

AccountManager ügyfélkezelő desktop applikáció

Programozási technológia 2.

Gyakorlatvezető: dr. Nagy Sára

Készítette: Szováty István Zsolt (PZ7KFQ)

Követelmény specifikáció

Bevezetés

Az AccountManager ügyfélszámla kezelő projekt, egy adott banki rendszer ügyfeleinek és azok számláinak kezelését és egyszerűen menedzselhetőségét teszi lehetővé.

A bank alkalmazottjai, ezen a rendszeren bejelentkezve tranzakciókat hajthatnak végre / hagyhatnak jóvá, illetve a program segítségével megjeleníthetik az ügyfelek, valamint azok számláinak adatait, tranzakcióit. A projekt célközönsége, olyan pénzügyi rendszerek, ahol magánszemélyek, illetve kis és nagy vállalatok számlakezelése folyik, valamint pénzügyi tranzakciókat vihetnek végbe.

Az AccountManager ügyfélkezelő rendszer segít ezen tranzakciók biztonságos, gyors, valamint egyszerű elektronikus úton történő megvalósítását.

Feladat leírása

Készítsünk programot, amellyel egy banki ügyfélszámla kezelést valósíthatunk meg az alábbi funkciókkal:

- A programba a munkatártnak előbb be kell jelentkeznie a felhasználónév és jelszó megadásával, csak úgy végezhet bármilyen műveletet, és láthat bármilyen adatot. 10 perc inaktivitás után a jelszót újra meg kell adni.
- A programban megtekinthetők az ügyfelek adatai (név, cím, telefon).
- Ügyfélt kiválasztva láthatók a bankszámlái (számlaszám, létrehozás dátuma, egyenleg). A bankszámla egyenlegét a tranzakciók alapján számoljuk ki. A bankszámla zárolható, ez azt jelenti, hogy nem lehet átutalás forrása, vagy célja, amíg a zárolást fel nem oldják.
- A bankszámlát kiválasztva látható a tranzakciók listája (dátum, forrás és cél számlaszám, összeg).
- Lehetőségünk van új tranzakció hozzáadására az összeg, illetve a forrás és cél számlaszámok megadásával (kiválasztásával). A dátum automatikusan íródjon bele. A program kérjen megerősítést a tranzakció végrehajtására, majd ezt követően rögzítse a program a tranzakciót (és módosítsa az egyenlegeket). A tranzakció szolgáltatói díja a forrás számlából levonásra kerül, amely az utalt összeg 0.05%-a.
- Ha egy ügyfél több számlával is rendelkezik, a tranzakciónál több forrás számla és számlánként az utalandó összeg is megadható. Ha egy ilyen több forrás számlát tartalmazó tranzakcióban hiba lép fel (bármely számla zárolt, vagy nem áll rendelkezésre elegendő egyenleg), az összes átutalás lépést vissza kell vonni.
- A számla egyenlege nem csökkenhet 0 alá, ha ez bekövetkezne a tranzakciót vissza kell utasítani.
- Tranzakciót kiválasztva lehessen azt sztornózni. Ekkor történik egy ellentétes irányú tranzakció ugyanazzal az összeggel. Tranzakciót sztornózni csak a létrehozást követő 12 órában lehet.

Az adatbázis az alábbi adatokat tárolja (ezek még nem feltétlenül a fizikai adattáblák):

- ügyfelek (név, cím, telefon);
- számlák (számlaszám, ügyfél, létrehozás dátuma, zárolt-e);
- tranzakciók (dátum, összeg, forrás, cél);
- munkatársak (név, azonosító, jelszó).

Követelmények

Funkcionális követelmények

Használati eset diagram (Use Case)

Alkalmazás funkciói:

1. Use Case 1: Ügyfélkezelő beléptetése

Az ügyfélkezelő (nincs bejelentkezett állapotban) accountant-ként bejelentkezhet az applikációba helyes felhasználónév / jelszó páros megadásával.

2. Use Case 2: Ügyfelek listázás / Ügyfelek számláinak kilistázása / Tranzakciók megjelenítése

Az ügyfélkezelő bejelentkezett állapotban megjelenítheti az rendszer ügyfeleit, valamint az ügyfelek számla adatait, és az azokhoz tartozó tranzakciókat.

3. Use Case 4: Tranzakció hozzáadása

A bejelentkezett ügyfélkezelő bejegyezhet nyithat / elfogadhat tranzakciót.

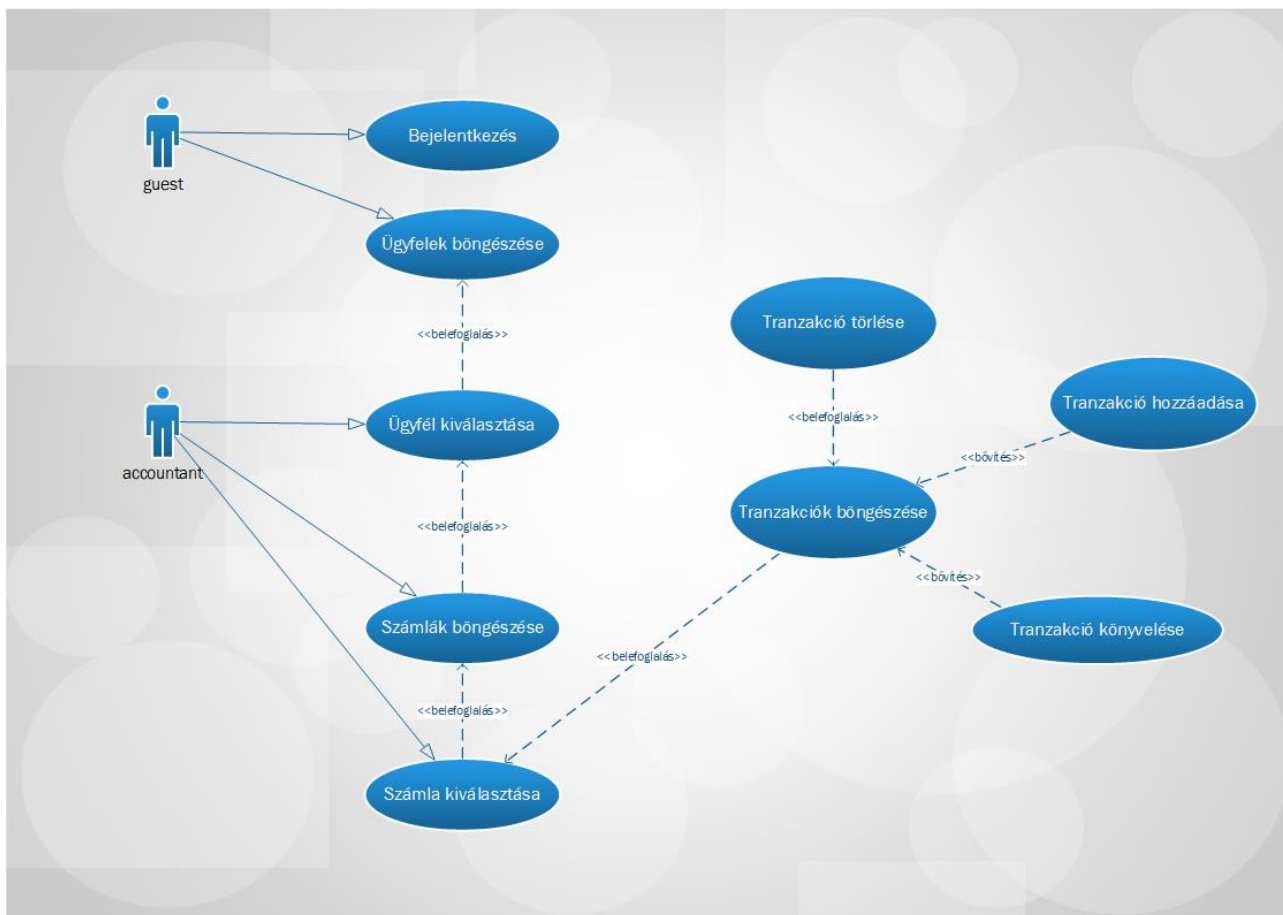
4. Use Case 3: Tranzakció törlése

A bejelentkezett ügyfélkezelő törölhet egy kiválasztott tranzakciót.

5. Use Case 5: Tranzakció könyvelése

A bejelentkezett ügyfélkezelő könyvelhet (jóváagyhat) egy kiválasztott tranzakciót

Használati esetek és interakciók illusztrációja használati eset diagrammon:



Felhasználói történetek (User story)

Felhasználó bejelentkezése / Felület megjelenítése (Use-Case 1-2):

Elsődleges Actor	Ügyfélkezelő / Accountant
Előfeltétel	A felhasználó nincs bejelentkezve.
Sikertelen eredmény	Hiba jelzése hiba üzenet küldéssel.
Sikeres eredmény	Az applikáció belépteti a felhasználót, megjelenítve egyben a felhasználói felületet.
Események	<ol style="list-style-type: none"> 1. Űrlap ellenőrzése, hogy kitöltöttük-e a felhasználónév illetve jelszó mezőt. 2. Beléptetés a rendszerbe, a login gombra való kattintással. 3. A rendszer ellenőrzi, a felhasználó helyességét.

	<p>4. A rendszer ellenőrzi a jelszó helyességét.</p> <p>5. A rendszer rögzíti a belépés időpontját.</p> <p>6. A rendszer belépteti a felhasználót, és megjeleníti a felhasználói felületet, tartalmazva az ügyfelek és azok számláit.</p> <p>7. A Use case sikeresen zárul.</p>
Kiegészítés	<p>3.a Hibás felhasználó név (és/vagy) jelszó esetén „Sikertelen belépés. Hibás felhasználónév és/vagy jelszó.” üzenetet küld a program, felugró ablak formában.</p>

Ügyfelek listázása

Tranzakció hozzáadása(Use-Case 3):

Elsődleges Actor	Ügyfélkezelő / Accountant
Előfeltétel	A felhasználó be van jelentkezve.
Sikertelen eredmény	Tranzakciós hiba jelzése.
Sikeres eredmény	Tranzakció létrehozása
Események	<p>1. Tranzakciós adatok ellenőrzése</p> <p>2. Tranzakció létrehozása a Tranzakció hozzáadása gomb segítségével.</p> <p>3. A rendszer ellenőrzi hogy a forrás számlán van elég fedezet.</p> <p>4. A rendszer ellenőrzi hogy a forrás fájl nincs e zárva.</p> <p>5. A rendszer elküldi feldolgozásra a tranzakciót!</p> <p>6. A rendszer levonja a számláról a kezelési költséget</p> <p>7. Use Case sikeresen zárul.</p>
Kiegészítés	<p>3.a – Az utalandó összeg nagyobb, mint a számlán lévő fedezet, a program hibaüzenetet küld. „Nincs elég fedezet a tranzakció végrehajtásához.”</p> <p>4.a - Zárt számla esetén hibaüzenet küldése. „Tranzakció hozzáadása sikertelen. A forrás számla zárt.”.</p> <p>5.a – Tranzakció feltöltésre kerül az adatbázisban, majd a program sikeres visszajelző üzenetet küld.</p>

Tranzakció törlése(Use-Case 4):

Elsődleges Actor	Ügyfélkezelő / Accountant
Előfeltétel	A felhasználó be van jelentkezve.
Sikertelen eredmény	Tranzakciós hiba jelzése.
Sikeres eredmény	Tranzakció jóváhagyása.
Események	1. Tranzakció kiválasztása 2. Megerősítés kérése felugró ablakként. 3. Megerősítést követően a tranzakció törlése. 4. Sikeresen zárul a Use case.
Kiegészítés	2.a - Felugró ablakban megerősítő kérdés küldése „Biztos törli a tranzakciót?”. 2.b - „OK” gombra kattintás esetén törli a tranzakciót az adatbázisból, és visszajelző üzenetet küld a program.

Tranzakció könyvelése (Use-Case 5):

Elsődleges Actor	Ügyfélkezelő / Accountant
Előfeltétel	A felhasználó be van jelentkezve.
Sikertelen eredmény	Tranzakciós hiba jelzése.
Sikeres eredmény	Tranzakció jóváhagyása.
Események	1. Tranzakció kiválasztása 2. Megerősítés kérése felugró ablakként. 3. Megerősítést követően a tranzakció könyvelése. 4. Célszámla adatainak frissítése 5. Sikeresen zárul a Use case.
Kiegészítés	2.a - Felugró ablakban megerősítő kérdés küldése „Biztos törli a tranzakciót?”. 2.b - „OK” gombra kattintás esetén törli a tranzakciót az adatbázisból, és visszajelző üzenetet küld a program.
Elsődleges Actor	Ügyfélkezelő / Accountant

Non-funkcionális követelmények

A projekt megvalósításához JDK 8-t futtató rendszerre van szükség.

Operációs rendszerkövetelmény: A rendszer platform független, azaz Unix/Linux IOS és Windows rendszereken egyaránt fut.

A program helyes működéséhez szükségeltetik egy MySQL adatbázis kiszolgáló. A programban szereplő entitások, adatai egy accountmanager névre hallgató MySQL adatbázisból származnak.

Használhatósági követelmény

A program egy zárt banki rendszer szimulációjára képes.

Hatékonyági követelmény

A program válasz ideje rövid. Az adatok az adatbázisból dinamikusan töltődnek be.

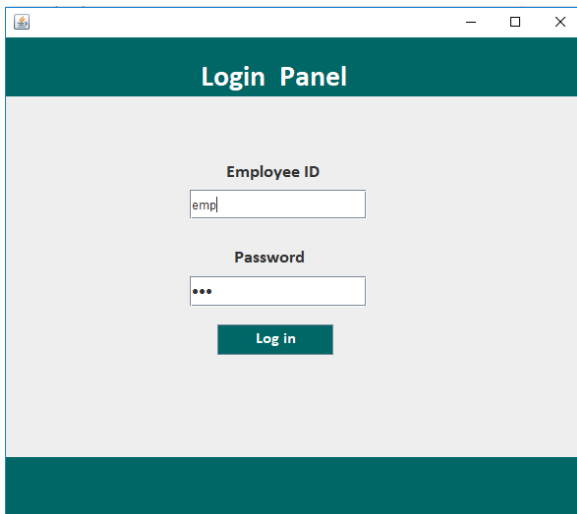
Biztonsági követelmény

Az adatbázisban lévő adatokat a szoftver biztonságosan tárolja. A jelszavakat hash függvény alkalmazásával titkosítva tároljuk. A rendszer alapvetően zárt és nem engedi kiszivárogtatni az információkat.

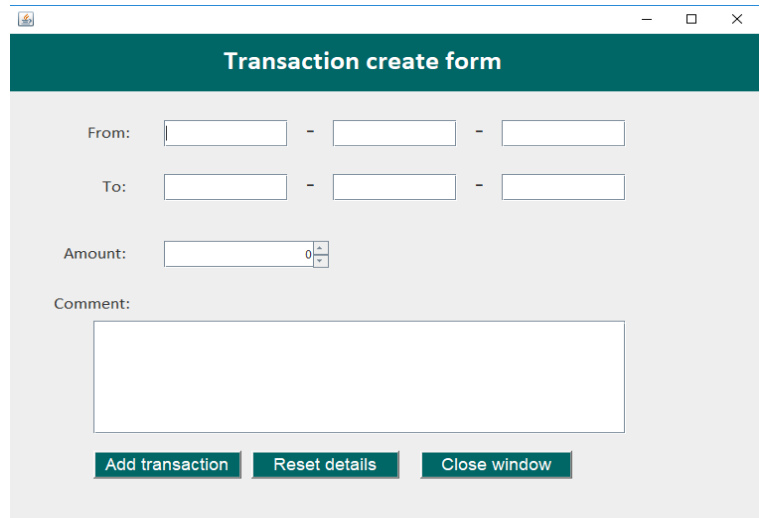
Hordozhatóság követelmény

A program bármely Java JDK-t futtató környezetben kompatibilis. Alapvetően platform független.

Látvány terv



The screenshot shows a web application window titled "Login Panel". It features a dark teal header bar with the title. Below the header, on a light gray background, there are two input fields: "Employee ID" with the text "empl" entered, and "Password" with three dots indicating a masked password. A dark teal "Log in" button is positioned below the password field. The window has standard OS controls (minimize, maximize, close) in the top right corner.



The screenshot shows a web application window titled "Transaction create form". It has a dark teal header bar with the title. The main area is light gray and contains several input fields: "From:" and "To:" each followed by three input boxes separated by dashes; "Amount:" followed by a single input box with a "0" and a small spinner icon; and a "Comment:" label above a large text area. At the bottom, there are three dark teal buttons: "Add transaction", "Reset details", and "Close window". The window includes standard OS controls in the top right corner.

Account Manager Panel

Signed in as emp

Customers

ID	First Name	Last Name	Address	Phone
1	Németh	Árpád	1118. Bp. Frankhegy ...	06209901142
2	Horváth	Kata	1221 Budapest, Pede...	06307894420
3	Kiss	Eszter	1035 Budapest, Bécs...	06207745546
4	Szabó	Zoltán	2400 Budaörs, Rozm...	06707798088

Selected Customer:

Account list:

Transaction List

Transaction create

Select customer

List all transactions

Log out

Statikus terv:

Tábla: accountmanager.customers

adatmező neve	mező típus	jelentése
id	egész , nem NULL, elsődleges kulcs	rekord azonosító
first_name	szöveges mező, nem NULL	ügyfél vezetékeve
last_name	szöveges mező, nem NULL	ügyfél keresztnéve
address	szöveges mező, nem NULL	ügyfél lakcíme
phone	szöveges mező, nem NULL	ügyfél telefonszáma

Tábla: accountmanager.accounts

adatmező neve	mező típus	jelentése
id	egész , nem NULL, elsődleges kulcs	rekord azonosító
account_number	szöveges mező, nem NULL	számlaszám
balance	egész	egyenleg
locked	logikai	zárolt-e a számla
customer_id	egész	ügyfél azonosító
date	szöveges mező	Számla létrehozásának dátuma.

Tábla: accountmanager.transactions

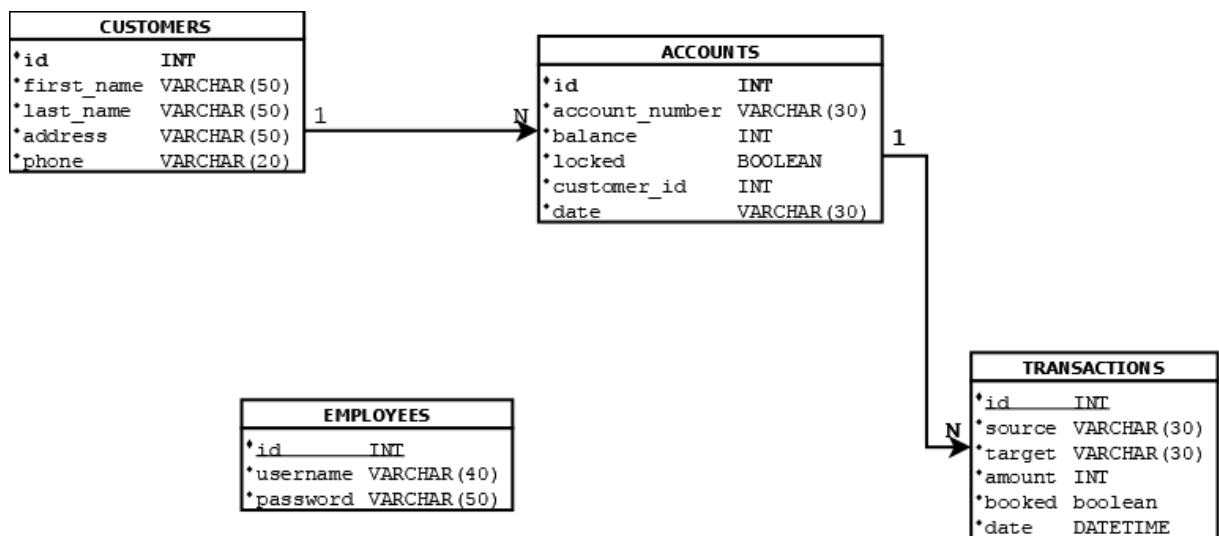
adatmező neve	mező típus	jelentése
id	egész , nem NULL, elsődleges kulcs	rekord azonosító
source	szöveges mező	forrás számla
target	szöveges mező	cél számla
amount	egész	utalandó összeg
booked	logikai	teljesített-e az utalás
date	szöveges mező	tranzakció létrehozásának dátuma

Tábla: accountmanager.employees

adatmező neve	mező típus	jelentése
id	egész, nem NULL, elsődleges kulcs	rekord azonosító
username	szöveges mező	alkalmazott felhasználóneve
password	szöveges mező	alkalmazott jelszava

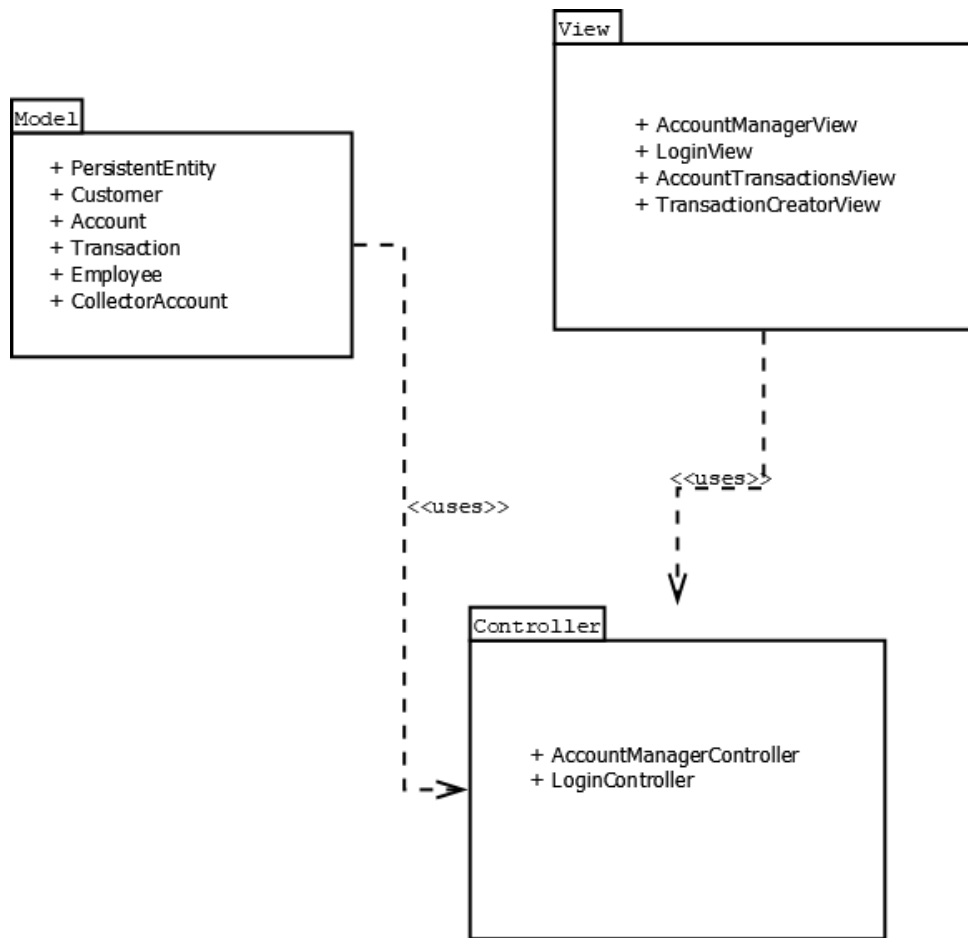
Egyed-kapcsolatok:

Az alábbi Egyed-Kapcsolat diagramm az adattáblák és a hozzájuk tartozó entitásokat tartalmazza, valamint a köztük lévő kapcsolatokat.



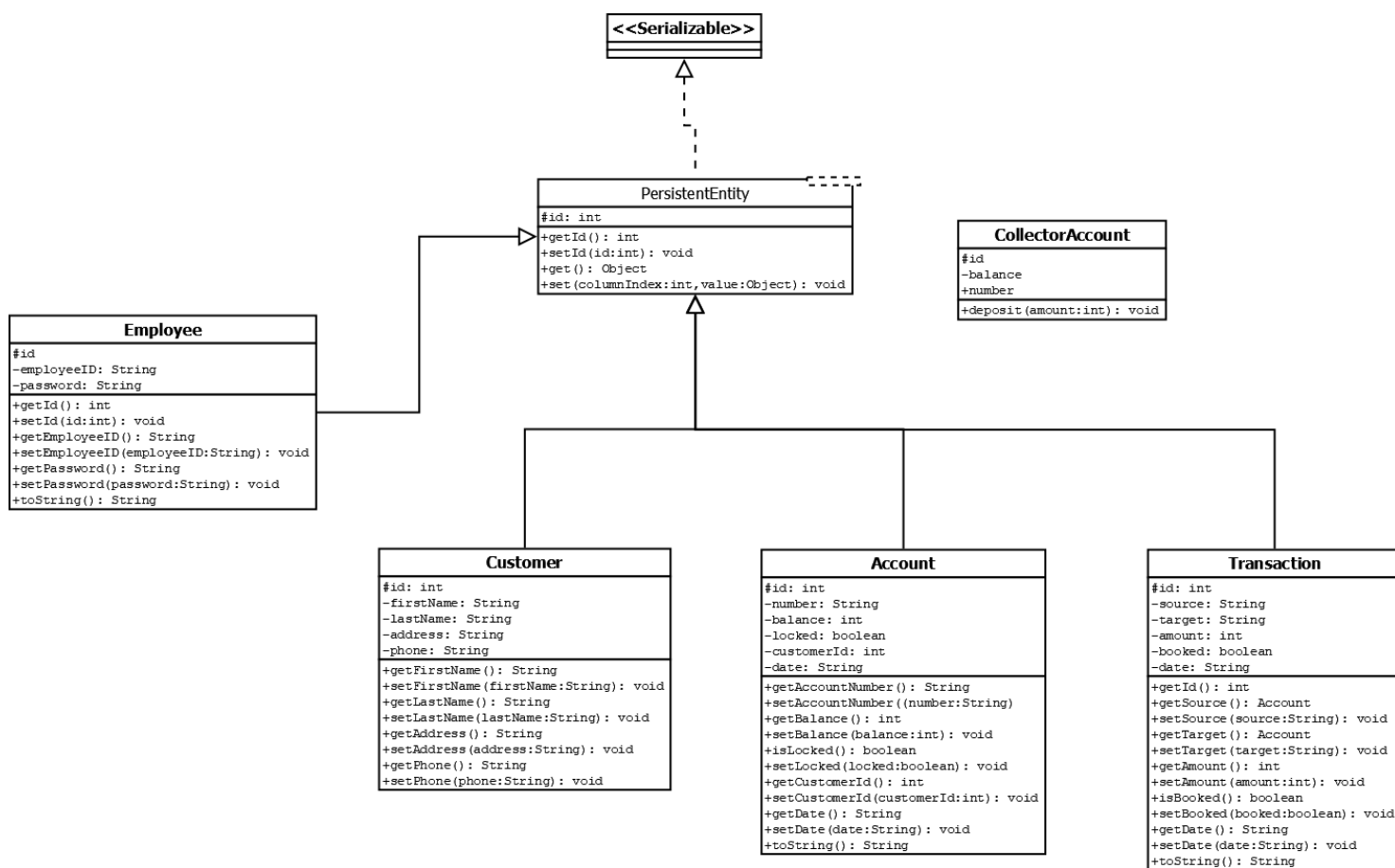
Osztályszerkezet

Az applikáció háromrétegű MVC (Model View Controller) architektúrának megfelelően készült. Az alábbi osztálydiagrammok az applikációban reprezentált osztályok hierarchiáját ábrázolja és írja le.



Model

A Model osztályai hozzáférnek az adatbázishoz. A diagramm csak a lényegesebb metódusokat tartalmazza.



PersistentEntity

Absztrakt entitás. A programban szereplő további entitások őseként funkcionáló osztály, mely tartalmaz egy egyedi azonosítót. Ezen kívül az entitások megvalósítják a Serializable interfészt. Az adatbázisban szereplő rekordokat(adatokat) felelteti meg a programban.

Customer

Az ügyfelek megkezelésére szolgáló entitás. Tartalmazza az alábbi információkat: egyedi azonosító, vezetéknév és keresztnévét, az ügyfél lakcímét, valamint telefonszámát.

Account

Az ügyfelek számláinak kezelését végzi. Rendelkezik egy egyedi azonosító illetve egyedi számlaszámmal, annak egyenlegével, a létrehozásának dátumával, a számlához tartozó ügyfél azonosítójával (customer_id), illetve tartalmazza azt az információt, hogy a számla zárolva van-e.

Transaction

A számlák közötti tranzakciók végrehajtásáért felelős entitás. Rendelkezik egy id „azonosítóval”, egy forrás és egy cél számla számmal, egy összeggel amelyet utalunk a forrás számláról, továbbá a tranzakció létrehozásának dátumával, illetve hogy a tranzakció le lett e könyvelve a rendszerben.

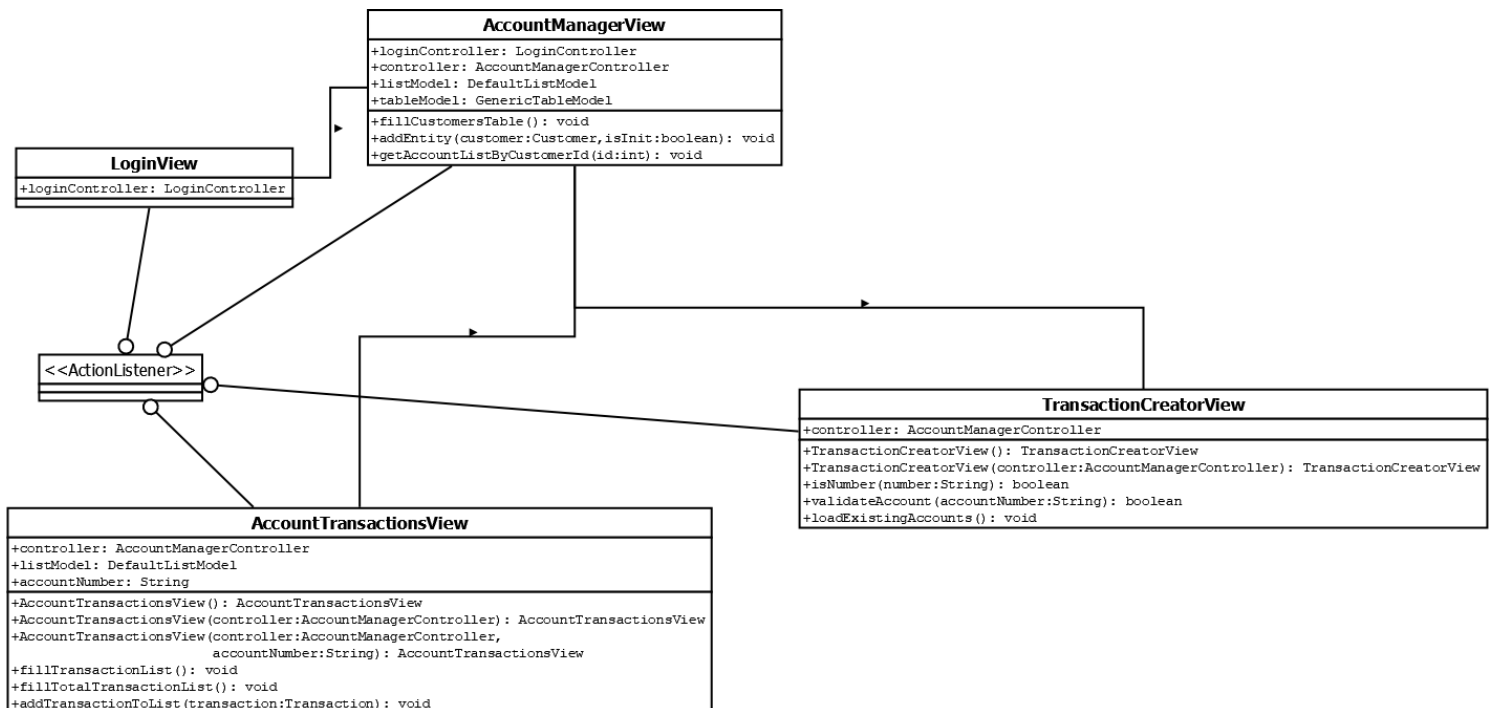
Employee

Alkalmazottakat reprezentáló osztály. Rendelkezik egy egyedi azonosítóval, egy alkalmazott azonosítóval (employee_id), illetve a hozzá tartozó jelszóval.

CollectorAccount

A rendszer gyűjtőszámláját megfelelő osztály. Rendelkezik egy deposit(int amount) metodussal, mellyel tranzakció létrehozása esetén a forrás számlára terhelt kezelési költséget gyűjti és tölti fel a számlára.

View



LoginView

A kezdő beléptető felület nézete. Az űrlapon történő megfelelő alkalmazott azonosító (employeeID) és a hozzá tartozó jelszó megadását követően, a login gomb-ra kattintva azonosíthatjuk magunk és nyerhetünk hozzáférést az AccountManagerView nézethez.

AccountManagerView

Az ügyfélkezelő program fő nézete. Tartalmaz egy táblázatot a rendszer ügyfeleinek adataival. Ezen táblázatból egyet kiválasztva a Select Customer gombra kattintás után a nézeten megjelenítődik egy listában, a kiválasztott ügyfél számla adatai. Ezen listából egyet ismét kiválasztva, és a List transactions gomb segítségével tudjuk megjeleníteni, egy AccountTransactionsView nézetben a kiválasztott számlához tartozó összes mint teljesített, mind teljesítetlen tranzakciót.

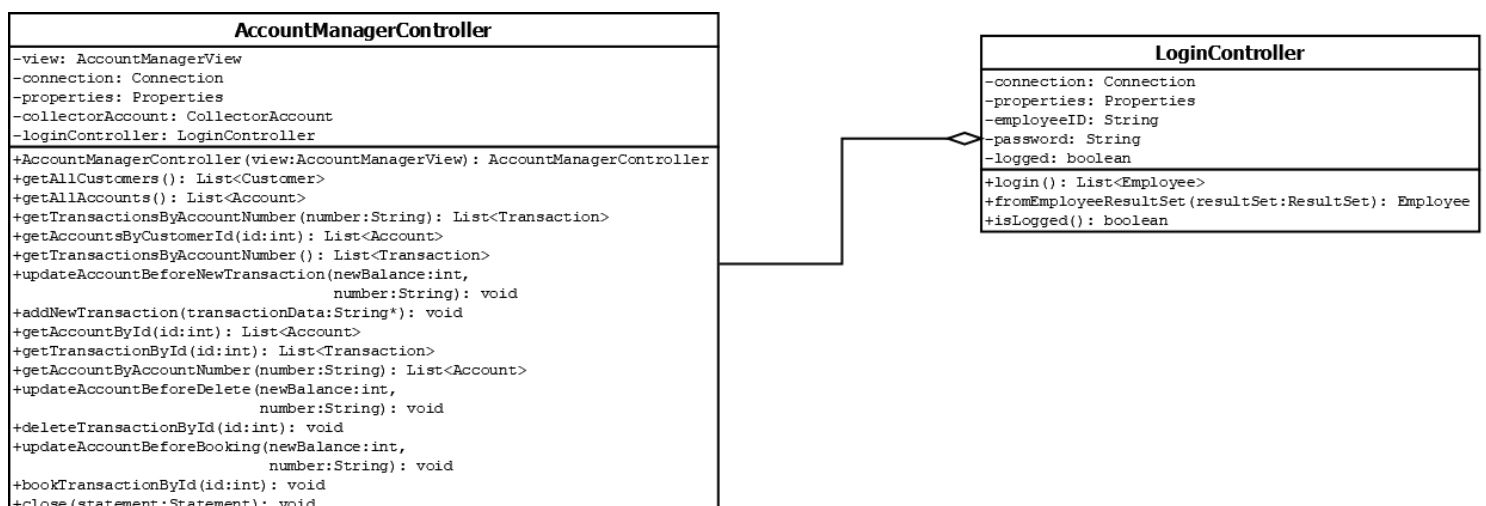
AccountTransactionsView

Az AccountManagerView nézeten korábban kiválasztott számlához tartozó vagy éppen az összes tranzakció megjelenítésére szolgáló nézet. Az AccountManagerView nézeten a List transactions gombra kattintásával hívhatjuk elő a nézetet. A Book transaction és delete transaction gomb használatával tudjuk a listából egy kiválasztott tranzakciót véglegesíteni, valamint sztorizni. A close gomb bezárja a nézetet.

TransactionCreatorView

Új tranzakció létrehozására szolgáló adatbeviteli űrlapot tartalmazó nézet. A „tranzakció indítása” gombra kattintva értesíti a Controllert, új tranzakció hozzáadásáról. Az űrlapon több hiba vétése esetére a reset details gombal törölhetjük a beviteli mezők már meglévő értékeit. A close gomb becsukja az ablakot.

Controller



LoginController

Az autentikációért felelős controller. Megvalósítja a háttérlogikát, illetve ez az osztály fér hozzá az adatbázishoz. A login() metódus segítségével egy alkalmazott a rendszerbe történő belépését teszi lehetővé.

AccountManagerController

Az AccountManagerView nézet vezérlő osztálya. Ez az osztály hozzáférést biztosít az adatbázishoz, tranzakciók törlése, hozzáadása, könyvelése esetén, módosítja az adatbázisban az adatokat, valamint frissíti a megfelelő nézetet is az adatokat. Emellett ez az osztály kezeli a CollectorAccount gyűjtőszámla osztály állapotait is.

Tesztelési terv:

Adatok dinamikus betöltése:

A szoftver teszteléséhez legalább szükséges, egy háttérben futó valamilyen mysql kiszolgáló, valamint a szoftver feltételezi, hogy létezik „accountmanager” névvel ellátott adatbázisunk.

Tesztelési segédlet:

mysql config:

host: localhost

user: root,

pass: (üresen hagyva)

beléptetéshez:

employeeID: employee1

password: 123

AccountManager:

- ✓ Adatbázis adatokkal való feltöltése
- ✓ LoginView kezdő nézet megjelenítése
- ✓ Controllerek létrehozása

LoginView:

- ✓ Beléptető felület megjelenítése
- ✓ Űrlap adatok feldolgozása
- ✓ Hibás illetve üres felhasználói név vagy jelszó megadás esetén a program felugró ablak formájában értesít minket a hibáról.
- ✓ Helyes adatok megadása esetén, belépteti az ügyfélkezelőt a rendszerbe, majd megjeleníti az AccountManagerView nézetet.

AccountManagerView:

- ✓ Ügyfélkezelő sikeres belépése esetén, megjeleníti egy táblázatban az rendszer ügyfeleinek adatait.
- ✓ Ügyfelet kiválasztva (select customer) lehet a táblázatban kijelölt ügyfélhez tartozó számlákat kilistázni.
- ✓ Amennyiben nem választottunk ki ügyfelet, a program felugró ablak formájában értesít ennek hiányosságáról.
- ✓ A list transactions gomb-ra kattintáskor, a program megjelenít egy AccountTransactionsView nézetet, tartalmazva benne a tranzakciók listáját, a kiválasztott számlához, ha korábban már kiválasztottunk egy számlát a megjelenített számlák listájából.
- ✓ Ha nincs a listában kijelölt számla, akkor a list transactions gombra történő kattintáskor az AccountTransactionsView nézet az összes tranzakciót megjeleníti.
- ✓ Create transactions gombra történő kattintáskor, megjelenik egy TransactionCreatorView űrlap nézet, melyen új tranzakciókat indíthatunk a rendszerben.
- ✓ Log out gombra történő kattintáskor, a rendszer kijelentkezteti az ügyfélkezelőt a programból és visszacod a LoginView nézetre.

AccountTransactionsView:

- ✓ Egy tranzakciót kiválasztva az összes, vagy épp szelektált tranzakciókból, a Book transaction gomb használatával könyvelhetjük le azt. Tranzakció teljesítése esetén, a tranzakciós összeg levonásra kerül a célszámláról, és frissülnek az adatok, mind az adatbázisban, mint a megjelenítési felületen.
- ✓ Egy tranzakciót kiválasztva az összes, vagy épp szelektált tranzakciókból, a Delete transactions gom segítségével törölhetjük a tranzakciót. Törlés esetén, a tranzakciós összeg visszakerül forrás számlára.
- ✓ Close gomb-ra a program bezárja a nézetet.

TransactionCreatorView:

- ✓ Megjeleníti a tranzakció indításához szükséges űrlapot.
- ✓ A forrás(from) és cél(to) számla mezők kitöltése, valamint az összeg (amount) megadása után, ha rákattintunk az add transaction gombra, a program ellenőrzi az adatok helyességét, azaz hogy pontosan 3x 8 karakter hosszú csak számsorozatot

adtunk-e meg, valamint hogy van e elég fedezet a számlán. Amennyiben megfelelőek, elindítja a tranzakciót.

- ✓ A Reset details gombbal több hibás elírás esetén, törölhetjük a már eddigi beírt adatokat az űrlapról.
- ✓ Close gomb-ra bezárja a program az ablakot.

Unit tesztek

A program tesztelése manuális tesztelés és debugolás mellett, a controller tesztelése unit tesztekkel, is végre lett hajtva. A unit tesztek JUnit 4.12-es verzió segítségével hajtottam végre. A unit tesztek a Test könyvtár alatt a default packages csomagban találhatóak. Tesztelés során, a különböző tesztesetek, egyaránt sikeresen zárultak.

AccountManagerControllerTest

1.teszteset: getAllCustomersTest()

Ellenőrizi hogy megfelelően létrehozta-e a program a customers táblát és feltöltötte-e adatokkal.

2.teszteset: getAllAccountsTest()

Ellenőrzi hogy az accounts tábla létezik és fel van-e töltve megfelelően adatokkal.

3.teszteset: getAllTransactionsTest()

Ellenőrzi hogy a transactions tábla létezik és fel van-e töltve megfelelően adatokkal.

4.teszteset: updateAccountBeforeDeleteTest()

Teszteli, hogy tranzakció adatbázisból való végleges törlése előtt, a tranzakciós összeg visszakérült-e a forrásszámlára.

5.teszteset: updateAccountBeforeBookingTest()

Teszteli, hogy a tranzakció könyvelése előtt a célszámlára valóban átutalódott az összeg.

6.teszteset: deleteTransactionByIdTest()

Tranzakció törlésénél, ellenőrzi, hogy a tranzakció ténylegesen törlődött az adatbázisból.

7.teszteset: bookTransactionByIdTest()

Ellenőrzi, hogy tranzakció könyvelése esetén a tranzakciónak megfelelő rekord booked mezője valóban módosítva lett.

8.teszteset: getAccountByAccountNumberTest()

Ellenőrzi, hogy adott számlaszámhoz csak egy rekord tartozik-e, és hogy megfelelően hajtotta-e végre a lekérdezést a controller.

9.teszteset: getTransactionById()

Ellenőrzi, hogy adott tranzakció azonosítóhoz csak egy rekord tartozik-e, és hogy megfelelően hajtotta-e végre a lekérdezést a controller.

10.teszteset: collectorAccountTest()

Teszteli, hogy tranzakció létrejöttkor, a program megfelelően kiszámolja-e a kezelési költséget(3%-a a tranzakciós összegnek) és h feltölti-e a gyűjtő számlára.