UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

Facultatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei

Proiect la disciplina

PROTOCOALE PENTRU INTERNET

Studenți:

Pop Diana-Elena

Țicală Roxana- Maria

Grupa 2243

Echipa 3

Coordonatori:

Prof.dr.ing. Virgil-Mircea DOBROTA

Ing. Gabriel LAZAR

----------client.c----------

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


int ClientMain()
{
         printf("Client program");
    int i,s,r;
         struct addrinfo  hints, *res,*p;
         char *text= "GET / HTTP/1.0\r\n\r\n";
         char buffer[2000];//dimensiunea max de date pe care o primeste
         FILE *fis;

         // Socket client
         s = socket(AF_INET6, SOCK_STREAM, 0); // AF_INET6- IPV6, 0 - selecteaza
protocolul pe baza socket-ului folosit, sock_stream - soceket de tip TCP
                  if(s ==-1) // in caz de eroare returneaza -1
                  {
                           printf("\n Socket not created");
                  }
         puts("\n Socket created ");

         // Socket agreed with server

    memset(&hints,0,sizeof hints); // hints  - indicator spre blocul de memorie

         hints.ai_family= AF_INET6; //selecteaza IPv6
         hints.ai_socktype = SOCK_STREAM; //socket de tip  TCP


         i= getaddrinfo("www.ja.net","80",&hints, &res); // returneaza adresa
Ipv6, 80 reprezinta portul pentru http
   if(i!=0)
   {
   printf("\n Fail  \n");
   return 0;
   }
   for( p=res; p!=NULL; p=p->ai_next) //ai_next este un pointer spre urmatoarea
strucrtura
     {

         if((s=socket(res->ai_family,res->ai_socktype,res->ai_protocol))==-1)
//verifica daca socket-ul nu este valid, iar daca este valid continua
         {
                  perror("socket");
                  continue;}
```

```
        // connect -  conecteaza socket-ul la adresa specificata
    if(connect(s,p->ai_addr, p->ai_addrlen)==-1) //returneaza in caz de succes 0
si in caz de eroare -1  si se conteaza
        {
               close(s);
                perror("connect");
                continue;
        }

        break;
        }

      if (p==NULL)
     {
        printf("failed to connect\n");
        return 0;}
    else { printf ("Connected\n");
        }
//Send data to server
   r=send(s, text, strlen(text),0); //trimite un mesaj serverului, 0 - seteaza
flag-urile in 0
   if(r==-1)
   {error("Failed");}
   else
        printf("Trimis \n");

   memset(buffer,0, 2000);
// Recieve data

   fis =fopen("index.htm","w"); //deschide fisierul, w pentru scriere
   if(fis==NULL) //verifica existenta fiserului
   {
            printf("Fisierul nu exista");

   }

   // verifica primirea datelor
  while(1)
  {r= recv(s, buffer ,2000-1,0); //primieste datele de la server (pagina html)

        if (r==-1) //verifica daca se receptioneaza informatii de la server
        {
        printf("Eroare \n");
        break;
        }

        if(r==0)
        {
        printf("conexiune incheiata \n");
        break;
        }
                printf("Mesaj server\n");
 buffer [r] ='\0'; //specifica finalul sirului
fprintf(fis,buffer); //scrie continutul buffer-ului in interiorul fisierului
//       puts(buffer); //afiseaza buffer-ul in consola
          }
return 0;
}
```

----------server.c----------

```
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<errno.h>
#include<string.h>
#include<sys/types.h>
#include<time.h>
#include "client.c"
//declaram metoda main
int main (int argc, char *argv[])
{
int serverPort=22313;// atribuie portul de conectare
int rc;
struct sockaddr_in serverSa;
struct sockaddr_in clientSa;
int clientSaSize;
int on=1;
char mess[2000];
char toCompareMessageCode[3] = "20#";
int isTweentyCommand;
int c,read_size;
int s= socket(PF_INET, SOCK_STREAM, 0);//creeaza un punct de com

//apeleaza Clientul
ClientMain();
// creaza un socket de tip server
rc= setsockopt(s,SOL_SOCKET,SO_REUSEADDR, &on, sizeof on);

memset(&serverSa, 0, sizeof(serverSa));
// seteaza parametri socket-ului de tip server IPv4
serverSa.sin_family=AF_INET;
serverSa.sin_addr.s_addr= htonl(INADDR_ANY);
serverSa.sin_port=htons(serverPort);

// leaga socket-ul de  port
rc=bind(s,(struct sockaddr *)&serverSa, sizeof(serverSa));
if(rc<0)
{
        perror("bind failed");
        exit(1);
}
else
printf("\n Bind succeeded");
rc= listen(s,10); // asculta cererile de conectare, nr de clienti in
asteptare
if (rc<0)
{
perror("listen failed");
return 0;
```

```
}
else
{
printf("\n Listen succeeded");
}
//asteapta conexiuni
while(1)
{
rc=accept(s,(struct sockaddr *)&clientSa,&clientSaSize); //accepta
conexiuni
if (rc<0)
{
perror("\n Accept failed");
exit(1);
}
else
printf("\n Accept succeeded");
// Revieve message
while(( read_size= recv(rc,mess,2000,0)>0))
{
        printf("\n Recieved");


        isTweentyCommand = 1;
        int i;
        for( i=0;i<sizeof(toCompareMessageCode);i++){
        // verifica  daca comanda primita este 20#
                if(mess[i] != toCompareMessageCode[i])
                {
                        isTweentyCommand = 0;
                        break;
                }else{
                        continue;
                }

        }

if(isTweentyCommand == 1){

        FILE * filePointer;
        //deschide fisierul Client
        filePointer = fopen("index.htm","r");
        char singleLine[255];
//citim linie cu linie pana ajunge la finalul fisierului
        while(!feof(filePointer)){

                fgets(singleLine,255,filePointer); //citeste linia curenta
                char *m = singleLine;
                send(rc,m,strlen(m),0); //trimite datele clientului IPv4
        }

        fclose(filePointer);// inchide fisierul

        }
```

```
        else{
        char *m = "Comanda NEIMPLEMENTATA"; //trimite mesajul in cazul in
care comanda nu este 20#
        send(rc,m,strlen(m),0);
        }
}
}
close(rc);
return 0;
}
```