DOCUMENTATIE

TEMA *2*

NUME STUDENT: Truța Andrei

GRUPA: 30221

CUPRINS

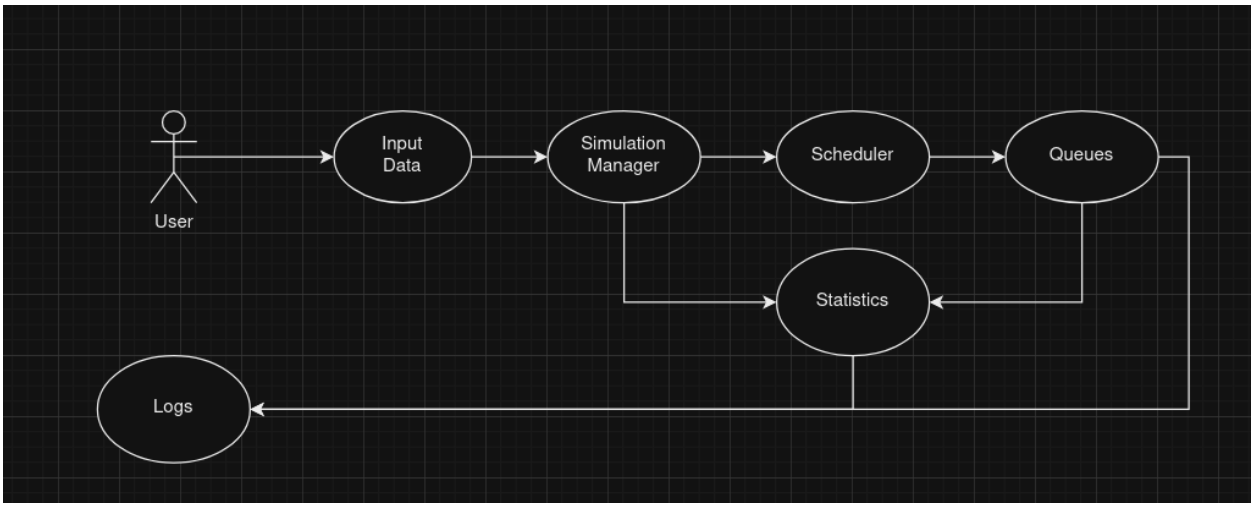
1. Obiectivul temei

Obiectivul principal al temei este de a ne familiariza cu conceptul firelor de execuție prin implementarea unui simulator de cozi paralele. Aplicația simulează N clienți care se împart în Q cozi și au un timp de servire pe care trebuie să-l aștepte până să poată ieși din coadă.

Pașii urmați au fost: definirea „programului” care simulează o coadă pe un thread, implementarea unui planificator cu două opțiuni de repartizare a clienților, apoi implementarea unui manager pentru încapsularea și simularea pe secunde a procesului.

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

Cerințele funcționale ale aplicației reprezintă totalitatea operațiilor pe managerul de simulare împreună cu planificatorul și flow-ul de date, care se leagă bineînțeles de interfața grafică. Managerul de simulare se ocupă de gestionarea timpului și selectarea clienților pentru a fi trimiși in queue. Planificatorul se ocupă de planificarea clienților în cozi, selectând-o pe cea mai optimă.

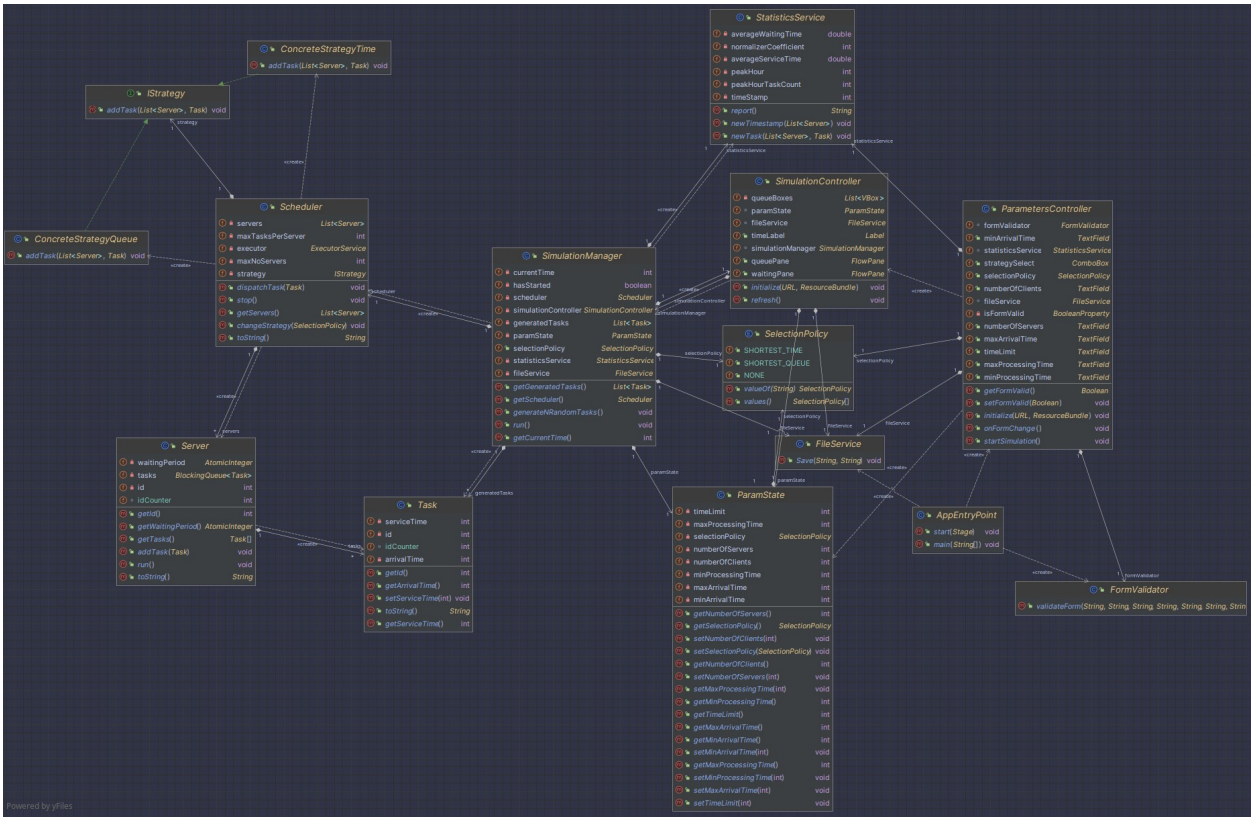


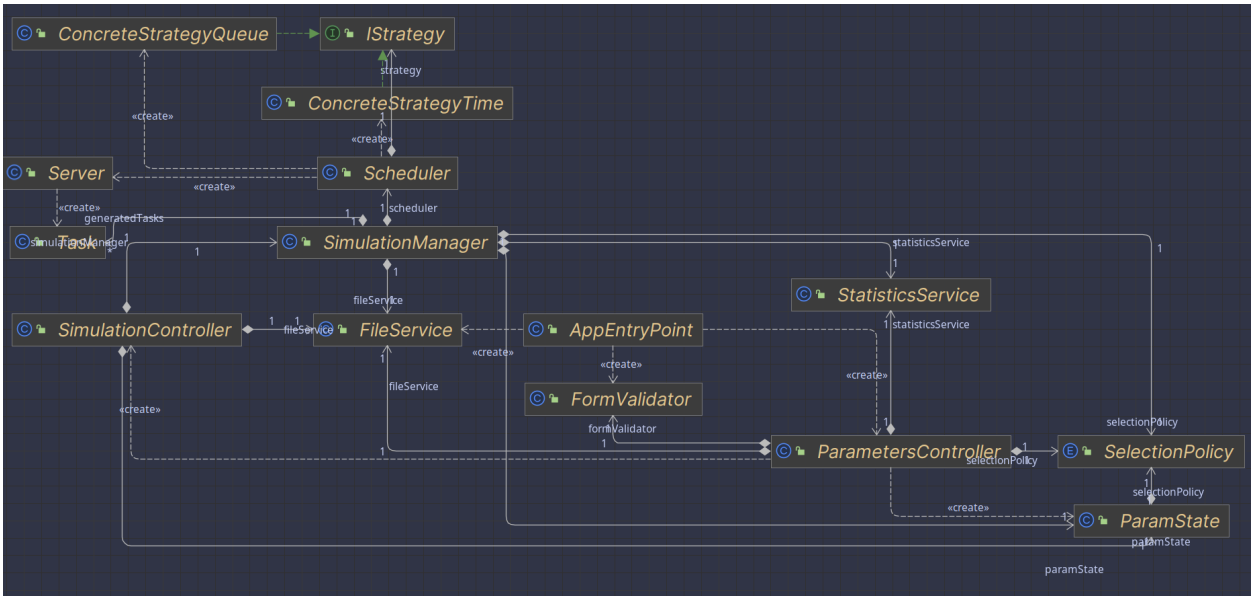
Flow-ul aplicației:

1. Utilizatorul introduce datele
2. Daca datele sunt valide se poate porni simularea
3. Se deschide o fereastră nouă în care se afișează progresul
4. În același timp se înregistrează în loguri

Cerințe nefunctionale: validarea datelor, modul de afișare a progresului.

3. Proiectare





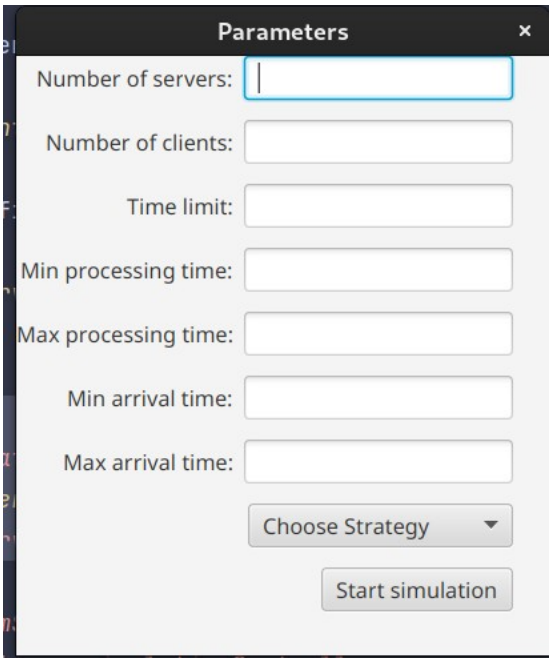
Algoritmii folosiţi sunt cei de gestionare a clienţilor în cozi, Shortest Time şi Shortest Queue.

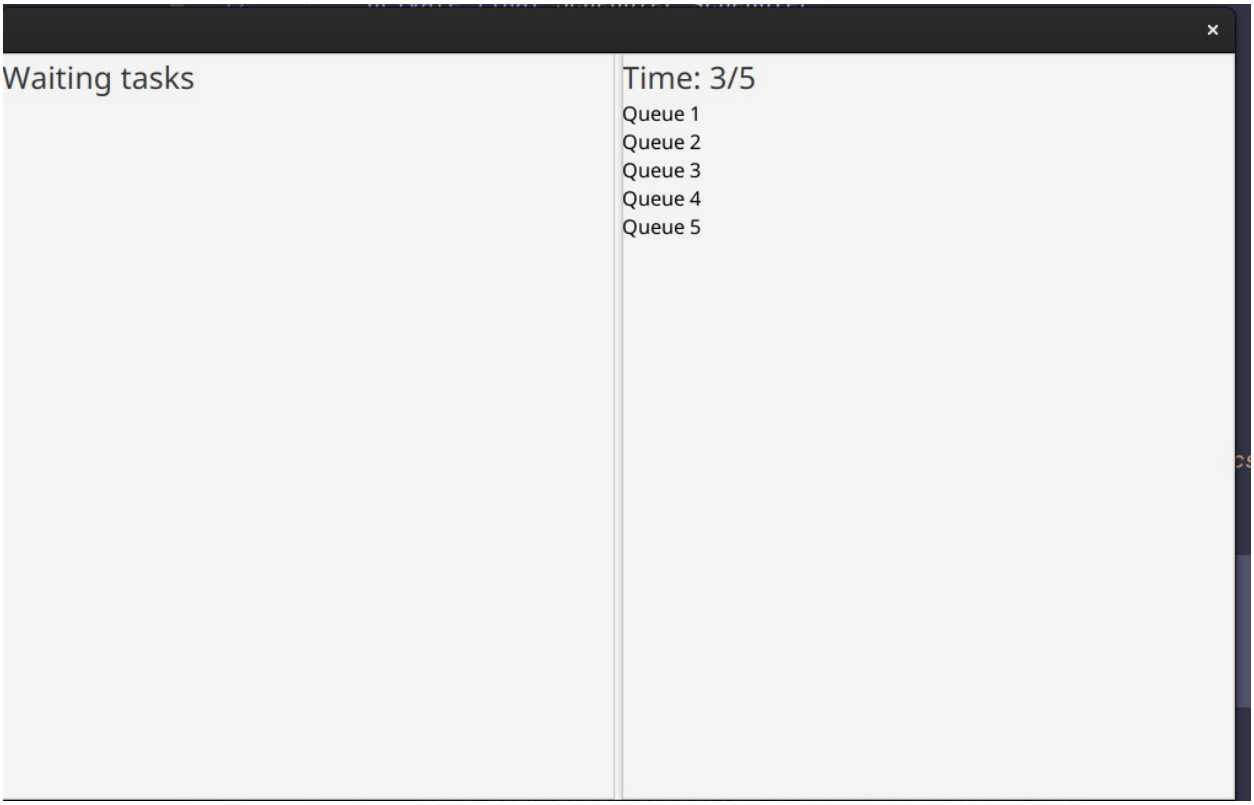
4. Implementare

Clasele folosite cu descriere:

- ParamState(int numberOfServers, int numberOfClients, int timeLimit, int maxProcessingTime, int minProcessingTime, int maxArrivalTime, int minArrivalTime, SelectionPolicy selectionPolicy) – model pentru reprezentarea stării curente a aplicației. Server() - se ocupă de gestionarea unei singure cozi
- Task(int arrivalTime, int processingTime) – model pentru reprezentarea unui client
- ParametersController(FormValidator formValidator, FileService fileService) – controller pentru pagina grafică de input de la utilizator
- SimulationController(ParamState paramState, FileService fileService) – controller pentru pagina de siimulare; instanțiază un SimulationManager și se ocupă de o simulare (pot exista mai multe).
- FileService - conține metoda Save() pentru a salva un string în mod de append într-un fișier, simulând un rolling file adapter de logging
- StatisticsService – se ocupă de statistici, precum average service time, ora de vârf, average waiting time
- ConcreteStrategyQueue – strategia de a adăuga un client în coada cea mai scurtă
- ConcreteStrategyTime - strategia de a adăuga un client în coada cu cel mai mic timp de așteptare
- FormValidator – validator pentru formularul de date de intrare de la utilizator
- Scheduler(int maxNoServers, int maxTasksPerServer) – planificator de clienți în cozi, se foloseşte de Istrategy cu dispatchTask()
- SimulationManager(ParamState paramState, SimulationController simulationController, FileService fileService) – clasa principală care se ocupă de gestiunea simulării

Interfața este implementată cu JavaFX. Are două ferestre,cea de Parameters creează una de Simulation, care la randul ei instanțiază Simulationmanager și face display la datele corespunzatoare.


Parameters
Number of servers:
Number of clients:
Time limit:
Min processing time:
Max processing time:
Min arrival time:
Max arrival time:
Choose Strategy
Start simulation


Waiting tasks
Time: 3/5
Queue 1
Queue 2
Queue 3
Queue 4
Queue 5

5. Rezultate

Pentru rezultatul testelor, a se vedea fisierul queues.log (4000+ linii).

6. Concluzii

Rezultatul acestei teme este o înțelegere mai bună a modului de funcționare ale firelor de execuție și a metodelor de sincronizare ale acestora, precum și aflarea de lucruri noi, precum Thread Pools și modul lor de funcționare.

Ulterior, interfața poate fi îmbunătățită; momentan se află într-un stagiu de nefinisare, lipsindu-i animații și efecte de îmbunătățire a aspectului. Task-urile ar trebui sa fie mai vizibile, în chenare proprii cu un design mai placut, etc.

7. Bibliografie

1. Introduction to Thread Pools in Java - https://www.baeldung.com/thread-pool-java-and-guavaare
2. Java Threads - https://www.w3schools.com/java/java_threads.asp
3. Guide to Blocking Queue - https://www.baeldung.com/java-blocking-queue
4. Idea for Logging - https://www.baeldung.com/java-logging-rolling-file-appenders