CMPE 322 – Project 2: Flight Reservation System Simulator Project Report

Yağmur Kahyaoğlu 2015400057

How to compile and run :

From the terminal run the following command:

```
$ g++ main.cpp -o project -std=c++11 -lpthread
$  ./project [number of clients]
```

I use a struct “client” to keep information such as the id of the client, seat number of the client, total number of seats, if client is seated and if the client is waiting for servers response. In the main I the number of seats as N from the arguments and allocate a memory for each seat as an array to control later if the seat is occupied. I create an array of client structs , an array of client threads and an array of server threads all sized N. In a for loop I create a client and a server thread using pthread_create and send the address of the corresponding struct element from the struct array as a parameter.

For the client threads I use the method clientMethod which takes the given struct from the parameter. While the current client is not seated in the critical section the program finds a random seat number and sets wait true and the critical section ends. Then the thread makes busy waiting until server either sets wait false or isSeated true. If the client is seated the program locks another mutex and writes to the file, then unlocks the mutex and the thread exits.

For the server thread I use the method serverMethod which also takes the given struct as the parameter. It makes busy waiting while the seat is 0 which means the client hasn’t chosen a seat yet. After the client chooses a seat the program enters another while loop which runs until client finds an available seat. In the loop mutexS is locked and if the seat that client chose recently is empty, server makes the seat full and makes isSeated true. Else, it makes wait boolean false to make client find another seat and makes busy waiting until client makes wait true. Then, mutexS is unlocked and after the while loop ends after client finds an available seat, the thread exits.

In the end I have another for loop to join all server and client threads in the main.