Assignment 9

Multithreaded Server and Clients

Objectives:

Build a multithreaded HotelServer that implements a supplied protocol.

Program:

In this Assignment, you will create a Server application that takes reservations for a small hotel. It will allow multiple clients to connect simultaneously. You will also write a Client application that can connect to the server to allow a user to make reservations.

Hint:

- For the client and server code, please use DataInputStream class to connect to the socket's input stream, and DataOutputStream class to connect to the socket's output stream.
- Check the DataInputStream and DataOutputStream API for the methods to read/write String objects and numbers from/to the sockets.

The Hotel Class

You are given a class called Hotel. Download and do not edit Hotel.java. Include the file in your submission so that the marker does not need to add it manually. The hotel tracks the current reservations for your hotel for the month of March only. It only maintains reservations for the 1st through 31st of March of the one room.

The Hotel class contains the following methods:

- boolean requestReservation(String user, int firstDay, int lastDay)
  - Attempt to make a reservation from the firstDay to the lastDay (inclusive) for the person specified by name
  - **Restrictions: a reservation will not be made if:**
    - The user already has a reservation
    - If the values for firstDay or lastDay are invalid in any way.
    - any of the requested days are not available
  - Return true if the reservation was made, false if it was not
- boolean cancelReservation(String user)
  - If the specified person has a reservation, it will be cancelled (all of their days become available) and returns true
  - Otherwise, returns false if they did not have a reservation to cancel
- String reservationInformation()

- Returns a string containing the full reservation information for the month. For each day displays either available or the name of the person who has the reservation for that day.

Actions:

- **Part1**: Create the HotelServer. The table below contains the protocol that the server should follow. After each command, the server should respond with the String output from the appropriate Hotel method.

  The server should handle exceptions so that even if one client connection crashes, the server continues to run

Protocol:

- In this protocol, the server always responds with a single String.
- **When a client connects, the server should send a welcome message to the client.**
  - The **client's first request should always be to set the user**, so that the server knows who is making reservations. If the client sends any other command first, respond with an error message and disconnect.
- The commands are sent as strings objects or integer numbers. Each 'parameter' is sent separately, requires two writes to the output stream. One for the command "USER" and one for the new name.

Client Request	Server Response	Description
`USER n`	`Hello, n`	`Change the current user to the String n. (Imagine this as logging in as a user)`
`RESERVE first last`	• `Reservation made: n from first through last` • `Reservation unsuccessful: n from first through last`	`Attempt to make a reservation for the current user from day first to day last (inclusive) first and last are sent as integers (not as a string)`
`CANCEL`	• `Reservations successfully canceled for n` • `Reservations not canceled for n, no current reservation.`	`Cancel any reservations made for the current user.`
`AVAIL`	`The full availability info`	`Send availability information as provided by the Hotel method.`
`QUIT`	`Closing Connection`	`Quit the connection.`
`Anything else…`	`Invalid command: Closing Connection`	`Close the connection.`

- **Part2**: Write a client program that allows a user to make reservations. When the client starts, it will connect to the server and asks the user for their name. Then repeatedly allow the user to use the server's basic commands (reserve, cancel, availability) until the user chooses to quit. The user should be shown a menu that allows them to choose any of the five commands

Submit a single ZIP (Compressed) file to BrightSpace containing:

Compress all of the Java source files and submit them to Brightsapce prior to the due date.

Marking Scheme

- [20] Multithreaded HotelServer
- [20] HotelClient
- [10] User friendly UI for the users to request for a service.
- [10] Break code into functions
- [5] Documentation: Javadoc for all the methods headers, purpose of the program, inline comments
- [5] Programming Style: Meaningful variable names and constants following the conventions, consistent indentation.