**P1 Client-Server PPD**

**Patras Sergiu Adrian, 235/2**

**Data predarii: 04.01.2022**

P1 - Client-Server

**Cerinta:**

**Sala spectacole**

O sala de spectacole 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, pret\_bilet).

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 (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’ poentru rezolvarea a taskurilor. Pentru acest mecanism este nevoie sa se poata specifica numarul de threaduri folosite.

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) a rezultatelor operatiilor de verificare executate periodic: data, ora, sold\_per spectacol, lista vanzarilor per spectacol, ‘corect/incorect’.

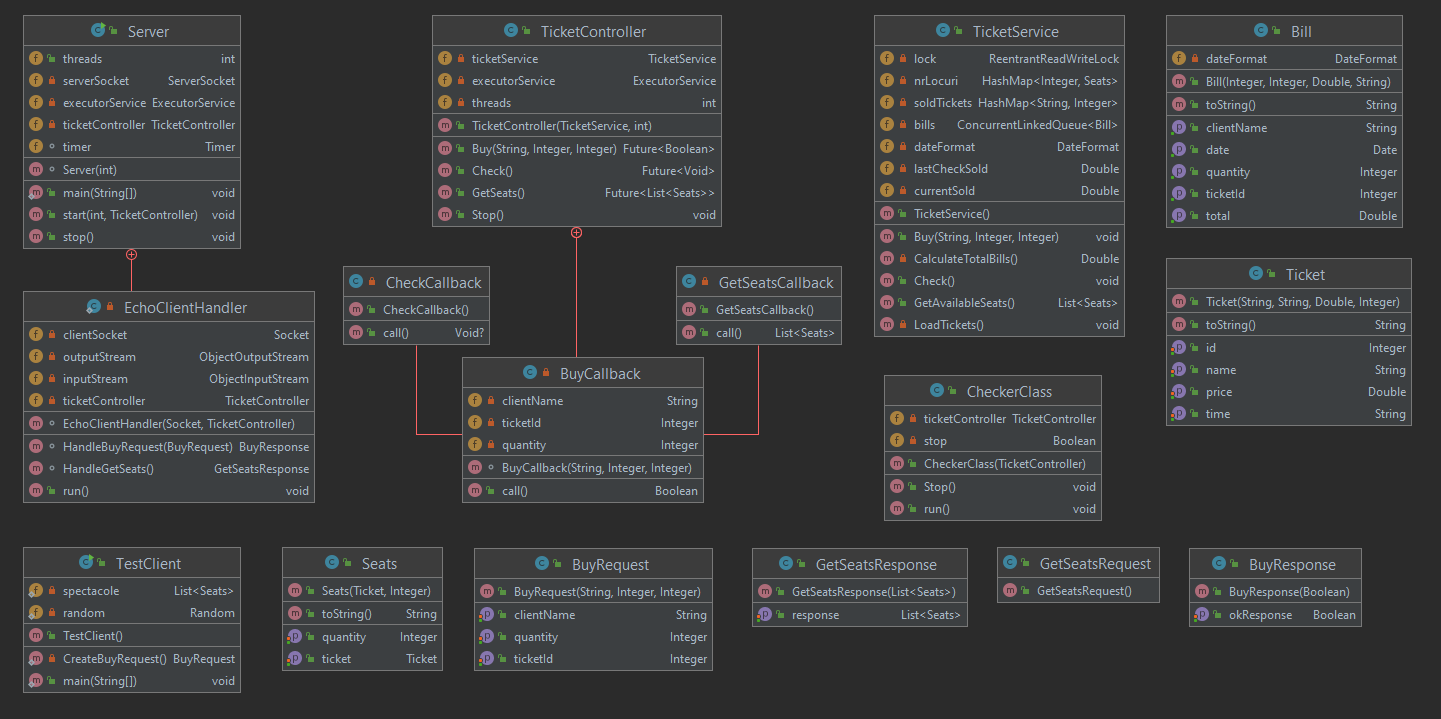
Serverul se inchide dupa un interval de timp precizat si notifica clientii activi referitor la inchidere.

**Implementare:**

Limbajul ales: Java

Pentru comunicarea server-client s-au folosit socket-uri. Serverul opereaza cu un obiect de tipul TicketController. Acesta va avea un obiect de tip ExecutorService(pentru managmentul thread-urilor) și va returna la toate operațiile un obiect de tipul *Future<T>*, execuția fiind asincronă, serverul va aștepta rezultatul operației ( *future.get()* ) și va trimite clientului un mesaj corespunzător.  
 Testarea se face folosind o clasa TestClient ce trimite cereri de cumparare periodice (la 2 secunde). O data la 5, respectiv 10 secunde(in functie de cazul de testare), sistemul face verificari ale tranzctiilor care apoi sunt salvate într-un fișier log.txt.

**Diagrama de clase:**

****

**Testare:**

Nr\_locuri =100; / 500

3 spectacole (S1, S2, S3)

S1 pret\_bilet=100;  
S2 pret\_bilet=200;  
S3 pret\_bilet=150;

Serverul lucreaza 2 minute.