**Q1) Design a network application protocol for this application and justify your design decision. In your protocol design, you should address the following. [35 points]**

**1. The communication patterns of the network application.**

This application makes use of a TCP Socket to allow for bi-directional communication between the Server and the Clients. How the communication of this application works follows this pattern: Server is constantly running looking for a connection to be made from a Client, Client connects to Server which causes Server to send message to Client to determine what role the Client is filling (Job Seeker or Job Creator) depending on the role the Client chooses this will determine what “functions” the Client can use (EX: Post Job or Accept Job). The connection with the Server will remain in tack until the Client has fulfilled all of it’s duties whether it be posting a job then accepting Job Seekers or accepting a job and submitting the files the Job Creator was looking for. Ideally this communication pattern would run with no error and can be repeated an infinite amount of times.

**2. Define the protocol design goals.**

Due to timeboxing issues the protocol design goals need to be broken into different phases of development.

Phase 1 (Completed):

Let n denote set of positive real numbers

* Creating a Server which can handle n Clients at a time
* Server can send and receive messages from Clients
* Server Thread was created to cause a Blocking Socket (Reason Disclosed in further Phases)

Phase 2 (Incomplete):

* Clients can distinguish their roles (EX: Job Seeker or Job Creator)
* Clients which are Job Creators can send a job to Server which will save this job to a .CSV file
* Clients which are Job Seekers can accept posted jobs (One job at a time)
* Clients which are Job Creators can start the job once all Job Seeker positions are filled

Phase 3 (Incomplete):

* Account Creation and Login function will be created to help distinguish Job Seekers from Job Creators (Make use of Salted hashes)
* Clients which are Job Seekers can submit files which will cause the job to be completed

Phase 4 (Incomplete):

* GUI development will start to allow for an easy to use interface
* Login Screen
* Job Posting Screen
* File Submission Screen

**3. Define the message format, structure, and semantics**

\*Client **(Job Creator)** Connects to Server\*

Server - - - - - >Client

* Sends Login Page

Client - - - - - >Server

* Sends Login Credentials
* Login Credentials Structure: Username, Password

Server - - - - - >Client

* Sends Job Posting Page

Client - - - - - >Server

* Sends Job to be posted
* Job Message Structure: Name, Description, #ofJobSeekers

\*Server saves job to .CSV file which will be referenced when needed\*

Server - - - - - > Client

* Sends confirmation message that job has been posted

\*Job Seeker/s Accept Job\*

* Sends message that Job Seeker spots are filled

Client - - - - - > Server

* Sends message to start job and is waiting for job completion

\*Job Seeker/s Finishes Job\*

Server - - - - - > Client

* Sends job completion message

Client - - - - - > Server

* Sends message accepting the completion message

\*Server Removes Job from Job Posting Page/.CSV File\*

\*Client **(Job Seeker)** Connects to Server\*

Server - - - - - >Client

* Sends Login Page

Client - - - - - >Server

* Sends Login Credentials
* Login Credentials Structure: Username, Password

Server - - - - - >Client

* Sends Job Posting Page

Client - - - - - >Server

* Sends Job accept request

\*Server saves Job Seekers Username to .CSV file which will be referenced when needed\*

\*Job Creator Accepts the Job Seeker\*

Server - - - - - > Client

* Sends confirmation of accept request

Client - - - - - > Server

* Sends message telling Server the Job Seeker is wait for job to start

\*Job Creator Starts Job\*

Server - - - - - > Client

* Sends job start message

Client - - - - - > Server

* Sends file that will complete the job

\*Server sends file to Job Creator\*

Server - - - - - > Client

* Sends job finished message

\*Job Seeker Terminates Connection\*

**4. Design the communication rules.**

Communication between the Server and Clients will consist of various Strings that cause certain functions to run, as well the Server will be sending objects over the TCP socket these objects being JPanels which are used for the GUI (Login Page and Job Posting Page). When sending object, they will need to be deconstructed into bytes and reconstructed on the other side the same goes for the Strings that are being sent between the Server and Clients. The Clients who identify as Job Seekers will need to send files across the TCP socket and the Clients who identify as Job Creators will need to accept these files from the Server through the TCP socket.

**Q2) Argue the need for a new application layer protocol for this network application instead of using existing standard protocols (e.g. HTTP, SMTP, WebSocket, etc.) [10 points]**

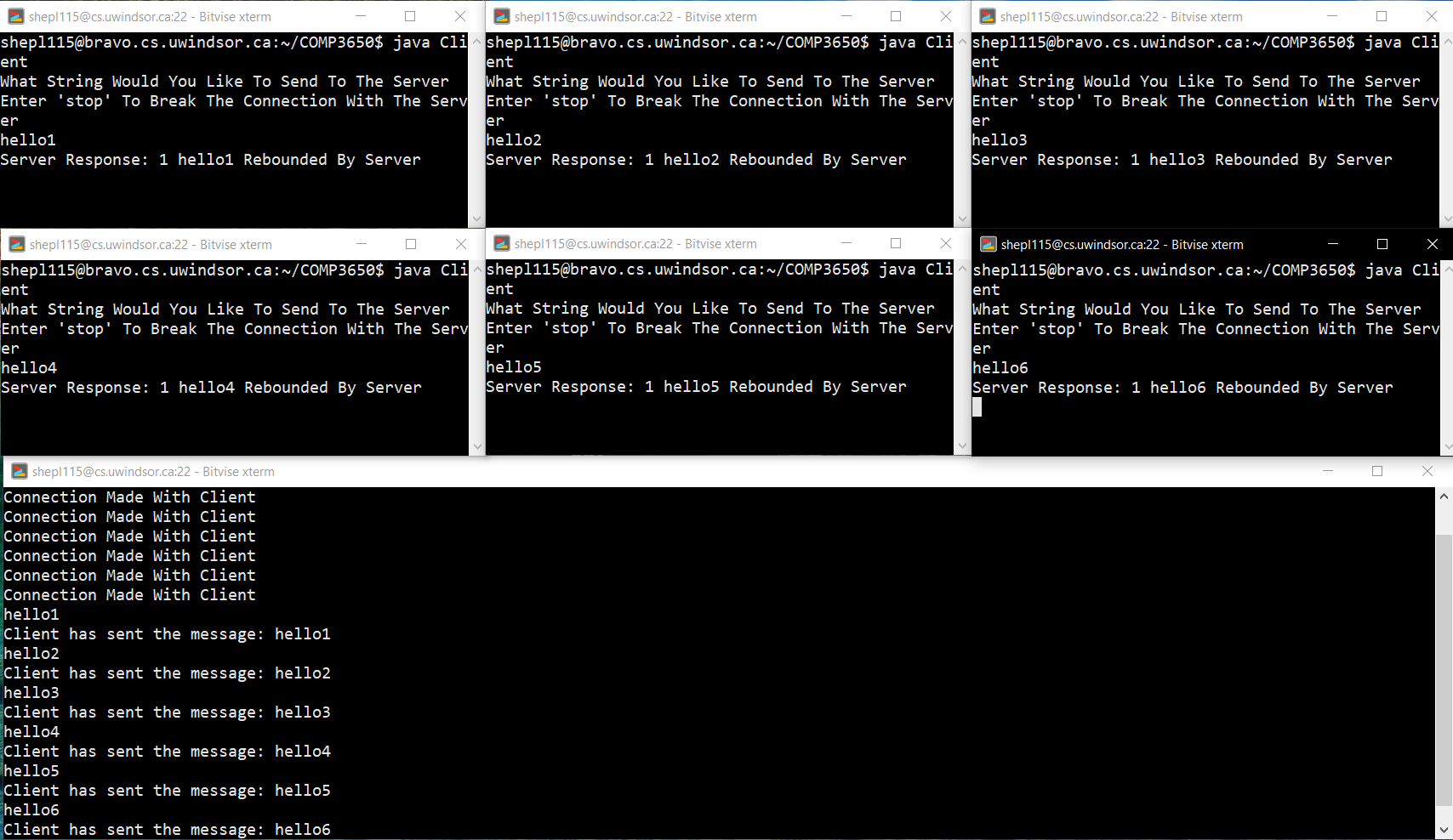
When talking about an application layer protocol there are a couple base functions that need to be satisfied. If we were to argue about needing a new application layer protocol for this network application the following must be kept in mind when talking about application layers; facilitates the user to use the services of the network, it can be used to develop network-based applications, provides services like; user login, naming network devices, formatting messages and transfer of files, as well error handling and recovery of messages/files.

Breaking down what this network applications actions are, they can be categorized into two majors’ components; File transferring/updating and communication between Server and multiple Clients. If there were a way to create a new application layer protocol the most useful function would be creating a global file store which is bound to the Server which can be accessed by every client, this protocol will be accompanied by Client priority numbers (Ex: 0 == Client can View Files, 1 == Client can Download Files, 2 == Clients can add Files, 3 == Client can Edit and Save Changes to Files, Each increase in priority number is given the functions of all previous priority numbers). Having this new application layer protocol would be perfect when considering what must be done for this network application, Having the Job Creators being of Priority 3 and the Job Seekers stating at Priority 0 once the Job Seeker accepts a job they will increase to Priority 1, after completion of this job they will move to Priority 2 then when the connection is terminated the Job Seeker will be set back to Priority 0 and the process can be repeated.

**Q3) Provide the implication source code of your network application protocol with sufficient test cases based on the design goals, message philosophy (format, structure, semantics), and Communication rule.[25 points]**

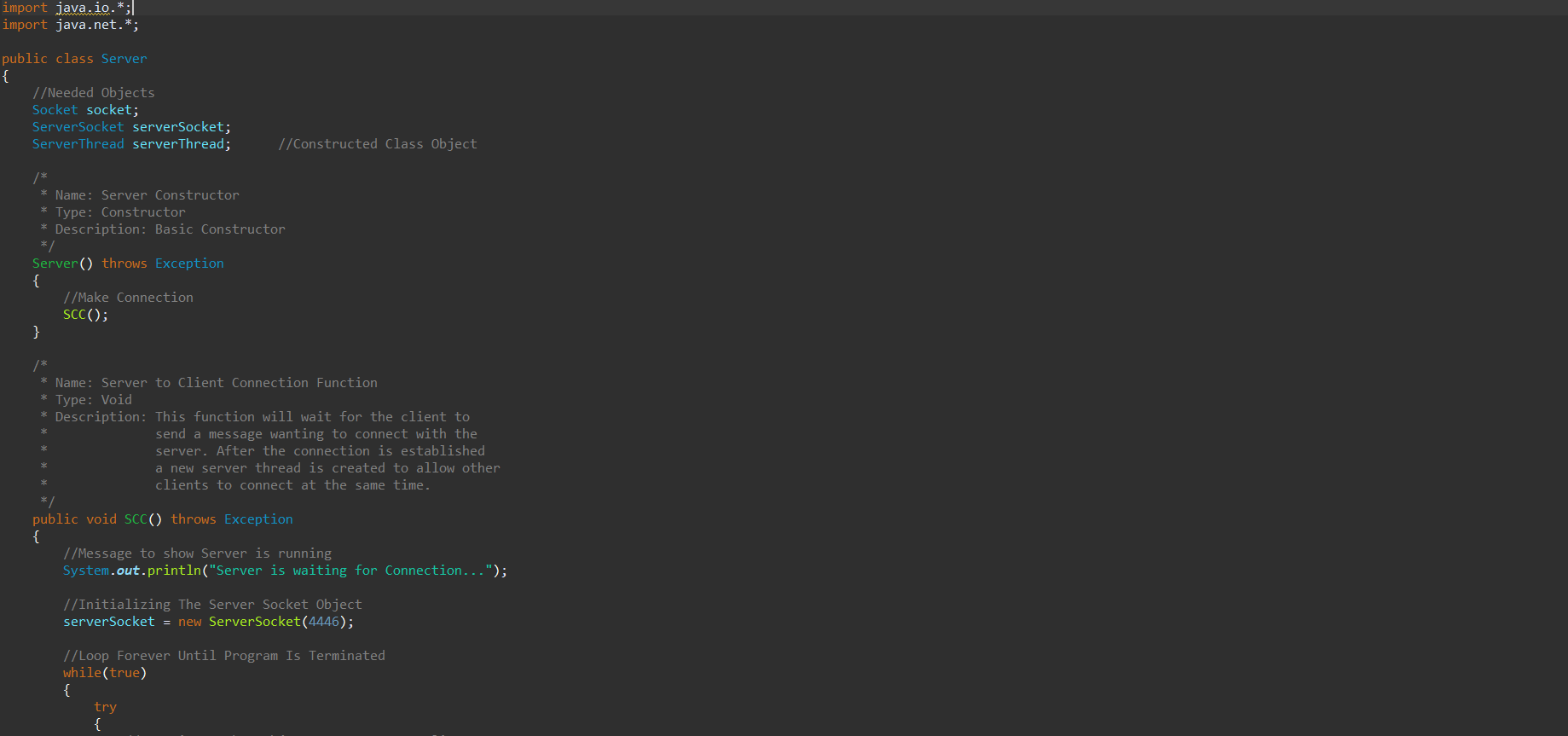
Figure: 0.1: Showing Server Running in a Blocking way with 6 different Clients each Client send its owe message then receiving a message back from the Server

(Server Message Structure: int messageCounter + String clientMessage + “Rebounded By Server”)



**Server Source Code Figures:**

**Figure 1.0:**

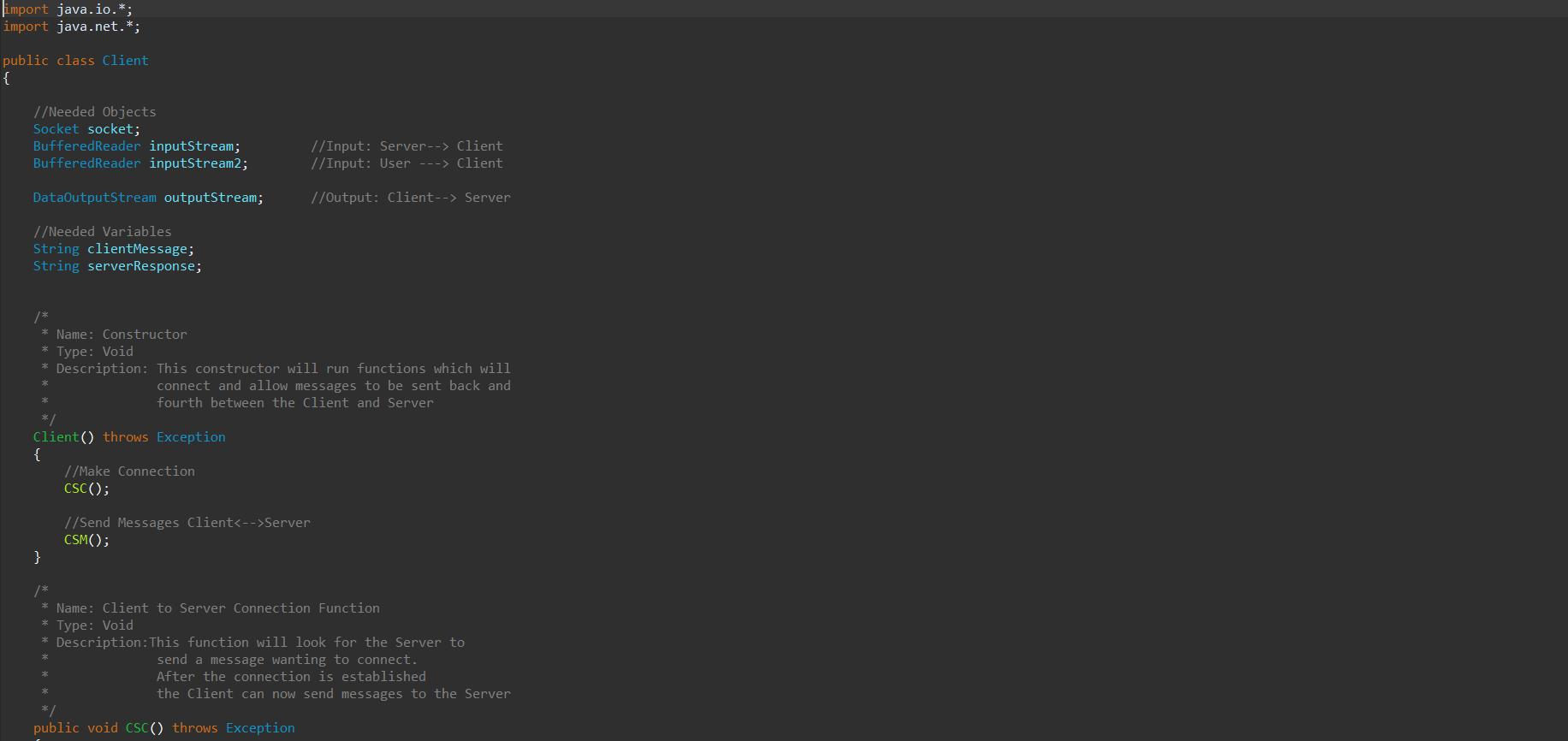


**Figure 1.1:**

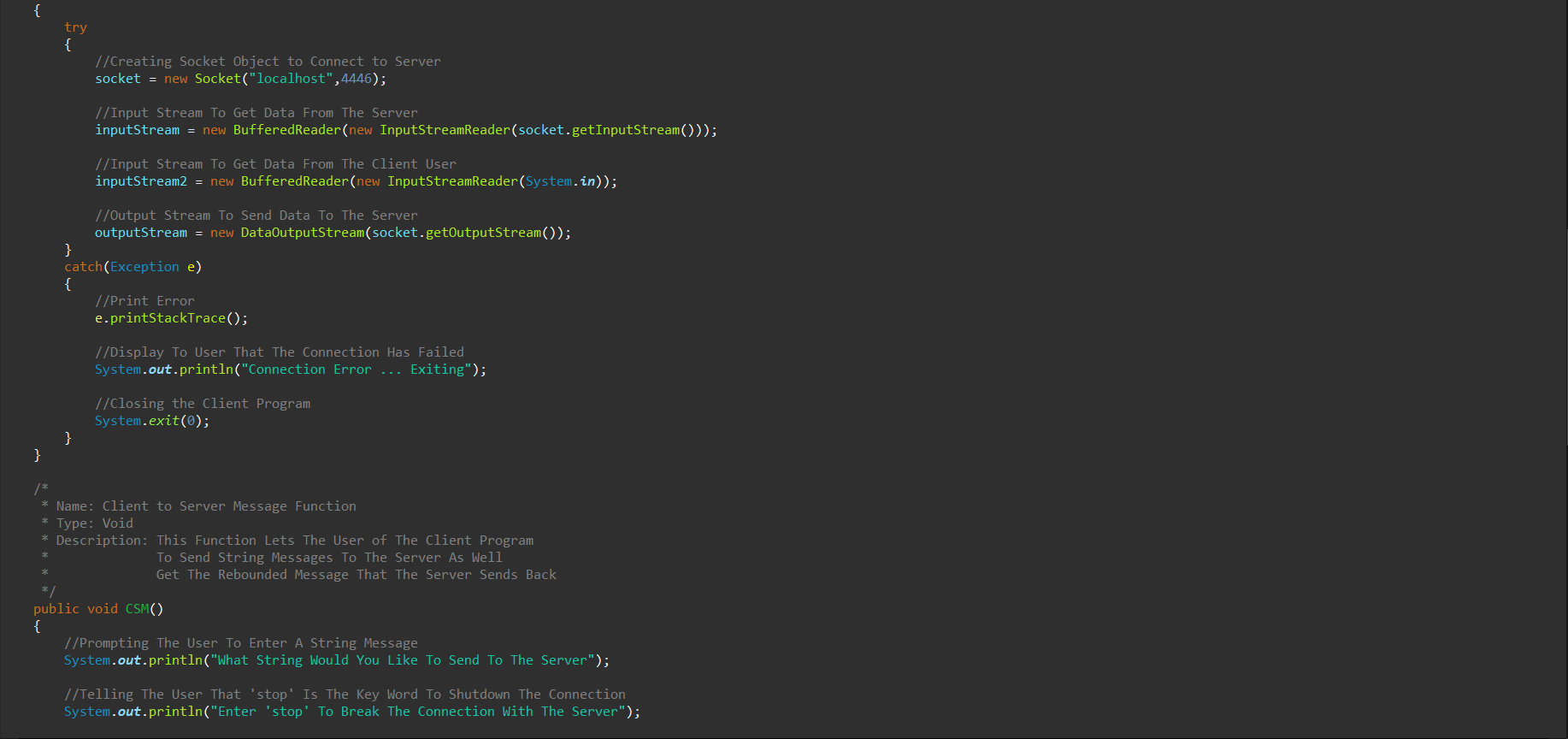


**Client Figures Source Code:**

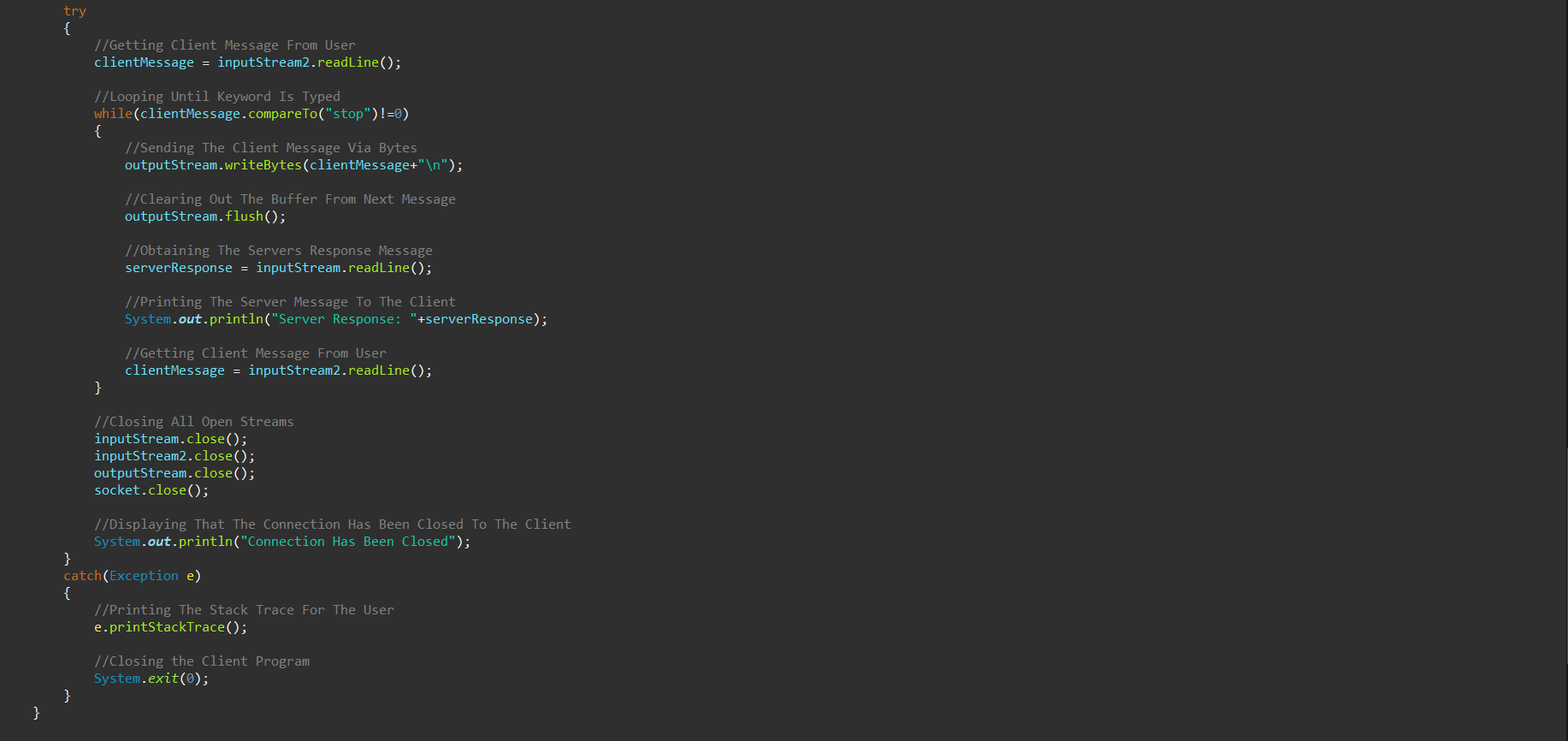
**Figure 2.0:**



**Figure 2.1:**

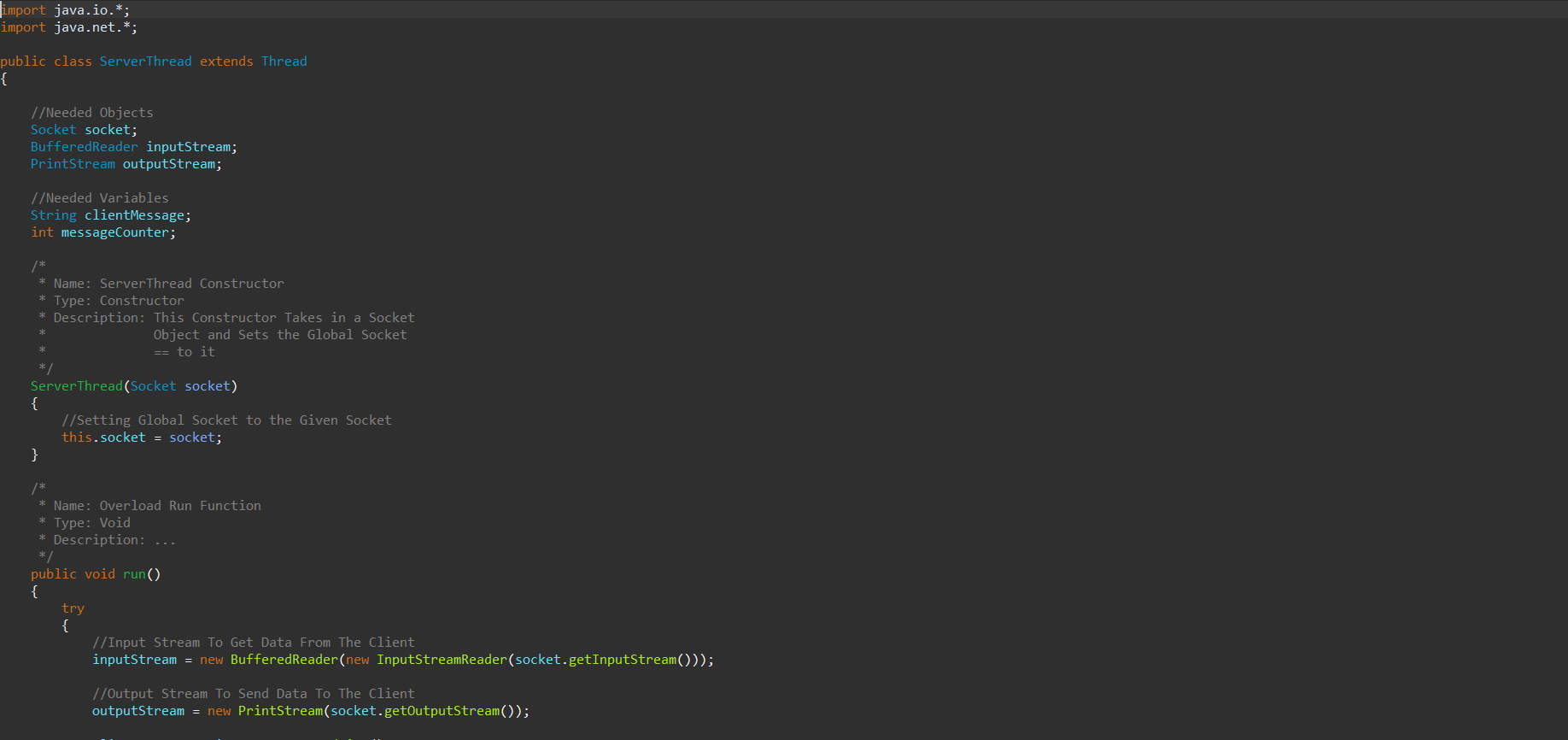


**Figure 2.2:**



**Server Thread Figures Source Code:**

**Figure 3.0:**



**Figure 3.1:**

