Distributed Systems Assignment

DT228

Lina Mir

C19366191

24TH of November 2022

Declaration

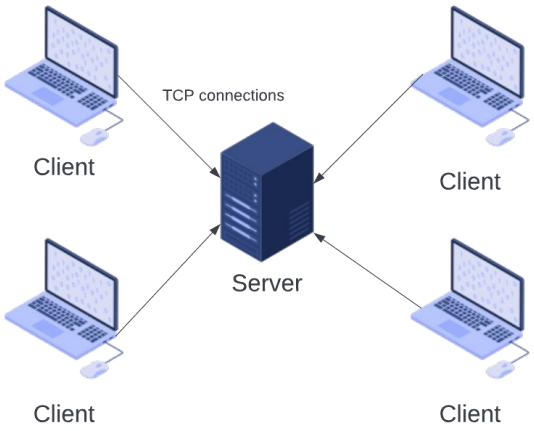
I declare that this work, which is submitted as part of my coursework, is entirely my own, except where clearly and explicitly stated.

Signed:

Lina Mir

Application Architecture

This is a distributed auction system with a server and multiple clients. It uses the TCP/IP protocol and java sockets to connect the clients and the server together. It uses multithreading, having one thread running at the same time.



The application architecture worked as below, the files and what they do are listed below.

AuctionClient

This is the client-side file that implements Runnable. This creates the thread by defining a single method which the thread code will be executed in. The socket, time for the bid and the client strings were created here. This code started the auction client by creating a new socket with the host and port number that was passed. Once this happens then the client joins auction. The item and the time will be displayed from AuctionItem. The BufferedReader will create an input stream for the client, while DataOutputStream will send outputs to the server. When joined, the clients' inputs are read, making sure the inputs match certain things. This file will also have the client leave the auction when they want by typing 'leave' and this will close the socket and the input and output streams on the client's end.

ClientThread

Here the thread is inherited. When this is a subclass, it implements runnable. The input streams for data and object are created and the socket. These things will help with the reading of the data from the client. The streams between the client and the server are opened here, using ObjectInputStream and DataInputStream. When the client is close, here the streams and sockets need to close too. The data is read from the server and the item can be displayed. The data is read from the server, and this is casted to an AuctionItem.

ClientHandler

This extends threads and will create a new thread to handle the connections for the clients. The data and object streams are created here and the socket. It waits for the data from the client and once done closes the streams and socket. It uses the ObjectOutputStream to send an AuctionItem and a string to the client.

AuctionHandler

This also extends the thread subclass. It will deal with the countdown of the auction. Th server is created, and the new auction items are called when the time runs out.

AuctionItem

An array of items and sold items are created here. This implements Serializable. This will convert the objects state to a byte stream, and this can be reverted back. The items are created here, details such as name, price and if sold.

AuctionServer

This implements runnable and creates an array of clients and a server socket. The server socket on the port is established and the start is called. The thread can stop it the run is broken. The main logic of the auction is dealt with here. The run method will accept new connections from clients and pass the socket to the thread. The auction starts and the details are sent to the clients. If another client joins, the information is sent to them.

AuctionItems.txt

This contains the list of auction items to be displayed to the clients. The title of the object and the price is listed here. This price is the reserve price. More auction items can be added here.

Setup

The code is compiled and run beforehand using the command javac -classpath . AuctionServer.java which complied the server code and then the client code was compiled using the command javac -classpath . AuctionClient.java

Client.bat

This contains the following code, 'java AuctionClient localhost 1234' to start and run the client for the auction quickly. It contains the host and the port that will be used to run the client. Once the code is complied, this can be typed, and the client can start.

Server.bat

This contains 'java AuctionServer 1234' which will server for the auction quickly. This has the port number where the server will run that the client will need to connect to. Once the code is compiled this is typed, and the server is started.

Running Instructions

Below are the instructions to run the auction system:

1. Open three command prompts.
2. Move to the folder where the code is.
3. Type server.bat and then press enter. This will start the AuctionServer on the port 1234.
4. In another command prompt type client.bat and press enter. This will start the AuctionClient on port 1234 using the local host.
5. In the third command prompt type client.bat and press enter.
6. The clients and server start, and the instructions are displayed on the client.

The menu will be displayed. To see the instructions, help can be typed. In order to leave the auction, the user can type leave. To see the time left during the auction, the user can type time.

Once two users are present, the auction begins. Numbers are typed by each bidder until the timer runs out. When the time runs out the bidder with the larger big will win the round. This keeps going until all the items have been sold.