
    struct sockaddr_in addr;
    addr.sin_addr.s_addr=INADDR_ANY;
    addr.sin_family=AF_INET;
    addr.sin_port=htons(80);

    if(bind(fd,(struct sockaddr*)&addr,sizeof(addr))==-1)
    {
        perror("error-BINDING FAILED ON SOCKET");
        exit(-1);
    }
    int backlog=10;
    if(listen(fd,backlog)==-1)
    {
        perror("listen failed on socket:");
        exit(-1);
    }
    int connfd;
    struct sockaddr_in cliaddr;
    socklen_t cliaddr_len=sizeof(cliaddr);
    connfd=accept(fd,(struct sockaddr*)&cliaddr,&cliaddr_len);
    if(connfd==-1){
        cout<<"error";
        exit(-1);
    }

pthread_t thread_handles[THREAD_COUNT];

//Create THREAD_COUNT number of Pthreads
for (long thread=0; thread<THREAD_COUNT; thread++) {
pthread_create(&thread_handles[thread], NULL, SendSM, (void*)&connfd);
pthread_create(&thread_handles[thread], NULL, RcvSM, (void*)&connfd);
}

cout<<"Hello from the server chatbox\n"<<endl;

//Join all created Pthreads
for (long thread=0; thread<THREAD_COUNT; thread++) {
pthread_join(thread_handles[thread], NULL);
}
//This line executes after completion of joined threads
cout<<"Server chat room closing\n";

    return 0;
}
```

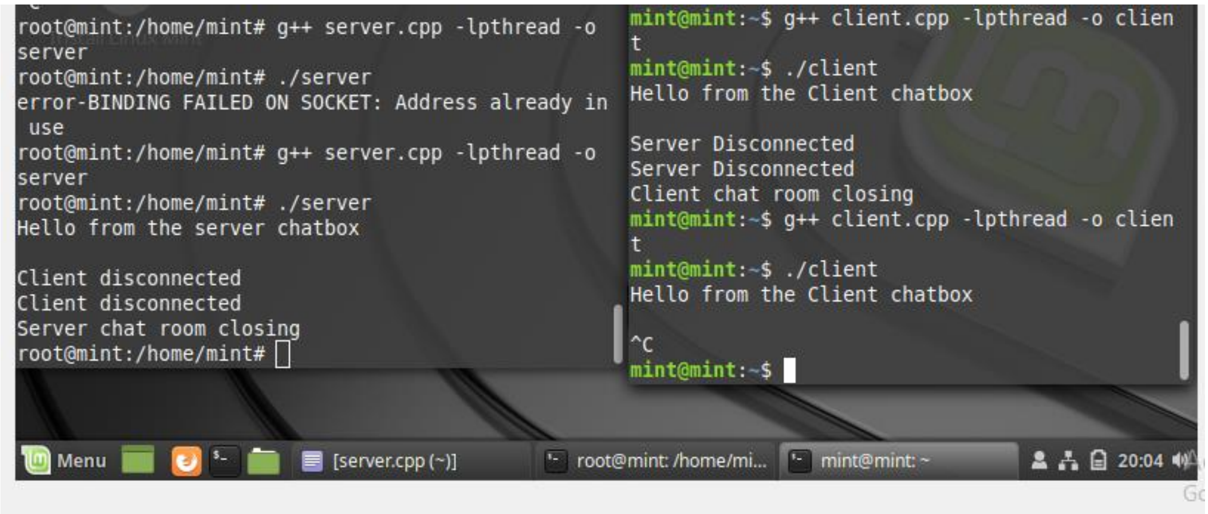
```
//Thread function, Check the return type and parameter list
void* SendSM(void *rank) {
long connfds = *((int*) rank);
while (1) {
        fgets(buffer,100,stdin);
        send(connfds,buffer,strlen(buffer),0);
     //   sleep(2);s
}
return NULL;
}

void* RcvSM(void *rank) {
long connfdr = *((int*) rank);
while (1) {
        int check = recv(connfdr,buffer,100,0);
        if (check < 1){
            cout << "Client disconnected" << endl;
            break;
        }
        cout << "Message rcvd on server is: " << buffer;
       // sleep(2);
}
return NULL;
}
```

OUTPUT:



```
root@mint:/home/mint# ^C
root@mint:/home/mint# g++ server.cpp -lpthread -o
server
root@mint:/home/mint# ./server
Hello from the server chatbox

Message rcvd on server is: hey
hi
sup
Message rcvd on server is: doing
Message rcvd on server is: good

wby
Message rcvd on server is: sem
Client disconnected
Client disconnected
Server chat room closing
root@mint:/home/mint#
```

```
mint@mint:~$ ^C
mint@mint:~$ g++ client.cpp -lpthread -o clien
t
mint@mint:~$ ./client
Hello from the Client chatbox

hey
Message rcvd on client is: hi
Message rcvd on client is: sup
doing
good
Message rcvd on client is: wby
sem
^C
mint@mint:~$
```



```
root@mint:/home/mint# g++ server.cpp -lpthread -o
server
root@mint:/home/mint# ./server
error-BINDING FAILED ON SOCKET: Address already in
 use
root@mint:/home/mint# g++ server.cpp -lpthread -o
server
root@mint:/home/mint# ./server
Hello from the server chatbox

Client disconnected
Client disconnected
Server chat room closing
root@mint:/home/mint# 
```

```
mint@mint:~$ g++ client.cpp -lpthread -o clien
t
mint@mint:~$ ./client
Hello from the Client chatbox

Server Disconnected
Server Disconnected
Client chat room closing
mint@mint:~$ g++ client.cpp -lpthread -o clien
t
mint@mint:~$ ./client
Hello from the Client chatbox

^C
mint@mint:~$ 
```