

Debreceni Egyetem

Informatikai Kar

# **INTEGRÁLT VÁLLALAT MENEDZSELÉSI ALKALMAZÁS FEJLESZTÉSE JAVA NYELVEN**

Készítette:

Löki Tamás

Programtervező informatikus

Témavezető:

Major Sándor Roland

Egyetemi tanársegéd

Debrecen

2020.

## Tartalomjegyzék

1. Témafelvetés .....	3
1.1 Indoklás.....	3
1.2 A program funkciói.....	4
2. Technológiai háttér áttekintés, felhasznált technológiák.....	5
2.1. Technológiák .....	5
2.2. Adatbázis .....	7
3. A program és adatbázis bemutatása.....	8
3.1 A programot felépítő néhány függvény .....	8
3.2 A programhoz használt adatbázis részletes bemutatása.....	13
3.2.1 A „wagemods” adatbázis .....	13
3.2.2 A „shop” adatbázis .....	14
3.2.3 Az „employees” adatbázis.....	20
4. Alkalmazás felhasználói szintű bemutatása .....	23
4.1. Programindítása, kezdőképernyő, bejelentkezési lehetőségek .....	23
4.2. A könyvelői felület .....	24
4.2.1. Könyvelői főoldal és lehetőségei .....	24
4.2.2. Adatbázis műveletek .....	27
4.2.3. Munka/Szabadság napok kezelése .....	29
4.3. Az értékesítői felület.....	31
4.3.1. Eladói főoldal és lehetőségei.....	31
4.3.2. Adatbázis műveletek .....	33
4.4. Vállalatvezetők lehetőségei .....	36
5. Összefoglalás .....	36
6. Irodalomjegyzék .....	37
6.1. Forráskód link .....	37
7. Köszönetnyilvánítás .....	38

# 1. Témafelvetés

## 1.1 Indoklás

Rengeteget gondolkodtam azon, hogy végül miből is írjam a szakdolgozatomat. Sok ötlet megfordult a fejemben, sok olyan, amit már előttem mások is elkészítettek, így nem akartam teljesen ugyanazt csinálni csak pepitában. Gondolok itt egy sima raktárkezelő programra, vagy egy asztali alkalmazásként elkészített boltra. Szerettem volna egy olyan programot elkészíteni, amit a való életben, persze kisebb-nagyobb pofozgatások, és fejlesztések után, is fel lehet használni éles körülmények között.

Végül az ötlet egy kedves barátoméék családi kisvállalkozása kapcsán jött miszerint, egy olyan programot fogok csinálni, amit kis, 2-5 főt foglalkoztató árukereskedéssel és értékesítéssel foglalkozó cégeknek lehetne eladni. Ezek a cégek éves szinten kicsi vagy alacsony árbevétellel rendelkeznek, így fontos a számukra, hogy ott spóroljanak, ahol csak tudnak. Így az olyan programok megvétele, mint egy bérszámfejtő program, ami éves szinten 99.000Ft-tól akár 199.990Ft-os [6] költséggel is rendelkezhet, funkcióktól függően vagy egy könyvelői program, ami 119.900 + áfától akár eget rengető 399.000 + áfás árcédulával is rendelkezhet [7] sok ilyen kisvállalkozás esetében elképzelhetetlen lenne beszerezni. Ha a raktárkezelést is digitális formában akarják szamon tartani a havidíjas szoftverrel, akkor az ilyen programok árai a 9.900 – 19.900Ft-os összegek között mozog, [8] szintúgy funkcióktól függően. Ráadásul ilyen kis cégek esetében sok funkciót egyáltalán, vagy csak nagyon ritkán tudnának kihasználni, így egyáltalán nem érné meg ezek beszerzése.

Így a célom ezzel a program elkészítésével azt lett volna, hogy ezeket a programokat kiváltva, egy kisebb vállalkozás igényeinek megfelelő, fölösleges funkciókat nem tartalmazó, így a program árát minimálisan tartva egy készletkezelő, nyilvántartó, dolgozókat és azok ledolgozott munkaóráit, szabadságon, vagy táppénzen töltött napjaikat, illetve az adott hónapra, persze megfelelő adminisztráció mellett, a bérszámfejtésüket is készítse el. Ez persze csak egy segédmunkó lenne, hiszen csak az adott hónapra számolja ezeket, illetve mutatja a dolgozó ledolgozott munkaóráit és szabadságait, de természetesen szükség esetén ezek az adatok az adatbázisban továbbra és visszamenőleg is elérhetőek.

## 1.2 A program funkciói

A program fő funkcióit tekintve 4 részre lehet bontani azt. A raktárkészlet nyilvántartás és frissítés, adás-vétel, bérszámfejtés, munkaórák és szabadságok kezelése.

Felépítését tekintve 3 fő ablakból áll. A bejelentkezés, a bérszámfejtő felület, és az adás-vétel. A bejelentkezési ablakban kell megadni a felhasználónevet, és a hozzá tartozó jelszót, illetve ki kell választani, hogy melyik felületre akarunk belépni. Az egyes felhasználók belépési jogosultságai korlátozva lehetnek a munkakörüknek megfelelően. A cégtulajdonos minden felületre beléphet a felhasználójával. Ezeket a felhasználóneveket és jelszavakat az adatbázis adminisztrátora hozza létre. A programon keresztül ezek nem módosíthatóak, illetve nem is törölhetők. A bérszámfejtő ablakból, ahonnan elérhetőek a munkaórák, illetve a szabadságok adminisztrációját ellátó felületek menüje, valamint a dolgozók és a hozzájuk tartozó adatok, és adózási információk felvitelére szolgáló ablakok. A bérszámfejtő rész csak akkor tekinthető hiteles információ forrásnak, ha azt megfelelően és felelősségteljesen vezetik és kezelik. A bérszámfejtést csak az adott naptári hónapra mutatja a program, se előre se hátra nem tudunk lapozni a hónapok között. Így fontos, hogy a bérszámfejtést a bérszámfejtő csak a saját munkájának ellenőrzésére használja, és ezt tegye meg még az adott hónapban. Továbbá az adás-vételi ablak, ahol a raktárba felvitt laptopok közül lehet keresgélni, azok adatait megváltoztatni, új laptopot felvinni az adatbázisba, leadni rendelést, illetve a már leadott rendelés mennyiségét megváltoztatni, vagy teljesen törölni azt. A laptopok adataival kapcsolatban semmilyen formai követelményt nem támaszt a program, azon kívül, hogy minden adatot ki kell tölteni, így a megfelelő formázást és formai követelményeket az adott cégen belül felállított elvárásoknak megfelelően kell felvezetni az adatbázisba.

Mint láthatjuk, a program rendelkezik, elég sok, akár kellően elegendő funkcióval ahhoz, hogy akár egy kisebb bolt igényeit ki tudja elégíteni, anélkül, hogy méregdrága funkció specifikus programok megvételéhez kellene a kisebb cégeknek folyamodniuk, rengeteget spórolva ezzel.

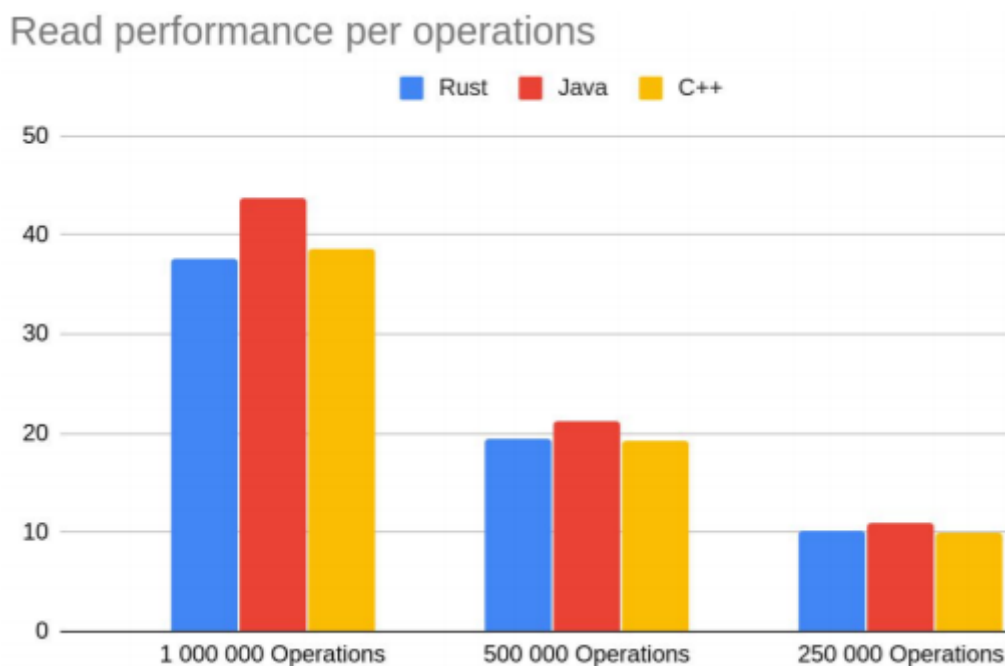
Ugyanakkor, egyes esetekben, így jelentkezhetnek hiányosságok, vagy egyéb specifikus dolgokat kívánhatnak más cégek, amik ebben a verzióban még nem elérhetőek.

## 2. Technológiai háttér áttekintés, felhasznált technológiák

### 2.1. Technológiák

Sokat gondolkodtam azon, milyen programozási nyelven készítsem el az alkalmazást, hiszen ennek többek között a sebességére, hordozhatóságára, illetve akár régebbi számítógépek esetén a teljesítményre, vagy különböző rendszereken való felhasználhatóságra is komoly hatásai lehetnek. Valamint az én tapasztalatom és tudásom is befolyásolta a döntésemet.

Teljesítmény szempontjából a C++ programozási nyelv döntően gyorsabb, mint a Java nyelv. Ez a sebességnövekedés reszponzívabb és gyorsabb felhasználást jelenthetett volna a felhasználók részéről. Ez persze egy elég komoly érv a C++ mellett hiszen, senki sem szeret a vásárlásnál csak állni és várni a technikára. Illetve az adminisztrációnál is fontos, hogy a kollégák minél kevesebb időt töltsenek



1. ábra: Java vs. C++ olvasási teljesítménye

Forrás: TRITA-EECS-EX; 2020:347

a programra várva miközben az adatokat vezeték fel az adatbázisba.

Ahogy a fenti ábrából is látszik, [9] minél több operáció elvégzésénél vizsgáljuk a két programozási nyelv sebességét, a különbség egyre szembe ötlőbb. Szerencsére az én esetemben nincsen szükség ilyen eget rengető sebességre, mivel a programom, a legrosszabb esetben is

csak maximum három adatbázishoz csatlakozik egyszerre. Valamint, korántsem fog egy másodpercben kétszázötvenezer műveletet elvégeztetni a processzorral. Így bátran kijelenthetem, szinte semmilyen érezhető sebességnövekedést nem értem volna el, ha a Java programozási nyelv helyett a C++-t választottam volna. Ugyanakkor ez az adat eléggé érdekes lehet, hogy az egyes programozási nyelvek között milyen sebesség különbségek lehetnek, és akkor még a memóriahasználatról nem is beszéltünk. Ebben a szegmensben a Java komoly hátrányból indul, hiszen ahhoz, hogy minden operációs rendszeren képes legyen bármilyen átírást nélkül akadálymentesen futni, virtualizálnia kell, ami komoly erőforrás igényt jelenthet. Ezt mi sem bizonyítja jobban, hogy az én programom is, fejlesztői környezetből futtatva, közel háromszáz megabájt RAM-ot használ fel, egy egyszerű grafikus felület megjelenítése és egy kisebb adatbázis lekérdezés elvégzése után. Ugyanakkor azt figyelembe véve, hogy a mai modern számítógépek már a legrosszabb esetben is négy gigabájtnyi RAM-mal rendelkeznek, ez a tény, úgy gondolom, nem fogja hátrányosan befolyásolni a programom futtatását semmilyen rendszeren.

Ugyanakkor a Java programozási nyelven írt programok, bármilyen operációsrendszeren is képesek futni. Így, ha a programot használó kliens a későbbiekben operációsrendszert vált, nem kell a program némely részét újraírni, hogy az új rendszeren is használhassa a cégének fontos szoftvert. Ez egy hatalmas előny, hiszen a programom egyik fontos alappillére, hogy minél olcsóbban elérhetővé tegye azokat a funkciókat, amiket más funkcióspecifikus szoftverek súlyos összegekért, hiszen azok esetében lehetséges, hogy újra kellene vásárolni az adott szoftver licencét. Ezzel szemben az általam Java-ban elkészített program, működik minden operációsrendszeren, így elég egyszer megvenni a használati jogot. Ezt követően lehet azt használni akármilyen számítógépen, akármilyen operációsrendszert futtatva.

A harmadik talán legfontosabb szempont a saját tapasztalatom a programozási nyelvekkel, így inkább a Java programozási nyelv mellett döntöttem. Hiszen ez az a nyelv, amit már évek óta tanulok, illetve ezt a nyelvet ismerem a legjobban. Valamint ez az a programozási nyelv, amiben készítettem már el több hasonló felhasználói felülettel rendelkező és adatbázishoz csatlakozó programot. Így mondhatjuk egész magabiztosan tudtam haladni a program megírása közben, illetve, ha ötletre volt szükségem a korábban általam megírt programok nagy segítségét tudtak nyújtani ebben.

A programot a Java SDK 11-es verziójával [3] készítettem el, de teljesen kompatibilis az 1.8-as vagy más néven Java 8-cal [4] is. A 8-as Java használata esetén nem szükséges külön Java FX telepítése hiszen az benne van a 8-as SDK-ban.

A grafikus felület elkészítéséhez a Java FX SDK 11.0.2-es verziót használtam. A Java FX legújabb verziója a 2020 szeptemberében kiadott Java FX 15. Illetve a gluon elemekhez [2] pedig a charm-glisten 4.4.1-es verziójú .jar-t. Ezek a gluon elemek főként a textFieldek voltak, amiknek nagy előnyük, hogy be lehet rajtuk állítani úgynevezett „floatingText” -et, ami annyit jelent, ha a felhasználó a mezőbe kattint, akkor egy előre meghatározott, addig a mezőben olvasható szöveg, a mező fölé fog csúszni, így továbbra is látható marad. Ez nagyban segíti a felhasználhatóságot, illetve egyedi külsőt is kölcsönöz ezeknek az elemeknek. Ugyan a gluon elemek a Java-hoz tartozó FX portjai az Android, illetve az IOS operációs rendszerekhez, de némely elemei használhatóak asztali alkalmazásokban is, mint például a „textField” -ek, amiket én is alkalmaztam a programom megírásához.

## 2.2. Adatbázis

Az adatbázis szerverhez Wampserver-t használtam. Ez egy ingyenes szoftver, ami támogatja a MySQL használatát [5]. A MySQL mellett, Apache web szerver, SSL támogatást és PHP programozási nyelvet is támogat. A WAMP mint szó egy négy szoftver nevének a kezdőbetűiből alkotott mozaik. Ezek a szoftverek a következők: **W**indows, **A**pache http server, **M**ySQL, **P**HP. A Windows, mint egyik legismertebb operációsrendszer a Microsoft által készített, az Apache a jelenleg legnépszerűbb nyílt forráskódú webszerver, a MySQL egy SQL adatbáziskezelő rendszer, ami többfelhasználós, illetve egyszerre többszálon fut, a PHP pedig egy a dinamikus weboldalak tervezéséhez készített programozási nyelv. A Wampserver grafikus felületét szinte bármely böngészőből el lehet érni a <http://localhost/phpmyadmin/> címen. Itt lehet belépni a root felhasználóval és üres jelszóval, miután kiválasztottuk, hogy MySQL, vagy MariaDB szerveret akarunk elindítani. Belépés után lehet létrehozni az adatbázisokat és azokban a táblákat. Valamint, már meglévő adatbázisok exportálásait is importálhatjuk a megfelelő menüfűl alatt, így egy tökéletes mását kaphatjuk az eredeti adatbázisnak.

### 3. A program és adatbázis bemutatása

#### 3.1 A programot felépítő néhány függvény

A program forráskódja elérhető az alábbi linken: <https://github.com/TomiMan7/szakdoga>

Az alábbiakban a vevő által leadott rendelések és azok törlésére használt függvényeket fogom bemutatni.

Az első függvény, ami a backend-ben erre szolgál az a „getOrders()” függvény.

```
public void getOrders()
{
    try {
        ArrayList customerData = Shop.getACustomer(customer.getText(), phone.getText(), email.getText(), city.getText(), street.getText(), hnumber.getText());
        String id = customerData.get(0).toString();
        ArrayList ordersData = Shop.getOrdersData( where: "customerId", id);

        ArrayList laptops = getAllLaptops(ordersData);

        ArrayList orderIds = getOrderId(ordersData);

        ArrayList laptopVendorName = getLaptopVendorName(laptops);

        setOrders(orderIds, laptopVendorName);
    }
    catch (Exception e){
        clearLabelsSmall();
        Wagemods.alert("Nincs ilyen korábbi vásárló!");
    }
}
```

#### 2. ábra: A „getOrders()” függvény

*Forrás: Saját készítés*

Ennek a függvénynek a szerepe a megrendelő adatainak a lekérése a GUI-ról és ezek alapján lekérje az eddigi rendeléseit. A függvény a vásárló adatainak megadása után a „Shop.getACustomer()” függvényt hívja meg, és annak a visszatérési értékét, ami egy „ArrayList”, a „customerData” -ba tölti be.

Ezután az „id” változóba teszi a „customerData” első elemét, ami a vásárló „id” -ja, így azonosíthatjuk őt a későbbiekben. Ezzel az „id” -val az „ordersData” -ba tölti az adott vásárló eddigi összes vásárlásának az adatát a „Shop.getOrdersData()” függvény segítségével. Ezután a „laptops” változóba tölti a felhasználó által rendelt eddigi laptopok adatait. A következő sorban szintén az „ordersData” változót használva az „orderIds” -ba tölti a rendelések „id” -ját a „getOrderId()” függvény segítségével. A következő sor a „getLaptopVendorName()” függvény segítségével a „laptopvendorName” változóba teszi a rendelt laptopok gyártóját és



fantázianevét. Ezt követően a „setOrders()” függvénnyel, illetve az „orderIds” és „laptopVendorName” változókkal betölti a kívánt vásárló eddigi vásárlásainak az „id” -jét, illetve a vásárlásban szereplő laptopok gyártójának és fantázianevének a megjelölését egy listába.

Ha bármelyik művelet, de legfőképpen a GUI-ról való adatbeolvasás meghiúsulna akkor a try-catch, catch ágában lefut a „clearLabelsSmall()” függvény, ami minden a GUI-n szereplő adatot töröl kivéve a vásárló adatait, illetve egy felugró ablak jelenik meg a megfelelő szöveggel.

A függvény által használt függvények áttekintésében először a „getACustomer()” a „Shop” osztályból. Az alábbi képen látható maga a függvény:

```
public static ArrayList getACustomer(String name, String phone, String email, String city, String street, String hnumber)
{
    ArrayList result = new ArrayList();
    try
    {
        PreparedStatement st = conn.prepareStatement( sql: "select * from customer where name = ?" +
            "and phone = ?" +
            "and email = ?" +
            "and city = ?" +
            "and street = ?" +
            "and hnumber = ?");

        st.setString( parameterIndex: 1,name);
        st.setString( parameterIndex: 2,phone);
        st.setString( parameterIndex: 3,email);
        st.setString( parameterIndex: 4,city);
        st.setString( parameterIndex: 5,street);
        st.setString( parameterIndex: 6,hnumber);

        rs = st.executeQuery();

        while(rs.next())
        {
            result.add(rs.getInt( columnIndex: 1));
            result.add(rs.getString( columnIndex: 2));
            result.add(rs.getString( columnIndex: 3));
            result.add(rs.getString( columnIndex: 4));
            result.add(rs.getString( columnIndex: 5));
            result.add(rs.getString( columnIndex: 6));
            result.add(rs.getString( columnIndex: 7));
        }
    }
}
```

### 3. ábra: A „getACustomer()” függvény

*Forrás: Saját készítés*

Ennek a függvénynek a szerepe, hogy az adatbázisból lekérdezze a kívánt megrendelő adatait. Ahogy az a képen is látszik, a függvény visszatérési értéke egy „ArrayList” lesz. Paraméterként megkapja a vásárló nevét, telefonszámát, e-mail címét, lakhelyének városát, az utcát és végül a házszámot. Ebből egyértelműen meghatározható melyik vásárlóról van szó, így megkaphatjuk

az „id” -jét, amit később használunk. A függvény a „shop” adatbázishoz csatlakozik és az „st” változóba kerülő SQL lekérdezést hajtja végre, a megfelelő paraméterekkel, amit később állítok be. Végül a „result” változóba kerülnek a lekérdezés eredményei, és ezekkel tér vissza a függvény.

A következő felhasznált függvény a „getOrdersData()”, ami a következő képen látható:

```
public static ArrayList getOrdersData(String where, String equals)
{
    ArrayList result = new ArrayList();
    try
    {
        PreparedStatement st = conn.prepareStatement( sql: "select * from orders where "+ where + " = ?");
        st.setString( parameterIndex: 1,equals);
        rs = st.executeQuery();
        while(rs.next())
        {
            result.add(rs.getInt( columnIndex: 1));
            result.add(rs.getInt( columnIndex: 2));
            result.add(rs.getInt( columnIndex: 3));
            result.add(rs.getString( columnIndex: 4));
            result.add(rs.getString( columnIndex: 5));
            result.add(rs.getString( columnIndex: 6));
            result.add(rs.getString( columnIndex: 7));
        }
    }
}
```

#### 4. ábra: A „getOrdersData()” függvény

*Forrás: Saját készítés*

Ennek a függvénynek a szerepe, hogy az adott rendelés részleteit lekérje az adatbázisból. Hasonlóan az előző függvényhez, ez is a „shop” adatbázishoz csatlakozik és az előzőleg lekérdezett vásárlói „id” segítségével lekéri a vásárló eddigi vásárlásainak összes adatait. Majd a helyileg létrehozott „result” változóba tölti ezeket, és visszatér ennek az értékével.

A következő használt függvény a „getOrders()” függvényben a „getOrderId()”. Ez a függvény az imént lekért rendelések adataiból kiválogatja a rendelések azonosítóit, azaz azok „id” -jét. Ezt a függvényt láthatjuk a következő képen:

```

public ArrayList getId(ArrayList ordersData)
{
    ArrayList orderIds = new ArrayList();

    for(int i = 0; i <= ordersData.size() - 1; i = i + 7)
    {
        orderIds.add(ordersData.get(i));
        if(i + 7 > ordersData.size() )
            return orderIds;
    }
    return orderIds;
}

```

### 5. ábra: a „getId()” függvény

*Forrás: Saját készítés.*

Ennek a függvénynek a szerepe a rendelések „id” -jának kiválogatása. A függvény visszatérési értéke egy „ArrayList” lesz. Ebben az „orderIds” változóban lesznek az egyes rendelések azonosítói. Egy egyszerű „for” ciklus van benne, ami a paraméternek megadott „ordersData” változón megy végig a 0. elemtől minden hetedik elemen, hiszen azok az adott rendelés azonosítói. Ezt később az adatbázis bemutatásnál láthatjuk. Az „if” elágazásban megvizsgálja, hogy a következő iterációban az „i” változó értéke nagyobb lesz-e, mint a paraméterként kapott „ordersData” változó mérete, és ha igen, akkor visszatér az „orderIds” változóval. A kiválogatott azonosítókat a helyileg létrehozott „orderIds” változóba rakja bele, és tér vissza ennek a tartalmával.

A következő használt függvény a „getLaptopVendorName()”. Ennek a függvénynek a szerepe az adott laptopok gyártójának és fantázianevének kiválogatása. Ez a függvény a paraméterként kapott változón végig menve, kiszedi a laptopok adataiból a gyártó nevét és a fantázianevet. Ezt a függvényt láthatjuk a következő képen:

```

public ArrayList getLaptopVendorName(ArrayList laptops)
{
    ArrayList laptopVendorName = new ArrayList();
    for(int j = 1; j <= laptops.size() - 1; j = j + 7)
    {
        laptopVendorName.add(laptops.get(j));
        laptopVendorName.add(laptops.get(j+1));
        if(j + 7 > laptops.size() )
            return laptopVendorName;
    }
    return laptopVendorName;
}

```

## 6. ábra: a „getLaptopVendorName()” függvény

*Forrás: Saját készítés.*

Ahogy láthatjuk egy „ArrayList” -et kap paraméterként, a „laptops” változót. Ebben a változóban vannak a vevő által rendelt laptopok „laptops” táblából lekérdezett adatai. Ezekből az adatokból jelenleg csak a laptopok gyártójára és fantázia nevére vagyunk kíváncsiak. Erre szolgál a „for” ciklus, ami végigmegy a paraméterként kapott változón, és kiválogatja belőle a szükséges adatokat. Az „if” elágazás itt is megtalálható, ha esetlegesen a „j” változó értéke a következő iterációban túllépné a paraméter méretét. Ha igen akkor visszatér a függvény.

A következő használt függvény a sorban a „setOrders()”. Ez az a függvény, ami a GUI-ra teszi ki a vásárló által rendelt laptopok nevét, és a rendelések „id” -jét. A következő képen láthatjuk magát a függvényt:

```

public void setOrders(ArrayList orderIds, ArrayList laptopVendorName)
{
    clearLabels();
    int j = 0;
    for(int i = 0; i <= orderIds.size() - 1; i++)
    {
        list.getItems().addAll( ...es: orderIds.get(i) + " " + laptopVendorName.get(j) + " " + laptopVendorName.get( j + 1))
        j = j+2;
    }
}

```

## 7. ábra: a „setOrders()” függvény

*Forrás: Saját készítés*

A függvény két paramétert kap, az „orderIds” -t és a „laptopVendorName” -et. Ezek tartalmazzák a számunkra szükséges információkat, amiket meg akarunk jelteni a felhasználói felületen. Az első sorban található „clearLabels()” függvény, az összes eddigi esetlegesen korábbi keresésből ott maradt értéket töröl. Utána egy egyszerű „for” ciklus segítségével végig megyünk a megfelelő adatokon és a „list” nevű listára visszük fel az adatokat, amit a programot kezelő munkatárs is látni fog.

## 3.2 A programhoz használt adatbázis részletes bemutatása

### 3.2.1 A „wagemods” adatbázis

Ebben az adatbázisban csak egyetlen egy tábla található. Ennek a neve az adatbázis nevével egyező „wagemods”. Ez az adattábla tartalmazza az egyes munkavállalókhoz tartozó a bérszámfejtéshez szükséges adózási adatokat.

wagemods	
id	int
* name	varchar
nyugdij	varchar
tb	varchar
szja	varchar
mpj	varchar
nyugdijtakarek	varchar

#### 8. ábra: A „wagemods” adattábla felépítése

*Forrás: Saját készítés*

Amint az a fenti ábrából is látszik, ez egy egyszerű adattábla. Nem sok adatot tartalmaz. Összesen 7 oszlopból áll. Az első oszlop az „id”. Ez ugyan elsődleges kulcs, de nem elsődlegesen ezzel vannak azonosítva a dolgozók, ugyanis a programom célközönségének, azaz a cégek méretéből kiindulva -2-től kb. 15 fő-ig- egyszerűbbnek láttam, ha a nevek alapján azonosítom a dolgozókat. Így a táblát kezelők sokkal emberközelibbnek érezhetik a dolgot, mintha csak egyszerű számokkal lennének a munkatársaik azonosítva.

A második oszlop a „name” azaz a név. Ez egy egyedi adatokat tartalmazó varchar típusú, azaz karaktereket és számokat tartalmazó oszlop. Mivel ebben az oszlopban csak egyedi adatok tárolhatóak, így ez az az oszlop, amivel azonosítom az adott sort, hiszen itt főleg az adott munkatársra történik a keresés.

A harmadik oszloptól kezdődően találhatóak az adózáshoz tárolandó adatok. A harmadik oszlop a „nyugdíj” oszlop. Ebben található az adott munkatársra vonatkozó nyugdíj adózási százalék. A tárolás típusa varchar. Igaz csak számokat tárolok benne, de a backend egyszerűsítése érdekében varchar-ként van az adatbázisban tárolva, hiszen így a lekérdezés és a GUI-ra való felvitel között nem kell típuskonverzió, szintúgy a GUI-ból való adatlekérés és az adatbázisba való felvitel között is kihagyható, hisz minden String, illetve varchar típusú.

A negyedik oszlop a „tb” nevű oszlop, ebben található az adott munkatársra vonatkozó társadalombiztosítási adózási százalék. A tárolás típusa varchar.

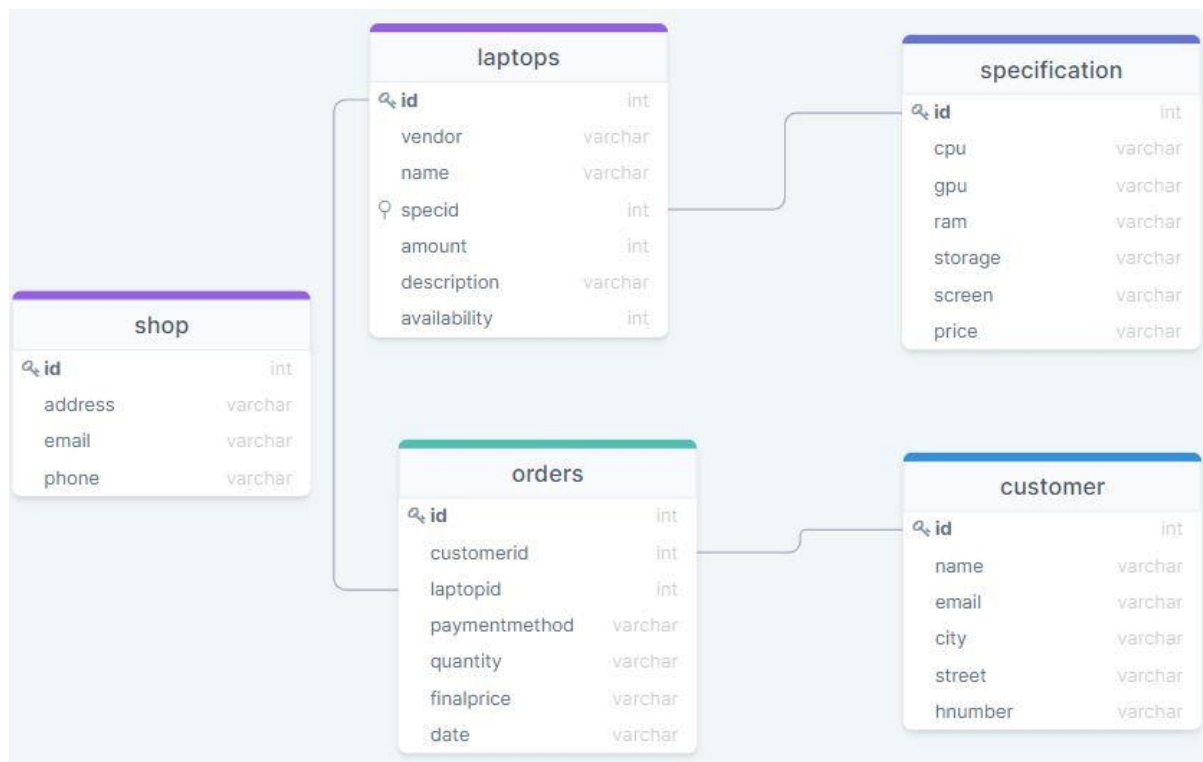
Az ötödik oszlop az „szja” nevű oszlop, ebben található az adott munkatársra vonatkozó személyi jövedelemadó adózási százalék. A tárolás típusa varchar.

A hatodik oszlop az „mpj” nevű oszlop, ebben az oszlopban található a munkaerőpiaci járulék. A tárolás típusa varchar.

A hetedik és egyben utolsó oszlop a „nyugdíjtakarek”. Ebben található az adott munkatárs által a nyugdíjtakarékjába utalt összeg nagysága. A tárolás típusa varchar.

### **3.2.2 A „shop” adatbázis**

Ebben az adatbázisban összesen 5 adattábla található. Három különböző részre lehet bontani. Egy a bolt adatait tároló adattábla, a vevők és a rendeléseiket tároló adattábla, illetve az árult laptopok és azok specifikációit tároló adattábla. Ez látható a következő ábrán.



9. ábra: A „shop” adatbázis és táblái

*Forrás: Saját készítés*

Ahogy a fenti ábrából is látszik a „shop” adattábla csak négy oszlopból áll. Az első oszlopban található egy azonosításra szolgáló „id”. Igaz ez nem kerül hasznosításra hiszen ahogy korábban is ismertettem a program célközönsége valószínűleg csak egy üzlethelységgel fog rendelkezni. A következő az „address” névvel rendelkező oszlopban található az üzlethelység teljes címe, utána az e-mail-je és telefonszáma. Ez a tábla csak adminisztráció szempontjából lett létrehozva, magában a programban nincs hasznosítva a tartalma.

A következő tábla a „laptops” nevet kapta. Ebbe az adattáblába kerülnek a cég által forgalmazott laptopok. Összesen hét adatoszlop található benne. Az első oszlop az „id”. Ez szolgál a laptopok azonosítására. Az „id” oszlopon kívül, a „vendor” és „name” oszlopok együttesével is egyértelműen azonosítani lehet egy-egy laptopot. Ugyan ezek nem egyedi kulcsok, de a laptopok névadásából kifolyólag, egy adott gyártó („vendor” oszlop) adott laptopjának a neve mindig egyedi („name” oszlop). Így nem volt szükség arra, hogy ezt az adatoszlopot egyedire állítsam. Ugyanakkor az értéke magától növekszik, minden új beszúrt sorral. Az „id” oszlop „int” típusként van tárolva.

Az adattábla következő oszlopa a „vendor”. Ebben az oszlopban van az adott laptop gyártójának a neve, mint például Asus, vagy Acer. A gyártók neve, az ő védjegyük, így egyedi. Ez azt jelenti, hogy erre az oszlopra se kellett beállítanom, hogy csak egyedi adatok kerüljenek bele. Ez nem csak felesleges, de hiba is lenne, hiszen ekkor egyetlen egy, laptopot vihetnénk fel egy adott gyártótól az adattáblába. Az oszlop szöveggént van tárolva.

A következő a „name” oszlop. Ebben az oszlopban van tárolva az adott gyártó egy adott laptopjának a neve. Ez minden laptop esetében egyedi, hiszen minden konfiguráció egyedi névvel rendelkezik, még akkor is, ha ez csak a fantázianév után található számsorral is van jelezve. Így ez az oszlop gyakorlatilag már önmagában is egyedileg azonosítja az adott sort. A könnyedebb keresés érdekében az adott laptop gyártójának a nevével való összefűzésével könnyedebben ki tudjuk keresni az adatbázisból a kívánt laptopot, anélkül, hogy az egyedi „id” azonosítójukat kellene használnunk. Ez nagyban megkönnyíti a program használatát hiszen, a fantázianevek könnyedebben megjegyezhetőek, mint az egyes laptopokhoz rendelt azonosítók az „id” oszlopból. A „name” adatoszlop szöveggént van tárolva.

Az adattábla következő oszlopa a „specid”. Ez egy egyedi azonosító, ami az adott laptophoz tartozó specifikáció táblában azonosítja a hozzá tartozó sort, azaz egy külső kulcs. A „specid” oszlopot int-ként tárolja az adatbázis.

Az adattábla következő oszlopa az „amount”. Ez az oszlop az adott laptopból elérhető mennyiséget tárolja az adatbázisban. Az, hogy ha ebben az oszlopban 0-ás szerepel nem jelenti azt, hogy az adott laptopot nem lehet kapni, csupán annyit, hogy nincs raktáron, de rendelhető. Az „amount” adatoszlop int-ként van tárolva az adatbázisban.

A következő oszlop a „description”. Ebben az oszlopban van az adott laptop egy rövid leírása, ami használható a reklámozására. Összesen 255 karaktert lehet ide írni. Ez az oszlop szöveggént van tárolva az adatbázisban.

Az adattábla utolsó oszlopa az „availability”. Ebben az oszlopban lehet jelezni, hogy az adott laptop elérhető-e, vagy sem. Ez azt jelenti, hogy még gyártják, vagy már kifutó darab. Ha az „amount” és „availability” oszlop tartalma is 0, akkor az a laptop már egyáltalán nem kapható. Ha az „amount” oszlop tartalma 0 és az „availability” oszlop tartalma 1, ugyan raktáron nincs már az adott laptop, de rendelhető. Az „availability” oszlop int-ként van tárolva az adatbázisban.



A „shop” adatbázis következő adattáblája a „laptop” táblához köthető „specification”. Ebben az adattáblában található a laptopok specifikációjának a részletezése.

Az adattábla első oszlopa az „id”. Ez a tábla kulcsa, egyedi. Ezzel lehet azonosítani az adott sort a táblában.

A következő oszlopa a táblának a „cpu”. Ebben az oszlopban van tárolva az adott laptopban található processzor típusa. Ide az adott cpu-nak a teljes nevét kell beírni, így az adott fantázianevet, típus megjelölést és a processzor sebességét is. Az oszlop tartalma maximum 255 karakter hosszú lehet. Az oszlop tartalma szöveggént van tárolva.

Az adattábla következő oszlopa a „gpu”. Ide vannak feljegyezve a laptopban található grafikus processzor adatai. A gyártója a neve, a sebessége és ha fontos a rajta található grafikus memória mérete és sebessége. Az oszlop maximum 255 karakter tud egyszerre tárolni. Szöveggént van tárolva.

A következő oszlop a „ram”. Itt található a laptopba szerelt memória mérete és sebessége. Az oszlop 255 karakter hosszú sztringet tud tárolni. Igény esetén a memória gyártója is tárolható, bár ezeket nem szokták feltüntetni. Az oszlop karaktersorozatként van tárolva.

A következő oszlopba kerül a „storage” név alá a laptopban található háttértár mérete és típusa. Több háttértár esetén azokat is ebbe az oszlopba kell felsorolni. Érdekes a későbbi keresés megkönnyítése érdekében egy megállapodást kötni, és egy adott formátumban felvinni az adatokat, azaz elől legyen a háttértár típusa és utána a mérete, vagy éppen fordítva. Az oszlop karaktersorozatként van tárolva és maximum 255 karakter tárolására képes.

A következő oszlop a „screen”. Ide kerül a laptopba szerelt kijelző adatai, mint például a mérete inch-ben, a kijelző felbontása és típusa. 255 karakter hosszú szöveggént van tárolva.

Az adattábla utolsó oszlopa a „price”. Ebben az oszlopba kerül a laptopnak az egységára. Nem kerül feltüntetésre, hogy az ár miben értetendő, így csak egy szám kerül bele. Ettől függetlenül karaktersorozatként van tárolva, és összesen 255 karakterből állhat.

Az adatbázis következő táblája a „customer”. Itt vannak tárolva a vevők adatai. Mint például a nevük, a telefonszámuk és címük.

A tábla első oszlopa az „id”. Ez az egyedi szám pontosan azonosítja az adott vevőt, így, ha két vevő azonos névvel rendelkezik, is biztosan azonosítani lehet őket. Továbbá, ha nincs név,

telefonszám és e-mail segítségével is pontosan be lehet azonosítani a vevőt, de ezt a módszert nem használtam a programban, inkább csak az összes kívánt adat megadása esetén lehet vevőre keresni, ezzel biztosítva, hogy csak az módosíthatja a vásárlást, aki pontos adatokkal rendelkezik a vevőről. Ez sok esetben maga a vételt leadó személy, vagy egy közeli hozzátartozó, vagy egy a vevő által megbízott személy.

A tábla következő oszlopába a „name” oszlopba kerül a vásárló neve. Nincs külön tárolva vezetéknév, avagy keresztnév se születéskori név. Az oszlop összesen 255 karakter befogadására képes, és karaktersorozatként van tárolva.

A következő az „email” oszlop. Ide kerül a vevő által kontaktfelvételhez megadott e-mailcím. Összesen 255 karakter hosszú lehet, és karaktersorozatként van tárolva az adatbázisban.

A következő a „city” oszlop. Ebben kell tárolni a vevő lakhelyéből a várost. 255 karakter hosszú lehet összesen, és karaktersorozatként van tárolva az adatbázisban.

Mellette található a „street” oszlop. Ide kerül az utca neve. 255 karakter hosszú lehet, és karaktersorozatként van tárolva az adatbázisban.

Az utolsó oszlop a táblában a „hnumber”. A neve a „house number” rövidítése, de ide nem csak a házszám, hanem igény esetén az emelet és ajtószám is kerülhet hiszen 255 karakter hosszú lehet, így bőven elfér benne ennyi adat. Karaktersorozatként van tárolva az adatbázisban.

Az adatbázis utolsó táblája az „orders”. Ide kerülnek be a rendelés adatai. Ki, mit, mennyit, mennyiért és mikor vásárolt, illetve készpénzzel vagy kártyával tette azt. Összesen 7 oszlop található ezeknek az információknak a tárolására.

Ezek közül az első oszlop az „id”. Ennek az értéke egyedi. Ez főleg az eddig összesen leadott rendelések számának a megadására lehet hasznos, de erre a funkcióra nem használtam. Egy adott rendelés pontos meghatározására, inkább a vevőt használtam, és az adott rendelésben leadott laptopot. Így pontosan ki lehet keresni az adott vevő által keresett rendelést, és lehet módosítani, vagy éppen törölni azt. Összesen 255 karakter hosszú lehet, és egész számként van tárolva.

A tábla következő oszlopa a „customerid”. Ez egy külső kulcs, így egyedi, a „customer” tábla „id” oszlopára mutat, így egyértelműen azonosítja a vevőt. Fontos szerepet játszik az adott vevő által leadott rendelések keresésekor. Összesen 255 karakter hosszú lehet és egész számként van tárolva.

A következő oszlop szintén egy külső kulcs lesz, a „laptopid”. Ez a külső kulcs mutat a „laptop” tábla „id” oszlopára, ezzel pontosan azonosítva a vevő által választott laptopot. Összesen 255 karakter hosszú lehet, és egész számként van tárolva.

A következő oszlop a „paymentmethod” tárolja a vásárláskor megadott fizetési módszert. Ha a vásárló készpénzzel fizetett, akkor az oszlopba egy 1-es kerül. Ha nem, akkor pedig nullás. Lehetett volna boolean-ként tárolni, de a backend egyszerűsítése érdekében szöveggént van tárolva.

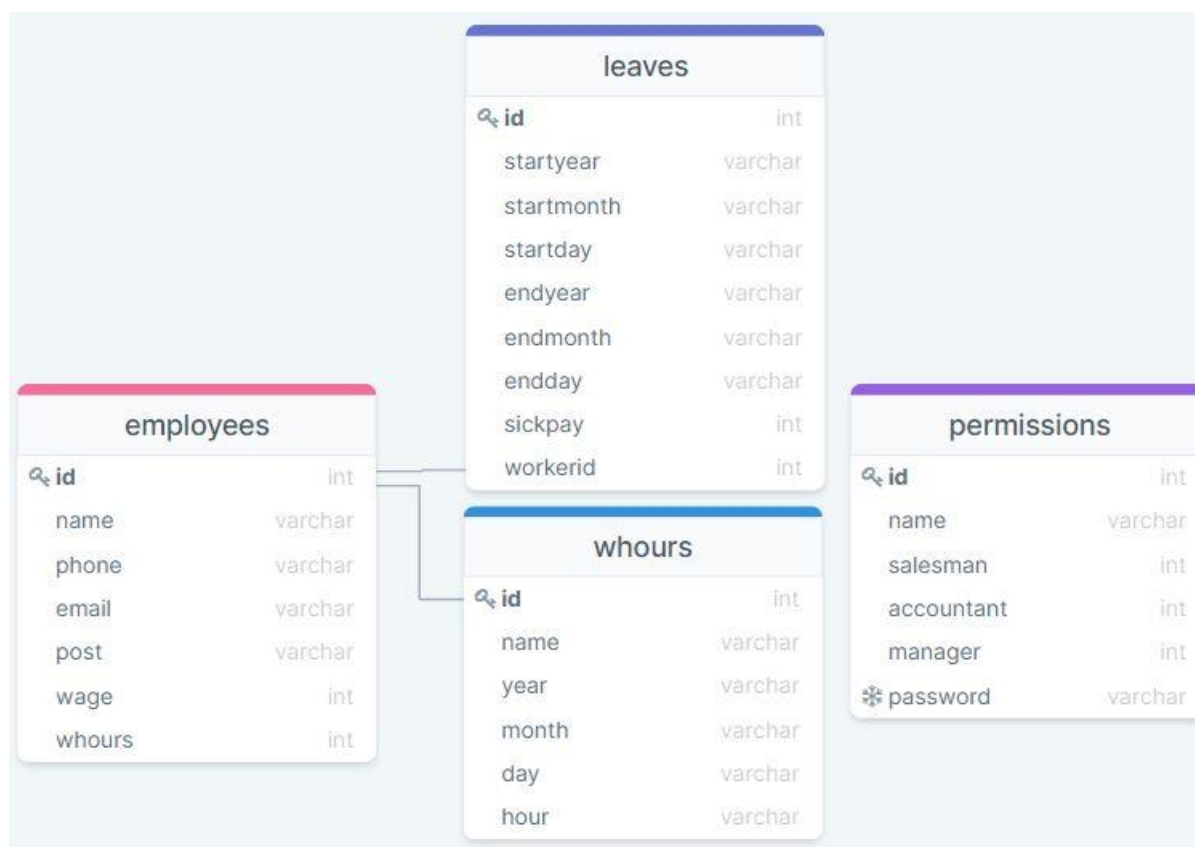
A következő oszlop a „quantity”. Ebbe az oszlopba kerül az adott laptopból vásárolt mennyiség. Szöveggént van tárolva és összesen 255 karakter hosszú lehet.

Az utolsó előtti oszlopa az adattáblának a „finalprice”. Ebbe az oszlopba kerül az laptop egységárának, és a vásárolt mennyiségnek a szorzata. Szöveggént van tárolva és összesen 255 karakter lehet. A pénznem nincs tárolva.

Az utolsó oszlopba a táblának a „date”. Ez a vásárlás dátumát tárolva. Szöveggént van tárolva, mert a backendből kerül ide az oszlop értéke, így elkerülendő az esetleges formázási hiba, valamint korábbi verziókban a rendelés törlésekor ide került egy „CANCELED” szöveg. Ezzel jelezve, hogy az adott rendelést törölték. Az oszlopba maximum 255 karaktert lehet írni, és szöveggént van tárolva.

### 3.2.3 Az „employees” adatbázis

Ebben az adatbázisban vannak tárolva a munkavállalók adatai, a ledolgozott munkaórák a szabadságon töltött napok számai, illetve a program használatához elengedhetetlen felhasználónevek és az azokhoz tartozó jelszavak. Az adatbázis összesen négy adattáblát tartalmaz. Ezek a következők: „employees”, „leaves”, „whours” és a „permissions”. Ezekről bővebb információt a következő képen találhatunk.



10. ábra: Az „employees” adatbázis és táblái

*Forrás: Saját készítés*

Az adatbázisból először a „permission” táblát mutatom be. Ebben a táblában találhatóak a belépéshez szükséges felhasználónevek, az azokhoz tartozó jelszavak, és az adott felhasználók jogosultságai.

Az adattábla első sora az „id”. Ez az adatsorok azonosítására szolgál, bár a programomban ennek nem veszem hasznát, hiszen a felhasználónév és jelszó párost használom egy adott sor kikeresésére.

A következő sor a „name”. Ebben a sorban találhatóak a felhasználónevek. Ezeket kell beírniuk a programba bejelentkezni kívánóknak.

A következő oszlop az egyik jogosultságot tároló oszlop a „salesman”. Ebben található, az adott felhasználóra vonatkozóan, hogy beléphet-e a programba, mint értékesítő, vagy sem. Ha az oszlop tartalma 0, akkor nem, ha 1, akkor igen. Az oszlopot, mint egész szám tárolja az adatbázis.

A következő oszlop egy másik jogosultságot tároló oszlop az „accountant”. Ebben található, az adott felhasználóra vonatkozóan, hogy beléphet-e a programba, mint könyvelő, vagy sem. Ha az oszlop tartalma 0, akkor nem, ha 1, akkor igen. Az oszlopot, mint egész szám tárolja az adatbázis.

A következő oszlop az utolsó jogosultságot tároló oszlop a „manager”. Ebben található, az adott felhasználóra vonatkozóan, hogy beléphet-e a programba, mint könyvelő, eladó, vagy sem. Ha az oszlop tartalma 0, akkor nem, ha 1, akkor igen. Ez a főnök, cégtulajdonosoknak fenttartott felhasználó, hiszen ez az, amivel bármelyik programrészbe be tud lépni az adott személy, és meg tudja nézni az alkalmazottai munkáját, vagy esetleg saját maga tudja elvégezni azt. Az oszlopot, mint egész szám tárolja az adatbázis.

Az adattábla utolsó oszlopa a „password”. Ebben az oszlopban vannak tárolva az adott felhasználóhoz tartozó jelszavak. Ezeknek a jelszavaknak egyedinek kell lenniük. Összesen 255 karakter hosszúak lehetnek. Szöveggént vannak tárolva.

Az adatbázis következő táblája az „employees”. Ebben az oszlopban találhatóak a munkavállalók adatai, illetve óráére és hogy hány órára vannak beosztva szerződés szerint.

A tábla első oszlopa az „id”. Ez szokás szerint egy sor azonosítására szolgál. 255 karakter hosszú egyedi számnak kell lennie. Egész számként van tárolva.

A következő oszlopa a „name”. Ebben az oszlopban van tárolva a munkatárs neve. A cél cégek méretéből kiindulva kevés az esély, hogy két teljesen megegyező nevű ember legyen,

így, ha ez mégis megtörténik esetleges számok hozzáfűzése a névhez megoldható, ez nagyobb vállalatoknál bevett szokás. Összesen 255 karakter hosszú lehet és szöveggént van tárolva.

A tábla következő oszlopa a „phone”. Itt tároljuk a munkatárs személyes, vagy céges telefonját, amin bármikor elérhetjük, ha erre szükség lenne. A teljes formátumot kell felvinni, azaz a +36/YY/XXXXXXX alakban. Összesen 255 karakter hosszú lehet és szöveggént van tárolva.

Az „email” a következő oszlop a táblázatban. Ebben tároljuk a munkatárs e-mail címét. Összesen 255 karakter hosszú lehet és szöveggént van tárolva.

A következő „post” nevű oszlopban tároljuk, a munkatárs beosztását. Eladó, könyvelő stb. Összesen 255 karakter hosszú lehet és szöveggént van tárolva.

Az utolsó előtti oszlop a „wage”. Ebben az oszlopban kerül tárolásra a munkavállaló bruttó órájára. Összesen 255 karakter hosszú lehet és egész számként van tárolva.

Az utolsó oszlopban van a szerződés szerinti munkaórák tárolására szolgáló hely. Ennek az oszlopnak a neve „whours”. Ebben, vagy 4,6,8,12 órát szoktak írni. Összesen 255 karakter hosszú lehet és egész számként van tárolva.

A következő adattábla a „whours” Ebben van tárolva az adott munkatárs által ledolgozott órák száma, és hogy melyik nap dolgozta le őket.

A tábla első oszlopa az „id”. Ez szokás szerint egy sor azonosítására szolgál. 255 karakter hosszú egyedi számnak kell lennie. Egész számként van tárolva.

A következő oszlop a „name”, ez a munkatárs neve, akihez egy adott napon a ledolgozott órák számát szeretnénk fűzni. Maximum 255 karakter hosszú lehet és szöveggént van tárolva.

A következő három oszlop a „year”, „month” és a „day”. Ezek szolgálnak a ledolgozott óra dátumának a bejegyzésére. Mind a három oszlop maximum 255 karakter hosszú lehet és szöveggént van tárolva.

Az utolsó oszlopba kerül az adott napon ledolgozott órák száma. Összesen 255 karakter hosszú lehet, és szöveggént van tárolva.

Az utolsó adattábla a „leaves”. Ebben vannak tárolva a szabadságolások.

Az első oszlopa az „id”. Ez szokás szerint egy sor azonosítására szolgál. 255 karakter hosszú egyedi számnak kell lennie. Egész számként van tárolva.

A következő három oszlop a „startyear”, a „startmonth” és a „startday” ezek szolgálnak a szabadságolás kezdetének a feljegyzésére. Mind a három oszlop maximum 255 karakter hosszú lehet és szöveggént van tárolva.

A következő három oszlop a „endyear”, a „endmonth” és a „endday” ezek szolgálnak a szabadságolás végének a feljegyzésére. Mind a három oszlop maximum 255 karakter hosszú lehet és szöveggént van tárolva.

Az utolsó előtti oszlop a „sickpay”. Itt kell jelezni, ha a szabadság az táppénz-e. Ha az értéke 0, akkor nem, ha 1 akkor táppénz. Egész számként van tárolva.

Az utolsó oszlop a „workerid”. Ez egy külső kulcs, ami mutat az „employees” tábla „id” oszlopára ezzel egyértelműen meghatározva a munkatársat.

## **4. Alkalmazás felhasználói szintű bemutatása**

### **4.1. Programindítása, kezdőképernyő, bejelentkezési lehetőségek**

Az alkalmazás az általánosan megszokott formában indítható el, az asztalon elhelyezett ikon segítségével, dupla kattintással. Indítás után az első ábrán látható felület fogadja a felhasználókat.

**11. ábra: Kezdőképernyő**

*Forrás: Saját készítés*

**12. ábra: Bejelentkezési lehetőségek**

*Forrás: Saját készítés*

Mint azt az ábrán is láthatjuk, a program betöltése után a kurzor, már egyből, az első felhasználó által kitöltendő mezőben, azaz, a felhasználónév mellett villog. Amennyiben a kurzor nem itt lenne látható, akkor a mező belsejében jelenne meg a felhasználónév segédfelirat. A lebegő szöveg (11. ábra) és a mező belsejében megjelenő segéd felirat (12. ábra) is a felhasználók könnyebb tájékozódását segítik a felületen. A jelszó mezőben értelemszerűen minden beírt karakter pontokkal helyettesítve jelenik meg.

A második ábrán az alkalmazásba történő bejelentkezési lehetőségek láthatóak. Az alkalmazotknak így lesz módja kiválasztani, hogy a könyvelői, vagy az értékesítói felületet szeretnék elérni majd. Természetesen, a saját fiókjukhoz hozzárendelt jogokkal rendelkeznek, így egy értékesítő munkatárs, nem tud majd könyvelői felületre a saját adataival bejelentkezni és fordítva.

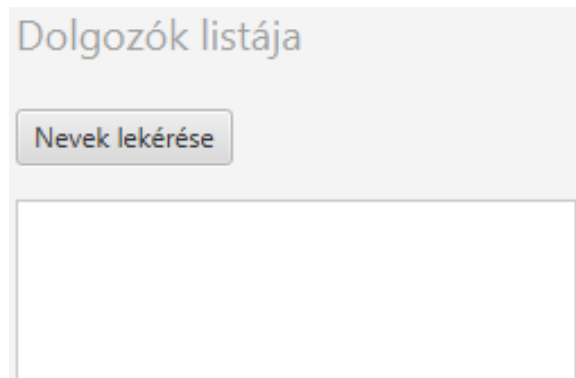
## **4.2. A könyvelői felület**

### **4.2.1. Könyvelői főoldal és lehetőségei**

A pénzügyekkel foglalkozó munkatársak, a könyvelői felületre történő bejelentkezés után a könyvelői főoldallal találkoznak majd először. A fő oldal felépítése a könnyebb áttekintés érdekében három fő oszlopból áll.

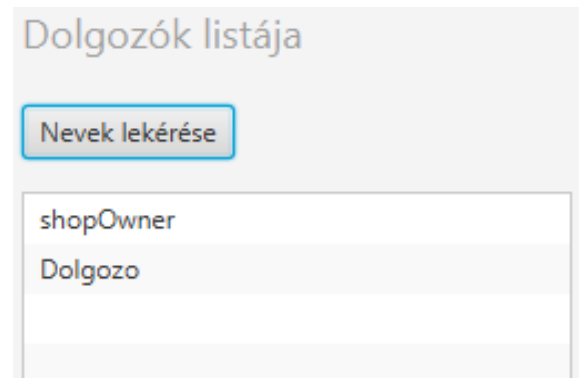


Az első oszlop a dolgozók névsorát tartalmazza. A listát, az adott munkatárs, a 'nevek lekérése' gombra kattintva tudja betölteni az adatbázisból (14. ábra). Miután ez a lista betöltött, a gomb alatti mezőben sorakoznak majd fel a nevek, a sorokat pedig a halványszürke és a fehér színek váltakozásával (bizonyos képernyőbeállítások mellett ennek az effektnek a láthatósága romolhat, akár el is tűnhet) lehet könnyen nyomon követni. Ezt követően kell annak a kollégának a nevét kiválasztani, akivel a továbbiakban foglalkozni szeretne.



**13. ábra: Adatok betöltése előtt**

*Forrás: Saját készítés*



**14. ábra: Adatok betöltése után**

*Forrás: Saját készítés*

A második oszlop (15. ábra) a különböző levonásokat tartalmazza a fizetésből. Ez minden egyes kollégára külön ki van számolva, így ezzel sem szükséges bajlódni, csak egyszerűen ki kell választani, az adott munkatárs nevét, és már is láthatóvá válik a levonások fajtája és mértéke százalékban és pontos értékben is.

Ez az oszlop, tehát a dolgozó nevével kezdődik, majd alatta tételek szerint lehet látni a különböző bérlevonásokat, mint például a nyugdíjjárulék, társadalombiztosítás, SZJA stb. Ezeknek láthatjuk mind a százalékos, mind a pontos értékét is. Az oszlop ezeken kívül még a bruttó fizetést is tartalmazza, és az utolsó sor a felhasználó munkájának könnyítése érdekében a levonások összértékét jeleníti meg.

## Bér és járulékok levonása

Dolgozó neve:

### Bér levonások

Levonás neve:

Százaléka:

Levonás mértéke a bérből:

**15. ábra: Könyvelői főoldal második oszlopa**

*Forrás: Saját készítés*

A könyvelői főoldal harmadik, egyben utolsó oszlopában, a munkavállaló adatai szerepelnek. Ide tartozik a munkavállaló neve, beosztása, telefonszáma, e-mail címe, és az órabére. Az adatokon kívül szintén ebben az oszlopban találjuk meg a hónapban ledolgozott összes óraszámát, szabadságon töltött napok számát, a táppénzen töltött napok számát, és az ezekből kiszámolt nettó fizetést is.

### Munkavállaló adatai

Név:

Beosztás:

Telefon:

### Munkavállaló adatai

Név: shopOwner

Beosztás: manager

Telefon: +36111111

**16. ábra: 3. oszlop adatok betöltése előtt**

*Forrás: Saját készítés*

**17. ábra: 3. oszlop adatok betöltése után**

*Forrás: Saját készítés*

A főoldalon nincs lehetőség az adatok szerkesztésére. A szerkesztő opciókat a fejlécben elhelyezett legördülő menük segítségével érheti el a felhasználó.

### 4.2.2. Adatbázis műveletek

A fejlécben található első legördülő menü, az adatbázis műveleteket tartalmazza. Az alkalmazottak itt érhetik el, az adatbázist szerkesztő felugró ablakokat. Az 'adatbázis műveletek' gombra kattintva két opcióból választhat a felhasználó. Az egyik lehetőség az adatbázis frissítése, a másik pedig az új adat felvitele.

Az 'adatbázis frissítése' opcióra kattintva egy felugró ablak jelenik meg a felhasználó előtt. Ezen a felületen lesz lehetőség a munkavállalók adatainak módosítására. Ilyen adatok például, a munkavállaló neve, telefonszáma, e-mail címe stb. Az egyes dolgozókról tárolt információkat, először az adatbázisból kell betölteni. Ezt a 'nevek lekérése' gombbal tudja megtenni a programot kezelő kolléga. Ekkor a 'Dolgozó' legördülő menüjében megjelenik a munkavállalók névsora. Itt lehet kikeresni, az adott munkatársat, akinek az adatait felül szeretné írni. Miután kiválasztotta a keresett nevet, a 'Dolgozó keresése' gombbal, az adott munkatárs adatai megjelennek a felületen. A gyorsabb kitöltés és könnyebb munkavégzés érdekében, azokra az adatokra, melyeket nem szükséges módosítani, elég csak ráklikkelni, és változatlan formában feltölti a program automatikusan a mellette lévő 'Frissített adat' mezőt. Természetesen a frissíteni kívánt adatokat, manuálisan kell feltölteni az adott kollégának.

**18. ábra: Frissítési felugró ablak, adatok betöltése előtt** //Forrás: Saját készítés

**19. ábra: Frissítési felugró ablak, adatok betöltése után** //Forrás: Saját készítés

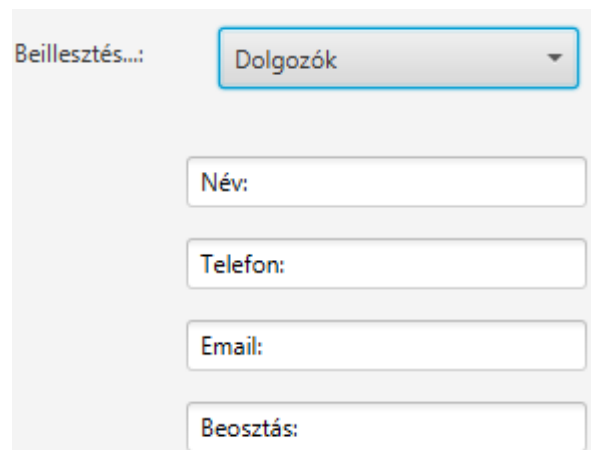
A 'Frissített adat' mezők kitöltése után, a dolgozóhoz tartozó új adatokat az 'Adatok frissítése' gombra kattintva lehet elmenteni és feltölteni az adatbázisba. A dolgozók adatain kívül, még a levonási adatok frissítésére is lehetőség van a felületen, a fentebb leírtakhoz hasonlóan, csak a 'Levonási adatok keresése' gomb segítségével. A leírtakon kívül, szintén ezen az oldalon van lehetőség, a dolgozó adatainak esetleges törlésére, amennyiben ez valamilyen indokolt esetben szükségessé válna.

Az adatbázis műveletek legördülő menü második opciója, az Új adat felvitele. Természetesen erre az opcióra klikkelve érheti el, az adott kolléga, azt a felugró ablakot, ahol új adatokkal tudja bővíteni a dolgozói adatbázist. Fontos, hogy mielőtt feltölti az új adatokat, ki kell választania a megfelelő adatállományt, azaz, dolgozók (munkatársak személyes adatait tartalmazza, például név, telefon, e-mail, beosztás stb.) vagy levonások (különböző bérlevonások fajtaíait, és mértékét tartalmazza, például nyugdíj, társadalombiztosítás, személyi jövedelemadó stb.) elnevezésű adatbázist. Egyszerre csak egy adatbázisba lehetséges új adatokat felvenni, így egy új kolléga adatainak felvételénél, mindkét adatállomány szerkesztése szükséges. Az új adatokat a 'Felvitel az adatbázisba.' elnevezésű gombbal lehet elmenteni. A megfelelő adatállomány kiválasztása után, a kitöltendő mezőkben megjelenő segédfeliratok segítik a felhasználók további tájékozódását (21. ábra).

The screenshot shows a web form titled 'Beillesztés....:'. At the top, there is a dropdown menu labeled 'Adatbázis választás' with a downward arrow. Below this menu, there are four empty text input fields stacked vertically.

**20. ábra: Adatfelviteli felugró ablak, a megfelelő adatbázis kiválasztása előtt**

*Forrás: Saját készítés*

The screenshot shows the same web form as in the previous image, but the dropdown menu now displays 'Dolgozók' with a downward arrow. Below the dropdown, there are four labeled text input fields: 'Név:', 'Telefon:', 'Email:', and 'Beosztás:'.

**21. ábra: Adatfelviteli felugró ablak, a megfelelő adatbázis kiválasztása után**

*Forrás: Saját készítés*

### 4.2.3. Munka/Szabadság napok kezelése

A fejlécben található második legördülő menü, a munkanapok és a szabadságon töltött napokat tartalmazza, minden kollégára név szerint. Ennél a menüpontnál is két különböző opcióból választhat a felszanaló. Az egyik lehetőség a 'Munkaórák felvitele' (22. ábra), a másik pedig a 'Szabadságok kezelése'.

Az előbb említett alternatívát választva, az a felugró ablak jelenik meg a programot kezelő kolléga előtt, amelyben a munkaórákat tudja hozzárendelni, adott munkatárshoz, ami az adott dolgozó bérszámfejtéséhez is egy nagyon jó ellenőrzési mód. A felületen, a már eddig is említett segédfeliratok, és lebegő feliratok könnyítik meg a felhasználók munkáját. Fontos különbség viszont, hogy az eddigiekkel ellentétben, itt nem csak az adatbázisból lehet betölteni a neveket, és utána kiválasztani a megfelelő munkatársat, hanem manuálisan is kilehet tölteni az adott mezőt. Miután sikerült kiválasztani vagy beírni a felhasználónak a megfelelő kolléga nevét, a következő cellában a napi ledolgozott óraszámot kell megadni, majd az azt követő lépésben a ledolgozott munkaórákhoz, a megfelelő dátumot kell kijelölni. A dátum kiválasztásában a 'dátum választó' mező jobb szélén elhelyezkedő naptárikon segíti a felhasználót. Erre rákattintva, egy kis naptár jelenik meg, amelyben a kezelő kiválaszthatja a ledolgozott órákhoz tartozó napot. A dátumot, és a hozzárendelt munkaórát, a 'Bejegyzés' gomb használatával lehet rögzíteni. A mellette jobbra látható 'Nevek.' feliratú gomb tölti be az alkalmazottak névsorát a felület jobb oldalán található listába.

22. ábra: Munkaórák felvitele panel //Forrás: Saját készítés

A legördülő menü második alternatívája, a szabadságok kezelésére szolgál. A felület felépítésében nagyon hasonló az előbb részletezett munkaórák rögzítésére alkalmas panelhez. A különbség abban látható, hogy ebben a felugró ablakban, nincs lehetőség manuálisan megadni a szabadságon lévő kolléga nevét, hanem először a 'Nevek' gomb segítségével be kell tölteni a névsort az adatbázisból, majd a felület jobb oldalán található listából kell kattintással kiválasztani azt. A kiválasztás sikerességét, az adott sor kék színnel való kiemelése jelzi a felhasználó számára. Miután megtörtént a kolléga nevének kiválasztása, a szabadság elszámlolásához szükséges dátumokat kell megadni. Ezt a szabadság kezdő dátumával és befejező dátumával lehet megadni. Abban az esetben, ha a szabadság idejében munkaszüneti nap is tartozik, akkor több részletben kell felvenni a szabadságot, pl.: ha a vállalatnál a szombat, vasárnap munkaszüneti nap, és egy adott kolléga esetleg péntektől, keddig lenne szabadságon, akkor ezt 2 részletben kellene rögzíteni, az egyikben csak a pénteki nap lenne benne, a másikban pedig a hétfő, kedd. A felületen a pontosabb számolás érdekében, meg lehet határozni, az adott szabadság fajtáját is, azaz, hogy esetleg betegség miatt volt a dolgozó távol. Ezt a 'Táppénz?' felirat mellett található mező kipipálásával teheti meg a felhasználó.

**23. ábra: Szabadságok kezelése panel** //Forrás: Saját készítés

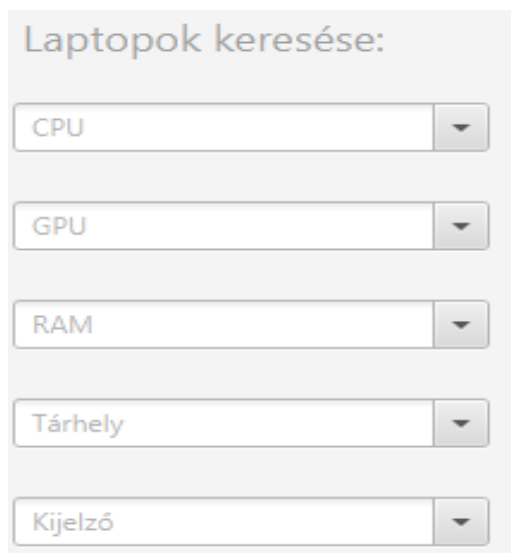
A két alternatívában, tehát a munkaórák, és a szabadságok kezelésére használt felületekben, az az egyik fontos különbség, hogy a munkaórát, bár nem mindennap kötelező, de mindennapra külön kötelező megadni, míg a szabadságokat akár egy részletben is be lehet írni, attól függően, hogy esik-e bele munkaszüneti nap vagy sem.

### 4.3. Az értékesítői felület

#### 4.3.1. Eladói főoldal és lehetőségei

Az eladással foglalkozó munkatársak, az értékesítői felületre történő bejelentkezés után az eladói főoldallal találkoznak majd először. A fő oldal felépítése a könnyebb áttekintés érdekében itt is három fő oszlopból áll, hasonlóan a könyvelői főoldalhoz.

Az első oszlopban, az értékesítendő laptopok kikeresésére van lehetősége a felhasználónak, a különböző specifikációk kiválasztásának segítségével. Több specifikáció beállítására is van mód, mint például: CPU, GPU, RAM, tárhely, kijelző, gyártó, név, és még az ár is beállítható. A kitöltendő mezők alatti 'Adatlekérés' gomb használatával lehet az információkat, az adatbázisból betölteni, de akár manuálisan is be lehet írni a keresendő specifikációt. A keresés sikerességéhez, nem feltétlenül szükséges minden adatot megadni, elég akár egyre rákeresni, így azokat a laptopokat kapjuk majd találatként, melyekben ez az adat megegyezik. A felhasználó tájékozódását, ezen a főoldalon is a segédfeliratok, és a lebegő feliratok segítik.



Laptopok keresése:

CPU

GPU

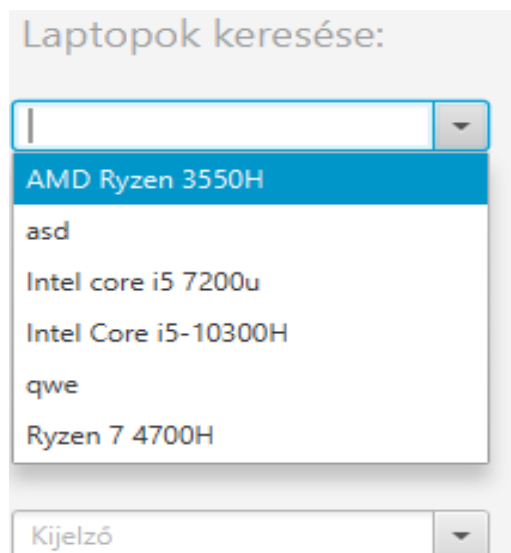
RAM

Tárhely

Kijelző

24. ábra: 1. oszlop adatok betöltése előtt

*Forrás: Saját készítés*



Laptopok keresése:

AMD Ryzen 3550H

asd

Intel core i5 7200u

Intel Core i5-10300H

qwe

Ryzen 7 4700H

Kijelző

25. ábra: 1. oszlop adatok betöltése után

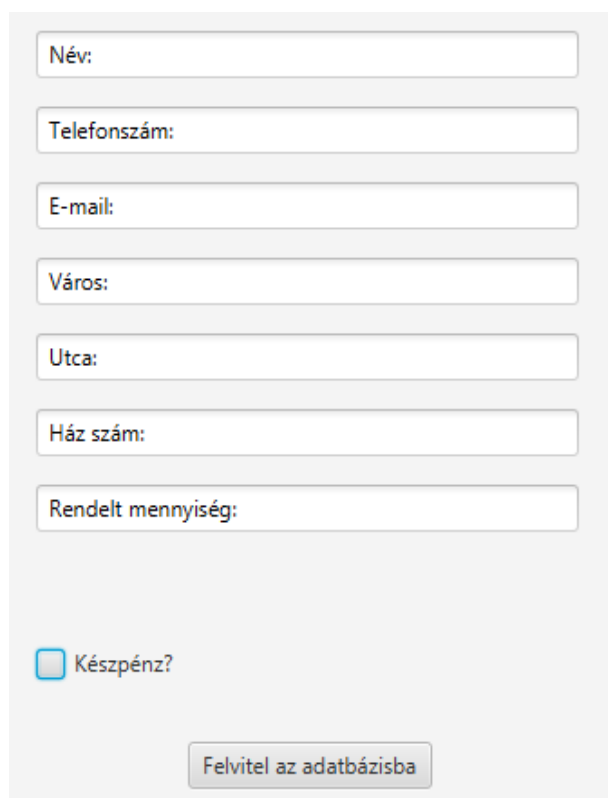
*Forrás: Saját készítés*

A listából való kiválasztást, ebben az esetben is a kék kiemelés könnyíti meg (25. ábra).

A keresési eredmények a második oszlopban található listában fognak megjelenni. A listában a sorokat itt is a halványszürke, és a fehér szín váltakozása fogja megkülönböztetni

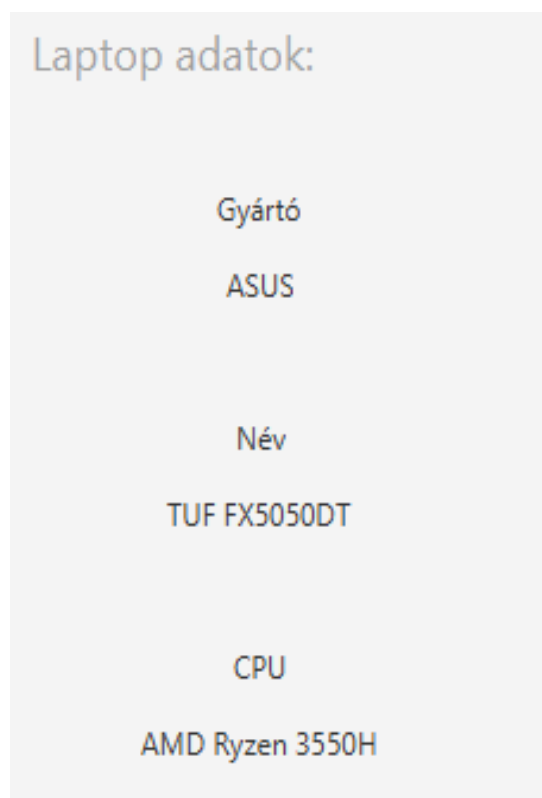
egymástól. Az adott laptop kiválasztását, pedig a sor háttérének kékszínre váltása jelzi majd a felhasználó számára. Miután a beállított specifikációk alapján megkapta az értékesítő munkatárs a keresési eredményeket, majd sikeresen kiválasztotta a keresett laptopot, az első oszlopban található 'Vétel..' gomb segítségével tudja rögzíteni a vásárlói igényt.

A gomb használatakor egy új felugró ablak jelenik meg a kezelő kolléga előtt, melyben a vásárló adatait (például: neve, címe, telefonszáma) lehet rögzíteni, és a rendelési mennyiséget. Ezeken kívül lehetőség van a fizetési módot is megjelölni, a 'Készpénz?' felirat melletti négyzet esetleges kipipálásával. Menteni itt is a 'Felvitel az adatbázisba' gomb segítségével lehet. Miután megadta a felhasználó ezeket az információkat, a program a kijelölt laptopot a vásárlóhoz rendeli. Az adatok esetleges módosítási lehetőségei a későbbiekben lesznek részletezve.



**26. ábra: Vásárlói igény felvételének panelje**

*Forrás: Saját készítés*



**27. ábra: Főoldal 3. oszlopa**

*Forrás: Saját készítés*

A főoldal harmadik oszlopa (27. ábra) pedig, a keresési eredményekből kiválasztott laptop adatait tartalmazza, mint például: gyártó, név, CPU, GPU stb. Ezen kívül még fontos, hogy az elérhető mennyiség is szerepel itt, ami pontos tájékoztatást ad az értékesítő kollégának és természetesen az ár is.



### 4.3.2. Adatbázis műveletek

A fejlécben található első legördülő menü itt is, az adatbázis műveleteket tartalmazza. Az alkalmazottak itt érhetik el, az adatbázist szerkesztő felugró ablakokat. Az 'adatbázis műveletek' gombra kattintva három opcióból választhat a felhasználó. Az egyik lehetőség az új laptop felvitele, a másik a laptop frissítése, a harmadik pedig a rendelés frissítése/ törlése.

Új termék esetén, természetesen, az új laptop felvétele elnevezésű felugró ablakot kell választania a programot kezelő kollégának. Ezen a felületen lehet minden, az új termékkel kapcsolatos, információt rögzíteni az adatbázisba. Az értékesítéssel foglalkozó kollégának manuálisan kell megadni a laptop egyes paramétereit, többek között a CPU, GPU, RAM, háttértár, kijelző adatait. Ezeken kívül az adatbázisban még szerepelnie kell a laptop gyártójának, nevének, raktáron lévő mennyiségének, elérhetőségének, árának és a laptopról készült rövid leírásnak. Mindezeket itt is a 'Felvitel az adatbázisba' gomb segítségével lehet elmenteni.

The form is titled 'Új laptop felvétele' and is used for entering new laptop data into the database. It consists of the following fields:

- Laptop Gyártó: (Text input)
- Laptop Neve: (Text input)
- Raktáron levő mennyiség: (Text input)
- Élrhető?(0:nem, 1:igen) (Text input)
- CPU: (Text input)
- GPU: (Text input)
- RAM: (Text input)
- Háttértár: (Text input)
- Kijelző: (Text input)
- Ár: (Text input)
- Laptop leírása: (Large text area)
- Felvitel az adatbázisba (Button)

**28. ábra: Új laptop felvitele** // *Forrás: Saját készítés*

A felületet kezelő munkatársnak, csak arra kell figyelnie, hogy az oldalon található kérdésre, miszerint elérhető-e az adott laptop vagy sem, számmal kell válaszolni. Az 1-es szám jelenti az igent és a 0-ás lesz a nem. Ez persze zárójelben feltüntetve szerepel az adott mezőben.

Ha egy olyan termék adatait szeretné a felhasználó módosítani, amely már szerepel az adatbázisban, akkor a 'Laptop frissítése' opciót kell választania. Ezen a felületen, a programot kezelő kolléga, a laptop gyártójával tudja ki keresni azt a bizonyos készüléket, melynek jellemzőit módosítani szeretné. Először persze az adatokat kell a megfelelő adatbázisból betölteni. Ezt a 'Keresési infó frissítése' gombbal tudja megtenni az alkalmazott. Ezt követően, a gomb mellett található legördülő menüből lehet kiválasztani a keresett gyártót. Ha ez megtörtént, az adott gyártóhoz tartozó laptopok neve, a panel jobb oldalán található listában jelennek meg. Az adott sorokat itt is halványszürke és fehér színek váltakozása különbözteti meg egymástól, a kijelölt sor háttere pedig, itt is kék színűre változik.

ASUS ▼ Keresési infó frissítése Info Frissítése

Gyártó:	ASUS	ASUS	TUF FX5050DT
Név:	TUF FX5050DT	TUF FX5050DT	
Spec. ID:	2	Spec. ID nem frissíthető!	
Elérhető db.szám:	3	Frissített infó	
Elérhetőség:(0:nem, 1:igen) 1		Frissített infó	
CPU:	AMD Ryzen 3550H	Frissített infó	
GPU:	nVidia GTX1650	Frissített infó	
RAM:	8GB DDR4	Frissített infó	
Tárhely:	256GB SSD	Frissített infó	
Kijelző:	15,6" IPS	Frissített infó	
Ár:	800	Frissített infó	

Tapasztald meg a simább és magával ragadóbb játékot az új ASUS TUF Gaming FX505 NVIDIA változattal.

**29. ábra: Laptop frissítése // Forrás: Saját készítés**

Miután ebből a listából is kiválasztotta a felhasználó a keresett készüléket, az ehhez a laptopoz tartozó adatok megjelennek a felületen, és innentől lehet őket módosítani. A gyorsabb munkavégzés érdekében, itt is lehetőség van arra, hogy azokat az adatokat, melyet a felhasználó nem kíván frissíteni, gyorsan változtatás nélkül betöltse a megfelelő mezőbe. Ehhez csak egyszerűen rá kell klikkelni az adott információra. Ahol pedig szükséges a javítás, ott

manuálisan kell kitölteni a kezelőnek a 'Frissített adat' mezőt. A laptopokhoz tartozó összes adat szerkeszthető, kivéve a készülék specifikáció azonosítója, mert az egy külső kulcs, egy másik adatbázisban, ami az adatbázis egy adott sorát referálja.

Az utolsó opció az adatbázis műveletek legördülő menüjében a rendelés frissítése/törlése. Ez akkor válik szükségessé, ha a vásárlói igény felvétele után a vevő esetleg módosítani, vagy akár elállni szeretne a rendelésétől. Ezen a felületen az programot kezelő kolléga, a már felvett rendelésben a mennyiséget tudja megváltoztatni. Először ehhez ki kell keresni a megfelelő vásárlót, amit az adatbázisban tárolt adatai (például: neve, telefonszáma, e-mail címe stb.) alapján lehetséges. A beállított információknak megfelelő keresési eredmények a felület középső részén található listában helyezkednek el. Ebből a találati listából kell a módosítandót kiválasztani. A sikeres kijelölés után két opció közül választhat az értékesítő munkatárs, az egyik, hogy módosítja a rendelési mennyiséget, amit manuálisan kell beírnia, a másik pedig, hogy törli az adott rendelést.

The screenshot shows a web application interface for managing orders. It features a search form on the left with fields for Name, Phone number, E-mail, City, Street, and Zip code. Below these fields are three buttons: 'Vásárlói rendelések keresése' (Search for customer orders), 'Vásárlás frissítése' (Update purchase), and 'Kijelölt vásárlás törlése' (Delete selected purchase). The central part of the interface is a large, empty rectangular area labeled 'Korábbi vásárlások:' (Previous purchases). To the right of this area is a details panel with fields for Manufacturer, Name, CPU, GPU, RAM, Storage, Monitor, Quantity (with a sub-field 'Mennyiség:'), Price, and Purchase ID. The interface is clean and functional, with a light gray background and white input fields.

**30. ábra: Rendelés frissítése/törlése // Forrás: Saját készítés**

Mind az értékesítői, mind a könyvelői felület letisztult, könnyen értelmezhető dizájnt képvisel, hogy egyszerűbben befogadható legyen a felhasználók számára.

#### **4.4. Vállalatvezetők lehetőségei**

A vállalatvezetők lehetőségei a program szempontjából nem sokban különböznek az alkalmazottakétól. Az egyszerű és gyors tanulhatóság érdekében az alkalmazás nem tartalmaz rejtett funkciókat, melyet csak speciális azonosítóval lehetne elérni. Különbséget egyedül a bejelentkezéshez kapott felhasználónév és jelszó páros tesz kizárólag. Ez lehetővé teszi számukra, hogy mind az értékesítők, mind a könyvelők felületre betudjanak jelentkezni. Ezáltal tudják ellenőrizni is akár az alkalmazottakat, vagy módosítani a kívánt adatokat, akár helyettesíteni is egy adott kollégát.

Az alkalmazás az egyszerűség, könnyen kezelhetőség, és költséghatékonyság szellemében készült, így vannak benne esetleges hiányosságok (például a könyvelői felületen), de kisebb, induló vállalkozások számára, nagyon hasznos lehet.

#### **5. Összefoglalás**

A célkitűzésem a program megírásával az volt, hogy egy olyan szoftvert hozzak létre, ami segíti a kisvállalkozások működését. Alacsony árával segítve azok spórolását, és ezzel a hosszútávú fejlődésüket, innovációjukat, hogy egyre nagyobb portfólióval rendelkezessenek a későbbiekben. Hiszen nagyon sok funkcióspecifikus szoftver, mint például egy bérszámfejtő program, ami éves szinten 99.000Ft-tól akár 199.990Ft-os költséggel is rendelkezhet, funkcióktól függően, vagy egy könyvelői program, ami 119.900 + áfától akár eget rengető 399.000 + áfás árcédulával is rendelkezhet sok ilyen kisvállalkozás esetében elképzelhetetlen, vagy egyenesen fölösleges lenne beszerezni, de muszáj, hiszen ezeket a funkciókat használniuk kell. Ha a raktárkezelést is digitális formában akarják számon tartani a havidíjas szoftverrel, akkor az ilyen programok árai a 9.900 – 19.900Ft-os összegek között mozog, szintúgy funkcióktól függően. Ráadásul ilyen kis cégek esetében sok funkciót egyáltalán, vagy csak nagyon ritkán tudnának kihasználni, így egyáltalán nem érné meg ezek beszerzése.

Így programom igyekszik ezeknek a funkcióknak, vagy részüknek megfelelni, egy kitalált vállalkozás által támasztott feltételekhez igazodóan. Ezzel sikerült a program árát minimálisan tartva egy készletkezelő, nyilvántartó, dolgozókat és azok ledolgozott munkaóráit, szabadságon, vagy táppénzen töltött napjaikat, illetve az adott hónapra, persze megfelelő adminisztráció mellett, a bérszámfejtésüket is készítse el. Ez persze csak egy segédmunkó

lenne, hiszen csak az adott hónapra számolja ezeket, illetve mutatja a dolgozó ledolgozott munkaóráit és szabadságait, de természetesen szükség esetén ezek az adatok az adatbázisban továbbra és visszamenőleg is elérhetőek.

Úgy érzem ezeknek a célkitűzéseknek sikerült eleget tennem, és mindent a kitalált cég igényeihez tudtam igazítani. Egy letisztult, könnyen érthető, reszponzív programot sikerült összehozzak, ami teljesen megfelelt a célkitűzéseimnek, illetve minden megrendelő is ilyen programot vár el.

Összességében teljes mértékben elégedett vagyok, hiszen minden a kezdetben kitűzött célt sikerült elérjem.

## **6. Irodalomjegyzék**

### **6.1. Forráskód link**

[1] Forráskód: <https://github.com/TomiMan7/szakdoga>

#### **Internetes adatgyűjtések:**

[2] Gluon dokumentáció: <https://docs.gluonhq.com/charm/2.0.0/>

[3] Java 11 dokumentáció: <https://docs.oracle.com/en/java/javase/11/>

[4] Java 8 dokumentáció: <https://docs.oracle.com/javase/8/javafx/api/toc.htm>

[5] MySQL dokumentáció: <https://dev.mysql.com/doc/>

[6] Bérszámfejtő program ára: [https://ajanlat.kulcs-soft.hu/kulcs-ber-programok?campaignid=1066795125&adgroupid=55239417747&gclid=Cj0KCQiAqdP9BRDVARIsAGSZ8AnseenbyRA8QaCCfpjPZaRXNSOG2zTfsAYunmIRfFkqEJUtvY3FTcaAlcUEALw\\_wcB](https://ajanlat.kulcs-soft.hu/kulcs-ber-programok?campaignid=1066795125&adgroupid=55239417747&gclid=Cj0KCQiAqdP9BRDVARIsAGSZ8AnseenbyRA8QaCCfpjPZaRXNSOG2zTfsAYunmIRfFkqEJUtvY3FTcaAlcUEALw_wcB)

[7] Könyvelői program ára: [https://armada.hu/programcsomagok-konyveloirodaknak/?gclid=Cj0KCQiAqdP9BRDVARIsAGSZ8AmWmsPFPkgdQwETrneq010qnRpSusZX1aUgWYQgD4MZGWqu2oI175gaAtcpEALw\\_wcB](https://armada.hu/programcsomagok-konyveloirodaknak/?gclid=Cj0KCQiAqdP9BRDVARIsAGSZ8AmWmsPFPkgdQwETrneq010qnRpSusZX1aUgWYQgD4MZGWqu2oI175gaAtcpEALw_wcB)

[8] Raktárkezelő program ára: <https://www.ovip.hu/araink/>

[9] Java vs. C++ sebességi cikk: <https://www.diva-portal.org/smash/get/diva2:1463855/FULLTEXT01.pdf>

## **7. Köszönetnyilvánítás**

Szeretnék köszönetet mondani a témavezető tanáromnak Major Sándor Rolandnak, aki időt nem kímélve adott tanácsokat a szakdolgozatommal kapcsolatban.

Szeretnék köszönetet mondani a barátnőmnek, Gulyás Viktóriának, aki végig mellettem állt miközben a dolgozatot készítettem.

Szeretnék köszönetet mondani a bátyámnak, Löki Krisztiánnak, aki elméleti és gyakorlati tanácsokat adott a forráskóddal kapcsolatban.

És végezetül, de nem utolsó sorban, szeretnék köszönetet mondani a családomnak, akik az egyetemi éveim alatt folyamatosan segítettek és támogattak.