**P1**

**Analiza cerintelor**

**Sala concerte**

O sala de concerte vinde bilete la spectacolele organizate printr-o aplicatie client-server.

Sala organizeaza cel mult un spectacol pe zi.

Sala de concerte are un numar maxim - ‘nr\_locuri’ - de locuri numerotate de la 1 la ‘nr\_locuri’.

Pentru fiecare spectacol avem informatii de tip (data, titlu).

Permanent sala mentine o evidenta actualizata pentru:

- informatii despre bilete pentru fiecare spectacol - (ID\_spectacol, lista\_locuri\_vandute);

- vanzarile efectuate: lista de vanzari; vanzare = (data\_vanzare, ID\_spectacol, numar\_bilete,

lista\_locurilor) ;

- soldul total per spectacol(suma totala incasata).

Periodic sistemul (2 cazuri testare: 5, 10 secunde) face o verificare a locurilor vandute prin verificarea corespondentei corecte intre locurile libere si vanzarile facute, sumele incasate per vanzare si soldul total.

Sistemul foloseste un mecanism de tip ‘Thread-Pool’ pentru rezolvarea a taskurilor.

Pentru testare se va considera ca fiecare client initiaza/creeaza la interval de 2 sec o noua cerere de vanzare bilete folosind date generate aleatoriu (nr\_de\_bilete, locuri) si se primeste de la server o notificare – vanzare reusita sau vanzare nereusita. Nu este necesara interfata grafica!

Pentru verificare se cere salvarea pe suport extern (fisier text, sau BD) a vanzarilor si a rezultatelor operatiilor de verificare executate periodic: data, ora, sold\_per spectacol, lista vanzarilor per spectacol, ‘corect/incorect’.

Serverul se inchide dupa un interval de timp precizat si notifica clientii activi referitor la inchidere.

**Proiectare**

Clase:

**Client:**

**Client** care contine: - clientSocket(Socket)

* out/in pentru scriere/citire din fisier ( PrintWriter/BufferedReader)
* os/oos pentru scriere/citire din socket ( OutputStream/ObjectOutputStream)

**ClientMain** care realizeaza requesturile catre server

**Server:**

**Server** care contine: - serverPort (int)

* serverSocket (ServerSocket)
* un threadpool de 10 threaduri (ExecutorService)
* repository – hall ( Hall)
* un timer (Timer)

**ServerMain** care porneste serverul timp de 2 minute si scrie rezultate la final in fisier

**Verify** care realizeaza verificarea corespondentei corecte ceruta in cerinta la un interval de 5/10 secunde.

**Worker** care contine: - un clientSocket (Socket)

* repository ul (Hall)

si realizeaza task-ul de vanzare a biletelor si trimite un raspuns clientului daca vanzarea s-a efectuat cu success.

**Model:**

**Hall** care contine: - seatNr – numarul de locuri din sala (int)

* showList – lista spectacolelor ( List<Show>)
* saleList – lista vanzarilor ( List<Sale>)

**Sale** care contine : - idShow (int)

* saleTime (data la care s-a efectuat vanzarea)
* soldSeatsNr (cate locuri s-au vandut)
* soldSeats ( lista cu locurile vandute)
* sum ( suma obtinuta din vanzarea biletelor)

**Show**  care contine : - id (id-ul showului)

* showDate ( data la care are loc spectacolul )
* title ( numele spectacolului )
* ticketPrice ( pretul unui bilet )
* soldSeats ( locurile ocupate la un anumit moment )
* sold ( soldul obtinut in functie de nr de locuri vandute )

Functii:

**checkFreeSeats(Show show)** : verifica daca numarul de locuri vandute la un anumit spectacol cu locurile vandute din vanzarile acelui spectacol

**checkSold(Show show) :** verifica daca suma obtinut din vanzarile la un anumit spectacol coincide ca soldul spectacolului.

**checkAvailability(SaleDTO sale)** : verifica daca vanzarea a avut succes, daca locurile cerute sunt libere la un anumit spectacol.

Detalii implementare:

Se creeaza serverul care ruleaza timp de 2 minute si asteapta requesturi din partea clientilor. Cand primeste un request un thread din Pool-ul de threaduri create initiale se va ocupa de task creand un nou **Worker** care va realiza vanzarea.

In acelasi timp la fiecare 2 secunde fiecare clientii trimit un nou request, iar la fiecare 5/10 secunde serverul va verifica daca locurile ocupate din cadrul vanzarilor coincide cu locurile ocupate din cadrul spectacolului respective, si suma la fel; apoi scrie in fisier daca verificarea a fost corecta/incorecta.

Dupa 2 minute in care serverul a rulat se scriu in fisier informatii despre vanzarile, spectacolele si sala care au rulat in acest timp.

Cazuri de testare:

Verificarea este corecta de fiecare data