

Budapesti Műszaki Szakképzési Centrum
Neumann János Számítástechnikai Szakgimnáziuma

Szakképesítés neve: Szoftverfejlesztő ***OKJ***

száma: 54 213 05

ZÁRÓDOLGOZAT

Zálogházi nyilvántartó program

Várkonyi Tibor

konzulens

Kövérné Gargya
Viktória

14.rsze

Budapest, 2020.

TARTALOMJEGYZÉK

Bevezetés.....	5
Témaválasztás	5
A probléma rövid ismertetése	5
Felhasználói dokumentáció	6
Rendszerigény	6
Beüzemelés.....	6
Bejelentkezés.....	6
Jogosultsági szabályok	7
Dolgozók jogosultságai	7
Zálogigazgató jogosultságai	7
Ügyfelek jogosultságai	7
Dolgozói felhasználói felület elemei, funkciói	8
Zálogjegy felvétel.....	8
Zálogjegy kezelés	9
Új ügyfél felvétel.....	10
Zálogigazgató felhasználói felülete és funkciói	11
Zálogfiókok nyilvántartása.....	11
Dolgozók nyilvántartása.....	12
Zálogjegyek bevonása kényszerértékesítésre	12
Ügyfelek felhasználói felülete és funkciói	13
Fejlesztői dokumentáció.....	15
Rendszerterv	15
Adatbázis terv.....	15
Adattáblák	16
Megvalósítás.....	17
Frontend forráskód	18
Backend forráskód.....	19
Config package.....	20
Controller package	21
Service package.....	21
Repository package	22
Zaloghaz pacage.....	22

Bevezetés

Biztonság – Spring Security	23
Tesztelés	23
Összegzés.....	25
Fejlesztési lehetőségek.....	25
felhasznált források.....	27

BEVEZETÉS

Témaválasztás

A záródolgozatom témájának azért választottam a zálogházi nyilvántartást, mert több éves tapasztalatom van zálogházi tevékenység működésében. Tisztában vagyok a zálogházi ügymenettel valamint működési nehézségeivel.

A probléma rövid ismertetése

Egy zálogháznak készítek egy belső nyilvántartói rendszert, valamint ügyfelek részére egy mobil webalkalmazást zálogjegyük egyszerű nyomon követéséhez.

Sokan megfordulnak zálogházba – vállalkozók, magánszemélyek – akiknek azonnali gyors kölcsönre van szükségük.

Szeretnék egy olyan rendszert létrehozni, ami a dolgozók és az ügyfelek igényeihez rugalmasan igazodik és kezeli a zálogházba felvett tárgyak adatait. Elzálogosítás esetén új tárgy felvitele az adatbázisba, hosszabbítás esetén az adatok módosítása, illetve kiváltás esetén az adatok törlése.

Az adatbázist egy központi szerveren szeretném tárolni, mivel különböző helyeken szeretném azokat elérni. Például, azért mert több ügyfél felvetette, hogy a hitelek hosszabbítása bármelyik zálogfiókba egyszerűen lebonyolítható lehessen, ne csak ahol elzálogosította a tárgyat.

Az adatbázis különböző hozzáférési jogokkal lehessen használni. A zálogigazgató hozzáférhet az egész adatbázishoz, le tud kérdezni belőle. A zálogfiók – ahol az elzálogosítás történik – tud felvinni új adatot az adatbázisba, azt módosítani (kiváltás, hosszabbítás), lekérdezéseket végezni. A zálogfiók csak a hosszabbítást tudja elvégezni a más fiókban elzálogosított tárgynál. Az ügyfél csak a saját elzálogosított tárgyait látja a mobil webalkalmazáson keresztül – mikor, melyik zálogfiókba, milyen kölcsönösszeggel lett elzálogosítva, valamint mennyi kamat van rajta az adott napon.

FELHASZNÁLÓI DOKUMENTÁCIÓ

Rendszerigény

- Internet kapcsolat
- Webböngésző (Mozilla Firefox, Google Chrome, Opera, Microsoft Edge, Safari)

Beüzemelés

Ahhoz, hogy be tudjuk üzemelni a rendszert futnia kell Tomcat, Apache és MySQL szervernek. Első indítás előtt létre kell hozni egy „zaloghaz” nevű adatbázist, amiben a program első indításakor létrehozza a táblákat.

A szerver beüzemelését az IntelliJ IDEA fejlesztői környezet segítségével lehet elvégezni.

Második indítás előtt az „application.properties”-ben át kell állítani a következő sort, hogy ne create legyen, hanem **update**:

```
spring.jpa.hibernate.ddl-auto=create
```

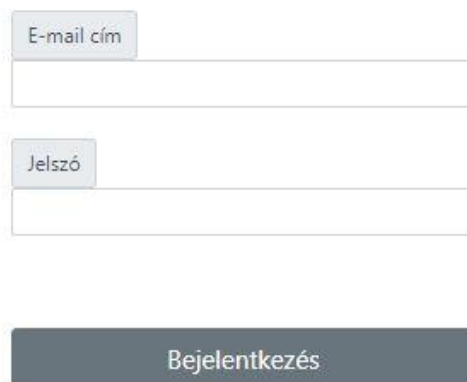
Ezt a JPA négy különböző funkcióval tudja meghívni:

- create (elkészíti az adattáblákat),
- create-drop (elkészíti az adattáblákat aztán törli azt),
- update (frissíti az adatbázist, ha van különbség)
- és validate (csak ellenőrzi, hogy van-e különbség, de nem végezz módosítást).

Bejelentkezés

A felhasználók e-mail címmel és jelszóval tudnak belépni az alkalmazásba (1. ábra), valamint a zálogfiók a címével. A felhasználóknak nincs lehetőségük saját magukat regisztrálni, az a programon belül történik meg. A zálogigazgató regisztrálja a zálogfiókot, és a dolgozók a zálogfiók felhasználói fiókon keresztül regisztrálják az ügyfeleket. A zálogfiók jelszavát a zálogigazgató állíthatja be, amit a zálogfiók vezetőjének eljuttat. Minden ügyfél a belépéshez szükséges jelszavát e-mail üzenetben kapja meg.

A kor szellemének megfelelően az adatbázis pusztán csak digitális verzióban működik, annak érdekében, hogy felesleges papírfelhasználás ne keletkezzen. Éppen ezért minden ügyfél köteles megadni az e-mail címét, a zálogjegye megtekintése érdekében.



The image shows a login form with two input fields. The first field is labeled 'E-mail cím' and the second field is labeled 'Jelszó'. Below these fields is a button labeled 'Bejelentkezés'.

1. ábra: Bejelentkező felület az alkalmazásba

Jogosultsági szabályok

Dolgozók jogosultságai

A dolgozók hozzá tudnak adni új ügyfeleket az ügyfél táblához, ezzel regisztrálva őket, hogy hozzáférjenek az alkalmazáson keresztül a zálogjegyeik lekérdezéseihez.

A becsüs feladata az új zálogtárgy elzálogosítása, ekkor hozzárendeli a zálogjegyet a megfelelő ügyfélhez.

A pénztáros feladata a hosszabbítás és a kiváltás lebonyolítása az ügyfél beazonosítása után.

Zálogigazgató jogosultságai

A zálogigazgató kezeli a zálogfiókok és dolgozók listáját. Nyithat új zálogfiókot, ezzel biztosítja a belépést az alkalmazásba, és be is zárhat. Valamint fel vehet új dolgozót, és meg is szüntetheti munkaviszonyát.

A zálogjegyeket le tudja kérdezni, és be tudja vonni kényszerértékesítésre.

Ügyfelek jogosultságai

Az ügyfelek bejelentkezési lehetőséget kapnak egy felületre, amit akár kényelmesen mobilról is nézhetnek. A felületen a saját ügyfélazonosítóval rendelkező zálogjegyek adatait, a hosszabbítás és kiváltás költségét kérheti le.

Dolgozói felhasználói felület elemei, funkciói

Ahogy az 2. ábra mutatja a felhasználói felületet több részre bontottam szét, azért hogy elkülönüljenek egymástól a különböző funkciók.



2. ábra: Dolgozói menüsor

Zálogjegy felvétel

A 3. ábra mutatja be az új zálogtárgy elzálogosítás folyamatát.

A screenshot of a form for 'Zálogjegy felvétel'. The form contains several input fields and a button. At the top, there are two date fields: 'Beadás dátuma' with the value '2020-05-11' and 'Lejárat dátuma' with the value '2020-08-09'. To the right of these is a dropdown menu. Below the dates is a large text area labeled 'Zálogtárgy leírása'. At the bottom, there are four more fields: 'Karát' with a dropdown showing '6', 'Darabszám' with the value '0', 'Súly' with the value '0,0', and 'Kölcsön összeg' with the value '0'. A dark grey button labeled 'Felvétel' is positioned at the bottom center of the form.

3. ábra: Elzálogosítás

A beadás dátuma az aznapi dátum, amikor az elzálogosítás történik. A lejárat dátum a beadás dátumától számított 90. nap. Ezt a két értéket a program maga generálja le.

A dátumok melletti lenyíló mező az ügyfelek listáját tartalmazza személyigazolvány számmal együtt. Ha az elzálogosító már regisztrálva van, akkor megjelenik a listába a neve, ellenkező esetben az „Új ügyfél felvétel” menüpont alatt kell regisztrálni. A zálogjegyet csak az elzálogosító válthatja ki.

„Zálogtárgy leírása” egy kötelezően kitöltendő mező, aminek minimum 5 karakter hosszúnak kell lennie.

A „karát”-nak a számát egy lenyíló listából lehet kiválasztani, lehetséges értékei: 6, 8, 10, 14, 18.

A „darabszám”-nak minimum 1-nek kell lennie, 1-esével lehet növelni vagy csökkenteni az értéket.

A „súly” század pontosságúra adható meg, minimum 0,5-nek kell lennie, 0,01-el lehet növelni vagy csökkenteni az értékét.

A „kölsön összeg” minimum 2000 Ft-nak kell lennie, 500-zal lehet növelni vagy csökkenteni az értékét.

A felvétel gombra való kattintáskor megtörténik az elzálogosítás, az adatok belekerülnek az adatbázisba.

Zálogjegy kezelés

A 4. ábra mutatja be a „Hosszabbítás” és a „Kiváltás” felületét.

4. ábra: Kezelés

A zálogjegy sorszámát és a kölcsönösszeget kötelező megadni a zálogjegy beazonosítása érdekében. Az ügyfél, a beadás dátuma, a lejárat dátuma, a zálogtárgy leírása, a karát, darabszám és a súly mezőket az „OK” gomb lenyomásakor kitölti.

A fizetendő összeg a kiváltás esetén a kölcsön összeg, az eltelt napokra a kamat és a kezelési költség összege.

„Kiváltás” gombra való kattintáskor törlődik az adatbázisból az adat.

Hosszabbítás esetén a fizetendő összeg az eltelt napokra a kamat és a kezelési költség összege.

„Hosszabbítás” gombra való kattintáskor a beadás dátuma megváltozik az aznapi dátumra.

Ahogy az 5. ábra mutatja, a lekérdezésnél van egy „kiváltás dátuma” mező, ahol tervezett kiváltási dátumot lehet megadni, és az alapján számolja ki a hosszabbítás illetve a kiváltás összegét.

Zálogjegy sorszáma	Kiváltás dátuma
<input type="text" value="0"/>	<input type="text" value="éééé. hh. nr"/>
Kölcsön összeg	OK
<input type="text" value="0"/>	

5. ábra: Lekérdezés

Új ügyfél felvétel

Az 6. ábra új ügyfelek regisztrációját valósítja meg.

Név	Személyigazolvány szám
<input type="text"/>	<input type="text"/>
Anyja neve	
<input type="text"/>	
Cím	
<input type="text"/>	
E-mail	
<input type="text"/>	
Új ügyfél	

6. ábra: Új ügyfél felvétel

Minden mező kötelezően kitöltendő mező. Az „e-mail” mezőn keresztül van biztosítva az ügyfélnek a hozzáférés a webalkalmazáshoz, ahol nyomon tudja követni zálogjegyeit.

A személyigazolvány számnak és az e-mail címnek minden ügyfélnek egyedinek kell lennie.

Az „Új ügyfél” gombra kattintva felkerül az ügyfél az adatbázisba, és az „Zálogjegy felvétel” felületre kerülünk át.

Zálogigazgató felhasználói felülete és funkciói

Zálog **Zálogfiók** **Dolgozók** **Zálogjegyek bevonása kényszerértékesítésre**

7. ábra: Zálogigazgatói menüsor

A zálogigazgatónak is szétszedtem a felületét – ahogy a 7. ábra mutatja –, hogy itt is elkülönüljenek egymástól a funkciók.

Zálogfiókok nyilvántartása

Ezen a felületen a zálogigazgatónak lehetősége van új zálogfiók rögzítésére – ha nyitnak egy újat – a zálogfiók címének és telefonszámának megadásával. Valamint jelszó megadásával, hogy be lehessen jelentkezni a webalkalmazásba és a zálogjegy kezelését lebonyolítani. A zálogfiókok listájának címét és telefonszámát egy táblázatban lehet látni.

The interface consists of three input fields for registration: 'Cím', 'Telefonszám', and 'Jelszó'. Below these is a table with two columns: 'Cím' and 'Telefonszám'. At the bottom right, there are two buttons: 'Új Zálogfiók' and 'Zálogfiók törlése'.

8. ábra: Zálogfiókok kezelése

Ahogy a 8. ábra mutatja lehetőség van zálogfiók törlésére is. Ez akkor szükséges, ha bezárnak egy zálogfiókot. Egy legördülő listából lehet kiválasztani a zálogfiók címét, és a „Zálogfiók törlése” gombbal lehet törölni. Akkor lehetséges a törlés, ha nincs a zálogfiókhoz tartozó dolgozó és/vagy zálogjegy.

Dolgozók nyilvántartása

Ahogy a 9. ábra bemutatja a dolgozói nyilvántartás nagyon hasonló a zálogfiókok kezeléséhez. Itt fel lehet vinni új dolgozókat

- „név”, „telefonszám”, „e-mail cím” kötelezően kitöltendő, aminek egyedinek kell lennie
- valamint a zálogfiókot és a beosztást egy legördülő menüből lehet kiválasztani.

Egy táblázatban látható a dolgozók listája és adatai.

The interface consists of several input fields and a table. At the top, there are five input fields: 'Név' (Name), 'Telefonszám' (Phone number), 'E-mail cím:' (Email address), 'Zálogfiók' (Mortgage account), and 'Beosztás' (Position). Below these is a table with the following columns: 'Id', 'Név', 'Telefonszám', 'E-mail cím', 'Zálogfiók', and 'Munkakör'. Below the table is a 'Név' dropdown menu. At the bottom right, there are two buttons: 'Új Dolgozó' (New Employee) and 'Dolgozó törlése' (Delete Employee).

9. ábra: Dolgozók kezelése

Itt is, mint a zálogfiók kezelésnél lehetőség van dolgozó törlésére. Egy legördülő menüből lehet kiválasztani a kilépett dolgozót.

Zálogjegyek bevonása kényszerértékesítésre

A zálogjegyek bevonása kényszerértékesítésnél (10. ábra) a lejárt zálogjegyeket lehet kezelni. A „beadás dátuma” automatikusan az aktuális dátumtól számított 90 naptári nappal előbbi dátumra van állítva. A „Listáz” gombra való kattintáskor kilistázza a zálogjegyeket és adatait egy táblázatba. A táblázat alatt található „Bevonás” gombra való kattintáskor a zálogjegyek törlődnek az adatbázisból és a zálogigazgató kényszerértékesítheti.

Beadás dátuma

2020. 02. 10.

Listáz

Sorszám	Ügyfél	Beadás dátuma	Leírás	Db	Súly	Karát	Kölcsön összeg
<div>Bevonás</div>							

10. ábra: Zálogjegy bevonása kényszerértékesítésre

Ügyfelek felhasználói felülete és funkciói

Zálog

Zálogjegy sorszáma

Kiváltás dátuma

2020. 05. 10.

Amennyiben a lejárat nap utáni dátummal szeretné kiváltani, abban az esetben nem tudjuk garantálni, hogy a zálogtárgy még nem került kényszerértékesítésre

OK

Zálogfiók címe:

Zálogfiók telefonszáma:

Elzálogosítás összege:

Elzálogosítás dátuma:

Lejárat dátuma:

Zálogtárgy leírás:

Hosszabbítás összege:

Kiváltás összege:

11. ábra Ügyfél felület

Az ügyfeleknek létrehoztam egy külön ügyfeleknek szóló felhasználói felületet, amit kényelmesen mobilról is nézhetnek, ahogy a 11. ábra mutatja, ha megfelelő okos mobiltelefonjuk van interneteléréssel.

A felület egy legördülő listában tartalmazza az adott ügyfél zálogjegyeinek sorszá-
mát. Valamint tartalmaz egy „Kiváltás dátuma” mezőt, ahol beállíthatja, hogy mikor ter-
vezi az adott zálogjegy kiváltását vagy hosszabbítását. Az alapértelmezetten beállított dá-
tum az aktuális dátum.

Az „OK” gomb megnyomása után a kiválasztott zálogjegy sorszáma alapján megje-
lennek a következő adatok:

- elzálogosítás helyszíne, és zálogfióknak a telefonszáma,
- az elzálogosított tárgy kölcsönösszege (elzálogosítás összege), mikor történt az elzálogosítás, mikor jár le a zálogjegy, és az elzálogosított tárgy leírása,
- valamint a beállított kiváltási napon felmerülő hosszabbítás és kiváltás költ-
sége.

FEJLESZTŐI DOKUMENTÁCIÓ

Rendszerterv

A zálogházi nyilvántartó programot két felületre terveztem. Az egyik az ügyfelek számára megtervezett mobil webalkalmazás, a másik felület a zálogházban dolgozók számára létrehozott alkalmazás, amin keresztül történik a zálogházi adminisztráció. Ez a két felület elkülönül egymástól, de ugyanahhoz az adatbázishoz férnek hozzá különböző jogosultságokkal.

A programon belül exportáló funkció nincs, éppen ezért az adatbázis biztonságos mentéséről a rendszergazdának kell gondoskodni.

Adatbázis terv

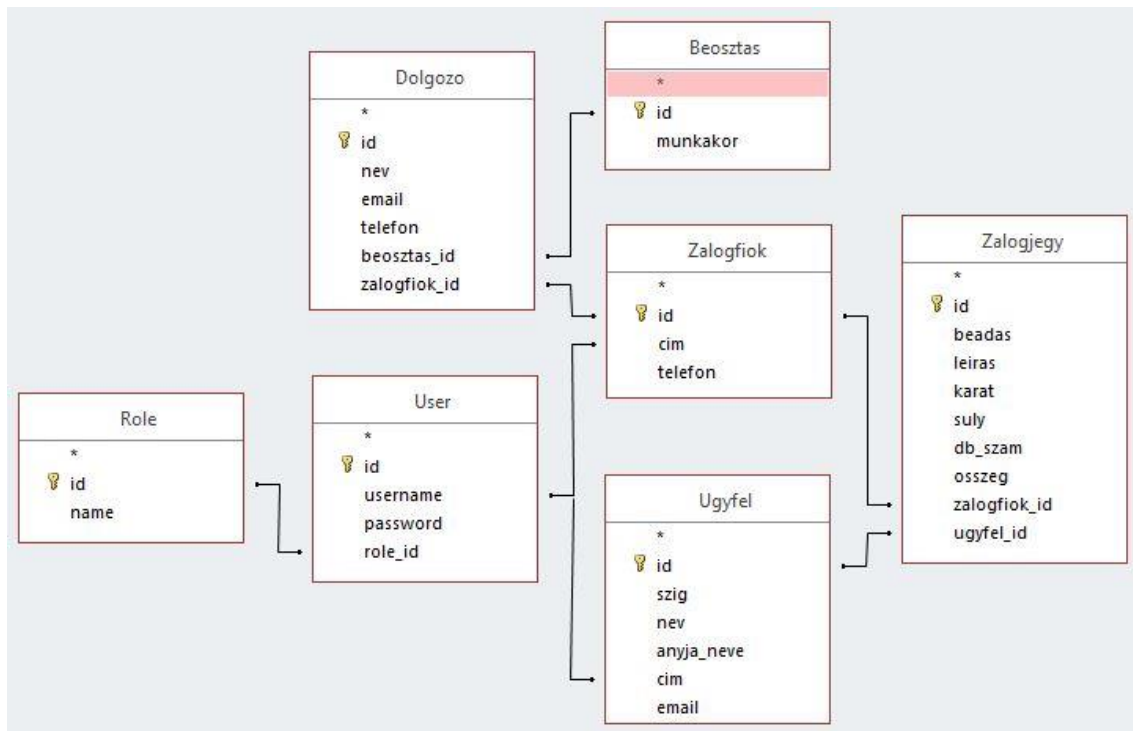
Első lépések egyike átgondolni, hogy kik fogják használni a rendszert, vagyis kik férnek hozzá az adatbázishoz és milyen jogosultsági szabályok szerint. Akik használják:

- zálogházi dolgozók
 - zálogfiókban dolgozó (pénztáros, becsüs)
 - zálogigazgató
- zálogházi ügyfelek

Valamint a másik fontos átgondolni való, hogy mire fogják használni az adatbázist. Jelen esetben egy akár több fiókkal rendelkező Zálogházi profillal rendelkező cég zálogjegyeinek kezelése a megvalósítandó feladat.

Az adatbázisban a duplikációk elkerülése érdekében normalizáltam az adattáblákat, így több, tömörebb különálló adattáblát hoztam létre.

Adattáblák



12. ábra: Relációs adatbázis-terv

Ahogy az adatbázis-terv táblán (12. ábra) látható, mindegyik tábla rendelkezik egy egyedi, numerikus típusú azonosítóval, „id”-val, ami automatikusan generálódik új rekord felvételekor.

A „**Dolgozo**” táblába került a dolgozó neve („nev”), az e-mail címe („email”), a telefonszáma („telefon”), amik karakter típusúak, valamint felkerült egy „Beosztas” táblához („beosztas_id”) és egy „Zalogfiok” táblához tartozó („zalogfiok_id”) idegen kulcs. Az adattáblába a zálogigazgató tud új rekordot felvinni és törölni.

A „**Beosztas**” táblába kerülnek a munkakörök („munkakor”), amik az alábbi értékeket tartalmazzák: becsüs, pénztáros.

A „**Zalogfiok**” táblába kerül a zálogfiók címe („cim”) és a zálogfiók telefonszáma („telefon”). Az adattáblába a zálogigazgató tud új rekordot felvinni és törölni.

A „**Zalogjegy**” tábla az elzálogosított tárgyak adatait tartalmazza, amik a következők:

- a beadás dátuma („*beadas*”), ami dátum típusú
- a zálogtárgy leírása („*leiras*”), ami karakter típusú
- a zálogtárgy karát száma („*karat*”), az elzálogosított tárgyak darabszáma („*db_szam*”) és a kölcsönösszege („*osszeg*”), ami egész szám típusú
- a tárgy/tárgyak súlya („*suly*”) lebegőpontos szám típusú.

Továbbá felkerült a táblába egy „Zalogfiok” táblához („*zalogfiok_id*”) és az „Ugyfel” táblához („*ugyfel_id*”) tartozó idegen kulcs. Az adattáblába a dolgozók tudnak új rekordot felvinni, módosítani (a beadási dátum módosítása), törlést (zálogjegy kiváltása) végezni.

Az „**Ugyfel**” tábla az ügyfelek beazonosításához szükséges adatokat tartalmazza karakter típusokban, amik a következők: ügyfél nevét („*nev*”), az ügyfél anyjának nevét („*anyja_neve*”), az ügyfél címét („*cim*”), valamint az ügyfél személyi igazolvány számát („*szi*g”) és az e-mail címét („*email*”), amiknek egyedinek kell lenniük. Az adattáblába a dolgozók tudnak új rekordot felvinni és zálogjegyhez kapcsolni.

A további két tábla a bejelentkezéshez („**User**”) és a szerepkörök („**Role**”) meghatározásához szükséges adatok miatt hoztam létre. A „User” táblához a „Zalogfiok” tábla a „*cim*” alapján, az „Ugyfel” tábla az „*email*” alapján kapcsolódik a „username”-hez.

Megvalósítás

A Java objektumok és a relációs adatbázis közötti adatok elérésére, kezelésére és megőrzésére **JPA** (Java Persistence API)-t használtam.

A JPA hídnak tekinthető az objektum-orientált tartománymodellek és a relációs adatbázis-rendszerek között.

Az adatbázisok csak skalár (pl stringek, egészek) értékeket tudnak tárolni és kezelni. Az ORM (Object Relational Mapping) valósítja meg, hogy az objektumokat egyszerű értékekre konvertálja, és így adatbázisban tárolhatóvá válnak. Az ORM a hagyományos adatelérési módszerekkel szemben lecsökkenti a megírandó kód mennyiségét és a kód hordozhatóbbá válik.

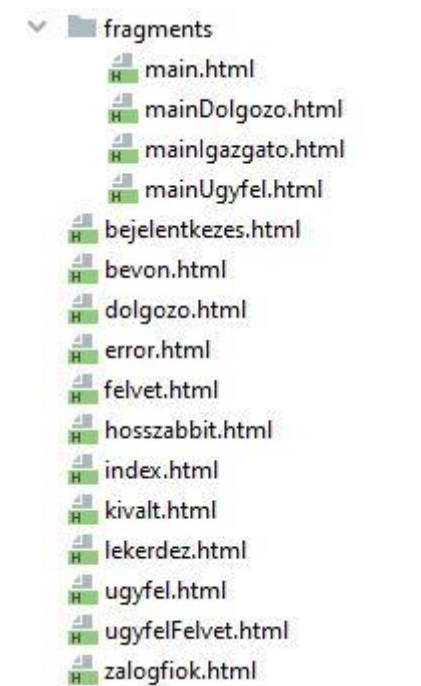
Mivel a JPA a Java egy specifikációja, ezért ORM eszközként a **Hibernate**-et használtam, ami végrehajtja a JPA specifikációit a relációs adatbázisban az adatok fennmaradása érdekében.

Ha a jövőben valami oknál fogva ORM eszközt szeretnénk váltani ezt könnyen meg tudjuk tenni a JPA-nak köszönhetően. Ugyanazon specifikáció megvalósításával minden ORM eszköz követi a közös szabványokat.

Külső adatbázisként a MySQL adatbázist választottam, ami céges felhasználásra is ingyen áll rendelkezésre. A MariaDB motor lehetőséget biztosít idegen kulcsokkal kapcsolatos szabályok definiálására. Az ilyen szabályok felállítása megkönnyíti az egymással kapcsolatban lévő táblák adataira az integritás következetes megőrzését. Pontosan ez az oka, hogy ezt az adatbázismotort választottam, mivel a feladat megköveteli, hogy az adatbázist több táblában tároljam. A több tábla és a feladat összetettsége szükségessé teszi a táblák szervezését, így létrejönnek szülő- és gyermektáblák. A szülőtábla neve abból adódik, hogy a kulcsmezőjét a gyermek táblában lehet megtalálni, mint idegen kulcsot ezzel a kulccsal tudjuk a táblázatokat összekötni.

Frontend forráskód

A frontend fájlok a „resources” mappa alatt lévő „templates” mappában találhatóak (13. ábra). A „fragments” mappába kerültek a különböző menüsorok.



13. ábra: Frontend fájlok

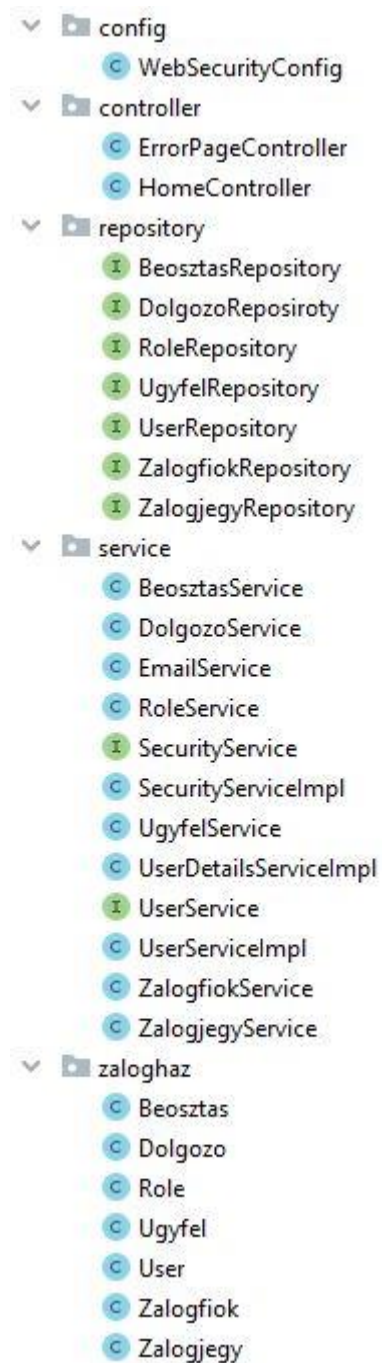
Azért választottam szét több részre a menüt, mert aki bejelentkezik, az ne láthassa azokat az oldalakat a menübe se, amihez nincs jogosultsága.

Ezekben a fájlokban az oldalon található állandó elemek vannak leírva, amik nem változnak, így nem kell minden oldalon megfogalmazni ugyanazt a részt. Ezzel elkerülhető a redundancia.

Az oldalak form tag-ek közé kerültek, amiben „submit” típusú gomb került, hogy a Thymeleaf el tudja küldeni az adatokat a Controller-nek.

Backend forráskód

A forráskódot a könnyebb átláthatóság miatt package-ekbe rendeztem, ahogy a 14. ábra mutatja a struktúrát, amit felállítottam.



14. ábra: Mappaszerkezet

Config package

A config package-be található a „WebSecurityConfig”, amiben szerepkörökhöz tartozó jogosultsági szabályok kerültek megfogalmazásra.

A következő szabályokat határoztam meg:

Az igazgató szerepkörrel rendelkező felhasználó hozzáfér a zalogfiókok, dolgozók és a zalogjegy bevonása felülethez.

A dolgozó szerepkörrel rendelkező felhasználók hozzáférnek a zálogjegy felvétele, hosszabbítása, kiváltása és lekérdezés felülethez, valamint az új ügyfél felvétele űrlaphoz, ahol jogosultságot adnak az ügyfélnek, hogy megtekinthesse a zálogjegyeit.

Itt hozom létre az első felhasználót, vagyis a zálogigazgatót, aki nem kerül be az adatbázisba csupán futási időben létezik.

Controller package

A controllerek valósítják meg a frontend (view) és a backend (service) közötti forgalom irányítást a Model interfész segítségével. Az adatokat POST metódussal juttatom a server oldalra, mert ekkor az átvitt adatok nem kerülnek be a böngésző URL-jébe. Ezáltal az oldalnak nem adhatóak meg jogosulatlan utasítások, emiatt biztonságosabb és könnyebben értelmezhető URL-t kapunk.

Service package

A service rétegben valósulnak meg a kalkulációk és a vállalati/üzleti logika, amik a controller segítségével kerülnek ki a frontend felületre.

A „ZalogjegyService”-ben létre hoztam négy konstans változót, ami segítségével számítom ki a hosszabbítás és a kiváltás költségét, valamint itt szabom meg a futamidőt is. Az esetleges futamidő-, kamat- és kezelési költség változást egyszerűen meg lehet változtatni, nem kell a kódsort végig böngészni, és lehetőséget adni, arra, hogy valahol nem sikerült megváltoztatni.

```
private final int futamido = 90;
private final double kamat = 0.2;
private final double napiKamat = kamat / futamido;
private final double kezelesiKoltseg = 0.05;
```

Mivel a futamidő lejáratát nem rögzítem az adattáblában, ezért a kiszámítását szintén itt valósítom meg:

```
public LocalDate futamidoLejarta(LocalDate beadas) {
    return beadas.plusDays(futamido);
}
```

Valamint a zálogjegy kezelését – felvétel, kiváltás, hosszabbítás, lekérdezés – ebben az osztályba valósítom meg.

Az „UgyfelServie”-ben létrehoztam az alábbi metódust, ami egy 8 véletlen karakteres jelszót generál:

```
public String jelszo() {  
    Random random = new Random();  
    char[] word = new char[8];  
    for (int i = 0; i < word.length; i++) {  
        word[i] = (char) ('a' + random.nextInt(26));  
    }  
    return new String(word);  
}
```

Továbbá a service package-be valósítom meg a bejelentkezéshez szükséges azonosításokat és az e-mail-ek küldését az ügyfelek részére.

Az email üzenet küldéséhez még szükséges az „application.properties”-ben beállítani a következő dolgokat, hogy megvalósuljon a küldés:

```
spring.mail.host= smtp.gmail.com  
spring.mail.port= 587  
spring.mail.protocol = smtp  
spring.mail.username=  
spring.mail.password=  
spring.mail.properties.mail.smtp.starttls.enable= true  
spring.mail.properties.mail.smtp.ssl.trust=smtp.gmail.com  
  
spring.mail.properties.mail.smtp.auth=true  
spring.mail.properties.mail.smtp.connectiontimeout=5000  
spring.mail.properties.mail.smtp.timeout=5000  
spring.mail.properties.mail.smtp.writetimeout=5000
```

Külön erre a projektre létrehoztam egy e-mail címet. A „spring.mail.username=”-nél a fiók e-mail címét kell megadni, a „spring.mail.password=” mezőhöz az e-mail fiókba való bejelentkezéshez tartozó jelszót kell beírni. A gmail-nél még külön figyelni kell, hogy engedélyezve legyen a „kevésbé biztonságos alkalmazásokhoz való hozzáférés” funkció.

Repository package

Itt interface-ek találhatók, amik az adatbázissal kommunikál. A CrudRepository<T,ID> valamint a JpaRepository<T,ID> kiterjesztéseit használtam.

Ezekben az interfészekben fogalmazom meg a lekérdezéseket, amiket a service rétegből hívok meg.

Zaloghaz package

Ebben az osztályban találhatók az úgynevezett POJO osztályok, ami alapján a JPA létrehozza az adattáblákat.

Biztonság – Spring Security

Az alkalmazás biztonságos használatához, hogy a felhasználók csak a jogosultságuknak megfelelő oldalakat láthassák és kezelhessék **Spring Security**-t használtam.

Első lépésként át kell gondolni, hogy milyen szerepkörrel lehessen bejelentkezni az alkalmazásba. A szerepkörök (roles) a következők:

- zálogigazgató
- dolgozó
- ügyfél

Az irányelv, hogy nem engedek semmit elérni senkinek. Majd lépésenként engedek hozzáférni dolgokhoz a felhasználóknak. Ezzel a módszerrel védem azt, hogy későbbiekben se férhessenek hozzá olyan dolgokhoz, amihez nem lenne szabad – esetleg ha valamit elfelejtünk szabályhoz kötni.

A bejelentkező felületet mindenki elérheti, ahol azonosítja magát az ügyfél az e-mail címével és a jelszavával, valamint a dolgozók a fiókhoz tartozó bejelentkezéssel. Regisztráció kizárólag a programon belül lehetséges.

A jelszavakat titkosítva tárolom az adatbázisban, aminek a kódolására a **BCrypt Password Encoder**-t használtam.

Tesztelés

Ahhoz, hogy a programot ne kelljen egy külső adatbázishoz csatolni fejlesztés és tesztelés során, **H2 Emdedded Database**-t használtam, ami egy Java nyelven írt relációs adatbázis-kezelő rendszer. A H2 Emdedded Database futási időben hozza létre az adatbázist, minden server újra indulásával törlődnek a felvitt adatok. Ezt azért használtam, mert a menet közben keletkező esetlegesen nem megfelelő adatokat nem kell manuálisan eltávolítani. Ezzel időt takarítottam meg a fejlesztés során.

Mivel az alkalmazás böngészőből érhető el, így a fejlesztés során folyamatosan manuálisan lett tesztelve különböző böngészőkből (Chrome, Firefox, Opera Mini, Safari Mobil) a funkciók helyes működése.

A végső tesztelés során a zálogigazgató fiókjával létrehoztam több zálogfiókot és dolgozókat is felvettem és töröltem. Teszteltem, hogy a zálogfiókot ne lehessen törölni, ha van hozzá kapcsolódó adat a kapcsolódó táblákban. Ilyenkor a táblákban változás nem

történik, hanem elküld egy hiba oldalra, ahonnan a böngésző vissza gombjával lehet visszatérni a felhasználói felülethez.

Ezután a fiókokkal léptem be felváltva, és hoztam létre új ügyfeleket, valamint zálogtárgyak elzálogosítását is végeztem különböző fiókoknál. Ha az ügyfél nem létező e-mail címet adott meg, akkor nincs lehetősége az alkalmazásba való belépéshez. Teszteltem a kiváltásokat, a hosszabbításokat és a lekérdezéseket. A kiváltás lehetőségét korrigáltam, hogy csak az elzálogosítást végző fiókból lehessen elvégezni. Amennyiben megpróbálják kiváltani a nem azon a helyen elzálogosított tárgyat esetleg nem megfelelő zálogjegy adatokkal keres, akkor elküld egy hiba oldalra, ahonnan a böngésző vissza gombjával lehet visszatérni a felhasználói felülethez.

Tapasztaltam egy olyan hibát, hogy a MySQL adatbázisba egy nappal előbbi dátum szerint menti el az adatokat. Ezt nem tapasztaltam a H2 adatbázis használata során, és a frontendre is jól küldi az adatot. Megoldásként a „ZalogjegyService”-be az „ujZalog” és a „hosszabbitZalogjegy” metódusba a beadáshoz hozzáadtam plusz egy napot (beadas.plusDay(1)). Adatbázis váltás esetén ezt külön meg kell nézni, és ha szükséges módosítani ezt a kódrészletet, hogy a megfelelő adat kerüljön be az adatbázisba.

Végül az ügyfél felhasználó bejelentkezését is teszteltem számítógépről és mobiltelefonról egyaránt.

A tesztelésem eredményei a várakozásnak megfelelően történtek.

ÖSSZEGZÉS

Fejlesztési lehetőségek

Az alkalmazásban rengeteg lehetőség van még a tovább fejlesztésre. A leghasznosabb biztonsági funkciónak gondolom, ha az ügyfelek az első bejelentkezés után megtudnák változtatni a jelszavukat, és egy jelszóemlékeztetővel is kibővíteném.

Egy nagyobb funkció beépítéseként a pénzkezelést tudnám a programban még elképzelni. Ebben a funkcióban nyomon lehet követni az aktuálisan a zálogfiókban lévő pénzösszeget. Még egy plusz funkciót is bele lehet tenni, hogy ha bizonyos összeg alá csökken a kézpénzállomány, akkor figyelmeztessen a program, hogy fel kell tölteni azt.

Egy másik nagyobb projekt a kényszerértékesítésre szánt tárgyakat webshopba lehetne exportálni, ezáltal nem csak a fiókokba lehet megtekinteni az ékszereket.

További apró változtatások például a beosztás tábla alapján további szerepkörök meghatározása, annak érdekében, hogy minden beosztott csak a saját feladatait tudja el látni.

FELHASZNÁLT FORRÁSOK

Baeldung. (2020. május). *Default Password Encoder in Spring Security 5*. Forrás:
<https://www.baeldung.com/spring-security-5-default-password-encoder>

Baeldung. (2020. május). *Redirect to Different Pages after Login with Spring Security*.
Forrás: https://www.baeldung.com/spring_redirect_after_login

javaTpoint. (2020. április). Forrás: <https://www.javatpoint.com/>

Ngo, G. (2020. május). *hellokoding*. Forrás: <https://hellokoding.com/registration-and-login-example-with-spring-security-spring-boot-spring-data-jpa-hsql-jsp/>

SanFranciscobólJöttem. (2020. március). *SFJ*. Forrás:
<https://sanfranciscoboljottem.com/>

Thymeleaf. (2020. május). *Tutorial: Thymeleaf + Spring*. Forrás:
<https://www.thymeleaf.org/doc/tutorials/2.1/thymeleafspring.html#dropdownlist-selectors>

Wikipedia. (2020. május). *Salt (cryptography)*. Forrás:
[https://en.wikipedia.org/wiki/Salt_\(cryptography\)?fbclid=IwAR1gHRN4Q1S4g3wiD_tIgSWjol9IWcx7Sl7X9_X67u11qr2-chtvD0duwQ](https://en.wikipedia.org/wiki/Salt_(cryptography)?fbclid=IwAR1gHRN4Q1S4g3wiD_tIgSWjol9IWcx7Sl7X9_X67u11qr2-chtvD0duwQ)