EXPERIMENT-10

A.S.SANJAY

IMPLEMENT CHAT APPLICATION

AIM: To implement chatting in C

CLIENT SIDE

ALGORITHM:

The steps involved in establishing a socket on client side are as follows.

1) Create a socket with the socket() system call

2) Connect the socket to the address of the server using the connect() system call.

3) Send and receive data. There are a number of ways to do this, but the simplest is to use the read() and write() system calls.

Scanned by CamScanner

```
#include<sys/socket.h>
#include <stdio.h>
#include <string.h>
#include <netdb.h>
#include <stdlib.h>
int  main()
{
char  buf[100];
int  k;
int  sock_desc;
struct  sockaddr_in  client;
memset (&client , 0, sizeof (client));
sock_desc = socket (AF_INET, SOCK_
 STREAM, 0);
 if (sock_desc == -1)
 {
  printf("Error in socket creation");
  exit(1);
 }
 client.sin_family = AF_INET;
client.sin_addr.s_addr = INADDR_ANY;
client.sin_port= 3002;
```

Scanned by CamScanner

```
k=connect(sock_desc,(struct sockaddr*)&
client,sizeof(client));
if(k==-1)
{printf("Error in connecting to server");
exit(1);
}
while(1)
{printf("\n Enter data to be sent to server:");
fgets(buf,100,stdin);
if(strncmp(buf,"end",3)==0)
break;
k=send(sock_desc,buf,100,0);
if(k==-1)
{printf("Error in ~~receiving~~ sending");
exit(1);
}
~~printf~~
k=recv(sock_desc,buf,100,0);
if(k==-1)
{printf("Error in receiving");
exit(1);
}
```

Scanned by CamScanner

```
printf ("Message got from server is : %s", buf);
}
close (sock_desc);
exit (0);
return 0;
}
```

O/P

Enter data to be sent to server

Message got from server: hai [illegible]

Enter data to be sent to server: i am fine

Message got from server: I am fine

Enter data to be sent to server: end

Scanned by CamScanner

~~AIM:~~

SERVER SIDE

ALGORITHM:

The steps involved in establishing a socket on Server side are as follows.

1) Create a socket with the socket() system call
2) Bind the socket to an address using the bind() system call for a server socket on Internet, an address consists of a port number on host machine.
3) Listen for connections with the listen() system call.

4) Repeatedly do the following:

- Call accept() to get a new socket for each client connection
- Communicate with the client via the new socket using send() and recv()
- Close the client connection using close()

Scanned by CamScanner

CAM

```
#include <sys/socket.h>
#include <stdio.h>
#include <string.h>
#include <netdb.h>
#include <stdlib.h>
int    main()
{  char    buf[100];
   int  k;
   socklen_t   len;
   int   sock_desc, temp_sock_desc;
   struct   sockaddr_in   server, client;
   memset  (&server, 0, sizeof (server));
   memset  (&client, 0, sizeof ( client));
  sock_desc = socket(AF_INET, SOCK_STREAM, 0);
  if ( sock_ddesc == -1)
  { printf(" Error   in   socket    creation");
     exit(1);
   }
  server.sin_family = AF_INET;
  server.sin_addr.s_addr = inet_addr("127.0.0.1");
 server.sin_port = 3002;
 k= bind( sock_desc, (struct   sockaddr*)&server,
    sizeof (server));
```

Scanned by CamScanner

```
if (k==-1)
{printf("Error in binding");
 exit(1);
}
k= listen(sock_desc, 20);
if (k==-1)
{
 printf(" Error in listening ");
 exit(1);
}
len = sizeof (client);
temp_sock_desc= accept(sock_desc, (struct
sockaddr*)&client, &len);
if (temp_sock_desc==-1)
{ printf(" Error in temporary socket creation");
  exit(1);
}
while(1)
{
k= recv(temp_sock_desc, buf, 100, 0);
if (k==-1)
{printf("Error in receiving");
exit(1);
}
printf("Message got from client is: %s",
   buf);
```

Scanned by CamScanner

```
printf("\n Enter data to be sent to
 client:");
fgets(buf, 100, stdin);
 if (strncmp(buf, "end", 3)==0)
  break;
  k=send(temp_sock_desc, buf, 100,0);
  if (k==-1)
  { printf("Error in sending");
     exit(1);
  }
  }
   close(temp_sock_desc);
   exit(0);
   return 0;
    }
```

Scanned by CamScanner

O/P.

~~En~~

```
Message got from client is: hi

Enter data to be sent to client: how
are you?

Message got from client is: I am
fine

Enter data to be sent to client: end
```

Scanned by CamScanner