Practical 8

Distributed Operating System

Name - Ritesh Parkhi

Roll No. – 46

Batch – 3

AIM:

Construct chat application to demonstrate the concept of echo client server application.

THEORY:

An Echo Server is an application that allows a client and a server to connect so a client can send a message to the server and the server can receive the message and send, or echo, it back to the client.

Server

A server is a computer or computer program that provides a service to another computer, or client. For example, the internet is based on web servers that respond to requests from clients via web browsers. How do you connect to a server to access it services? Often, you will connect to a server over a computer network, which is a group of computers connected together in order to share resources. To connect to a server, you need to know the port number that the server socket is listening on, which is an endpoint for communication between the client and the server. Each port number is associated with an IP address, a numerical label assigned to each device connected to a computer network, and a protocol type, preset rules and guidelines for communicating data. With the port number and IP address, you can attempt to establish a connection to a server. It is important to note that when building an echo server, it is not a requirement to be connected within a network. The server can be run locally on a computer and a client can connect to the server via the same computer.

Client

A client is responsible for sending a message to the server, and is also where the message is echoed back to. Once the server is running, or listening for connections, a client is able to send a request to the server. With the correct port number and IP address, and if everything goes to plan, the server accepts the connection and creates a new socket. Through the newly created socket, the client and server can now communicate by writing to or reading from the socket.

PROGRAM:

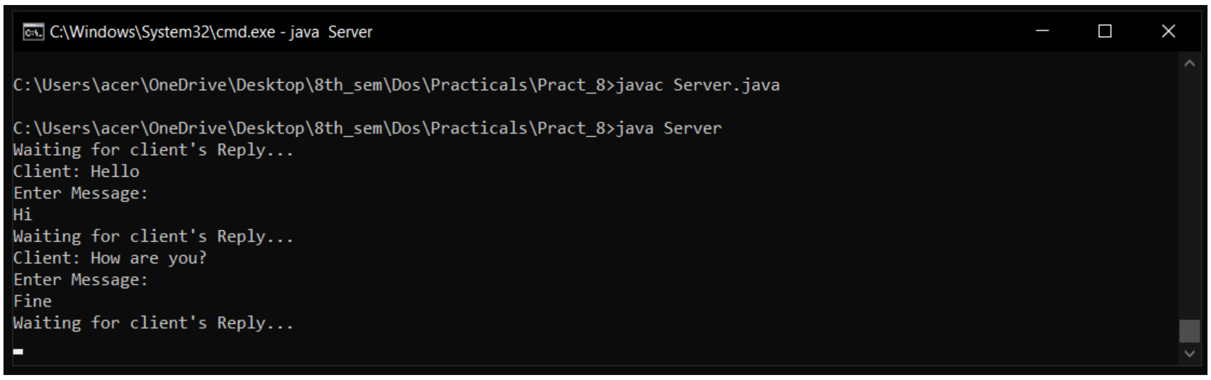
Client.java

```java
import java.net.*;
import java.io.*;
public class Client  {

    public static void main(String[] args)  throws Exception {

        Socket s=new Socket("localhost",8088);

        DataInputStream din=new DataInputStream(s.getInputStream());
        DataOutputStream dout=new DataOutputStream(s.getOutputStream());

        BufferedReader br=new BufferedReader(new
InputStreamReader(System.in));

        String str="",str2="";

        while(!str.equals("stop")){
            System.out.println("\nEnter Response: ");
            str=br.readLine();
            dout.writeUTF(str);
            dout.flush();
            System.out.println("Waiting for Server's Reply...");
            str2=din.readUTF();
            System.out.println("Server says: "+str2);
            }

            dout.close();
            s.close();

    }
}
```

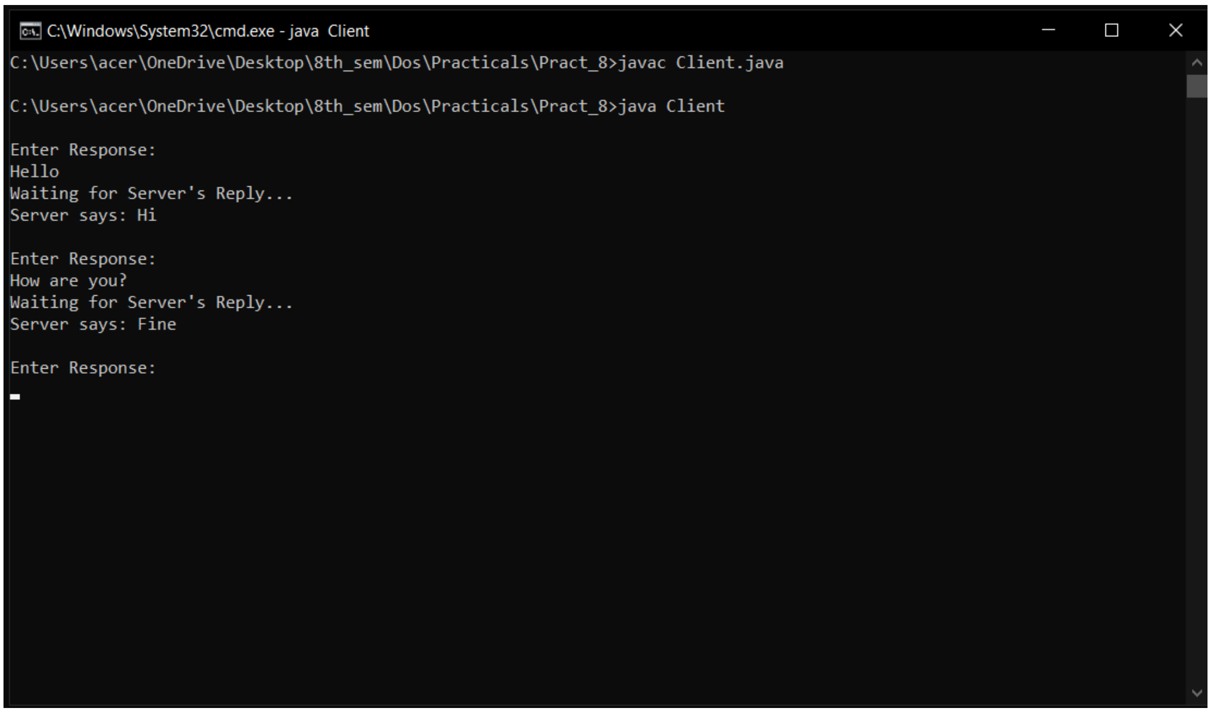
Server.java

```java
import java.net.*;
import java.io.*;
public class Server {
    public static void main(String args[]) throws
Exception,UnknownHostException{

        ServerSocket ss=new ServerSocket(8088);

        Socket s=ss.accept();;

        DataInputStream din=new DataInputStream(s.getInputStream());
        DataOutputStream dout=new DataOutputStream(s.getOutputStream());

        BufferedReader br=new BufferedReader(new
InputStreamReader(System.in));

        String str="",str2="";

        while(str!="stop")
        {
            System.out.println("Waiting for client's Reply...");
            str=din.readUTF();
            System.out.println("Client: "+str);
            System.out.println("Enter Message:");
            str2=br.readLine();
            dout.writeUTF(str2);
            dout.flush();
        }

        din.close();
        s.close();
        ss.close();
    }
}
```

OUTPUT:



```
C:\Windows\System32\cmd.exe - java  Server

C:\Users\acer\OneDrive\Desktop\8th_sem\Dos\Practicals\Pract_8>javac Server.java

C:\Users\acer\OneDrive\Desktop\8th_sem\Dos\Practicals\Pract_8>java Server
Waiting for client's Reply...
Client: Hello
Enter Message:
Hi
Waiting for client's Reply...
Client: How are you?
Enter Message:
Fine
Waiting for client's Reply...
```



```
C:\Users\acer\OneDrive\Desktop\8th_sem\Dos\Practicals\Pract_8>javac Client.java

C:\Users\acer\OneDrive\Desktop\8th_sem\Dos\Practicals\Pract_8>java Client

Enter Response:
Hello
Waiting for Server's Reply...
Server says: Hi

Enter Response:
How are you?
Waiting for Server's Reply...
Server says: Fine

Enter Response:
```

CONCLUSION:

Hence we have successfully built a program to implement echo client server application.