Documentatie Tema 4

Bank Management

Negrea Malina-Adela

Grupa 30224

Profesor laborator: Pop Claudia

Cuprins

1. Obiectivul temei
2. Analiza problemei
3. Proiectare
4. Implementare
5. Testare
6. Concluzii
7. Bibliografie
8. Obiectivul temei

**Obiectiv principal**

Se considera o apliatie BankManagement pentru procesarea conturilor si clientilor bancii , precum si a operatiunilor bancare se retragere si adaugare de numerar.

Datele bancii sunt stocate cu ajutorul unui fisier serializabil.

**Obiective secundare**

* Cazuri de utilizare si scenarii de utilizare . . . . . . . . .capitolul 2
* Alegerea structurilor de date . . . . . . . . . . . . . . . . . . capitolul 3
* Impartirea pe clase . . . . . . . . . . . . . . . . . . . . . . . . . . capitolul 3
* Dezvoltarea algoritmilor . . . . . . . . . . . . . . .. . . .. . . . capitolul 4
* Implementarea solutiei . . . . . . . . . . . . . . . . . . . . . . capitolul 4
* Testare . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . . . . capitolul 5

1. Analiza problemei

Cazuri de utilizare:

* Inregistrarea de noi client
* Stergerea unui client
* Actualizarea datelor unui client
* Adaugarea de noi conturi
* Actualizarea informatiilor din cont
* Stergerea de conturi
* Retragere de numerar pentru un anumit cont
* Depunere de numerar pentru un anumit cont

Scenarii de utilizare:

1. Modificarea listei de clienti:

Pentru a putea accesa informatii despre clienti se apasa butonul Clients din fereasta principal .

Pentru adaugarea unui noi client se completeaza campurile Name , Address si CNP si se apasa butonul Add. Dupa apasarea butonului noul client va fi vizibil in tabela din partea stanga a ferestrei.

Pentru stergerea unui client se selecteaza din tablela clientilor clientul dorit si se apasa butonul Delete.

Pentru actualizarea datelor despre un client se selecteaza din tabela clientul dorit si se modifica campurile din partea dreapta ce trebuie ce trebuie actualizate, dupa aceea se apsa butonul Update.

Pentru a reveni la fereasta de pornire de apasa butonul Close din coltul dreapta jos.

1. Modificarea listei de conturi:

Pentru a putea accesa informatii despre produse se apasa butonul Account din fereasta principal .

Pentru adaugarea unui nou cont se completeaza campurile ID si Amount, se allege din lista de client din partea dreapta proprietarul contului si se apasa butonul Add. Dupa apasarea butonului noul cont va fi vizibil in tabela din partea stanga a ferestrei.

Pentru stergerea unui cont se selecteaza din tablela contul dorit si se apasa butonul Delete.

Pentru actualizarea datelor despre un cont se selecteaza din tabela contul dorit si se modifica campurile din partea dreapta ce trebuie actualizate, dupa aceea se apsa butonul Update.

Pentru a reveni la fereasta de pornire de apasa butonul Close din coltul dreapta jos.

1. Efectuarea de operatiuni bancare:

Pentru a putea accesa informatii despre produse se apasa butonul Operations din fereasta principal .

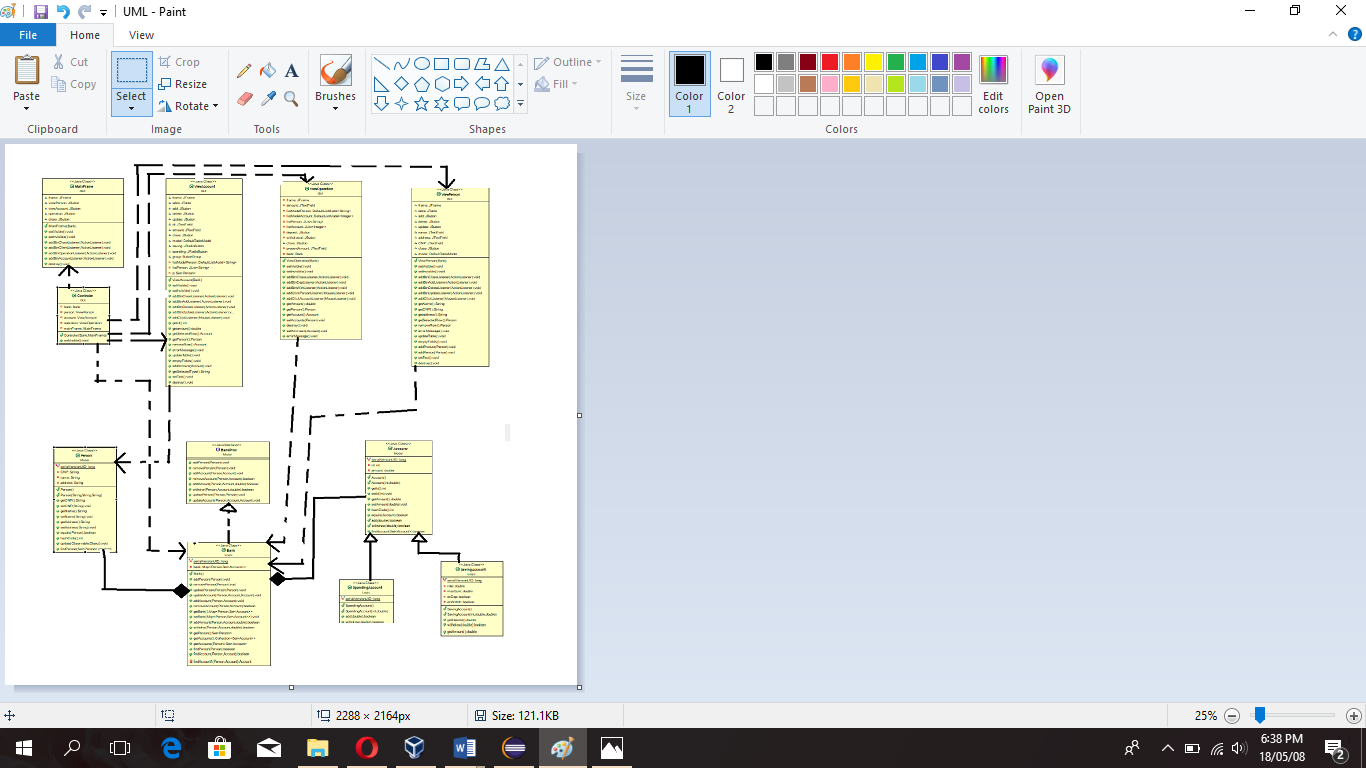
Se allege din lista clientilor proprietarul contului, iar apoi contul dorit si se completaza cu suma dorita campul Amount. In campul available amount se poate vedea suma disponibila din cont. Pentru operatia de adaugare se apasa butonul Deposit , iar pentru operatiune de retragere se apsa butonul Withdraw. In cazul in care operatiunea se poate realiza suma actualizata va fi vizibila in campul available amount, in caz contrat va aparea un mesaj de eroare.

Pentru functionarea corecta a aplicatiei facem urmatoarele presupuneri legate de datele de intarare:

* User-ul introduce informatii corecte in campurile de date pentru client
* User-ul introduce informatii corecte in campurile de date pentru cont si selecteaza probritarul contului din lista
* User-ul nu apasa butoanele delete si update fara a selecta o linie din tabele
* User-ul nu apsa butoanele deposit si withdraw fara a selecta clientul si contul si a introduce o suma valida in compul amount ( o valoare intraga sau zecimala pozitiva, mai mica decat suma disponibila in cont pentru operatiunea de retragere de numerar)

1. Proiectare

Diagrama UML



Aplicatia este impartita in 3 pachete:

Pachetul Model ce contine clasele Account (abstracta) si Person si interfata BankProc.

Pachetul Logic contine clasele SavingAccount si SpandingAccount ce mostenesc clasa Account, si clasa Bank. In plus contine si clasa App ce contine metoda main.

Pachetul GUI contine clasele Controler, MainFrame, ViewAccount , ViewPerson si ViewOperation.

Pentru constructia aplicatiei au fost folosite pe langa structurile simple de date si un map de tip HashMap pentru retinerea informatiilor despre client. Acesta are ca si chei persoanale din banca si ca si valori un Set de tip HashSet pentru retinerea conturilor affiliate persoanei respective.

Interfata grafica are un aspect simplu , usor de folosit. Acesta este formata din patru ferestre : prima fereasta este cea principal . Acestra contine patru butonane , si anume Clients, Accounts, Operations si Close. Prin apasarea primelor trei butoane se vor deschide noi ferestre , iar cel de-al treilea are rolul de a inchide aplicatia . Cea de-a doua ferestara este fereastra Clients ce se deschide prin apasarea butonului Clients. Acesta este formata dintr-un table ce infatiseaza client din banca aflat in partea dreapta, trei campuri editabile ce au rolul de a complete informatii despre client , trei butoane pentru a manipula tabela si un buton de inchidere. La apasarea celui din urma se deschide din nou feresatra principal. Cea de-a treia feresta este este fereastra Accounts ce se deschide prin apasarea butonului Account. Acesta este formata dintr-un table ce infatiseaza produsele din baza de date aflat in partea dreapta, doua campuri editabile ce au rolul de a complete informatii despre produs , o lista cu clientii bancii, trei butoane pentru a manipula tabela si un buton de inchidere. La apasarea celui din urma se deschide din nou feresatra principal. Ultima feresta este ferestra Operations. Acesta este formata din doua liste, una ce infatiseaza client si una ce infatiseaza conturile clientului selectat, acesta din urma devenind vizibila doar in momentul selectarii unui client, un camp Amount pentru introducerea suma dorita , un camp Available Amount ce afiseaza suma din cont, acesta aparand doar in momentul alegerii unui cont, butoanele Deposit si Withdraw pentru operatiiloe bancare associate si butonul Close pentru a reveni in fereasta principla.

Pe langa aceste fram-uri principale , aplicatia mai contine si frame-uri pentru afisarea uni mesaj de eroare in cazul introduceri de date sunt gresite.

Nu au fost folositi algoritmi ce trebuie documentati .

1. Implementare

Aplicatia contine 11 clasa si o interfata

1. Clasa ViewPerson – deseneaza interfata de manipularea clientilor
2. Clasa ViewAccount - deseneaza interfata de manipularea conturilor
3. Clasa ViewOperation - deseneaza interfata de manipulare a produselor
4. Clasa Controler – controleza interactiunea dintre parte de logica aplicatiei si intefata grafica
5. Clasa Person – mapeaza clientul bancii
6. Clasa Account – clasa abstracta ce mapeaza conturile bancii
7. Clasa SavingAccount – extinde clasa abstata Account si mapeaza contul de tip economii
8. Clasa SpandingAccount – extinde clasa abstata Account si mapeaza contul de tip cont de chieltuieli
9. Clasa App – contine metoada main
10. Interfata BankProc – contine toate metodele necesare operatiilor bancare
11. Clasa Bank – implementeza interfata BankProc, reprezentand entitatea principala a aplicatiei
12. Clasa MainFrame – deseneaza frame-ul principal
    1. Clasa ViewPerson

Acesta deseneaza interfata grafica pentru interactiunea cu clientii.

Ea contine un JTabel , patru JButton , trei JLable-uri si trei JTextField-uri.

Acesta nu gestioneaza evenimentele.

* 1. Clasa ViewAccount

Acesta deseneaza interfata grafica pentru interactiunea cu conturile.

Ea contine un JTabel, patru JButton , trei JLable-uri, un JList si trei JTextField-uri.

Acesta nu gestioneaza evenimentele.

* 1. Clasa ViewOperation

Acesta deseneaza interfata de operatiuni bancare.

Ea contine doua JList-uri, doua Jlabel-uri si doua JTextField-uri si trei butoane.

De asemena ea contine si doua messageDialog-uri.

Ea nu gestioneaza evenimentele.

* 1. Clasa MainFrame

Deseneaza feresta principal cu patru butoane , unul pentru inchidere sit trei pentru accesarea celorlalte ferestre .

Ea nu gestioneaza evenimentele.

* 1. Clasa Controler

Acesta interactioneza atat cu interfata grafica cat si cu zona de logica.

Acesta gestioneaza evenimentele din toate cele patru ferestre prin clasele interne ce implementeaza interfata ActionListener , precum si cele 4 clase ce implementeza MouseListener.

* 1. Clasa Person

Acesta mapeaza un client al bancii.

Peroana are ca si attribute CNP , Name si Addres. Clasa contine gettere si settere pentru acestea. In plus ea suprascrie (@override) metodele equals si hashCode. Pentru calcularea hashCondului sunt folosite hashCodeurile celor trei attribute.

Clasa implementeaza interfetele java.io.Serializable, Observer, atat pentru a putea fii serializat cat si pentru a fi notificat in urma oricarei modificari aparute asupra oricarui dintre conturile sale.

* 1. Clasa Account

Acesta este o clasa abstract ace mapeaza contul unui client.

Acesta are ca si attribute id-ul si amount. Clasa contine gettere si settere pentru acestea. In plus , acesta suprascrie metodele equals si hashCode. De asemenea, clasa defineste metodele withdraw si add ce vor fi implementate de clasele SavingAccount si SpandingAccount in mod diferit.

Clasa Account extinde de asemenea clasa Observable si implementeaza interfata java.io.Serializable. In acest fel modificarile aspura contului pot fi sesizate catre client.

* 1. Clasa SavingAccount

Acesta mosteneste clasa Account.

Acesta contine pe langa atributele clasei Account, campurile rate , maxSum si flagurile enDep si en Withdr. Primele doua sunt folosite pentru a calcula dobanda ce se adauga sa suma din cont , iar cele din urma sunt folosite pentru a valida operatiunile.

Atat operatiunea de retragere cat si cea de depunere se realizeaza atata timp suma ce trebuie procesata nu este mai mare decat maxSum si suma este pozitiva , in caz contrar operatiunea se realizeaza doar in cazul in care o alta operatiune cu sume mari a fost realizata . Controlul se realizeaza cu flagurile enDep si enWithdr ce sunt setate dupa o operatiune cu sume mari. In plus, la operatiune de retragere se verifica si daca sunt fonduri necesare.

* 1. Clasa SpandingAccount

Acesta este o clasa concreta ce extinde clasa Account.

Ea nu are attribute in plus fata de clasa Account .

Operatiile de retragere si depunere nu sunt limitate decat de suma din cont.

* 1. Clasa App

Contine metoda main.

* 1. Inerfata BankProc

Acesta defineste toate operatiile ce pot fi effectuate de banca : adaugare de client , stergere de client , adaugare de cont , stergere de cont , actualizarea datrelor unui client , actualizarea datelor unui cont , operatiunile de adaugare si retragere de numerar.

Interfata contine de asemenea preconditii si post conditii pentru fiecare metoda (desing by contract).

* 1. Clasa Bank

Acesta reprezinta entitatea principala a aplicatiei.

Ea are ca atribut un HashMap ce are ca si chei persoanele si ca si valori seturi de conturi.

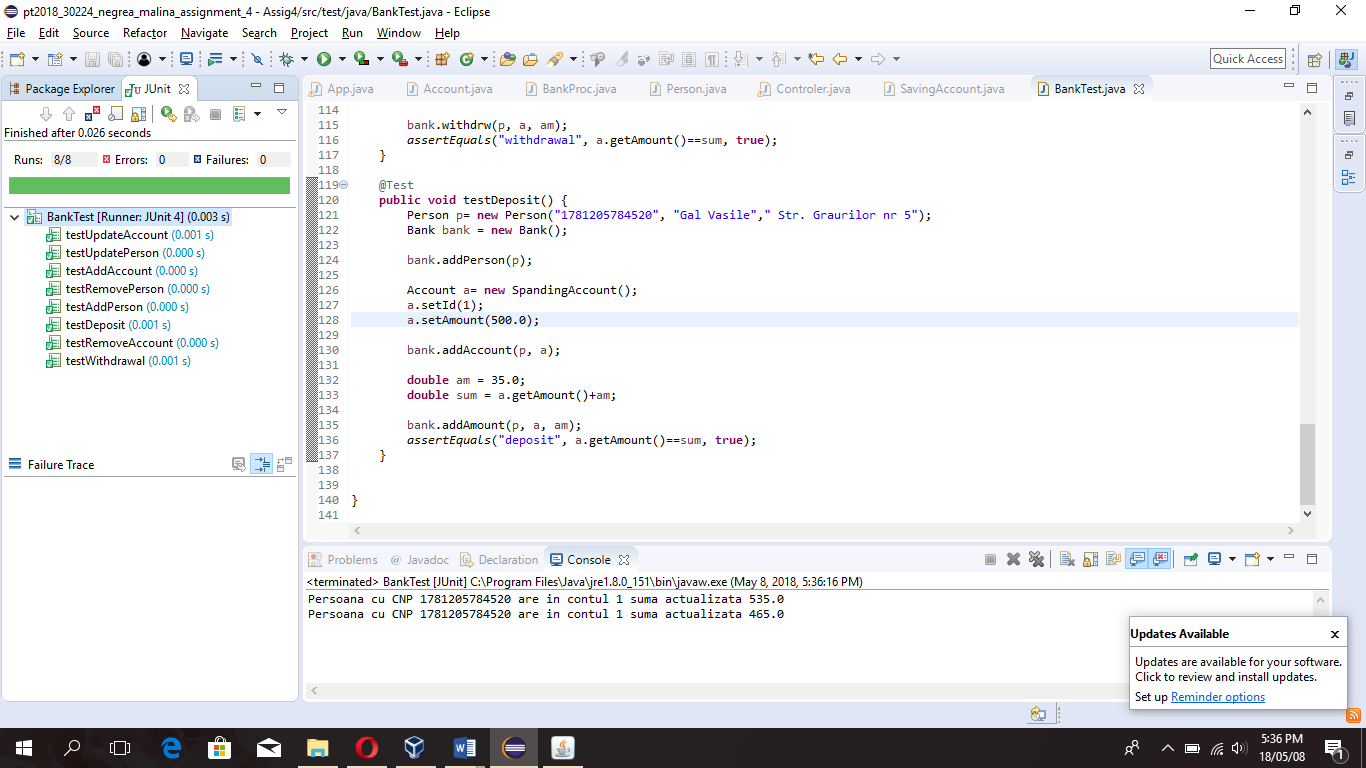
Ea implementeaza metodele din inferfata BankProc: adaugare de client , stergere de client , adaugare de cont , stergere de cont , actualizarea datrelor unui client , actualizarea datelor unui cont , operatiunile de adaugare si retragere de numerar.

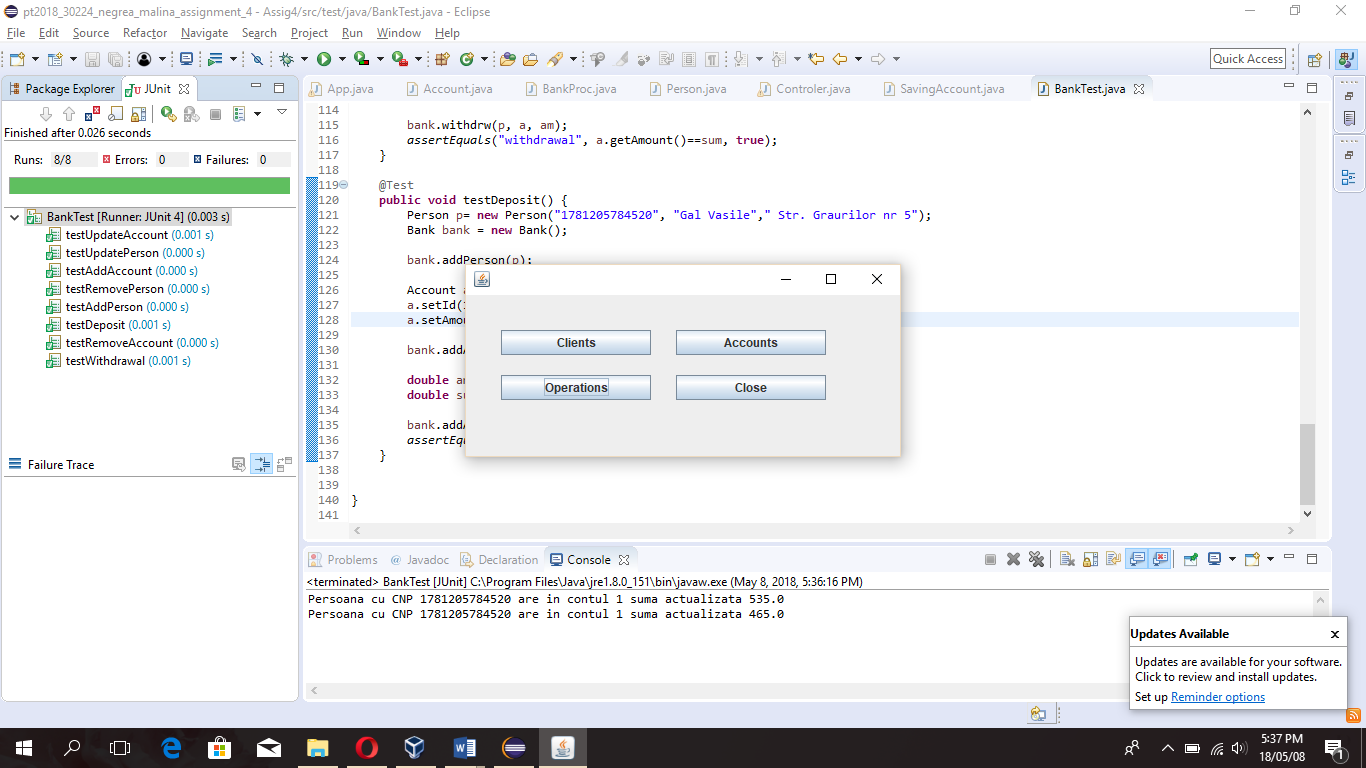
De asemenea, ea implemementeaza si preconditiile si post conditiile din interfata cu ajutorul asertiunilor.

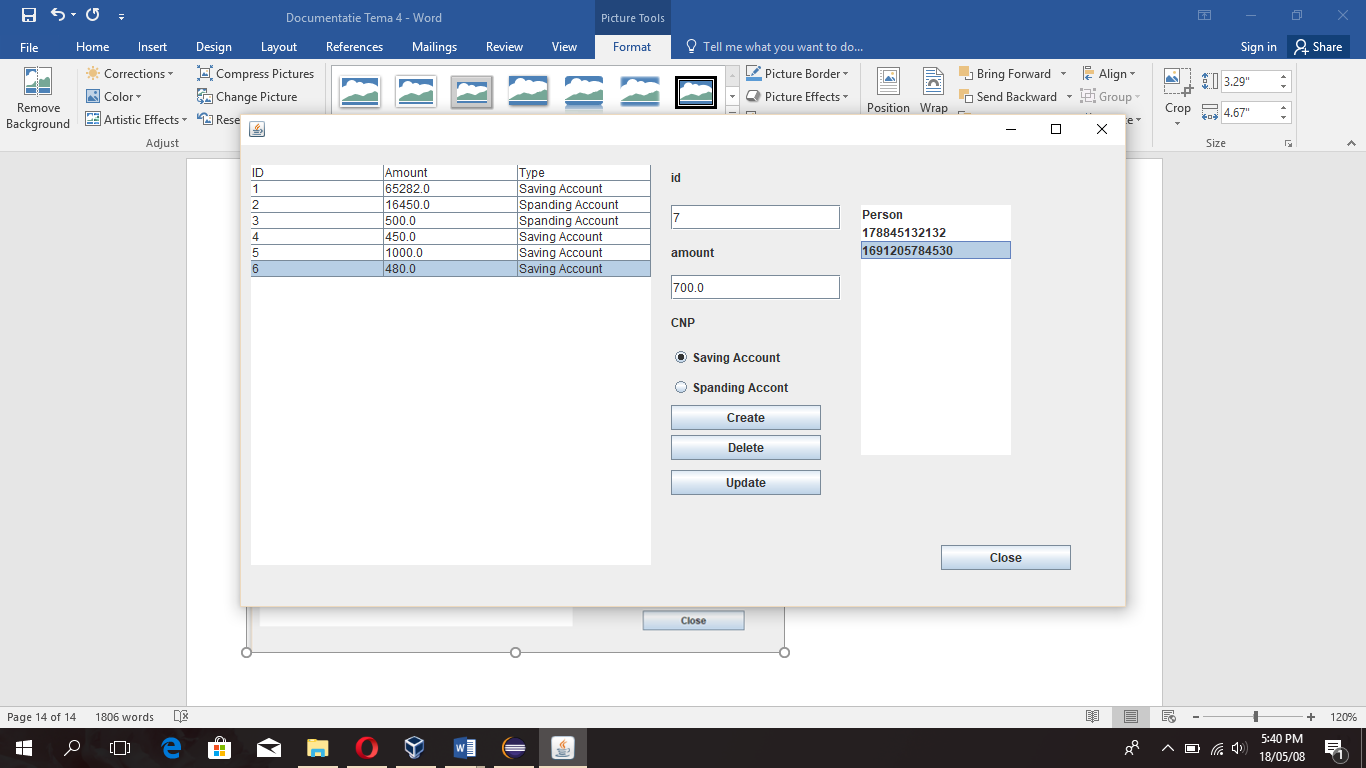
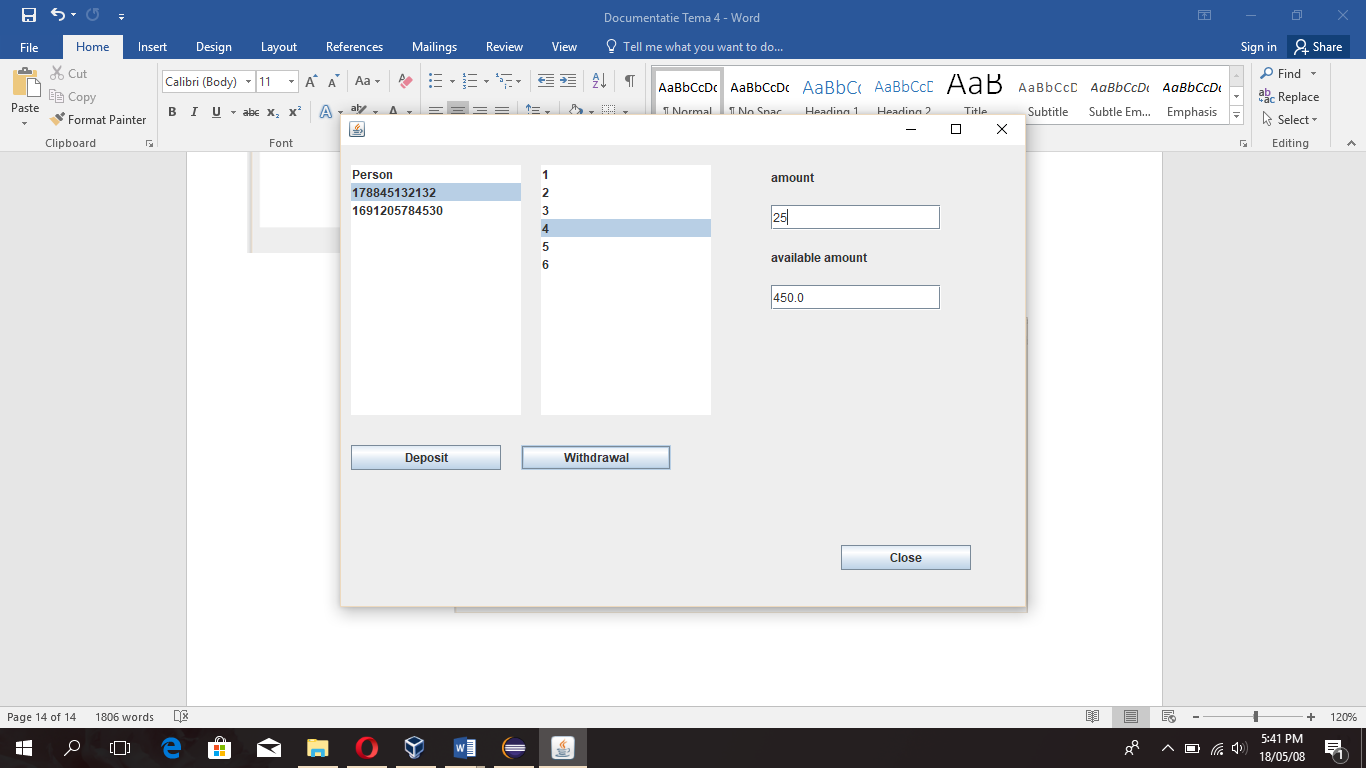
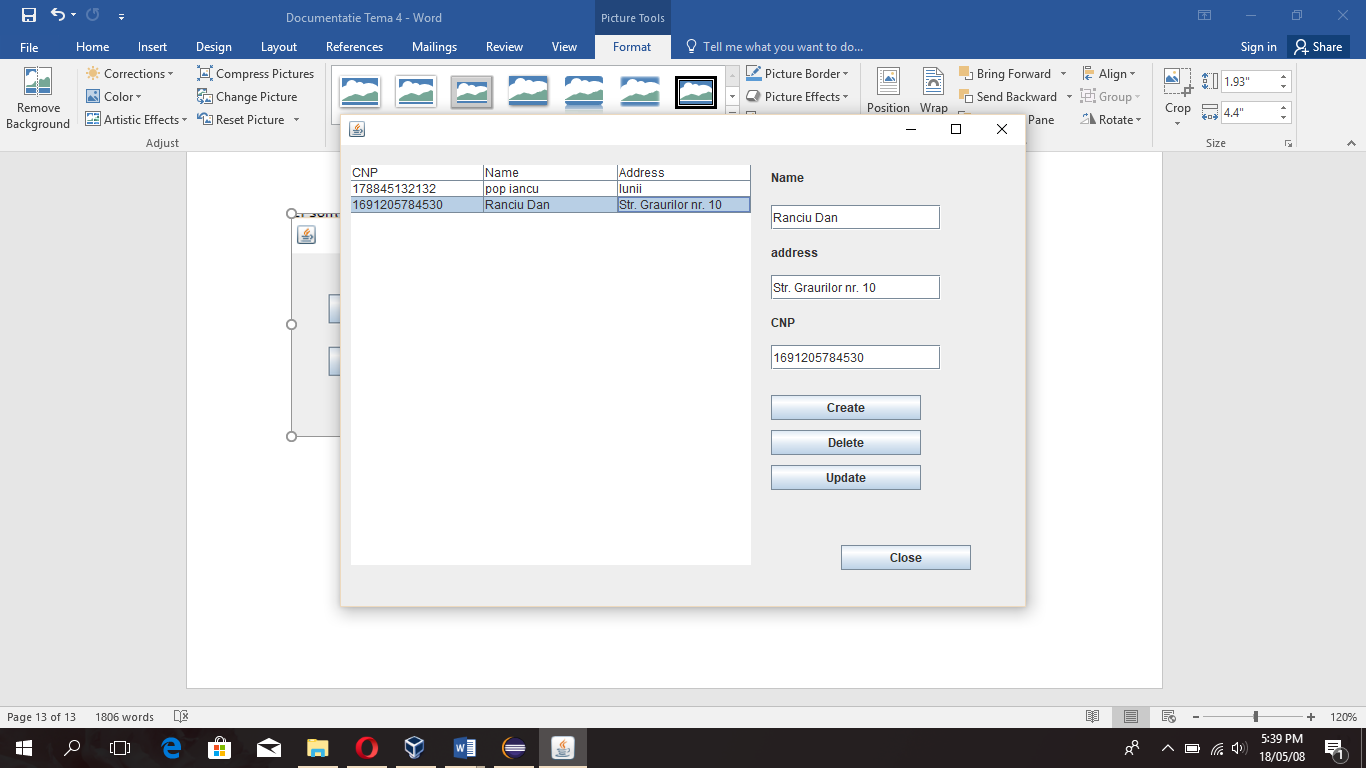
Obiectul de tip bank este serializat ca si un intreg.

Invarinatul clasei este proprietatea atrbutului bank de a ramane o instant a lui Map.

1. Testare





1. Concluzii

Realizarea proiectului a insemnat familiarizarea cu conceptul de design by contract, preconditii , postdonditii si asetiuni, precum si cea de Map si Set, mai exact cu cea de HashMap si HashSet , idea de suprascriere a functiilor de hashCode astfel in cat aplicatia sa actionize optimal. De asemenea , a insemanat o reamintire a conceptului de action listener si in special de mouselistener (click listener) .

Pentru o extindere ulterioara a proiectului se pot lua in considerare adugare de noi campuri in clasa Person pentru o mai buna mapare, precum si verificatori pentru ca datele introduce de utilizator sa fie corecte sis a nu fie nevoie de o eventuala corectie a acestora. De asemenea, clasa Saving Account ar putea fi extinsa de mai multe clase astfel incat fiecare clasa sa fie dedicate unei perioada specifice a depositului. Mai mult , se poate notifica clientul in momentul in care depositul expira.