NAME: G.Shiva Shankar reddy

REG.NO.: 192211486

CODE:CSA0734

EXPERIMENT: 34

AIM:TO impement application tcp server.

PROGRAM:

```
#include <stdio.h>

#include <netdb.h>

#include <netinet/in.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#include <sys/types.h>

#include <unistd.h> // read(), write(), close()

#define MAX 80

#define PORT 8080

#define SA struct sockaddr


// Function designed for chat between client and server.

void func(int connfd)

{

        char buff[MAX];

        int n;

        // infinite loop for chat

        for (;;) {
                bzero(buff, MAX);
// read the message from client and copy it in buffer
                read(connfd, buff, sizeof(buff));
                // print buffer which contains the client contents
                printf("From client: %s\t To client : ", buff);
                bzero(buff, MAX);
                n = 0;
                // copy server message in the buffer
                while ((buff[n++] = getchar()) != '\n')
                        ;

                // and send that buffer to client
                write(connfd, buff, sizeof(buff));

                // if msg contains "Exit" then server exit and chat ended.
                if (strncmp("exit", buff, 4) == 0) {
                        printf("Server Exit...\n");
                        break;
                }
        }
}

// Driver function
int main()
{
        int sockfd, connfd, len;
        struct sockaddr_in servaddr, cli;
```

```
// socket create and verification

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));


// assign IP, PORT

servaddr.sin_family = AF_INET;

servaddr.sin_addr.s_addr = htonl(INADDR_ANY);

servaddr.sin_port = htons(PORT);


// Binding newly created socket to given IP and verification

if ((bind(sockfd, (SA*)&servaddr, sizeof(servaddr))) != 0) {

        printf("socket bind failed...\n");

        exit(0);

}

else

        printf("Socket successfully binded..\n");


// Now server is ready to listen and verification

if ((listen(sockfd, 5)) != 0) {

        printf("Listen failed...\n");
```

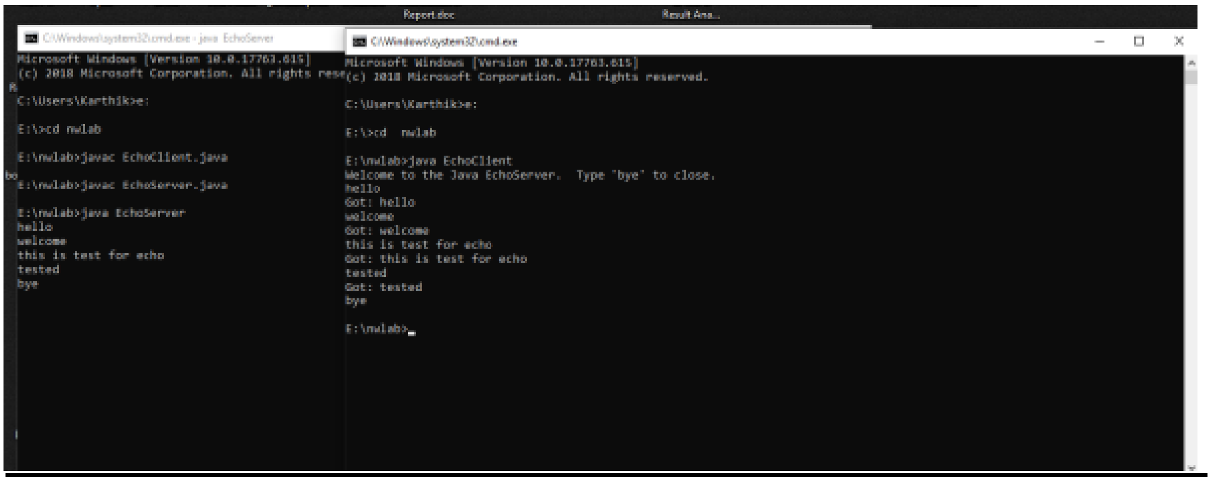
```
		exit(0);
	}
	else
		printf("Server listening..\n");
	len = sizeof(cli);

	// Accept the data packet from client and verification
	connfd = accept(sockfd, (SA*)&cli, &len);
	if (connfd < 0) {
		printf("server accept failed...\n");
		exit(0);
	}
	else
		printf("server accept the client...\n");

	// Function for chatting between client and server
	func(connfd);

	// After chatting close the socket
	close(sockfd);
}
```

OUTPUT:



```
C:\Windows\system32\cmd.exe - java EchoServer
Microsoft Windows [Version 10.0.17763.615]
(c) 2018 Microsoft Corporation. All rights rese

C:\Users\Karthik>e:

E:\>cd nwlab

E:\nwlab>javac EchoClient.java

E:\nwlab>javac EchoServer.java

E:\nwlab>java EchoServer
hello
welcome
this is test for echo
tested
bye
```

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.17763.615]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Karthik>e:

E:\>cd  nwlab

E:\nwlab>java EchoClient
Welcome to the Java EchoServer.  Type 'bye' to close.
hello
Got: hello
welcome
Got: welcome
this is test for echo
Got: this is test for echo
tested
Got: tested
bye

E:\nwlab>
```

RESULT: Therefore , applications using TCP echo server and client has been successfully excecuted.