

UNIVERSITATEA TEHNICĂ CLUJ- NAPOCA

AUTOMATICĂ ȘI CALCULATOARE, AN II



Lucrare laborator – Assignment V

Aplicatie care analizeaza comportamentul
unei persoane pe baza de senzori

Profesor curs: Prof. Dr. Ing. Cristina Pop

Profesor laborator: Teodor Petrican

Student: Jitaru Andrei

Grupa: 30229

CUPRINS

1. Introducere – Obiectivul temei
2. Analiza problemei
 - a. Asumpții
 - b. Modelare
 - c. Scenarii
 - d. Cazuri de utilizare
3. Proiectare
 - a. Diagrama UML
 - b. Structuri de date utilizate în implementare
 - c. Proiectare clase
 - d. Interfata Utilizator
4. Implementare
5. Rezultate
6. Concluzii și dezvoltări ulterioare

1. Introducere – obiectivul proiectului

Obiectivul acestui proiect este de a verifica constintele studentului referitoare la utilizarea expresiilor lambda, acestea trebuind imbinat cu conceptul de procesare de stream-uri. Intregul proiect ofera posibilitatea de a ne familiariza cu diferitele caracteristici și proprietăți ale conceptelor care produc realizarea sa. Privind mai in detaliu tema, s-a dorit implementarea în limbaj de programare Java a unei aplicatii care sa analizeze comportamentul unei persoane. Acest comportament este urmarit de catre un set de senzori, iar datele transmise de acestia vor fi stocate intr-un fisier de unde aplicatia le va prelua. Proiectul a fost realizat cu scopul de a demonstra atât înțelegerea asupra temei și cerintelor date cât și etalarea aptitudinilor de implementare a acestora în contextul limbajului de programare mai sus menționat.

Toate aspectele enumerate anterior au fost verificate prin implementarea unei aplicații simple. Aceasta a fost realizata conform cerintelor temei, fiind adoptat un stil cat mai minimalist pentru o profunda intelegere a utilizatorului.

2. Analiza problemei

Proiectul, care în speță reprezintă un sistem care faciliteaza urmarirea eficienta a activitatilor unui individ pe parcursul unei perioade de timp, sugerează un proces complex care necesită atenție la fiecare pas. O astfel de abordare este absolut necesară deoarece fiecare operație este unică în ceea ce priveste functionalitatea sa.

a. Asumptii

Pentru a obine rezultate optime in urma utilizarii aplicatiei este necesara impunerea unor asumptii. Aceste asumptii se aplica datelor din fisierul "Activities.txt" care reprezinta trackingul activitatilor individul de-a lungul

perioadei de timp. In acest sens, s-a decis faptul ca o activitate nu trebuie sa dureze mai mult de 24 de ore. De asemenea, ultima activitate din logger-ul de activitati nu trebuie sa aiba StartTime-ul intr-o zi si EndTime-ul in urmatoarea. De exemplu, aparitia liniei "2011-12-11 15:43:51 2011-12-12 21:41:48 Spare_Time/TV " este ilegala din cauza incalcarii ambelor cerinte enumerate mai sus.

b. Modelare

Cerința temei și anume de a se realiza un sistem care sa urmareasca comportamentul unei persoane pe o perioada limitata de timp care sa faciliteze in acelasi timp analiza si gestionarea acestui comportament poate fi implementata in variate moduri. Prima si cea mai evidenta metoda care ar putea fi abordata de catre un programator ar fi citirea si parsarea datelor din fisier prin metode clasice de citire/parsare. Datele ar putea fi stocate in cele din urma intr-o lista iar de acolo, pentru implementarea cerintelor temei, ar trebui realizate o suita de operatii costisitoare care ar reduce din performantele aplicatiei.

Din aceasta cauza, obiectivele temei ofera o noua abordare care sa reduca pierderile legate de performanta proiectului. Daca privim problema din punct de vedere al vitezei de lucru dar si a eficientei, atunci fara indoiala o solutie baza pe expresii lambda si procesare de streamuri este cea evidenta. Astfel, s-a optat pentru citirea datelor din fisier ca stream si modelarea acestora in cadrul sau. Operatiile care vor fi aplicate in continuare pe aceste date vor fi realizate de asemenea prin modelare si gestionare de streamuri.

c. Scenarii

O serie de etape trebuie urmate pentru ca aplicatia să fie capabilă de a returna rezultate clare și corecte. Totalitatea acestor etape dau naștere unei serii de scenarii care definesc funcționalitatea programului. În acest sens, pentru a obține un anumit rezultat, diferite evenimente trebuie să aibă loc. Interfata simplista este alcatuita doar din butoate care prin apasarea lor realizeaza anumite task-uri ale temei.

Pasi de utilizare ai aplicatiei:

- Daca se doreste numararea zilelor distincte in cadrul carora a fost urmarita activitatea
 - Se apasa butonul "TASK 1";
 - Se verifica fisierul "OperationResult" din folder-ul aplicatiei pentru vizualizarea rezultatului operatiei.
- Daca se doreste numararea aparitiilor fiecarei activitati de-a lungul perioadei de tracking
 - Se apasa butonul "TASK 2";
 - Se verifica fisierul "OperationResult" din folder-ul aplicatiei pentru vizualizarea rezultatului operatiei.
- Daca se doreste cautarea activitatilor care au o durata totala mai mare de 10 ore
 - Se apasa butonul "TASK 3";
 - Se verifica fisierul "OperationResult" din folder-ul aplicatiei pentru vizualizarea rezultatului operatiei.
- Daca se doreste numararea aparitiilor fiecarei activitati pentru fiecare zi in cadru careia s-a efectuat urmarirea activitatii
 - Se apasa butonul "TASK 4";
 - Se verifica fisierul "OperationResult" din folder-ul aplicatiei pentru vizualizarea rezultatului operatiei.
- Daca se doreste filtrarea activitatilor care au 90% din durata de monitorizare mai mica de 5 minute
 - Se apasa butonul "TASK 1";
 - Se verifica fisierul "OperationResult" din folder-ul aplicatiei pentru vizualizarea rezultatului operatiei.

d. **Cazuri de utilizare**

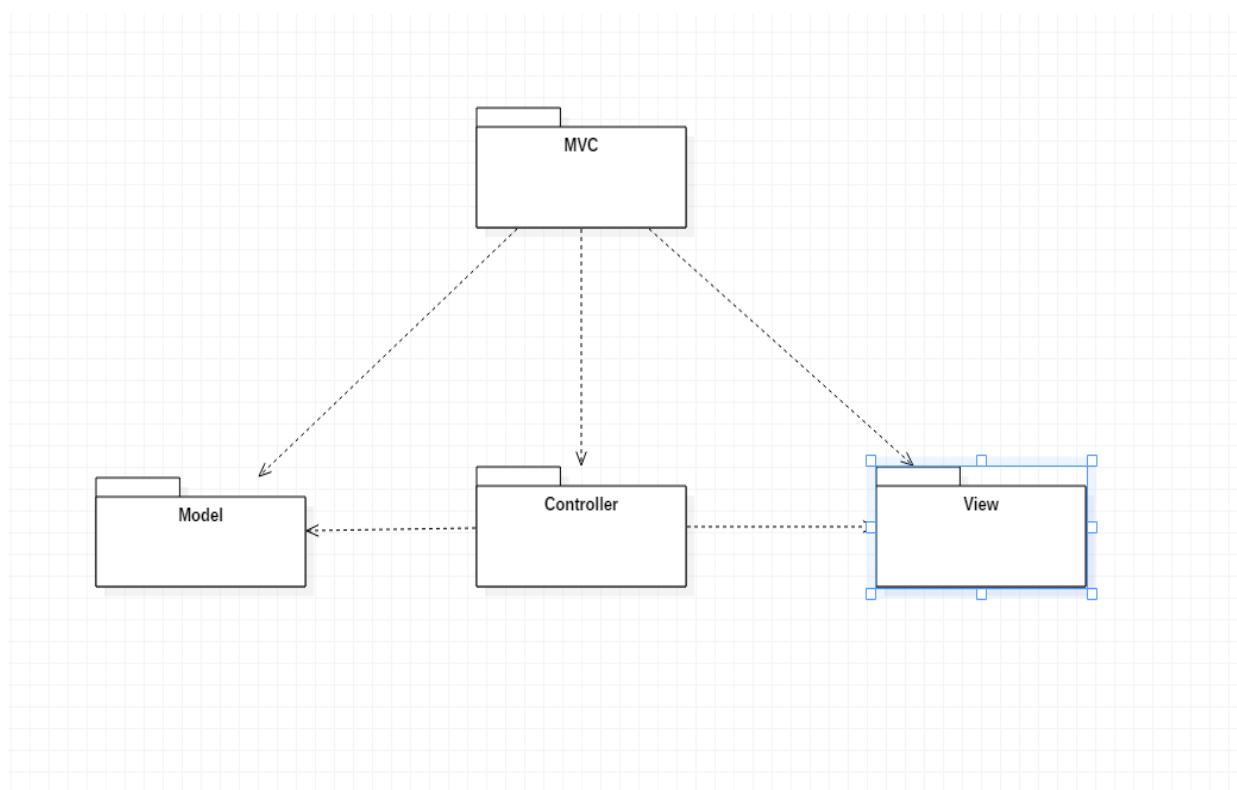
Acest proiect este realizat cu scopul de a urmări comportamentul unei persoane pe o perioadă limitată de timp dar și pentru a facilita analiza și gestionarea acestui comportament. Luând în considerare aceste detalii putem

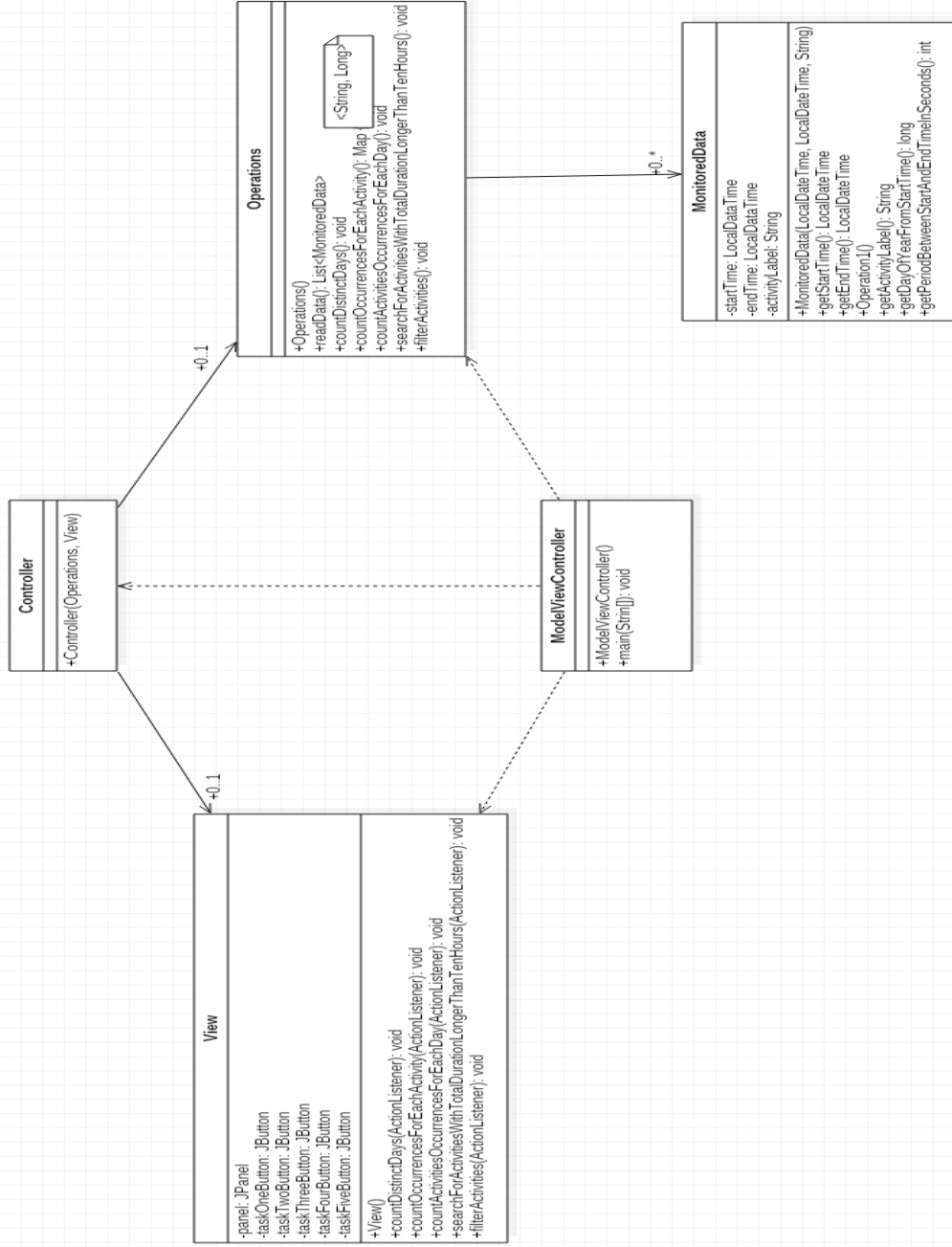
spune fara tragere de inima ca aplicatia ar putea fi utilizata in scopuri practice. Din momemnt ce dispune de o interfă grafică simplă cu o utilizare la fel de banală, putem deduce ca ar putea fi folosită de orice persoană care ar dori sa gestioneze datele primite de exemplu de la o bratara fitness sau de la un telefon.

3. Proiectare

a. Diagrame UML

În diagramele UML de mai jos sunt prezentate toate pachetele si clasele, cu attributele și relațiile dintre acestea. O analiză detaliată a acestor diagrame este prezentată în capitolul 4 al acestei documentații.





b. Structuri de date utilizate în implementare

Funcționarea aplicației este strâns legată de utilizarea de structuri de date în cadrul implementării acesteia. Astfel, pentru a putea rezolva task-urile impuse de tema au fost utilizate diferite structuri de date. Preponderent au fost folosite structuri de tip Map însă, în funcție de operația de implementat au fost utilizate și structuri de tip List.

c. Proiectare clase

Proiectul, ca un tot unitar, este divizat în 5 clase care permit transpunerea cerinței și temei în limbaj de programare Java. Clasele au dimensiuni, alcătuiri și roluri variate.

Clasa „MonitoredData” reprezintă scheletul proiectului. Aceasta conține mijlocul de reprezentare al datelor din fișierul în care s-a salvat desfășurarea activităților individului în funcție de intervalul de timp determinat de începerea unei activități și finalizarea acesteia. Clasa "Operations" stochează operațiile care pot fi realizate cu datele obținute din fișier. Astfel, pot fi realizate 5 operații, pentru fiecare fiind creată câte o metodă: numărarea zilelor distincte din perioada de monitorizare, numărarea aparițiilor fiecărei activități de-a lungul perioadei de tracking, căutarea activităților care au o durată totală mai mare de 10 ore, numărarea aparițiilor fiecărei activități pentru fiecare zi în cadrul căreia s-a efectuat urmărirea activității, filtrarea activităților care au 90% din durata de monitorizare mai mică de 5 minute. Clasa View este cea care realizează interfața grafică a programului și permite interacțiunea cu utilizatorul. Aceasta și clasele din pachetul Model sunt legate una de celelalte prin clase Controller. Astfel conexiunea operațiilor la interfață este realizată. Clasa Mvc este cea care pune la loc toate componentele aplicației.

d. Interfața utilizator

Interfața cu utilizatorul are un aspect minimalist pentru a putea fi utilizată cât mai ușor de orice persoană în parte. S-a optat doar pentru introducerea strictului necesar. Astfel, interfața este alcătuită doar dintr-un meniu cu 5 butoane, fiecare având un nume caracteristic pentru operația care o efectuează. Prin apăsarea acestor butoane vor fi efectuate operații specifice task-urilor temei, rezultatele acestora fiind afișate în fișierul "OperationResult" din folderul aplicației.

4. Implementare

Întreaga aplicație este implementată aferent modelului architectural "Model-View-Controller". Alegerea acestui model se datorează izolării logicii față de considerentele interfeței cu utilizatorul, rezultând o aplicație unde aspectul vizual și nivelele inferioare sunt mai ușor de modificat, fără a afecta alte nivele.

Clasa pivot a proiectului este fără îndoială clasa `MonitoredData`. Aceasta conține mijlocul de reprezentare al datelor din fișierul în care s-a salvat desfășurarea activităților individului. `MonitoredData` are 3 atribute private: `startTime`, `endTime` și `activityLabel`. `StartTime` și `endTime` sunt atribute de tip `LocalDateTime` și înfățișează momentul în care activitatea a început, respectiv când a fost definitivată. Atributul `activityLabel` este de tip `String` și reprezintă numele activității care are loc în intervalul de timp determinat de câmpurile `startTime` și `endTime`. Pentru această clasă au fost create metodele accesoare: `public LocalDateTime getStartTime()`, `public LocalDateTime getEndTime()` și `public String getActivityLabel()` care returnează atributele `startTime`, `endTime` și `activityLabel` pentru un obiect de tipul `MonitoredData`. De asemenea, a fost creată metoda `public long getDayOfYearFromStartTime()` care returnează numărul zilei din `startTime` raportat la un an de zile (ex: ziua 3 = 3 ianuarie). Metoda `public int getPeriodBetweenStartAndEndTimeInSeconds()` returnează perioada în secunde dintre `startTime` și `endTime`.

Clasa `Operations` dictează operațiile care pot fi efectuate la nivelul operației. Aceasta are un singur atribut, și anume o listă de obiecte de tip

MonitoredData care reprezinta datele preluate din fisierul de tracking. Datele sunt obtinute in momentul apelarii constructorului din clasa MonitoredData care la randul lui apeleaza metoda statica readData pentru a oferi atributului clasei valorile din fisier. Pentru inceput metoda preia datele din fisier intr-un stream. Dupa care valorile vor fi mapate in functie de o operatie de split care creeaza un map cu obiecte de tip vector[String]. O a doua Mapare va avea loc pentru a transforma obiectele din stream in obiecte de tip MonitoredData, acestea urmand a fi colectate intr-o lista.

Prima operatie care poate fi executata este cea de numarare a zilelor distincte. Cu alte cuvinte, dorim sa numaram cate zile distincte sunt in loggerul de activitati. In acest scop a fost creata metoda public void countDistinctDays(). Astfel, lista care contine elementele care au fost citite din fisier este transformata din nou intr-un stream asupra caruia este realizata o mapare care returneaza o noua structura. Aceasta contine numarul fiecarei zile raportat la un an de zile. Pentru a obtine numarul de zile distincte din logger va fi aplicata o operatie de distinct urmata de o operatie de count. Rezultatul va fi scris intr-un fisier.

Urmatoarea operatie care poate fi executata este cea de numarare a aparitiilor fiecarei activitati. Cu alte cuvinte, dorim sa numaram de cate ori apare fiecare activitate in loggerul de activitati. In acest scop a fost creata metoda public Map<String, Long> countOccurrencesForEachActivity(int permission). Aceasta primeste ca argument un intreg care reprezinta o permisiune. Lista care contine elementele care au fost citite din fisier este transformata din nou intr-un stream asupra caruia se apeleaza collect pentru a grupa intr-un map activitatile si numarul de aparitii ale acestora. Daca permisiunea primita ca parametru are valoare 1 atunci rezultatul va fi scris intr-un fisier.

Cea de-a treia operatie care poate fi executata este cea de cautare a activitatilor care au durata totala mai mare de 10 ore. In acest sens a fost creata metoda `public void searchForActivitiesWithTotalDurationLongerThanTenHours()`. Astfel, lista care contine elementele care au fost citite din fisier este transformata intr-un stream asupra caruia se apeleaza collect pentru a grupa fiecare activitate cu perioada totala a acesteia. Cheia pentru acest map va fi numele activitatii iar valoarea,

perioada totala. Pentru a obtine rezultatul cerut apelam entrySet urmat de stream pentru a aduce valorile noului map in stream. Asupra acestora se apeleaza filter pentru a obtine doar activitatile care au o perioada mai lunga de 10 ore. Rezultatul va fi convertit intr-un map care va avea ca valoare numele activitatii si perioada convertita intr-un obiect de tip Period. Rezultatul va fi scris intr-un fisier.

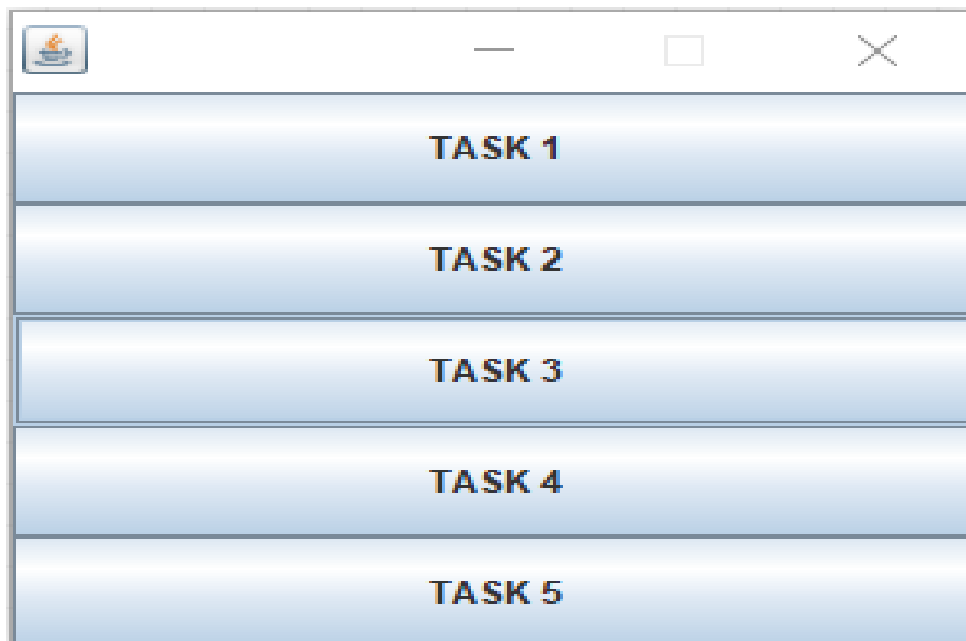
Cea de-a treia operatie care poate fi executata este cea de numarare a aparitiilor activitatilor pentru fiecare zi. In acest sens a fost creata metoda public void countActivitiesOccurrencesForEachDay(). Astfel, lista care contine elementele care au fost citite din fisier este retransformata intr-un stream asupra caruia va fi apelata metoda de collect care grupeaza activitatile si numarul de aparitii a acestora pentru fiecare zi distincta din logger. Rezultatul este scris intr-un fisier.

Ultima operatie care poate fi executata este cea de filtrare a activitatilor care au 90% din durata de monitorizare mai mica de 5 minute. In acest sens a fost creata metoda public void filterActivities(). Astfel, lista care contine elementele care au fost citite din fisier este retransformata intr-un stream asupra caruia va fi aplicat un filter pentru a obtine doar activitatile care au durata de monitorizare mai mica de 5 minute. Urmeaza apelul unui collect pentru a grupa numarul de aparitii al acestor activitati in functie de nume. Map-ul rezultata va fi retransformat in stream si va fi filtrat astfel incat sa ramana doar activitatile care au 90% din durata de monitorizare mai mica de 5 minute. Valorile rezultate vor fi colectate intr-o lista si scrise intr-un fisier.

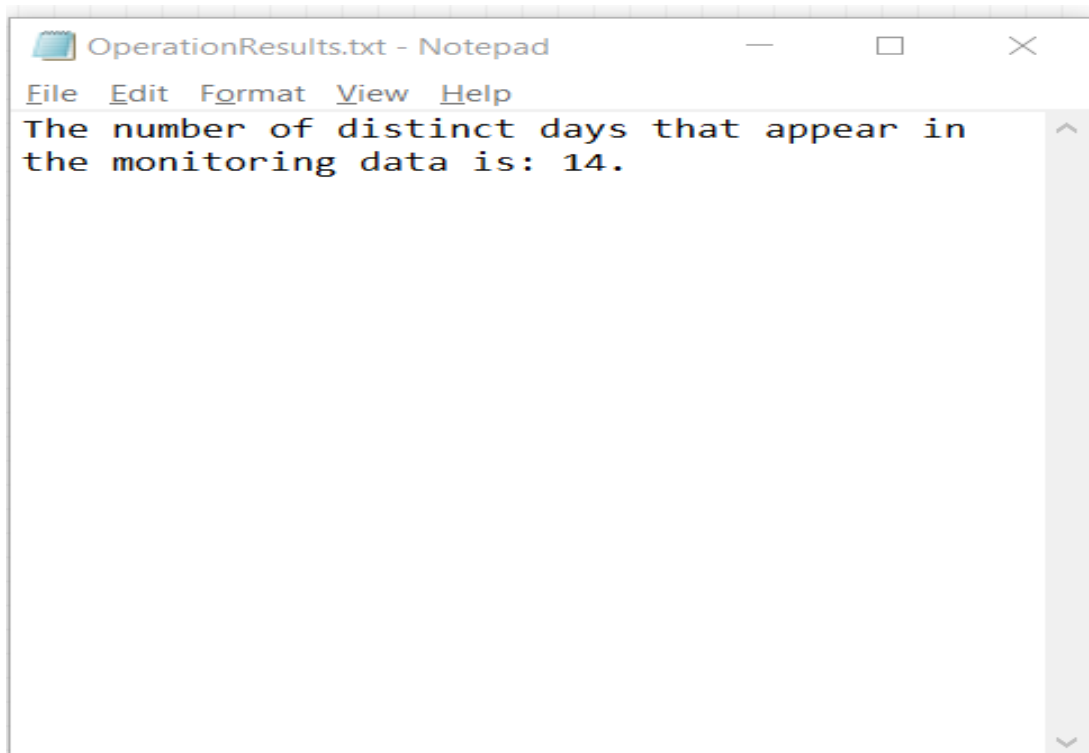
6. Rezultate

Au fost executate o serie de teste pentru a exemplifica diferitele rezultate care se pot obtine in urma interactiunii utilizatorului cu interfața.

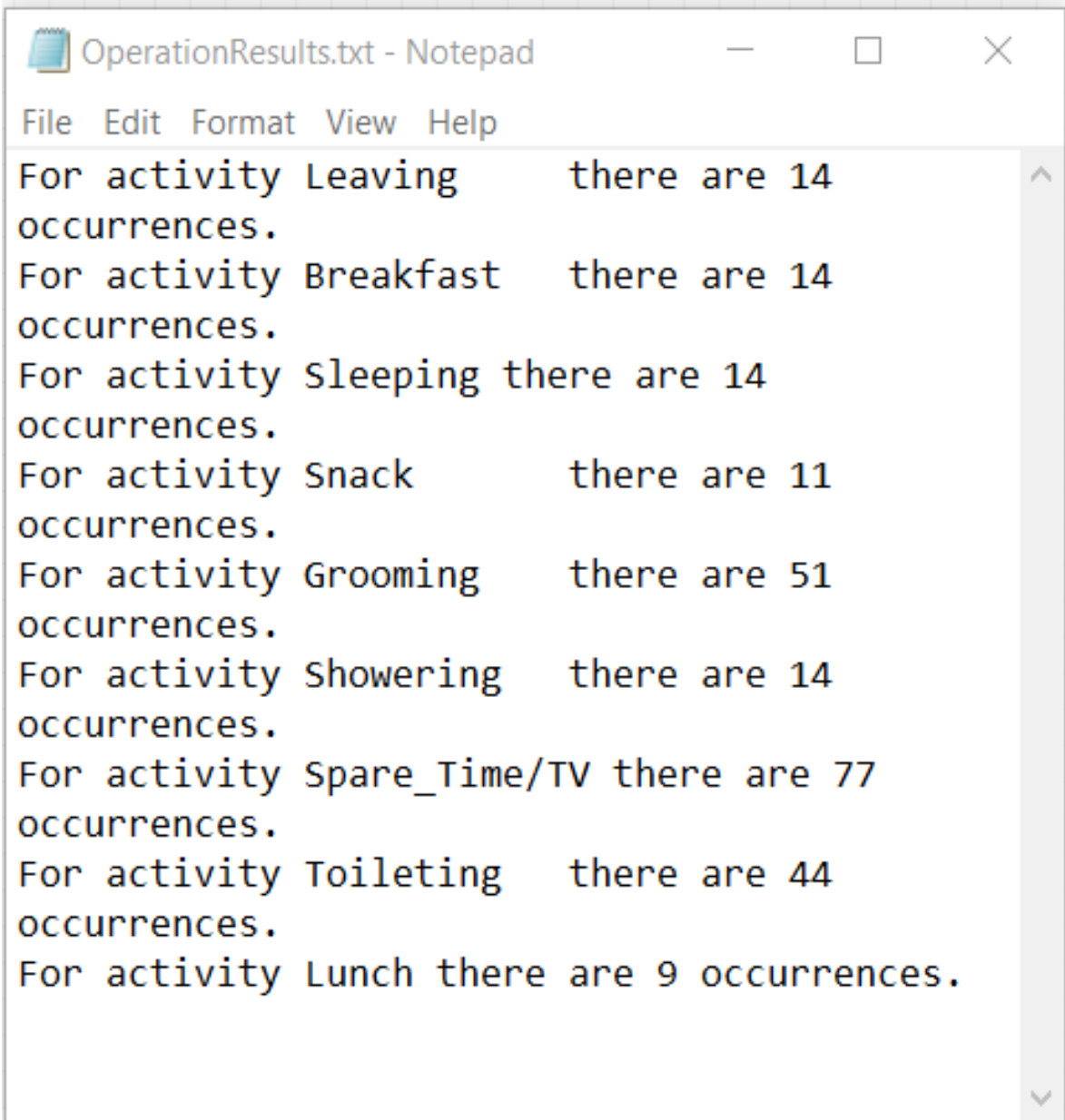
Urmatoarele rezultate vor fi returnate in urma apasarii succesive a butoanelor din interfata incepand cu TASK1 si culminand cu TASK5.



- numararea zilelor distincte in cadrul carora a fost urmarita activitatea



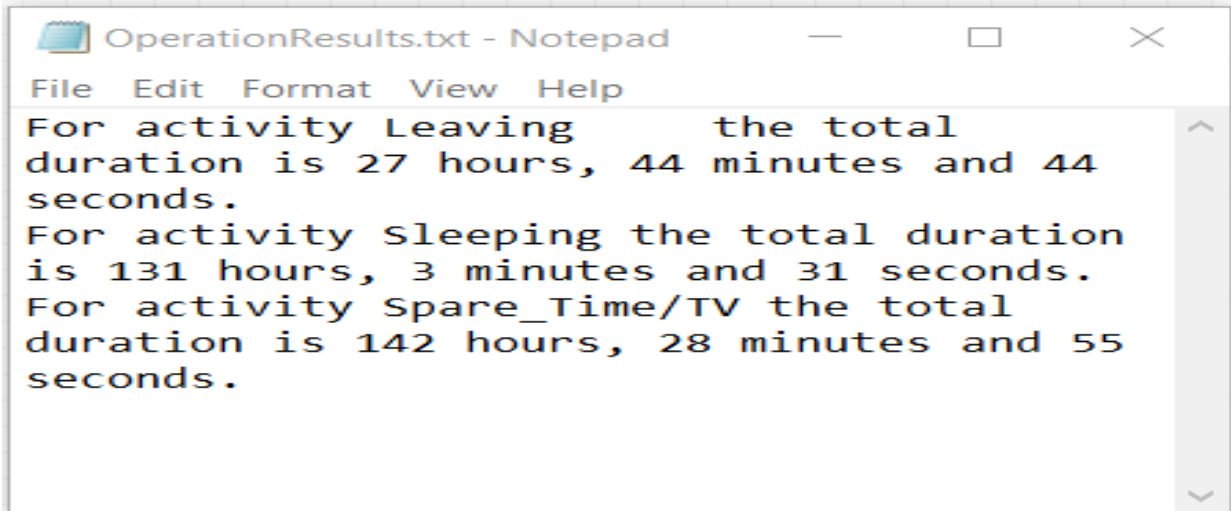
- numararea aparitiilor fiecărei activitati de-a lungul perioadei de tracking



The screenshot shows a Notepad window with the title bar 'OperationResults.txt - Notepad'. The menu bar includes 'File', 'Edit', 'Format', 'View', and 'Help'. The text content lists activities and their occurrence counts, with each entry on a new line. The activities and their counts are: Leaving (14), Breakfast (14), Sleeping (14), Snack (11), Grooming (51), Showering (14), Spare_Time/TV (77), Toileting (44), and Lunch (9).

```
For activity Leaving      there are 14
occurrences.
For activity Breakfast    there are 14
occurrences.
For activity Sleeping     there are 14
occurrences.
For activity Snack        there are 11
occurrences.
For activity Grooming     there are 51
occurrences.
For activity Showering    there are 14
occurrences.
For activity Spare_Time/TV there are 77
occurrences.
For activity Toileting    there are 44
occurrences.
For activity Lunch        there are 9 occurrences.
```

- cautarea activitatilor care au o durata totala mai mare de 10 ori



```
OperationResults.txt - Notepad
File Edit Format View Help
For activity Leaving the total
duration is 27 hours, 44 minutes and 44
seconds.
For activity Sleeping the total duration
is 131 hours, 3 minutes and 31 seconds.
For activity Spare_Time/TV the total
duration is 142 hours, 28 minutes and 55
seconds.
```

- numararea aparitiilor fiecărei activitati pentru fiecare zi in cadru careia s-a efectuat urmarirea activitatii

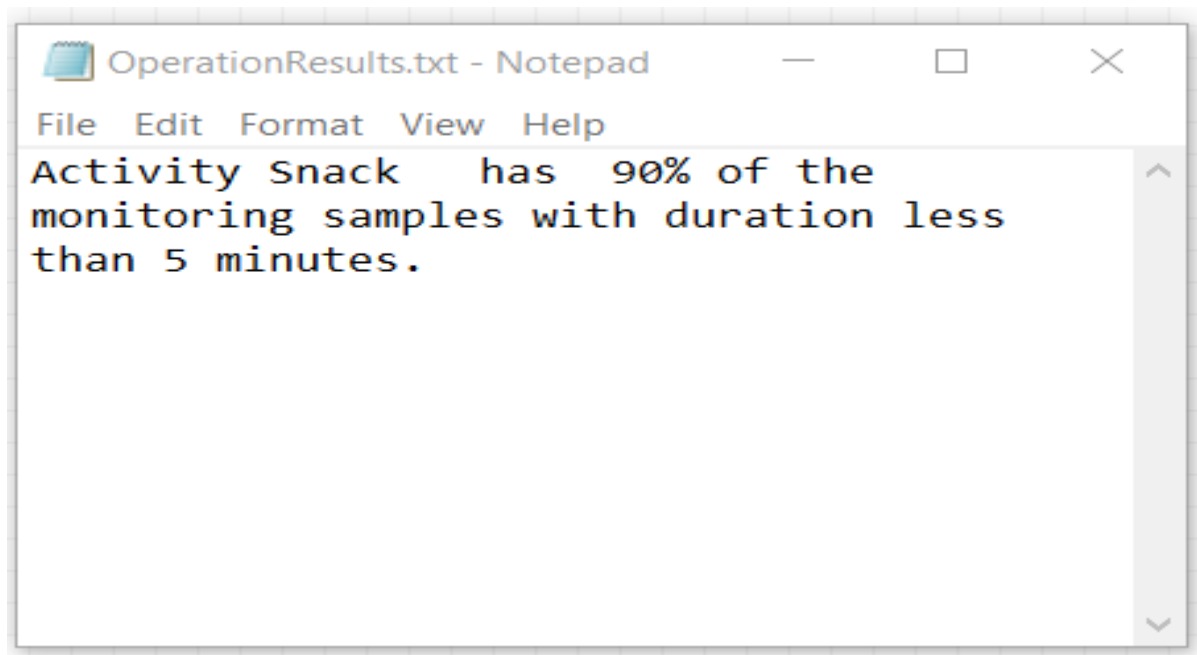
```
OperationResults.txt - Notepad
File Edit Format View Help
For day 332: activity Leaving    occurred 1
times
activity Breakfast             occurred 1 times
activity Sleeping occurred 1 times
activity Snack    occurred 1 times
activity Grooming             occurred 2 times
activity Showering            occurred 1 times
activity Spare_Time/TV occurred 4 times
activity Toileting            occurred 3 times
activity Lunch occurred 1 times

For day 333: activity Leaving    occurred 1
times
activity Breakfast             occurred 1 times
activity Sleeping occurred 1 times
activity Snack    occurred 1 times
activity Grooming             occurred 3 times
activity Showering            occurred 1 times
activity Spare_Time/TV occurred 6 times
activity Toileting            occurred 4 times
activity Lunch occurred 1 times

For day 334: activity Leaving    occurred 1
times
activity Breakfast             occurred 1 times
activity Sleeping occurred 1 times
activity Snack    occurred 2 times
activity Grooming             occurred 2 times
activity Showering            occurred 1 times
activity Spare_Time/TV occurred 8 times
activity Toileting            occurred 6 times
activity Lunch occurred 1 times

For day 335: activity Leaving    occurred 1
times
activity Breakfast             occurred 1 times
```

- filtrarea activitatilor care au 90% din durata de monitorizare mai mica de 5 minute



7. Concluzii si dezvoltari ulterioare

Sunt de parere ca aplicatia de față prezintă posibilitatea de a fi utilizată cu ușurință de absolut orice persoană interesată de trackingul propriilor activități pe parcursul unei perioade limitate de timp. Ca dezvoltări ulterioare aș sugera crearea unei interfețe utilizator mai complicate dar și a unei aplicații de tracking care să permită mai multe operații.