DOCUMENTAȚIE

Tema 2

NUME: Stroia Anton-Călin

GRUPA:30225

Cuprins

1.Obiectivul temei

Obiectivul principal al acestui proiect este de a implementa si simula un numar variabil de cozi, în care pot intra un anumit numă de clienti, fiecare client prezentand un id, un timp de ajungere și un timp de servire.

Pentru a se putea realiza acest proiect, următorii pași au fost necesari de parcurs:

A. Definirea clienților, prin declararea atribuțiilor fiecăruia(id, arrivalTime, serviceTime)

B. Creearea cozilor și simularea acestora pentru a putea determina acțiunea fiecărui client

C. Implementarea unei interfețe grafice cu scopul de a face posibilă interacțiunea cu un utilizator

2.Analiza problemei,modelare, scenarii, cazuri de utilizare

Această aplicație functioneză prin introducerea de date de catre utilizatot. Cum ar fi:

Intervalul de timp în care poate ajunge

Intervalul de timp în care poate sa fie servit

Timpul de simulare

Numărul de cozi

Numărul de clienți

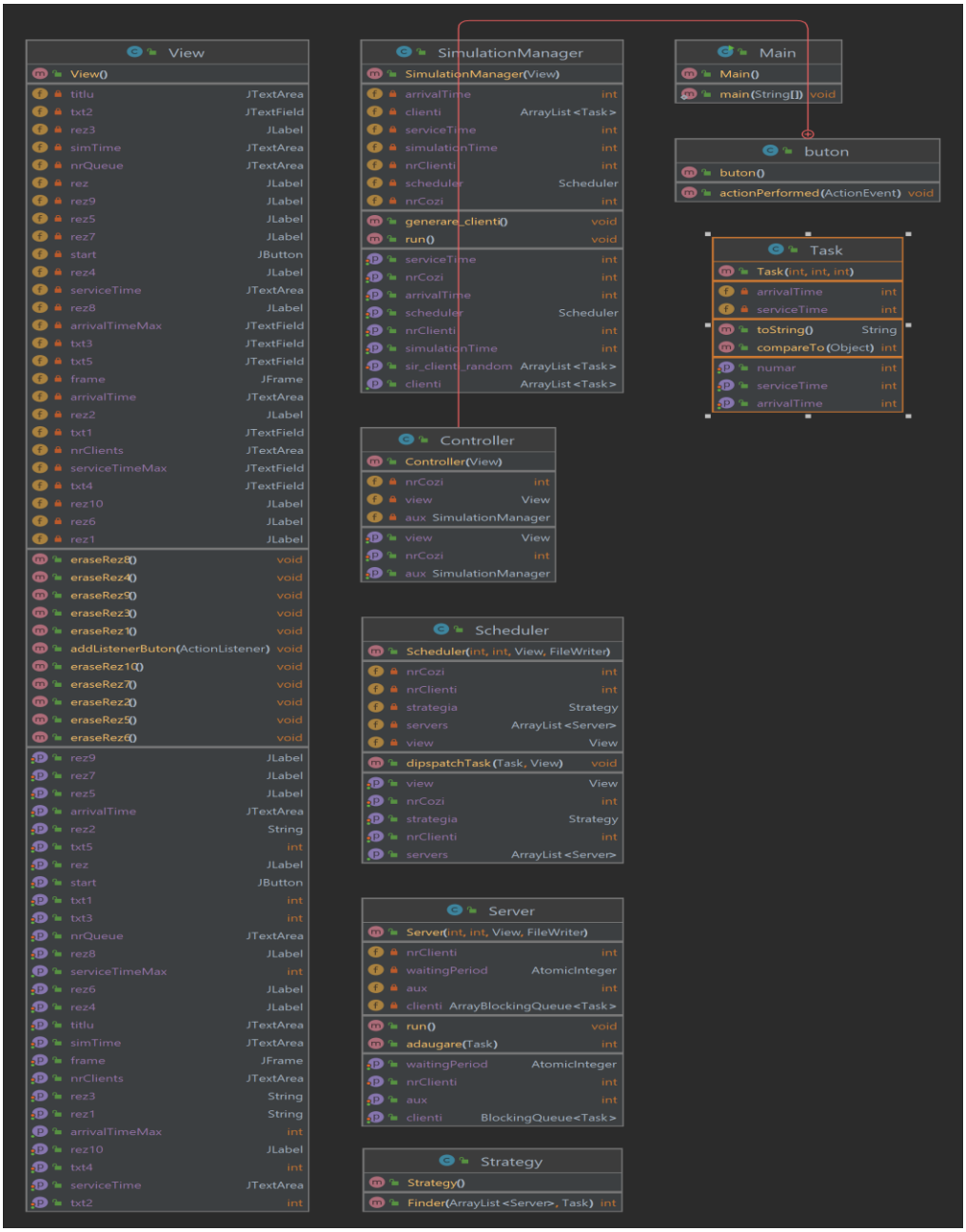
În urma apăsării butonului “start” se poate analiza cum fiecare client, intră si iese din cozile generate.

3.Proiectare

Pentru a face posibilă reprezentarea cozilor, a fost nevoie de folosirea unui concept aparte din POO, si anume “Threads”. Aceste Threadu-ri au scopul de a forma un șir de execuție în interiorul căruia se pot realiza diferite actiuni.

Thread-urile au capacitatea de a aștepta celelalte thread-uri, astfel putând să se realizeze reprezentarea și simularea cozilor noastre. Pentru a scoate un client din coadă, pe baza unei strategii, se poate determina coada care are timpul de asteptare cel mai mic, realizându-se ulterior introducerea clientului, pe baza indexului determinat.

Diagrama UML



4.Implementare

Pentru realizarea acestui proiect s-au creat următoarele clase:

1.Task-> această clasă are scopul de a putea reprezenta clineții ce urmează să fie introduși in cozi.

2.Server->această clasa are scopul de a reprezenta cozile aici declarându-se numărul de clienți, sirul de clienți și alte atribute necesare implementării

3.Scheduler-> această clasă realizează numărul de cozi ce trebuie realizate,si de asemenea determina ce clienți au intrat in rând.

4.Controller-> are rolul de a realiza functionalitatea butonului “start”

5.Main-> pornirea programului

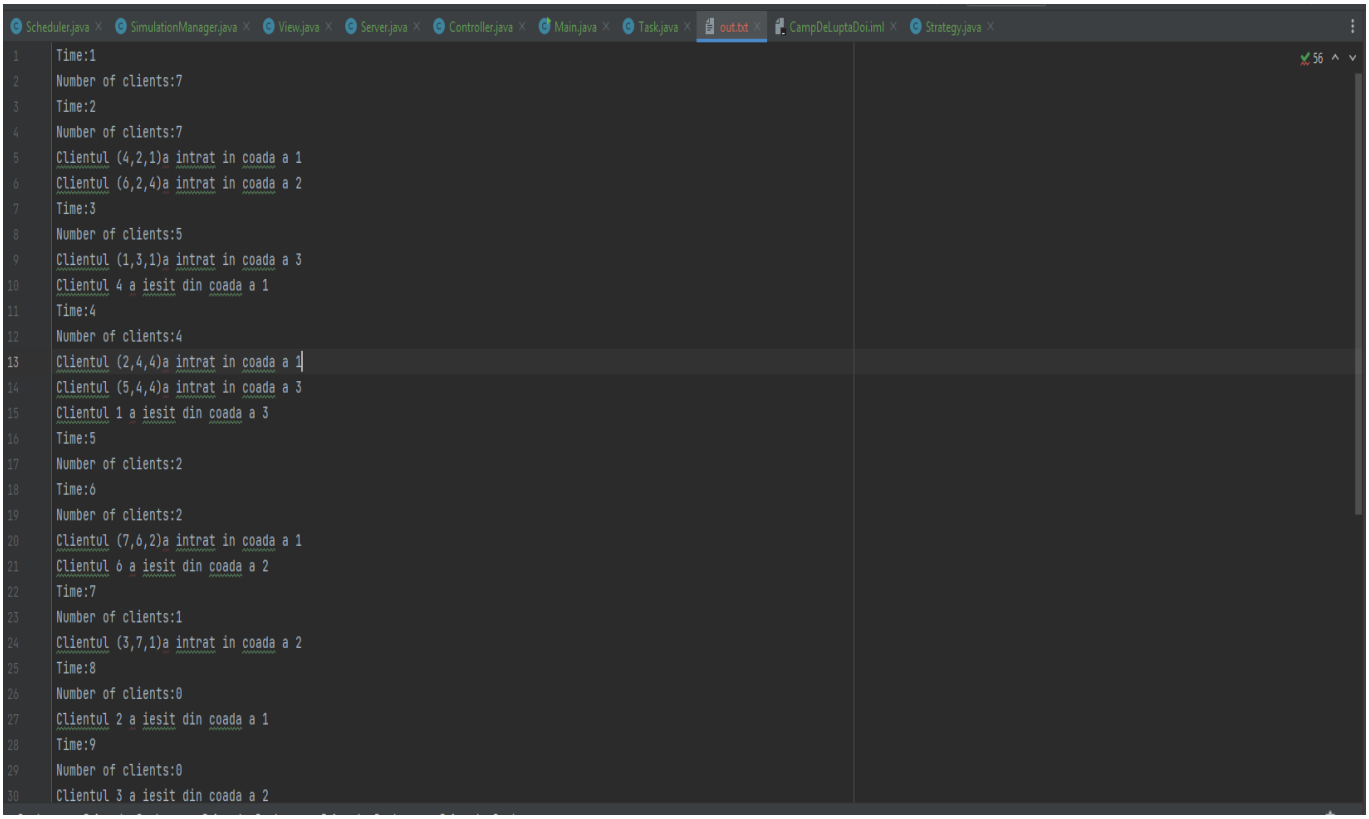
6.SimulationManager -> această clasă are rolul de a genera random clientii, și de a putea coordona timpul in care se realizează acțiunea cozilor. De asemenea în interiorul acestei clase se genereaza fisierul .txt care prezintă rezumatul a ceea ce se întâmplă

6.Strategy -> această clasă are rolul de determina unde se vor introduce fiecare clint, fiecare coada avand un anume waiting time. Strategia constă în introducerea clienților in coada cu timpul de așteptare cel mia mic

5.View-> această clasă reprezintă ceea ce vede utilizatorul, Aici sunt prezentate fiecare componentă a interfeței și locul specific al acesteia.

Interfața grafică

Concluzii


Scheduler.java
SimulationManager.java
View.java
Server.java
Controller.java
Main.java
Task.java
out.txt
CampDeLuptaDoi.iml
Strategy.java
Time:1
Number of clients:7
Time:2
Number of clients:7
Clientul (4,2,1)a intrat in coada a 1
Clientul (6,2,4)a intrat in coada a 2
Time:3
Number of clients:5
Clientul (1,3,1)a intrat in coada a 3
Clientul 4 a iesit din coada a 1
Time:4
Number of clients:4
Clientul (2,4,4)a intrat in coada a 1
Clientul (5,4,4)a intrat in coada a 3
Clientul 1 a iesit din coada a 3
Time:5
Number of clients:2
Time:6
Number of clients:2
Clientul (7,6,2)a intrat in coada a 1
Clientul 6 a iesit din coada a 2
Time:7
Number of clients:1
Clientul (3,7,1)a intrat in coada a 2
Time:8
Number of clients:0
Clientul 2 a iesit din coada a 1
Time:9
Number of clients:0
Clientul 3 a iesit din coada a 2

Bibliografie

https://www.javatpoint.com/how-to-create-a-thread-in-java

https://www.geeksforgeeks.org/java-program-to-create-a-thread/

https://dzone.com/articles/java-thread-tutorial-creating-threads-and-multithr

https://www3.ntu.edu.sg/home/ehchua/programming/java/j4a_gui.html

https://github.com/