DOCUMENTATIE

TEMA 2

NUME STUDENT: MARGINEAN TEODOR IOAN

GRUPA: 6

# CUPRINS

[1. Obiectivul temei 3](#_Toc95297885)

[2. Analiza problemei, modelare, scenarii, cazuri de utilizare 3](#_Toc95297886)

[3. Proiectare 4](#_Toc95297887)

[4. Implementare 5](#_Toc95297888)

[5. Rezultate 7](#_Toc95297889)

[6. Concluzii 8](#_Toc95297890)

[7. Bibliografie 8](#_Toc95297891)

# Obiectivul temei

Obiectivul principal al acestei teme este realizarea unui program care analizeaza cozile de Task care asteapta si determina cea mai optima coada la care se poate aseza urmatorul Task in functie de timpul de asteptare sau de numarul de Task existenti deja in coada. Interfata va avea o functie de pornire a programului care va trebui implementata si folosita. Obiectivele secundare sunt:

* + Realizarea de interfata grafica pentru prluarea numarului de cozi, numarului de Task, timpul de desfasurare a programului, minimul si maximul timpului de asteptare si minimul si maximul timpului de venire;
  + Generarea random de Taski in functie de minimele si maxilmele introduse de catre utilizator;
  + Generarea random de cozi in functie de minimele si maximele introduse de catre utilizator si un thread aferent fiecarei cozi;
  + Realizarea de scheduler si realizarea de Service Manager care contine si un atribut de GUI;
  + Realizarea clasei main pentru pornirea programului;
  + Testarea pe cele trei fisiere de intrare propuse;
  + Generarea fisierelor de iesire (loguri);
  + Implementarea a doua tipuri de strategii: shortest time & shortest queue;

Clasa de timp GUI ale proiectului face afisarea in timp real a cozilor, secunda cu secunda si are denumirea de GUI.

Clasa Controller Start a proiectului incepe o instanta de simulation manager care porneste programul intr-o bucla infinita.

In Bussines Logic avem o enumeratie care stabileste cele doua tipuri de strategii care poti fi applicate pentru punerea unui Task la coada: Shortest Time, Shortest Queue. O interfata care are ca metoda neimplementata metoda de adaugare in coada. Doua clase care dau override la acea metoda si implementeaza strategii ptentru strategiile de mai sus. Si inca doua clase Simualtion Manager care reprezinta (creierul) din spatele programului si Scheduler care reprezinta magazinul la care exista n cozi.

In Model se regasesc clasele de tip Task(Task) si de tip Server.

# Analiza problemei, modelare, scenarii, cazuri de utilizare

Sunt noua use-case uri in

Primul use-case este nr de Taski in urma carauia se preia numarul de Task prin cadrul interfetie grafice cu ajutorul Controller Start.

Al doilea use-case este timpul de rulare in urma carauia se preia timpul de rulare prin cadrul interfetie grafice cu ajutorul Controller Start.

Al treilea use-case este timpul minim de asteptare in urma carauia se preia timpul minim de asteptare prin cadrul interfetie grafice cu ajutorul Controller Start.

Al patrulea use-case este timpul maxim de asteptare in urma carauia se preia timpul maxim de astgeptare prin cadrul interfetie grafice cu ajutorul Controller Start.

Al cincelea use-case este timpul maxim de procesare in urma carauia se preia timpul maxim de procesare prin cadrul interfetie grafice cu ajutorul Controller Start.

Al saselea use-case este timpul minim de procesare in urma carauia se preia timpul minim de procesare prin cadrul interfetie grafice cu ajutorul Controller Start.

Al saptelea use-case este numarul de cozi in urma carauia se preia numarul de cozi prin cadrul interfetie grafice cu ajutorul Controller Start.

Al optulea use-case este shortest time strategy care incarca Taskii in cozi in functie de timpul fiecarei cozi si le ordoneza in scheduler.

Al noualea use-case este shortest queue strategy care incarca Taskii in cozi in functie de numarul de Task al fiecarei cozi si le ordoneza in scheduler.

Al zecelea use-case este afisarea peak-ului in interfata grafica cat is in fisierele de tip log.txt;

Al unsprezecelea use-case este afisarea timpului avrege de venire in interfata grafica cat is in fisierele de tip log.txt;

Al doisprezecelea use-case este afisarea timpului averege de prelucrare in interfata grafica cat is in fisierele de tip log.txt;

# Proiectare

In cadrul acestei teme au fost folosite patru pachete denumite: Business Logic,GUI , Model .Pachetul Business Logic are patru clase denumite Scheduler, Shortest Queue Strategy, Shortest Time Strategy, Simulation Manager si o Strategy si o enumeratie Politica Selectie. Pachetul Model are doua clase: Server si Task. Pachetul GUI are trei clase: Controller Start, GUI si MainTest.

# Implementare

Proiectul este un proiect complex care are mai multe tehnici de programare prezente in acesta.

Tot proiectul pleca de la metoda main in clasa GUI Log In din pachetul gui. Aceasta instantiaza un gui care prezinta trei butoane. Fiecare buton are un controller instantiate in constructorul view-ului. Prin apasarea butonului Log In se instantiaza un nou gui care prezinta mai multe textfield-uri care trebuie comeltate pentru a putea continu executia programului. Exista alte patru butoane care realizeaza operatii aspura tabelului GuiTask din baza de date.

Prin apasarea butonuluilist, se intra in metoda insert din clasa Simulation Manager, in care se iau textul se inlocuiesc elementele notate din text field-uri cu elementele transmise ca parametrii, elemente preluate in controllerTaskAdaugare, functia continua executia si face legatura la lista de date prin transform din Simulation Manager si scrie in consola o noua instanta de lista.

Prin apasarea butonului Delete, se intra in metoda delete din clasa Delivery Sercice, in care se iau textul din text frame se inlocuiesc elementele cu elementele transmise ca parametrii, elemente preluate in controllerTaskAdaugare, functia continua executia si face legatura la baza de date prin intermediul clasei ConnectionFactory si sterge din tabelul Task instanta de Simulation Manager cu id-ul transmis ca parametru.

Prin apasarea butonului Modify, se intr in metoda update din clasa Simulation Manager , in care se iau textul se inlocuiesc elementele? cu elementele transmise ca parametrii, elemente preluate in controllerTask Adaugare, functia continua executia si face legatura la baza de date prin intermediul clasei ConnectionFactory si modifica din tabelul Task instanta de Task cu id-ul egal cu id-ul trimis ca parametru si valoare de pe coloanal age este inlocuita cu valoara transmisa ca parametru.

Aceasta este functiomalitate asociata tabelului Task. Astfel se satisfac toate cerintele care apar in preznetare pentru tabele List.

Prin apasarea butonului List se instantiaza un nou gui care prezinta mai multe textfield-uri care trebuie comeltate pentru a putea continu executia programului. Exista alte patru butoane care realizeaza operatii aspura tabelului List din baza de date.

Prin apasarea butonului Select, se intr in metoda insert din clasa Simulation Manager, in care se iau textul se inlocuiesc elementele cu cu elementele transmise ca parametrii, elemente preluate in controllerTaskAdaugare, functia continua executia si face legatura la baza de date prin intermediul clasei ConnectionFactory si returneaza toate datele referitaore la Task-ul cu id-ul transmis ca parametru. Aceasta metoda din Simulation Manager se apeleaza de mai mutle ori pentru a fi preluate toate datele din tabel. Dupa preluarea datelor cu ajutorul metodei refelction se determina antetul tabelului si este instantiate o noau clasa de tip viewTabel in care se introduc intr-un JTable taote datele si afisat pe ecran.

Prin apasarea butonului Add, se intra in metoda insert din clasa Simulation Manager, in care se iau textul se inlocuiesc elementele cu cu elementele transmise ca parametrii, elemente preluate in controllerListAdaugare, functia continua executia si face legatura la baza de date prin intermediul clasei ConnectionFactory si adauga in tabelul List o noua instanta de List.

Prin apasarea butonului Delete, se intra in metoda delete din clasa Simulation Manager, in care se iau textul se inlocuiesc elementele cu cu elementele transmise ca parametrii, elemente preluate in controllerTaskAdaugare, functia continua executia si face legatura la baza de date prin intermediul clasei ConnectionFactory si sterge din tabelul List instanta de List cu id-ul transmis ca parametru.

Prin apasarea butonului Edit, se intr in metoda update din clasa Simulation Manager, in care se iau textul se inlocuiesc elementele cu cu elementele transmise ca parametrii, elemente preluate in controllerListAdaugare, functia continua executia si face legatura la lista prin intermediul metodei transform si modifica din mao-ul de Order instanta de Order cu id-ul egal cu id-ul trimis ca parametru si valoare de pe coloanal age este inlocuita cu valoara transmisa ca parametru.

Prin apasarea butonului List se instantiaza un nou gui care prezinta mai multe textfield-uri care trebuie comeltate pentru a putea continu executia programului. Exista alte patru butoane care realizeaza operatii aspura tabelului List din baza de date.

Prin apasarea butonului Start, se intr in metoda insert din clasa Simulation Manager, in care se iau textul se inlocuiesc elementele cu cu elementele transmise ca parametrii, elemente preluate in controllerListAdaugare, functia continua executia si face legatura la baza de date prin intermediul clasei ConnectionFactory si returneaza toate datele referitaore la List-ul cu id-ul transmis ca parametru. Aceasta metoda din Simulation Manager se apeleaza de mai mutle ori pentru a fi preluate toate datele din tabel. Dupa preluarea datelor cu ajutorul metodei refelction se determina antetul tabelului si este instantiate o noau clasa de tip viewTabel in care se introduc intr-un JTable taote datele si afisat pe ecran.

Aceasta este functionalitatea programului asociata tabelului List. Astfel se satisafac taote criterile din cerinta legate de tabelul Task.

Prin apasarea butonului Task se instantiaza un nou gui care prezinta mai multe textfield-uri care trebuie completate pentru a putea continu executia programului. Exista un butoan care realizeaza crearea unei noi comenzi.

Prin apasarea butonului Generare, se inta in metoda insert din clasa Simulation Manager, in care se iau textul se inlocuiesc elementele cu cu elementele transmise ca parametrii, elemente preluate in controllerListAdaugare, functia continua executia si face legatura la baza de date prin intermediul clasei ConnectionFactory si adauga in tabelul ordered doar daca ceea ce introduce ca parametru este mai mica decat stocul din produs, pentru a putea compara cele doua se foloseste o apelare a metodei select din Simulation Manager, care retureneaza sub forma de string toate datele necesare pentru produsul cu id-ul timis ca parametru. In urma verificarii daca rezultatul este pozitiv se peleaza o metoda de afisare care va genera cu ajutorul textwriter-ului un text care va contine Taskul, prdusul si cantitatea care va fi solicitata.

Astfel se satisfac toate cerintele necesare pentru acest proiect.

Rezumat:

Clasa MainTest porneste un Simulation Manager care porneste un GUI. Din GUI se peiau datele introduse in TextField-uri prin intermediul sett-eri, get-eri. Prin intermediul clasei ControllerStart se porneste un Simualtion Manager care porneste u Scheduler care este format din mai multe Servere stocate intr-o lista. Fiecare Server are o lista de Task uri si un waiting time. In functie de strategia aleasa in fiecare secunda din metoda run a Simulation Manager se realizeaza transpunerea clientilor in functie de timpul de venire in cozile din Servere. Clientii sunt generate random cu ajutorul metodie randomInt care este folosita atat pentru generarea de timp de venire cat si pentru generarea de timp de prelucrare. La fiecare itratie se afiseaza in Text area secunda cozile si clientii din fiecare coada plus clientii care urmeaza sa vina.

# Rezultate

In urma realizarii programului, am reusit sa indeplinesc taote cerintele. Am implementat si pentru tipul de strategie shortest queue.

* Sa realizez o interfata grafica;
* Sa realizez preluarea datalor din interfata grafica;
* Sa realizez strategia de tip shortest queue;
* Sa realizez strategia de tip shortest time;
* Sa realizez afisarea in interfata grafica.
* Sa realizez peak-ul;
* Sa realizez average time pentru venire;
* Sa realizez average time pentru prelucrare;
* Sa realizez afisarea acestora la terminarea programului;
* Sa realizez afisarea intr-un fisier text denumit Log.txt;
* Sa realizez exemplul unu;
* Sa realizez exemplul doi;
* Sa reazliez exemplul trei;
* Sa le afisez separat pe fiecare intr-un log;

# Concluzii

In concluzie in urma realizarii temei, am invatat sa folosim conversie de string prin intermediul parsarii, am revenit asupra conceptelor invatate si le-am reimprospatat, am refacut un model architectural de tip model controller view. Am invatat sa:

* avem un proiect structurat;
* sa lucrez cu threaduri;
* sa fac legatura dintre un view si o alta clasa;
* sa realizez serealizarea, dar si deserealizarea intr-un proiect cu diferite fisiere;

In urma acestui proiect am invatat sa lucrez sub presiune, sa imi ordonez timpul astefel

incat sa nu mai am pe ultima suta de metrii atat de mult de lucru. Am invatat din greseli, am amanat teme dar totusi am reusint intr-un final.

# 7. Bibliografie

<https://dsrl.eu/courses/pt/>

<https://cadredidactice.ub.ro//sorinpopa/files/2018/10/L1_diagrame_use_case.pdf>

<https://muhaz.org/io-introducere.html?page=2>