ACADEMIA DE STUDII ECONOMICE DIN BUCUREȘTI
FACULTATEA DE CIBERNETICĂ, STATISTICĂ ȘI INFORMATICĂ ECONOMICĂ
PROGRAMUL DE MASTER eBusiness

PROIECT LA DISCIPLINA

PROIECTARE ȘI PROGRAMARE ORIENTATE OBIECT

MoneyMaster – APLICAȚIE DE GESTIUNE A TRANZACȚIILOR ÎN SISTEMELE DE E-BANKING

Profesor coordonator: Conf. univ. dr. CIUREA Cristian-Eugen

Student: ARON Alin-Costin

Grupa: 1102

BUCUREȘTI, 2019

CUPRINS

INTRODUCERE

Trăim în secolul informației, perioadă marcată de o creștere colosală a proceselor automatizate, ce implică din ce în ce mai puțin aport și efort uman. Rolul personalului fizic a fost preluat, integral sau parțial, de computere inteligente, capabile să realizeze operații complexe, cu o marjă de eroare foarte mică.

Sectorul bancar se bucură în continuare de un avans tehnologic irefutabil, de necontestat. Prezența unui angajat fizic responsabil cu intermedierea tranzacțiilor a devenit deja facultativă în contextul foarte multor furnizori de servicii bancare.

Aplicația MoneyMaster are întocmai rolul de a facilita gestiunea tranzacțiilor în sistemele de e-banking, prin oferirea unui meniu extrem de simplist, dar complet și complex din punct de vedere funcțional.

Așadar, aplicația permite autentificarea atât ca administrator, cât și ca simplu client. Administratorul poate vizualiza întreaga listă de tranzacții, pentru toți utilizatorii sau particular, prin introducerea unui cod numeric personal unic. De asemenea, din contul administratorului pot fi generate rapoarte privind numărul total de conturi deschise, precum și valoarea totală și valoarea maximă a tranzacțiilor.

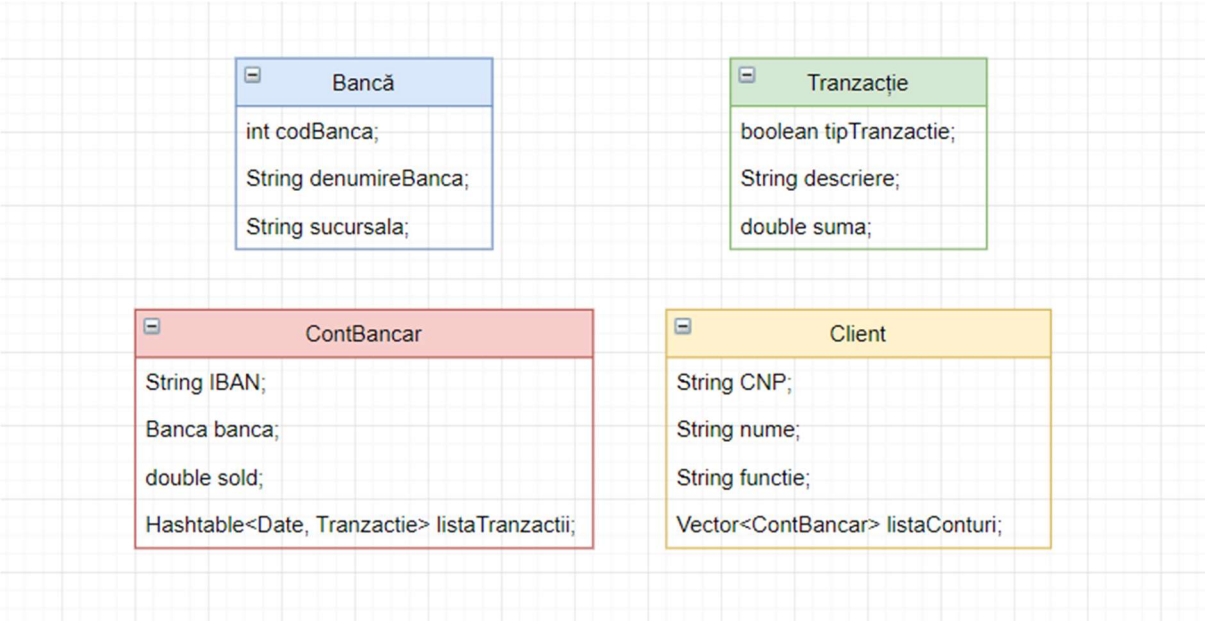
Prin introducerea CNP-ului, clientul are acces la lista conturilor bancare disponibile. Acesta poate opta pentru deschiderea unui nou cont la banca dorită. După selectarea contului de interes, utilizatorul are acces la o gamă foarte diversă de opțiuni financiar-bancare, de la interogarea soldului și până la depunerea sau retragerea de numerar.

După cum menționam și în cele de mai sus, MoneyMaster oferă o perspectivă de ansamblu asupra tranzacțiilor efectuate, pentru un management mai eficient al resurselor financiare, dar și de timp.

DESCRIEREA PROBLEMEI

Aplicația MoneyMaster cuprinde patru clase, și anume: Bancă, Tranzacție, ContBancar și Client. Clasa Bancă conține informațiile referitoare la băncile disponibile în aplicație (codBancă - de tip întreg, denumire și sucursală - ambele de tip String). Clasa Tranzacție conține atributele: tipTranzacție - de tip boolean, descriere - de tip String și sumă - de tip double. tipTranzactie ia valoarea *true* în situația în care operațiunea este pe debit, respectiv *false*, dacă tranzacția este pe credit. Clasa ContBancar stochează datele cu privire la conturile deschise. Un cont bancar este identificat prin câmpurile: IBAN - de tip String, bancă - de tip Bancă, sold - de tip double și listaTranzacții - de tip Hashtable <Date, Tranzactie>. Ultima clasă, Client, conține atributele: CNP, nume, funcție - toate de tip String și listaConturi, de tip Vector <ContBancar>.

Așadar, un client poate deține mai multe conturi, iar un cont poate fi utilizat pentru efectuarea mai multor tranzacții, unice la un moment dat (nu sunt aprobate tranzacțiile duplicate, efectuate la același moment de timp).



În implementarea aplicației, au fost utilizate mai multe masive și colecții de date. Printre masivele folosite, se pot număra: vectori de String - String[] credentiale și String [] banci, vectori de întregi - int [] numarConturi, sau vectori de double - double [] soldTotal.

Colecțiile utilizate sunt: List<Banca>, List<Client>, Hashtable<Date, Tranzactie>, Vector<ContBancar>, Set<Date> - pentru preluarea setului de chei din Hashtable și Collection<Tranzactie> - pentru obținerea valorilor din Hashtable.

IMPLEMENTAREA APLICAȚIEI

MoneyMaster este o aplicație de tip consolă, implementată în limbajul de programare Java. Obiectivul acesteia este facilitarea procesului de management al finanțelor, prin oferirea unui meniu de opțiuni variate.

În ordinea naturală, programul se deschide cu un meniu de autentificare, din care utilizatorul poate selecta logarea fie ca administrator, fie ca simplu client. Lista clienților este încărcată din fișierul *clienti.txt*. Secvența de cod responsabilă cu citirea fișierului și încărcarea listei este:

```
private static List<Client> readFile(String file) throws FileNotFoundException{
		List<Client> listaClienti = new ArrayList<Client>();
		Scanner scanner = new Scanner(new File(file));
		DateFormat formatter = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
		while(scanner.hasNext()) {
			String CNP = scanner.nextLine();
			String nume = scanner.nextLine();
			String functie = scanner.nextLine();
			int numarConturi = Integer.parseInt(scanner.nextLine());
			Vector<ContBancar> listaConturi = new Vector<ContBancar>();
			for(int i=0;i<numarConturi;i++) {
				Hashtable<Date, Tranzactie> listaTranzactii = new Hashtable<Date,
Tranzactie>();
				String IBAN = scanner.nextLine();
				String [] banci = (scanner.nextLine()).split(",");
				int cod = Integer.parseInt(banci[0]);
				String denumire = banci[1];
				String sucursala = banci[2];
				Banca banca = new Banca(cod,denumire,sucursala);
				double sold = Double.parseDouble(scanner.nextLine());
				int numarTranzactii = Integer.parseInt(scanner.nextLine());
				for(int j=0;j<numarTranzactii;j++) {
					Date dataTranzactie = null;
					try {
						dataTranzactie = (Date)formatter.parse(scanner.nextLine());
					} catch (ParseException e) {
						System.err.println("Eroare la conversia datei! Fisierul
contine date eronate...");
					}
					boolean tipTranzactie = Boolean.parseBoolean(scanner.nextLine());
					String descriereTranzactie = scanner.nextLine();
					double sumaTranzactie = Double.parseDouble(scanner.nextLine());
					try {
					if(sumaTranzactie < 0)
						throw new ExceptieSold();
					}
					catch (ExceptieSold e) {
						sumaTranzactie = 0;
					}
					Tranzactie tranzactie = new Tranzactie(tipTranzactie,
descriereTranzactie, sumaTranzactie);
					listaTranzactii.put(dataTranzactie, tranzactie);
				}
				ContBancar contBancar = new ContBancar(IBAN, banca, sold, listaTranzactii);
				listaConturi.add(contBancar);
			}
			Client client = new Client(CNP, nume, functie, listaConturi);
			listaClienti.add(client);
		}
		return listaClienti;
	}
```

În situația în care acest fișier nu este găsit, va fi declanșată o excepție, tratată prin inițializarea cu null a listei în speță, astfel:

```
List<Client> listaClienti;
try{
	listaClienti = readFile("clienti.txt");
}
catch (FileNotFoundException e) {
	System.err.println("Fisierul de clienti nu a fost incarcat...");
	listaClienti = new ArrayList<Client>();
}
```

```
Bun venit la MoneyMaster, managerul tau financiar virtual!
Selectati optiunea dorita:
1. Autentificare ca administrator
2. Autentificare client
3. Inchidere aplicatie
```

Pentru autentificarea cu succes, administratorul este nevoit să introducă un username și parola asociată, preluate din fișierul text *credentiale.txt*. După logare, administratorul poate opta pentru vizualizarea tranzacțiilor sau a rapoartelor aferente acestora, atât la nivel general, cât și particular, prin introducerea CNP-ului.

```
Username:
admin
Parola:
admin123
Selectati optiunea dorita:
1. Vizualizare tranzactii
2. Vizualizare rapoarte
3. Inchidere aplicatie
```

Afișarea tranzacțiilor se face de forma:

```
Clientul Anghel Marian identificat prin CNP 1890413454232 detine urmatoarele conturi:
IBAN: RO32434INGB la banca ING, sucursala Piata Spaniei (380)
Valoare sold: 14294.40 lei
Tranzactii:
 - 13/11/2019 23:01:09
Credit: -425.9 lei
Plata POS
 - 29/10/2019 18:43:26
Debit: 70.0 lei
Decont transport
```

```
Clientul Mares Cristina identificat prin CNP 2901214019492 detine urmatoarele conturi:
IBAN: RO2321UNI091 la banca Unicredit, sucursala Tineretului (880)
Valoare sold: 188.10 lei
Tranzactii:
 - 30/10/2019 19:57:39
Credit: -4000.0 lei
Retragere numerar

IBAN: ROBT90321LEI la banca Banca Transilvania, sucursala Ferentari (1020)
Valoare sold: 1249.20 lei
Tranzactii:
 - 12/11/2019 15:12:25
Debit: 391.4 lei
Retur
 - 10/11/2019 12:00:31
Debit: 1900.0 lei
Incasare salariu
```

Afișarea rapoartelor, pe de altă parte, se face astfel:

```
Soldul total al clientilor este: 25189.90 lei
Numarul total de conturi din aplicatie este: 8
Cea mai mare tranzactie inregistrata in aplicatie este: 4000.00 lei
```

Respectiv:

```
Introduceti CNP-ul:
1980204245211
Soldul total al clientului Popa Ion este: 5517.40 lei
Numarul total de conturi al clientului Popa Ion este: 3
Cea mai mare tranzactie inregistrata de clientul Popa Ion este: 3500.00 lei
```

Revenind la meniul principal, utilizatorul are posibilitatea de a intra în contul de client, ceea ce presupune introducerea codului numeric personal. În situația în care utilizatorul nu apare în fișierul de intrare, *clienti.txt*, acesta va primi un mesaj sugestiv. În caz contrar, îi vor fi listate pe ecran conturile deținute, având totodată posibilitatea de a deschide un nou cont.

```
Introduceti CNP-ul:
1980205142342
Nu exista niciun client cu acest CNP!
```

```
Introduceti CNP-ul:
1980204245211
Selectati contul dorit:
1. RO992414BRD - BRD
2. RO424253ING - ING
3. RO999900CEC - CEC
4. Deschidere cont nou
```

Selecția unui cont este succedată de afișarea unui meniu de tranzacții, după cum urmează:

```
Selectati optiunea dorita:
1. Vizualizare extras de cont
2. Retragere numerar
3. Solicita imprumut
4. Plata catre comercianti
5. Depunere numerar
6. Interogare sold
7. Inchidere aplicatie
```

Opțiunea "Vizualizare extras de cont" afișează integral informațiile contului bancar, cu detalierea tranzacțiilor, astfel:

```
Vizualizare extras de cont
IBAN: RO424253ING la banca ING, sucursala Piata Spaniei (380)
Valoare sold: 4973.30 lei
Tranzactii:
 - 10/11/2019 20:29:13
Credit: -89.1 lei
Plata catre comercianti (Apanova)
 - 10/11/2019 20:07:28
Credit: -250.0 lei
Retragere numerar
 - 10/11/2019 12:19:52
Debit: 3500.0 lei
Incasare salariu
```

La selectarea unei opțiuni din listă, clientului îi este solicitată suma tranzacționată. În situația în care această sumă este negativă, va fi declanșată o excepție definită de utilizator, *ExceptieSold*, ce va fi tratată prin anularea operațiunii.

```
Introduceti suma dorita:
-250.9
Suma este eronata! Tranzactia a fost anulata...
```

În situația unei tranzacții de credit (retragere numerar sau plată către comercianți), dacă suma de plată depășește soldul disponibil, se va afișa un mesaj sugestiv, iar operațiunea nu va fi acceptată:

```
Introduceti suma dorita:
100000
Fonuri insuficiente
```

În caz contrar, tranzacția va fi aprobată, iar soldul va fi actualizat:

```
Sold disponibil: 4973.30 lei

  2
  Introduceti suma dorita:
  300

  Tranzactie acceptata!

Sold disponibil: 4673.30 lei
```

La închiderea aplicației, informațiile for fi actualizate în fișierul *clienti.txt*.

```
  Aplicatia a fost inchisa!
```

CONCLUZII

Implementarea unei aplicații de management al tranzacțiilor în sistemele de e-banking reprezintă un real progres tehnologic, dar și o chestiune de incontestabilă utilitate. Activitățile anterior specific umane au fost preluate, în diverse proporții, de roboți și dispozitive inteligente.

În implementarea aplicației informatice, am utilizat diverse metode, tipuri de date, colecții și clase de obiecte.

Una din constatările personale este posibilitatea folosirii unei colecții de tip ArrayList<ContBancar>, în detrimentul colecției Vector<ContBancar>, în clasa Client. Principala diferență între cele două este faptul că metodele din Vector sunt sincronizate, în timp ce metodele din ArrayList nu sunt sincronizate. Executarea de metode sincronizate poate fi foarte costisitoare, din punct de vedere al resurselor și al timpului, motiv pentru care utilizarea colecției de tip Vector poate conduce la scăderea performanței, în raport cu ArrayList. Fiind o colecție de dimensiuni reduse însă, nu există o descreștere notabilă a eficienței. În plus, pentru varietatea exemplului, în sens demonstrativ, am decis utilizarea unei astfel de colecții.

BIBLIOGRAFIE

1. Core Java Volume I – Fundamentals (11th Edition)**,** Cay S. Horstmann

2. Resurse Internet:

- www.stackoverflow.com (Stack Overflow)
- www.stackabuse.com (Stack Abuse)
- www.callicoder.com (CALLICODER)
- www.javatpoint.com (Java T Point)
- www.tutorialspoint.com (Tutorials Point)
- www.geeksforgeeks.org (GeeksForGeeks)