Budapesti Műszaki Szakképzési Centrum

Neumann János Számítástechnikai Szakgimnáziuma

***Szakképesítés neve:*** Szoftverfejlesztő ***OKJ száma:*** 54 213 05

**ZÁRÓDOLGOZAT**

**Zálogházi nyilvántartó program**

|  |  |
| --- | --- |
| Várkonyi Tibor | Kövérné Gargya Viktória |
| konzulens | 14.rsze |

Budapest, 2020.

Tartalomjegyzék

[Bevezetés 2](#_Toc37614807)

[Témaválasztás 2](#_Toc37614808)

[A probléma rövid ismertetése 2](#_Toc37614809)

[Felhasználói dokumentáció 2](#_Toc37614810)

[Webalkalmazás 2](#_Toc37614811)

[Rendszerigény 2](#_Toc37614812)

[Beüzemelés 2](#_Toc37614813)

[Felhasználói felület elemei, funkciói 2](#_Toc37614814)

[Telefonos alkalmazás 2](#_Toc37614815)

[Rendszerigény 2](#_Toc37614816)

[Beüzemelés 2](#_Toc37614817)

[Felhasználói felület elemei, funkciói 2](#_Toc37614818)

[Fejlesztői dokumentáció 2](#_Toc37614819)

[Rendszerterv 2](#_Toc37614820)

[Jogosultsági szabályok 2](#_Toc37614821)

[Adatbázis terv 2](#_Toc37614822)

[Adattáblák 2](#_Toc37614823)

[Megvalósítás 2](#_Toc37614824)

[Biztonság 2](#_Toc37614825)

[Tesztelés 2](#_Toc37614826)

[Összegzés 2](#_Toc37614827)

[Fejlesztési lehetőségek 2](#_Toc37614828)

[Hivatkozások 2](#_Toc37614829)

# 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 alkalmazást 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 illetve kiváltás esetén az adatok módosítása.

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 tárgyát.

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. Más zálogfiók adatait nem tudja lekérdezni. Az ügyfél csak a saját elzálogosított tárgyait látja egy mobil applikáción keresztül – mikor, melyik zálogfiókba, milyen kölcsönösszeggel lett elzálogosítva, valamint mennyi kamat van rajta az adott napon.

# Felhasználói dokumentáció

## Webalkalmazás

### Rendszerigény

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

### Beüzemelés

### Felhasználói felület elemei, funkciói

## Telefonos alkalmazás

### Rendszerigény

* Android rendszerű mobiltelefon
* Internet

### Beüzemelés

### Felhasználói felület elemei, funkciói

# Fejlesztői dokumentáció

## Rendszerterv

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

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, ellenőrző becsüs)
  + zálogigazgató
* zálogházi ügyfelek

### Jogosultsági szabályok

Ügyfél jogosultságai:

* mobil alkalmazáson való bejelentkezés
* olvasási jog a zálogjegy táblán lévő ügyfél azonosítójával rendelkező rekordokra
* saját jelszó változtatása a mobil alkalmazásra való bejelentkezéshez

Dolgozói jogosultságok munkakör alapján:

Zálogfiókban dolgozók jogosultságai:

* ügyfél tábla adatainak módosítása (az összes zálogfiók egyhez fér hozzá), új ügyfél hozzáadása
* zálogjegy tábla olvasási jog (csak az adott zálogfióké)
* megjegyzés hozzáadása a zálogjegyhez

*Pénztáros jogosultságai:*

* zálogjegy tábla olvasási és módosítási jog – hosszabbítása, kiváltása (helyi zálogfiók)
* zálogjegy tábla olvasási és módosítási jog – hosszabbítás (más zálogfiók)

*Becsüs jogosultságai:*

* zálogjegy táblába új zálogtárgy felvitele
* pénztáros jogosultságai

Ellenőrző becsüs jogosultságai:

* olvasási jog a zálogjegy táblán
* megjegyzés hozzáadása a zálogjegyhez

Zálogigazgató jogosultságai:

* ügyfél- és zálogjegy tábla olvasási jog (összes fiók)
* zálogfiók- és dolgozó tábla olvasási és írási jog
* zálogjegy bevonása kényszerértékesítésre, ha a lejárati időt túllépte

## Adatbázis terv

### Adattáblák

### 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 váljanak. 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.

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 szerver újra indulásával törlődik a felvitt adat.

## Biztonság

## Tesztelés

# Összegzés

## Fejlesztési lehetőségek

# Hivatkozások

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

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