**JAV745 Porject2**

Yuhang Zhao 150467199

1. **Instructions on how to run your program:**

For the WebServer:  
The command **java -jar jav745p2server.jar <port number> <input file>** should start the WebServer of this application. You can use my sample server input file called “serverInformation.csv”.

All transaction results, regardless of successful or failed, would be written into a newly created file “serverTranscation.txt”.

For the Client:

The command **java -jar jav745p2client.jar <server address> <server port> <configurable files>** should start the Client of this application. You can use my sample client input file called “orderInformation1.csv” or “orderInformation2.csv”.

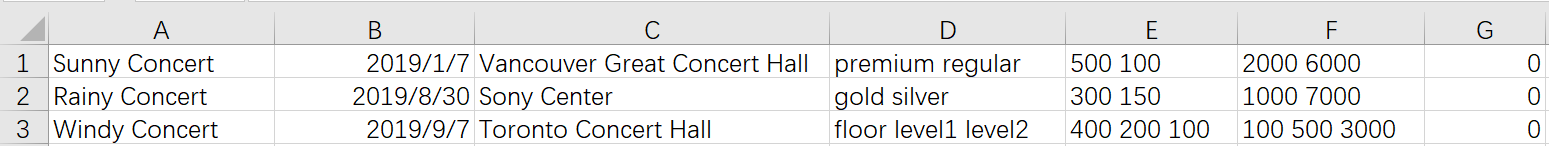
If you want to test if my application can process requests from multiple clients concurrently, you can open two command line window, use respectively “orderInformation1.csv” and “orderInformation2.csv” file to feed this command to run the Client.

All transaction results, regardless of successful or failed, would be written into a newly created file “clientTransaction.txt”.

1. **The structure of my WebServer input file and Client configure file**
2. WebServer input file:

My WebServer input file is called serverInformation.csv, which is to feed the WebServer.class.

The structure of this input file is like below:



Each line can be seen as one concert information.

The data in column A means the name of concert.

The data in column B means the date of every concert.

The data in column C means the venue selected for every concert.

The data in column D means different seat types for every venue, different seat types are separated by space.

The data in column E means the seat price for every corresponding seat type. (The order of seat price should be same as that of the corresponding seat type. Numbers are separated by space).

The date in column F means the total seat number for every seat type. (The order should be same as the order of the corresponding seat type. Numbers are separated by space).

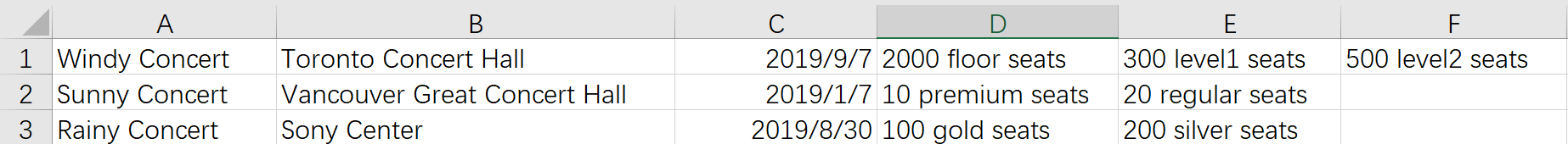
The data in column G means the initial balance for every concert, which will get the payment from clients once the they paid for their orders.

1. Client configurable files:

My Client configurable file are two files, orderInformation1.csv and orderInformation2.csv, each one containing some requests.

Here I have two order files since I can use them to test if my application can process requests from multiple clients concurrently.

The structure of configurable file is like below:



Each line can be regarded as one request from one client.

The column A means the concert name client chose in his order.

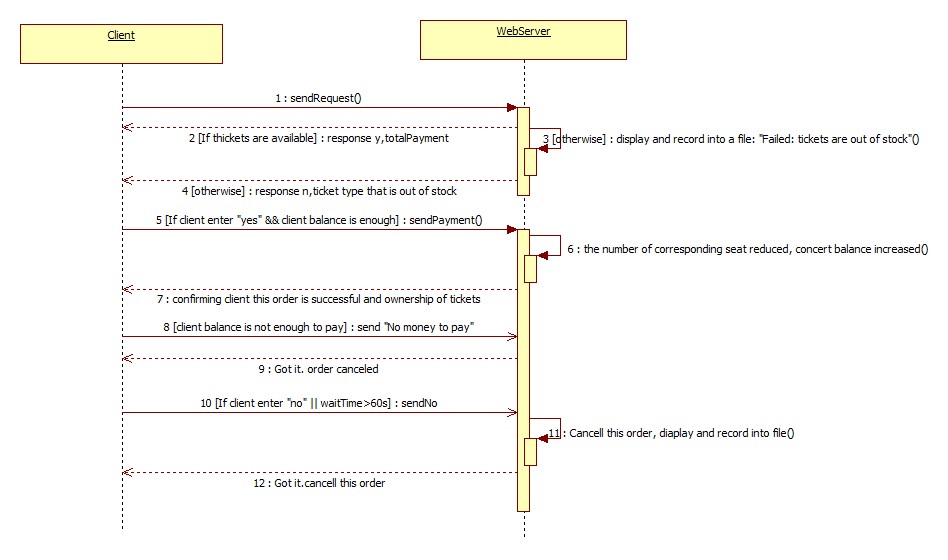
The column B means the venue name for this concert.

The column C means the date of this concert held.

The column D, E, F mean how many seats this client wants to order for different seat types. The order is from top-class seat type to low-level seat type. The first word in every column must be a number. The words are separated by space.

1. **Detailed description of client-server protocol**

The below UML sequence diagram describe the client-WebServer exchange.



Firstly the Client will send the order information to WebServer. Once this WebServer gets these data, it will check if the tickets that this request orders are available. If yes, return “y, total payment for this order” to Client, otherwise, return “n and the ticket type that is out of stock”.

Then if the client receives “n and the ticket type that is out of stock”, this order will be canceled.

Otherwise, after seeing this prompt “Do you want to pay?”, if client enters “yes” and his balance is enough to pay this order, client will send payment information to WebServer. Once WebServer got payment, the number of corresponding seat is reduced, the corresponding concert balance is increased. Finally, WebServer returns confirmation message to Client.

If Client balance is not enough to pay, it will send “No money to pay” to WebServer. Then WebServer would response Client “Got it. Order canceled”.

If Client enters “no” OR wait time exceeding 60 seconds, it will send “No” to WebServer. The WebServer cancels this order and responses Client “Got it. Order canceled”.

One another thing if one concert promoter decides to only use 50% of the seats in the chosen venue, you can just open the serverInformation.csv file, change every number in the 6th column in half, which is the easiest way to do that for my application.