Extra Assignment

- **TITLE :** Demonstrate Socket Programming in JAVA by implementing client-server chat application using Networking concepts.

- **CODE:**

1. Client

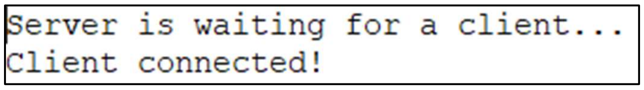
```
package networking;
import java.io.*;
import java.net.*;

public class TCPClient {
    public static void main(String[] args) {
        try {
            Socket socket = new Socket("localhost", 5000);
            System.out.println("Connected to Server!");

            BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
            PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
            BufferedReader userInput = new BufferedReader(new
InputStreamReader(System.in));

            Thread readThread = new Thread(() -> {
                try {
                    String serverMessage;
                    while ((serverMessage = in.readLine()) != null) {
                        System.out.println("Server: " + serverMessage);
                    }
                } catch (IOException e) {
                    System.out.println("Connection closed.");
                }
            });

            Thread writeThread = new Thread(() -> {
                try {
                    String message;
                    while ((message = userInput.readLine()) != null) {
                        out.println(message);
                    }
                } catch (IOException e) {
                    System.out.println("Error in sending message.");
                }
            });

            readThread.start();
            writeThread.start();

            readThread.join();
            writeThread.join();

            socket.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2. Server

```
package networking;
import java.io.*;
import java.net.*;

public class TCPServer {
    public static void main(String[] args) {
        try {
            ServerSocket serverSocket = new ServerSocket(5000);
            System.out.println("Server is waiting for a client...");

            Socket socket = serverSocket.accept();
            System.out.println("Client connected!");

            BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
            PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
            BufferedReader userInput = new BufferedReader(new
InputStreamReader(System.in));

            Thread readThread = new Thread(() -> {
                try {
                    String clientMessage;
                    while ((clientMessage = in.readLine()) != null) {
                        System.out.println("Client: " + clientMessage);
                    }
                } catch (IOException e) {
                    System.out.println("Connection closed.");
                }
            });

            Thread writeThread = new Thread(() -> {
                try {
                    String message;
                    while ((message = userInput.readLine()) != null) {
                        out.println(message);
                    }
                } catch (IOException e) {
                    System.out.println("Error in sending message.");
                }
            });

            readThread.start();
            writeThread.start();

            readThread.join();
            writeThread.join();

            socket.close();
            serverSocket.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

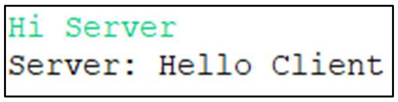
Output –

Connection Establishment



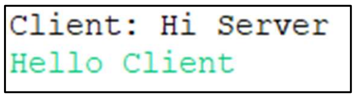
```
Server is waiting for a client...
Client connected!
```

Start of Chat Application via Client



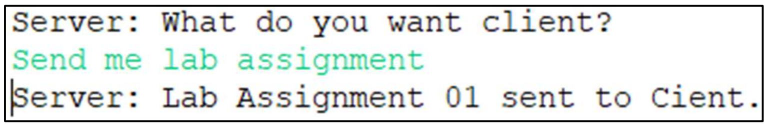
```
Hi Server
Server: Hello Client
```

Start of Chat Application via Server

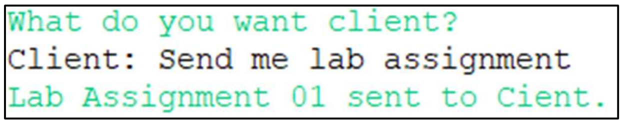


```
Client: Hi Server
Hello Client
```

Small Conversation between them!



```
Server: What do you want client?
Send me lab assignment
Server: Lab Assignment 01 sent to Cient.
```



```
What do you want client?
Client: Send me lab assignment
Lab Assignment 01 sent to Cient.
```

- **Conclusion –** In this assignment, I have studied and used Socket Programming principles and classes to create a simple chat interface between client and server.