```java
/* A simple client with a single window, two textfields and buttons
 27th March 02

 The buttons: disconnect from the server,
             send the employee name in the textfield to the server

 The server will send the email address which is shown in the second textfield.
 */

import java.awt.*;
import java.awt.event.*;
import java.io.*;
import java.net.*;

public class ClientExample extends Frame implements ActionListener {

    TextField userName, emailAddress;
    Button startProcessing, quit;

    InputStream is = null;
    OutputStream os = null;
    PrintWriter pw = null;
    BufferedReader br = null;
    Socket s;

    ClientExample(String title) {
        super(title);
        userName = new TextField(10);
        emailAddress = new TextField(20);
        startProcessing = new Button("Start");
        quit = new Button("Quit");
        setLayout(new GridLayout(4, 1));
        add(new Label("User Name"));
        add(userName);
        add(new Label("Email Adress"));
        add(emailAddress);
        add(startProcessing);
        add(quit);
        setSize(150, 300);
        setVisible(true);

        startProcessing.addActionListener(this);
        quit.addActionListener(this);

        // Set up connection to the server on the loop back address
        // and the same port number as the Server is expecting
        try {
            s = new Socket("127.0.0.1", 2000);
            is = s.getInputStream();
            os = s.getOutputStream();

            pw = new PrintWriter(os, true);
            br = new BufferedReader(new InputStreamReader(is));
        } catch (IOException e) {
            System.out.println("Error connecting wth the Server  " + e);
        }
    }  // end of constructor

    public void actionPerformed(ActionEvent ae) {
        String buttonTest = ae.getActionCommand();
        String typedName;
        String receivedAddress;
        try {
            if (buttonTest.equals("Quit")) {
                System.out.println("Exiting from Server");
                pw.println("Exit");
                is.close();
                os.close();
                pw.close();
                br.close();
                s.close();
                System.exit(0);
            }
            if (buttonTest.equals("Start")) {
                typedName = userName.getText();

                // Send to Server
                pw.println(typedName);

                // Receive reply
                receivedAddress = br.readLine();

                emailAddress.setText(receivedAddress);
            }
        } catch (IOException e) {
            System.out.println("Problem contacting the server to send/receive");
        }
    }  // end of actionPerformed method

    public static void main(String[] args) {
        new ClientExample("Client Example");
    }  // end of main method

}  // end of ClientExample class
```

```java
//A sample server
//27th March 02

import java.io.*;
import java.util.*;
import java.net.*;

public class ServerExample {

    public static void main(String args[]) {
        // Streams definition for connection
        InputStream is = null;
        OutputStream os = null;

        //Writers and readers for communication
        PrintWriter pw = null;
        BufferedReader br = null;

        int connectionCount = 0; // Count of clients connecting
        String lineRead = ""; // String read from client
        Object o = null; // Used for assessing the Hashtable
        String reply = ""; // Reply to be sent to the client

        System.out.println("Server starting");

        // Set up the database;
        Hashtable<String, String> names = new Hashtable();
        names.put("Fred Smith", "F.Smith@cov.ac.uk");
        names.put("Joe Bloggs", "J.Bloggs@cov.ac.uk");
        System.out.println("Database done");

        // Establish Server Socket
        try {
            ServerSocket ss = new ServerSocket(2000);
            while (true) {
                // Listen for a connection and return new Socket if one is made
                Socket s = ss.accept();  // Blocks until connection made
                connectionCount++;
                System.out.println("Connection " + connectionCount + " made");
                is = s.getInputStream();
                os = s.getOutputStream();
                pw = new PrintWriter(os, true);
                br = new BufferedReader(new InputStreamReader(is));
                System.out.println("System set up");

                //Read and process names until the client tells the server
                //the service is no longer required.
                lineRead = "";
                while (true) {
                    lineRead = br.readLine();
                    if (lineRead.equals("Exit")) {
                        break;
                    }
                    o = names.get(lineRead);
                    if (o == null) {
                        reply = "User not known";
                    } else {
                        reply = (String) o;
                    }
                    pw.println(reply);
                }
                pw.close();
                br.close();
                is.close();
                os.close();
                System.out.println("Closed down");
            }
        } catch (IOException e) {
            System.out.println("Trouble with connection" + e);
        }
    }  // end of main method

}  // end of SeverExample class
```