

# **Szak Dolgozat Dokumentáció**

**Selejt Lista**

**Ardai Márk**

**Szoftverfejlesztés 2/14-SL**

**Konzulens:**

**Bogdán Marianna**

## ***I.Bevezetés:***

A Programról röviden:

A programom Javában íródott, 8.2 Netbeans alatt tulajdonképpen egy raktárkezelő felület volt előttem, amit meg kell, valósítsak.

MySQL-ben is dolgoztam, amit a Wamp program segítségével értem el a Localhost/phpmyadmin weboldalon. Javában egy úgy nevezet MySQL-Connectort használtam, hogy ezt meg tudjam valósítani. Azért a Java mellett döntöttem, mert később ebbe szeretnék dolgozni és érdekesnek tűnt az a cél, hogy ebben a programnyelvben írjam meg. A használata hétköznapi van egy bejelentkező felület egy regisztrációs felület és maga a lista felvevő felület.

Mi adta az ihletet, kitűzött célok:

A célom ezzel tulajdonképpen az volt, nagyon egyszerűen, hogy megkönnyebbítsem annak az embernek a dolgát aki esetleg, leltárt csinál, listát vezet a kidobott dolgokból, vagy éppen le ellenőrzi mi is van a raktárba illetve a hűtőkamrába és persze az, hogy programozzak valami újat, önállót. A tovább fejlődés célja sokat közre játszott nálam. Az ihletet az adta, hogy gyors étteremben dolgozom és mindennap hiány volt valamiből vagy nem találtak valamit, tudtam, hogy van már ehhez hasonló program, de úgy gondoltam én is írok egyet. Akkor terveztem úgy, hogy miért ne csinálhatnék egy selejt listát, amire esetleg a kidobot zöldségeket, húsokat, vagy esetleg a be nem érkezett árukat fel lehet jegyezni.

## ***II. A témaválasztás indoklása:***

Személyem szerint nagyon nehéz volt, az első időszakba, hogy miről is kéne, írjak programot vagy mi az, amelyik program nyelvvel szeretnék foglalkozni. Az elején úgy gondoltam a C#(Sharp) közelebb áll hozzám, mert nagyon tetszett a Windows applikációs dizájnok, és második év elején mikor megismerkedtem a Java programozási nyelvvel vacilláltam magamba, hogy melyik is lenne számomra megfelelőbb. Végül hetek múlva a Java mellett döntöttem, mert számomra át láthatóbb volt a kódolás rész, könnyebben tudtam kezelni és néhol beszédesebbnek találtam, mint a C#(Sharp)-ot. Rengeteget kell, még tanuljak programozás terén, de úgy érzem erre a Java lesz számomra a megfelelő nyelv. A szakdolgozatom írás közbe sok újat tanultam, sok mindennek néztem utána és sikerült rá jönnöm egy-két remek dologra, ami segítheti a jövőben a munkásságomat, bármit is hoz a jövő. Még nem tudom, hogy konzol applikációkat szeretnék írni, vagy esetleg maradok a keret projekteknel, amivel a szakdolgozatom is készült.

## ***III. Fejlesztői dokumentáció***

### ***3.1. Az alkalmazott fejlesztői eszközök:***

## Ardai Márk 2/14SL Szak Dolgozat Dokumentáció

A programom Netbeans Java nyelven írtam JFrame Formában, a programom-összekötöttem mysql-connectorral a Wamp program segítségével a localhost/phpmyadmin oldalon csináltam egy adatbázist. A dokumentációmat Wordben írtam és át konvertáltam pdf fájlba. A programomhoz segítséget nyújtott a Google, a Youtube és azon belül is 1BestCsharp felhasználó. A StackOverflowt is használtam nagyon sokat.

### 3.2. Felületterv (opcionális):

Egy belépési funkcióval kezdtem, aminek az alján csináltam egy linket, ami át visz a regisztrációs felületre. A bejelentkezéshez egy felhasználóra és egy jelszóra van szükség.



#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
1	f_id	int(11)			Nem	Nincs		AUTO_INCREMENT	Módosítás Eldobás Tobb
2	f_username	varchar(128)	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Tobb
3	f_password	varchar(128)	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Tobb
4	f_birthdate	date			Nem	Nincs			Módosítás Eldobás Tobb
5	f_firstname	varchar(128)	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Tobb
6	f_lastname	varchar(128)	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Tobb

Amit az adatbázisban egyeztet, amivel beregisztráltunk előtte. MySQL-be két táblát csináltam az egyik a felhasználó, aminek van egy ID (int11 típusú tehát szám ezt automatikusan állítja be) oszlopa egy felhasználónév oszlop (varchar 128, ami 128 karaktert jelent), egy jelszó (varchar 128, szintén 128 karakter lehet) oszlop egy

születési dátum (date típusú, azaz csak dátumot fogad el),  
vezetéknév (varchar 128), és keresztnev (varchar 128).

Erre azért volt szükségem, mert innen érem el majd a bejelentkezésnél a felhasználómat felhasználónév és jelszó alapján.

A másik táblázatom a selejt lista maga, amiket feljegyzetelnek, akik használják persze a programot. Itt is szintén az ID (int 11, automatikus beállítással), oszloppal kezdtem. Majd egy Selejt cím (megnevezés) (varchar 500, 500 karakter hosszú lehet) oszlop. Egy Leírás (varchar 500, szintén karakter hosszú lehet) oszlop, egy Hosszú Leírás (varchar 500, szintén 500 karakter hosszú lehet) oszlop, egy Kategória (varchar 500, szintén 500 karakter hosszú lehet) oszlop, egy Állapot (varchar 500, 500 karakter hosszú lehet) oszlop és egy Dátum (date típusú, csak dátumot elfogadó) oszlop.



The screenshot shows the MySQL Table Structure window for the 'selejt\_felvetele' table. The table has 7 columns: st\_id, st\_cim, st\_leiras, st\_hleiras, st\_kategoria, st\_allapot, and st\_datum. The columns are defined with their respective data types, collations, and constraints.

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
1	st_id	int(11)			Nem	Nincs		AUTO_INCREMENT	Módosítás Eldobás Több
2	st_cim	varchar(500)	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Több
3	st_leiras	varchar(500)	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Több
4	st_hleiras	varchar(500)	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Több
5	st_kategoria	varchar(500)	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Több
6	st_allapot	varchar(500)	utf8_hungarian_ci		Nem	Nincs			Módosítás Eldobás Több
7	st_datum	date			Nem	Nincs			Módosítás Eldobás Több

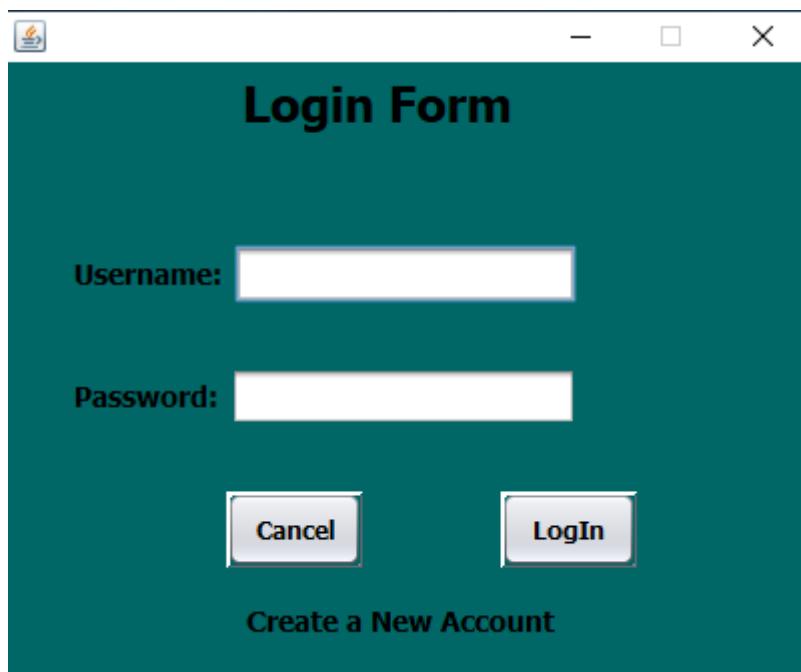
A Java programomat a mysql-connector-java-5.1.47 jar fájlal kötöttem össze az MySQL adatbázissal. Itt egy külön class-t (osztályt) használtam, hogy a programom elérje az adatbázist.

Amibe megadtam egy eljárást, ennek a connector fájlnak a segítségével érje el az adatbázist, a port és a felhasználó segítségével.

```
public class MyConnection {  
    public static Connection getConnection() {  
  
        Connection con = null;  
  
        try {  
            Class.forName("com.mysql.jdbc.Driver");  
            con = DriverManager.getConnection("jdbc:mysql://localhost:3308/szakdoga?useSSL=true", "root", "");  
        } catch (Exception ex) {  
            System.out.println(ex.getMessage());  
        }  
  
        return con;  
    }  
}
```

Sok keresés után találtam rá erre a módszerre, úgy érzem, van még rajta fejleszteni való, próbáltam pár számítógépen és mindig az én elérésemet láttam a fájlknál (jcalendar1.4, mysql-connector-java-5.1.47). Ha lesz, még rá módok alakítani a programom úgy szeretném, hogy ne kelljen a két fájlt hozzá használni, hanem csatlakozzon hozzá egy bizonyos metódussal.

A Java programom, mint írtam ez előtt egy bejelentkező JFrame formával kezd.



**Login Form**

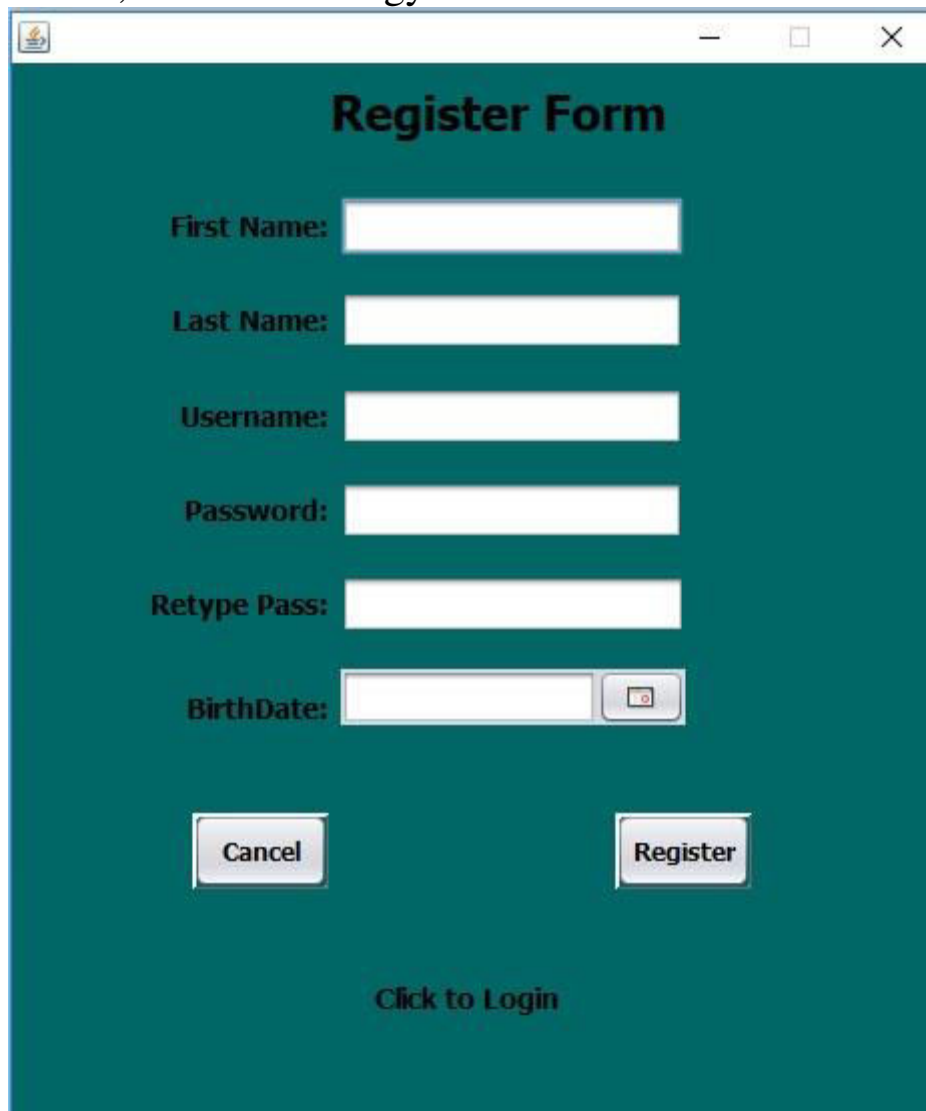
Username:

Password:

Create a New Account

Itt ugye egy felhasználó nevet, és egy jelszót vár, amit az adatbázissal egyeztet, a felhasználó táblából. Csináltam hozzá egy írást „Create a New Account” azaz ha ide

kattint, csinálhatunk egy másik felhasználót.

A screenshot of a web browser window displaying a 'Register Form'. The form has a dark teal background. It contains six input fields: 'First Name:', 'Last Name:', 'Username:', 'Password:', 'Retype Pass:', and 'BirthDate:'. The 'BirthDate' field includes a date picker icon. Below the fields are two buttons: 'Cancel' and 'Register'. At the bottom center, there is a link that says 'Click to Login'. The browser window has a standard title bar with minimize, maximize, and close buttons.

Ezen a formán hozhatunk létre új felhasználót, amihez meg kell adnunk a vezetéknévünk, a keresztnévünk, egy gondolt felhasználó nevet, egy jelszót megerősítéssel, és egy születési dátumot. Mint azt írtam az elején, az adatbázisomba menti le ezeket az adatok a felhasználó táblába. Itt is csináltam egy írást alulra, hogy vissza lehessen térni a bejelentkezés formára az egyszerűsítés kedvéért a „Click to Login”-ra kattintva. Ha beregisztráltunk, és visszatérünk a bejelentkezés részre, bejelentkezünk és megjelenik a listaforma.

Itt már fel tudjuk venni a kívánt selejtet, megnevezés szerint, kategória szerint, állapot szerint, leírást adhatunk neki, majd egy hosszabb leírást, és végül a dátumot,

## Ardai Márk 2/14SL Szak Dolgozat Dokumentáció

amikor történik a selejt vagy a kívánt termék felvétel.

**Selejt Lista**

Id:

Megnevezés:

Kategória:

Állapot:

Leírás:

Hosszú Leírás:

Dátum:

Add Update Delete

ID	Megnevezés	Leírás	Hosszú Leírás	Kategória	Állapot	Dátum
34	Saláta	5kg saláta át v...	Hétfőn jött áru és kaptunk 5kg salátát, ami lem...	Zöldség	Friss	2012-08-06

Itt már az adatbázisban a selejt listával van összekötve. Itt adhatjuk hozzá a selejtünket, módosíthatunk rajta illetve törölhetjük. Amit felvettünk vagy esetleg módosítottunk az a lenti táblába fog megjelenni értelem szerűen. A felületről egy kicsit részletesebben szeretnék beszélni. A bejelentkező felületen találkozhatunk a felhasználónévvel és a jelszóval, ezeknek ugye csináltam két úgy nevezet JTextFieldeket (Java Szöveg mezőket). Található itt két gomb is amiket Eventel (Eseménnyel) láttam el, ugye értelem szerűen a Cancel-el kiléphetünk a programból, a Loginnal bejelentkezünk és tovább visz, a Selejt Lista felületre ez ugye össze van kötve az adatbázis lekérdezésemmel, ha talál ilyen felhasználó nevet, akkor lépjen tovább, és ha helyesen adtuk meg a felhasználó nevet és a jelszót (következő fejezetben illusztrálok képpel). A „Create a New Account” kiírásról már



## Ardai Márk 2/14SL Szak Dolgozat Dokumentáció

korábban írtam, ugye ez arra szolgál, ha egy fiókot szeretnénk csinálni, akkor tovább, visz minket a regisztrációs felületre, amit szintén egy Eventtel (Eljárással) csináltam meg.

```
private void jLabelRegisterMouseClicked(java.awt.event.MouseEvent evt) {  
    RegisterForm rgf = new RegisterForm();  
    rgf.setVisible(true);  
    rgf.pack();  
    rgf.setLocationRelativeTo(null);  
    rgf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    this.dispose();  
}
```

Itt ugye annyit láthatunk, hogy létrehoz egy új ablakot és a regisztrációs formám fog látszani. A regisztrációs formán szintén JTextFieldekét használtam. A Cancel gomb megnyomásával szintén bezárjuk az egész programot a Register gomb segítségével létrehozunk egy új fiókot/felhasználót, amit korábban is írtam, hogy felveszi a felhasználók adatbázis táblámba. A regisztrációs formán is csináltam ugye egy „Click to Login” kiírást erre is szintén írtam egy Eventet ha rá kattintunk jelenjen meg a bejelentkezés formám.

```
private void jLabelRegisterMouseClicked(java.awt.event.MouseEvent evt) {  
    LoginForm lgf = new LoginForm();  
    lgf.setVisible(true);  
    lgf.pack();  
    lgf.setLocationRelativeTo(null);  
    lgf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    this.dispose();  
}
```

Itt már egyszerűbb dolgom volt, mert ez ugyan az az eljárás, mint amit a bejelentkezés formán írtam a regisztrációs kattintásra csak át kellett írnom az első sort, hogy a bejelentkező formát lássam. A bejelentkezés formán a LogIn gombra kattintva megjelenik a Selejt Lista szintén egy Eventre volt szükségem, hogy a Selejt Listát lássam. Amit illusztrálni fogok a következő fejezetben, mert itt végeztem is egy ellenőrzést, ha van ilyen nevű felhasználó és jelszó regisztrálva, csak akkor jelentkezzen be, különben dobjon fel hiba üzenetet. Ha

beléptünk és minden rendben talál, megjelenik a Selejt Lista forma. A Selejt Lista formán szintén JTextFieldeket használtam. Az adat be írásához. A dátumhoz teljesen mást, használtam külön leszedtem egy ki egészítő jcalendar1.4 jar fájlt, ami elérhetővé tett nekem egy úgy nevezet JDateChoosert naptár dizájnt, így sokkal egyszerűbb kiválasztani, hogy melyik nap is vettük fel a selejtet.

### 3.3. Részletes feladatspecifikáció, algoritmusok

A bejelentkező felületen csináltam egy vizsgálatot, ha esetleg az adatbázisba nem regisztráltam be egy felhasználót, vagy rossz jelszót írtam a be regisztrált felhasználóhoz akkor dobjon fel egy üzenet dobozt, hogy valami hibás.

```
try {
    ps = con.prepareStatement("SELECT `f_username`,`f_password`,`f_id` FROM `felhasznalok` WHERE `f_username`= ? AND `f_password` = ?");
    ps.setString(1, uField.getText());
    ps.setString(2, String.valueOf(pField.getPassword()));
    rs = ps.executeQuery();

    if (rs.next()) {
        MyContactsForm mcf = new MyContactsForm();
        mcf.setVisible(true);
        mcf.pack();
        mcf.setLocationRelativeTo(null);
        mcf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        this.dispose();
    } else {
        JOptionPane.showMessageDialog(null, "Valami hiba történt.");
    }
} catch (SQLException ex) {
    Logger.getLogger(LoginForm.class.getName()).log(Level.SEVERE, null, ex);
}
```

A lényege, hogy elkezd futni a ciklus és le ellenőrzi, van-e az adatbázisba ilyen nevű felhasználó ezzel a jelszóval és csak akkor lép tovább a selejt lista felületre, ha mindent rendben talált.

A Selejt Lista felületemről a feladatspecifikáció:

Az Add gombra egy bizonyos Insert (beillesztés) függvényt alkalmaztam, ez ugye az adatbázisoknál egy függvény, mint az adatbázisban itt is meg kell adnom melyik táblába, szeretném felvenni az adott selejtet vagy terméket, a mysql-connector segítségével egy con nevezetű változóra meg tudtam hívni az Insert függvényt.

```
public boolean insertContact(contact cont)
{
    boolean contactIsCreated = true;
    Connection con = MyConnection.getConnection();
    PreparedStatement ps;
    try {

        ps = con.prepareStatement("INSERT INTO 'selejt_felvetele'('st_cim', 'st_leiras', 'st_hleiras', 'st_kategoria', 'st_allapot', 'st_datum') VALUES (?, ?, ?, ?, ?, ?)");
        ps.setString(1, cont.getStcim());
        ps.setString(2, cont.getStleiras());
        ps.setString(3, cont.getSthleiras());
        ps.setString(4, cont.getStkategoria());
        ps.setString(5, cont.getStallapot());
        ps.setString(6, cont.getStdatum());

        if (ps.executeUpdate() != 0) {
            JOptionPane.showMessageDialog(null, "Új selejt hozzáadva!");
            contactIsCreated = true;
        }
        else{
            JOptionPane.showMessageDialog(null, "Valami hiba");
            contactIsCreated = false;
        }

    } catch (SQLException ex) {
        Logger.getLogger(contactQuery.class.getName()).log(Level.SEVERE, null, ex);
    }
    return contactIsCreated;
}
```

Található itt még egy Update gomb, ezt is szintén az adatbázisból való függvénnyel sikerült megcsinálnom, ugye ezzel tudom, módosítani a Selejtemet vagy akármilyen termékemet ki mire szeretné használni. Ahogy az adatbázisban itt is meg kell adnom melyik Selejtet vagy terméket szeretném módosítani. A lényege az, hogy amit felvettünk korábban selejt azt lehívja és lehet módosítani.

```
public void updateContact(contact cont)
{
    Connection con = MyConnection.getConnection();
    PreparedStatement ps;
    String updateQuery = "UPDATE `selejt_felvetele` SET `st_cim` = ?, `st_leiras` = ?, `st_hleiras` = ?, `st_kategoria` = ?, `st_allapot` = ?, `st_datum` = ? WHERE `st_id` = ?";
    try {
        ps = con.prepareStatement(updateQuery);

        ps.setString(1, cont.getStcim());
        ps.setString(2, cont.getStleiras());
        ps.setString(3, cont.getSthleiras());
        ps.setString(4, cont.getStkategoria());
        ps.setString(5, cont.getStallapot());
        ps.setString(6, cont.getStdatum());
        ps.setInt(7, cont.getStid());
        if (ps.executeUpdate() != 0) {
            JOptionPane.showMessageDialog(null, "Selejt módosítva lett!");
        }
        else{
            JOptionPane.showMessageDialog(null, "Valami hiba");
        }
    } catch (SQLException ex) {
        Logger.getLogger(contactQuery.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

Az utolsó gomb a Delete ezt is szintén az adatbázis függvénnyel van összekötéssel ugyan, úgy ahogy ott szeretnénk törölni valamit úgy adtam meg itt is egy bizonyos ID kiválasztással tegye meg a kívánt Selejt vagy terméktörlést.

```
public void deleteContact(int st_id)
{
    Connection con = MyConnection.getConnection();
    PreparedStatement ps;
    try {
        ps = con.prepareStatement("DELETE FROM `selejt_felvetele` WHERE `st_id` = ?");
        ps.setInt(1, st_id);

        if (ps.executeUpdate() != 0) {
            JOptionPane.showMessageDialog(null, "Selejt törölve.");
        }
        else{
            JOptionPane.showMessageDialog(null, "Valami hiba");
        }
    } catch (SQLException ex) {
        Logger.getLogger(contactQuery.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

Ezeket a metódusokat a `contactQuery` nevezetű osztályba írtam meg és így hívtam meg a `Selejt Listakezelés` felületemre, a gombokra kattintva.

A `Selejt Lista` táblázatott, egy `Jpanel` segítségével csináltam, ami a `Javaban` egy úgy nevezet tulajdonság. Erre helyeztem rá a beírni kívánt szöveget, a gombokat ugyan úgy, mint a többi ablakomon. Található még itt egy tábla tulajdonság ahol megjelenik a felvett `Selejt` vagy termék. Ahhoz, hogy ez sikerülhessen írtam egy külön osztályt `myModel` néven, ahol amint látszódik az első sorba a megnevezések, és, hogy mit hol helyezzen el mind itt állítottam be, hány oszlop legyen és az adatbázisból lekért adatok során hány sor. Ezt az osztályom meghívtam a tábla tulajdonságra, így megjelenítve az adatokat. A táblának csináltam még egy metódust, hogy ha rá kattintunk egy `selejt`re a táblába az összes adat megjelenjen a bekívánt írt helyre, ezt úgy tudtam elérni, hogy kattintásra csináltam egy eseményt hogy az adatokat fogja meg és illessze be az adott

helyekre.

```
private String [] columns ;
private Object[][] rows ;

public myModel(){}

public myModel(Object [][] data, String [] columnsName){
    this.columns = columnsName;
    this.rows = data;
}

@Override
public Class getColumnClass(int col)
{
    if (col == 7) {return Icon.class; }
    else{
        return getValueAt(0, col).getClass();
    }
}

@Override
public int getRowCount() {
    return this.rows.length;
}

@Override
public int getColumnCount() {
    return this.columns.length;
}

@Override
public Object getValueAt(int rowIndex, int columnIndex) {
    return this.rows[rowIndex][columnIndex];
}

@Override
public String getColumnName(int col){
    return this.columns[col];
}
```

A regisztrációs felületemre több ellenőrzést is csináltam, elsőnek kezdem az üres mezőkkel.

Csináltam egy olyan ellenőrzés ciklust, amely megnézi, hogy mely mezők, üresek és feldob egy hiba ablakot, hogy például: a vezetéknév és utónév hiányos, nem adtunk meg felhasználó nevet, nem adtunk meg jelszót, vagy esetleg a két jelszó nem egyezik.

```
public boolean verifData()
{
    if (fnF.getText().equals("") && lnF.getText().equals("") || uField.getText().equals("") || String.valueOf(pF1.getPassword()).equals("")) {
        JOptionPane.showMessageDialog(null, "Egy vagy több mező üres!");
        return false;
    } else if (!String.valueOf(pF1.getPassword()).equals(String.valueOf(pF2.getPassword())))
    {
        JOptionPane.showMessageDialog(null, "Helytelen jelszó!");
        return false;
    } else
    {
        return true;
    }
}
```

Csináltam még egy olyan ciklust, ami le ellenőrzi, hogy egy adott felhasználónév már szerepel az adatbázisban és nem lehet vele még egyszer regisztrálni, itt is szintén hiba üzenetet dob fel „A felhasználónév már létezik”.

```
public boolean inUsernameExist(String un)
{
    boolean uExist = false;
    Connection con = MyConnection.getConnection();
    PreparedStatement ps;
    ResultSet rs;

    try {
        ps = con.prepareStatement("SELECT * FROM `felhasznalok` WHERE `f_username`= ? ");
        ps.setString(1, uField.getText());

        rs = ps.executeQuery();

        if (rs.next()) {
            uExist = true;
        }

    } catch (SQLException ex) {
        Logger.getLogger(LoginForm.class.getName()).log(Level.SEVERE, null, ex);
    }

    return uExist;
}
```

Erre úgy gondolom, azért van szükség, ha esetleg, használtban lesz a programom valahol jó, ha van figyelmeztetés, hogy hol hibáztunk, mit nem töltöttünk ki.



### ***3.4. Tesztelési Dokumentáció***

A gépemen a wamp program beállított egy saját ip-t a localhost/phpmyadminra ez a 3308-as port, amit minden gépen át kell, állítsak a csatlakozási osztályomban.

Ezen kívül nem tapasztaltam hibát, amikor le futott csak az elején voltak, hogy például nem mindegy melyik java connector fájlt használok, mert vannak elavultak, amik nem kompatibilisek a wamp programmal, legalábbis ahogy utána néztem ezt véltem fel fedezni.

Ahogy írtam korábban a biztonságra is gondoltam csináltam pár ellenőrző metódust, ami például jelez, ha rossz a felhasználó vagy egy esetleg a jelszó, regisztrációnál jelez, ha nem adtunk volna meg valami, vagy ha két jelszó nem egyezik.

Valamiért a jar fájlknál meg jegyezi a saját gépemen tárolt helyet és ezt minden indításnál meg kell adni melyik mappába, találja, az exe fájl, amit csináltam hozzá az kicsomagolja a programom mappáját és abba bele illesztettem e két fájlt.

Nem jöttem rá, hogy-hogy is lehetne megoldani, hogy ez alapból benne legyen ez a két jar fájl és ne kelljen ezzel időt húzni, hogy másképpen indulásnál ezt be importáljuk a programba. A 8.0.2-es NetBeans verziónál furcsa mód nem tapasztalom ezt a hibát itt is szintén az gépemet érzékeli, hogy az E:\-ről importáltam be a használt jar fájlokat, de lefut hiba üzenet nélkül.

A selejt táblázatomban sajnos nem írja ki teljesen a hosszú leírásnál a leírást, de ha a csúszkával ki húzzuk, akkor látszódik minden, hogy mit is írtunk oda. Ha nincs be kapcsolva a wamp server, akkor nem tudjuk futtatni a programot.

### ***3.5. Továbbifejlesztési lehetőségek***

Amit szerettem volna még csinálni a felhasználó váltás gomb. Ha netalán több felhasználó használná a programot egy gépről,

## Ardai Márk 2/14SL Szak Dolgozat Dokumentáció

akkor csak rá nyomna a felhasználó váltás gombra, amit el helyeznék a selejt listán valahol, a célja az lenne rá nyomunk és egy kis ablak le gördül, hogy be írjuk a felhasználónevet és a jelszót. Ugyan úgy lenne rajta egy bejelentkező gomb és váltana egy Selejt Lista nézetet, amit például a másik felhasználó vett fel az jelenne meg.

Gondoltam még olyanra is, hogy minden selejthez esetleg lehetne képed be fűzni egy adott mappából. Ez azért is lehet hasznos ha estleg kidobásra került valami áru, akkor hozzá lehet adni a képet, hogy mi is volt a probléma vele, vagy kapunk egy terméket, amivel nem vagyunk elégedettek nem olyan a minősége és szeretnénk a főnöknek megmutatni. Arra gondolok, hogy van egy írás mező, amire létre hozok, egy úgy nevezet csatolt gombot ahonnan be hívom, a képet mondjuk, és viszem fel az adatbázisba. Lehetne csinálni rá egy eljárást, ha rá kattintok, a programba ki nagyítja, vagy fel kínálja a lehetőséget, hogy küldés, módosítás vagy törlés. Mit értek küldés alatt? Ha mondjuk, egy nagy cégnél dolgozol, akkor mondjuk a vezetőségnek is el lehet küldeni a képet, vagy a beszállítónak is elküldheted, amivel nem vagy elégedet, lejárt rohadt a termék.

A gomboknak valamilyen ikont adni esetleg bele valami képet. Ez mondjuk ilyen kisebb módosításnak jó pofa. Igényes háttér adni, amihez passzolnak a gombok kinézetei. Mind háttér mind formailag. Sajnos most csak egy alapszín van benne, de ha használatba lenne, akkor sokkal jobban érezné magát az ember szerintem, ha igényesen van össze rakva a háttér is a ki írások és a gombok.

A selejt táblának is gondoltam egy ki egészítést, háttérszín valami dizájn, keresési funkció, amit ugyan úgy az adatbázisból kérne le, gondolok, itt mennyi saláta lett ki dobva csak rá keress fel dobja az értéket és a táblába megmutatja hány darab van. Így egyszerűbb lenne a használnak mondjuk havi leltárt írni, vagy hetit attól függ. Ehhez is gondoltam írni, egy kisebb programot, hogy ki számolja a havi selejtet összesítve, termékekre lebontva. Ugyan úgy a selejt lista részen lenne valahol elhelyezve és

## **Ardai Márk 2/14SL Szak Dolgozat Dokumentáció**

tovább vinne egy másik felületre és be írnánk a kívánt terméket, a mennyiséget mondjuk, ami kidobásra szánt és lenne egy gomb, ami kiszámolná. Vagy egy átlagot számolna melyik hónapba volt több selejt egy termékből vagy esetleg összesítve.

Sajnos nem sikerült egy dolgot bele raknom, hogy ugye több felhasználó használja és a termékek, amiket felvesznek, az ne azonos legyen, hanem minden felhasználónak külön, külön legyen a felvett selejt. Ezt úgy tudtam volna elérni utána néztem, hogy létre hoztam volna egy változót és meg adtam volna neki a felhasználó azonosítóját (id-ját), és amit arra az azonosítóra vettek fel azt jelenítse meg. Mert úgy gondoltam, nálunk is az étterembe úgy van, hogy egy ember be regisztrál, és azt használják többen. De ha mondjuk adunk különböző jogokat a programhoz, használatra nézve lesz egy admin akinek mindenhez joga van, lesz egy moderátor, akinek úgy mond, joga van mindenhez, de mégis kevesebb, mint az adminnak és lenne a sima felhasználó, akinek csak alap jogok vannak meg. Ezt mondjuk, aki a programozó a cégnél az állítaná, be kinek milyen joga lenne, az adatbázisban, úgy lehetne elérni, hogy a regisztrálásnál lenne egy ilyen opció, hogy választhat a regisztráló, admin, moderátor, felhasználó közül.

Lehetne egy olyan opció is, hogy a felhasználóknak lehetne úgy nevezet profilképet be állítani. Ez is ilyen mellékes kis fejlesztés.

Gondoltam olyanra is, ha regisztrálunk, akkor a jelszó mezőre kattintva megjelenne egykis infó buborék, ami le írná mi is kéne, a jelszóba milyen hosszú legyen, speciális karakter, kis betű nagybetű, számokat tartalmazzon.

Egy nyomtató gombra gondoltam még a selejt listánál, ha esetleg ki szeretnének nyomtatni, össze lehetne kötni a nyomtató programmal, hogy hozza be és be állíthassuk, mit szeretnének ki nyomtatni. Könnyebb egyeztetni, ha megvan papír formában is, úgy gondolom.

## **Ardai Márk 2/14SL Szak Dolgozat Dokumentáció**

Nem tudom, hogy lehet-e ilyet javában de gondoltam még olyanra is hogy lehessen nagyítani a selejt listán, lenne egy gomb, ami ki nagyítja így elkerülni, hogy nem látunk valamit. A selejt lista felület elé rakná és ide szintén egy kicsinyítés gombot.

Gondoltam egy regisztrálás megerősítés e-mail címre, de szerintem ez már weblapfejlesztéssel lenne keverve. Kapunk egy e-mailt, amire kattintva megerősítenénk a regisztrációt, és ha rá kattintunk, ki írná, hogy sikeres regisztráció. Ebből ki indulva egy elfelejtett jelszó gombra is gondoltam, rá kattintva kérhetnénk egy új jelszót, de ez is szintén weblappal össze kötött kapnánk egy oldalt ahol meg adhatjuk az új jelszavunkat, és az adatbázisba frissítené az adatunkat, mert lehetne azt is, hogy a programozó kitöröli az adatbázisból az adott felhasználót, aki elfelejtette a jelszavát, de akkor el veszik az összes selejt, amit felvett.

Gondoltam egy kategorizálás gombra is, külön megnyitná a zöldségeket, vagy a hús termékeket, estleg más termékeket doboz stb. ezzel is egyszerűsíteni a keresést. Ugyan úgy a selejt lista felületen el helyezve a gombot.

Szerettem volna csinálni egy olyat a selejt táblába, hogy hónapra szét szedi a felvett selejteket, időtől függetlenül, az átláthatóság egyszerűsítésért.

## **IV. Felhasználói Dokumentáció**

### **4.1. A program általános specifikációja**

A program tulajdonképpen 3 fő ablakból áll, és 4 osztályból. Amikor elindítjuk a programot egy bejelentkező felülettel találkozunk ezt, egy regisztráció követ, majd ha be regisztráltunk, akkor kezdődhet el az adatfelvétel, módosítása, törlése. A bejelentkezéshez regisztráláshoz szükségünk van egy külső adatbázisra, hogy ez megtörténhessen, én itt Localhost/phpmyadmin oldal segítségével dolgoztam ahhoz, hogy ezt elérjem szükségem volt egy programra aminek Wamp

## Ardai Márk 2/14SL Szak Dolgozat Dokumentáció

a neve. Ez a program segít nekem, hogy adatot tudjak felvenni, vagy épp felhasználót tudjak regisztrálni. Mint írtam van, 4 fő osztályom ezekre azért van szükségem, hogy a programba tudjak dolgozni. Az első osztályom, egy csatlakozási osztály, amit azért csináltam, hogy elérjem az adatbázisom a programmal. Mert ha nincs, kapcsolat a kettő között akármit írhatnék bele nem fog történni semmi, nem tudok adatot (selejtet) felvenni, nem tudok be regisztrálni, tehát elengedhetetlen, hogy legyen egy külső tárhelyünk, ha így tetszik.

Ez után következik a másik fő osztályom ahhoz, hogy tudjak példányosítani az adatbázisomból, mire gondolok, itt az adatbázisba megvannak, adva mi szerint veszi fel az adatot, szám esetleg, szöveg változó, vagy mint például itt dátum. Tehát létrehoztam egy contact nevű osztályt, ahol az adatbázisomból felvettem az ott lévő változókat, hogy amikor be viszem, az adatot tudjak, rá hivatkozni mit kötök össze mivel.

A harmadik fő osztályom nem jöhet létre, ha nincs a második fő osztály. Itt arra gondolok, hogy van egy csatlakozásom a program és az adatbázis között erre szolgál az első fő osztály majd a második osztály, a kapott adatokat egyeztetni az adatbázisban lévő adatokkal és annak függvényében veszi fel, vagy módosítja, esetleg töröli, ha a két megnevezés, amit én adtam, és amire hivatkozok, az adatbázisból megegyezik. Persze ezeket ellenőriztetni kell, hogy minden helyesen van-e megadva. Egy selejtet vagy egy raktári készletet akkor tudunk felvenni, ha a bevitt adatok egyeznek, mert ha fel cserélünk valami adatot, akkor hibát dob a program. Úgy hiszem ez a legfőbb kapcsolat, ami nagyon fontos, hogy működjön a programnál, az adatbázis és a program közötti kapcsolat.

A negyedik és egyben utolsó osztályom a megjelenésre szolgál. Ha felvesszünk valami adatot a táblába ennek köszönhetően jeleníthetem meg, vagy ha rá kattintok és az adatok vissza fejtem a beírni kívánt helyre ennek a módszernek köszönhető. A táblát tulajdonképpen e-nélkül is lehetne használni, de jobbnak láttam, hogy ha ezt is kézzel adom meg hány oszlop és hány

sorom legyen, vagy mi legyen ki írva az első oszlopba, mert a Java ezt már be állította megának, és így egyszerűbb vele dolgozni, ha megunk állítjuk be, hogy mit melyik sorba szeretnénk látni, vagy melyik oszlopba.

A három fő felületem, ezekre épül rá, hogy mit hova írok és az hova kerül. Bejelentkezésnél az adatbázisból a felhasználó táblát használja. Regisztrációnál szintén ezt a táblát használja, és az adat felvételhez egy selejt\_felvetele nevű táblát használ. Itt ugye mindent meg kell adni feltétel nélkül nevünk, születési évünk, egy kitalált felhasználónév hozzá egy jelszó, és már használható is a program. Mindenről tájékoztatást add a program, hogy sikerülte, vagy nem amit szeretünk volna tenni benne, gondolok itt a helyes regisztrációra és az adatok felvételére (selejt, esetleg raktári termék).

### **4.2. Rendszerkövetelmények**

A Netbeans program futtatásához a minimális rendszerkövetelmény (utána néztem az adatoknak).

Windows Vista SP1/ Windows 7 alatt:

Processzor: 800MHz Intel Pentium III vagy azonos

Memória: 512 MB

Hely használat: 750 MB szabad hely kapacitás

A maximális rendszerkövetelményt is megtaláltam bár kicsit soknak tartom az igényt hozzá.

Windows 7 Professional/ Windows 8/ Windows 8.2

Processzor: Intel Core i5 vagy azonos

Memória: 2 GB (32-bit), 4 GB (64-bit)

Hely használat: 1.5 GB szabad hely kapacitás

## Ardai Márk 2/14SL Szak Dolgozat Dokumentáció

Az én konfigurációm:

Windows 10

Processzor: Intel I3 (8th Generáció)

Memória: 8GB (64-bit)

Tárhely: 120GB SSD és 1TB HDD

A programom futását feladatkezelőben néztem meg, és nagyon alacsony volt a használsága mind a Netbeansnek mind a programnak. .NET Framework 4 alatt van használva, DirectX 12, Java verzió jre 1.8.0\_202 (legfrissebb), jdk 12 (szintén a legfrissebb). A program adatbázis használatához szükség van a Wamp program használatához. Erről nem találtam információt rendszerkövetelményként csak annyit, hogy minden Windows fajtán lefut, 43 megabyte a telepítője és 457 megabyte-t foglal.

> Java(TM) Platform SE binary	11,7%	3,0%	0%	0%	0,4%	GPU 0 - 3D	Mérsékelt
> Microsoft Word	0%	2,5%	0%	0%	0%		Nagyon alacsony
> NetBeans IDE	0%	22,8%	0%	0%	0%		Nagyon alacsony

Itt az látható, hogy a program a processzort 11,7 %-ban használja ki, ez miután betöltött mindent rögtön le ment 0%-ra.

A memóriát 3,0%-ban használja ki ez maradt így végig, amíg futtattam.

A következő a lemez kihasználtság az 0%-ról indult és körülbelül ott is maradt.

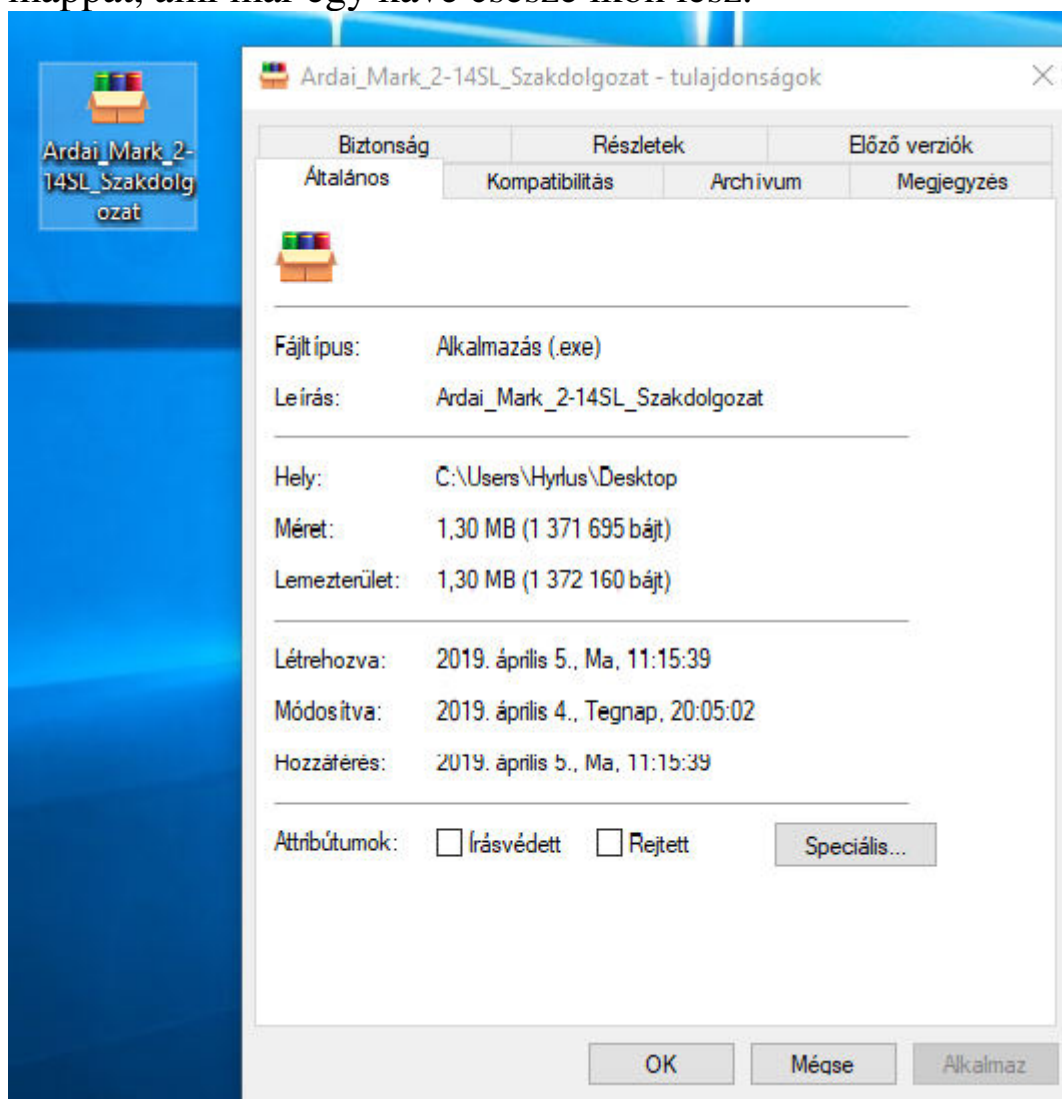
A videokártyát 0,4% mozgatta, meg ami szintén vissza ment 0%-ra.

Az energiafogyasztás itt a képen látva mérsékeltre becsülte, de mikor betöltött mindent akkor azt írta nagyon alacsony.

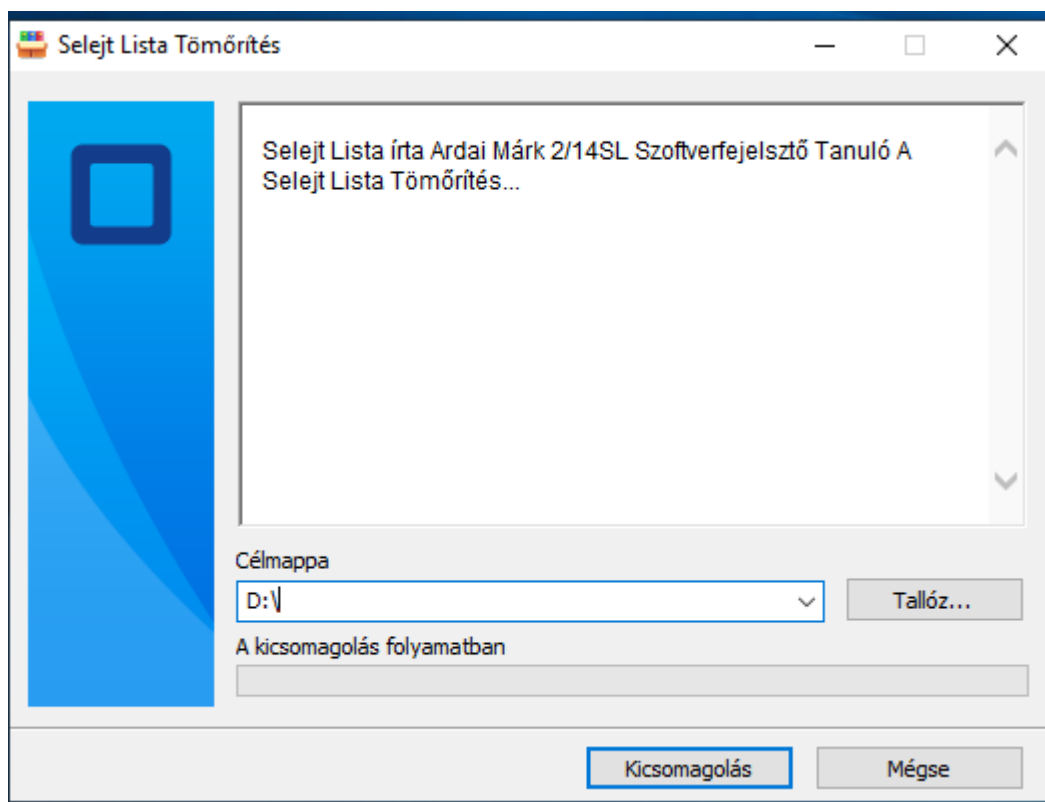
Ami nagyobb ki használtságot élvez az a Netbeans program, de ott is a memóriát használja, ki néha a processzort mikor be tölt valamilyen programot, és ugye a videokártyát is mikor betölt.

### 4.3. A program telepítésének és konfigurálásának a leírása

A programomat archiváltam Winrarral így tudtam megoldani a telepítése részt. Kapunk egy exe fájlt, ami ki tömöríti nekünk a megadott helyre a program könyvtárát benne a program adatait, majd ezt a Netbeansen belül kell futtatnunk, ezt úgy tudjuk, hogy megnyitjuk a projectet, ezt úgy tudjuk elérni, hogy Netbeansben bel felül File, Open project és ott kiválasztjuk a mappát, ami már egy kávé csésze ikon lesz.



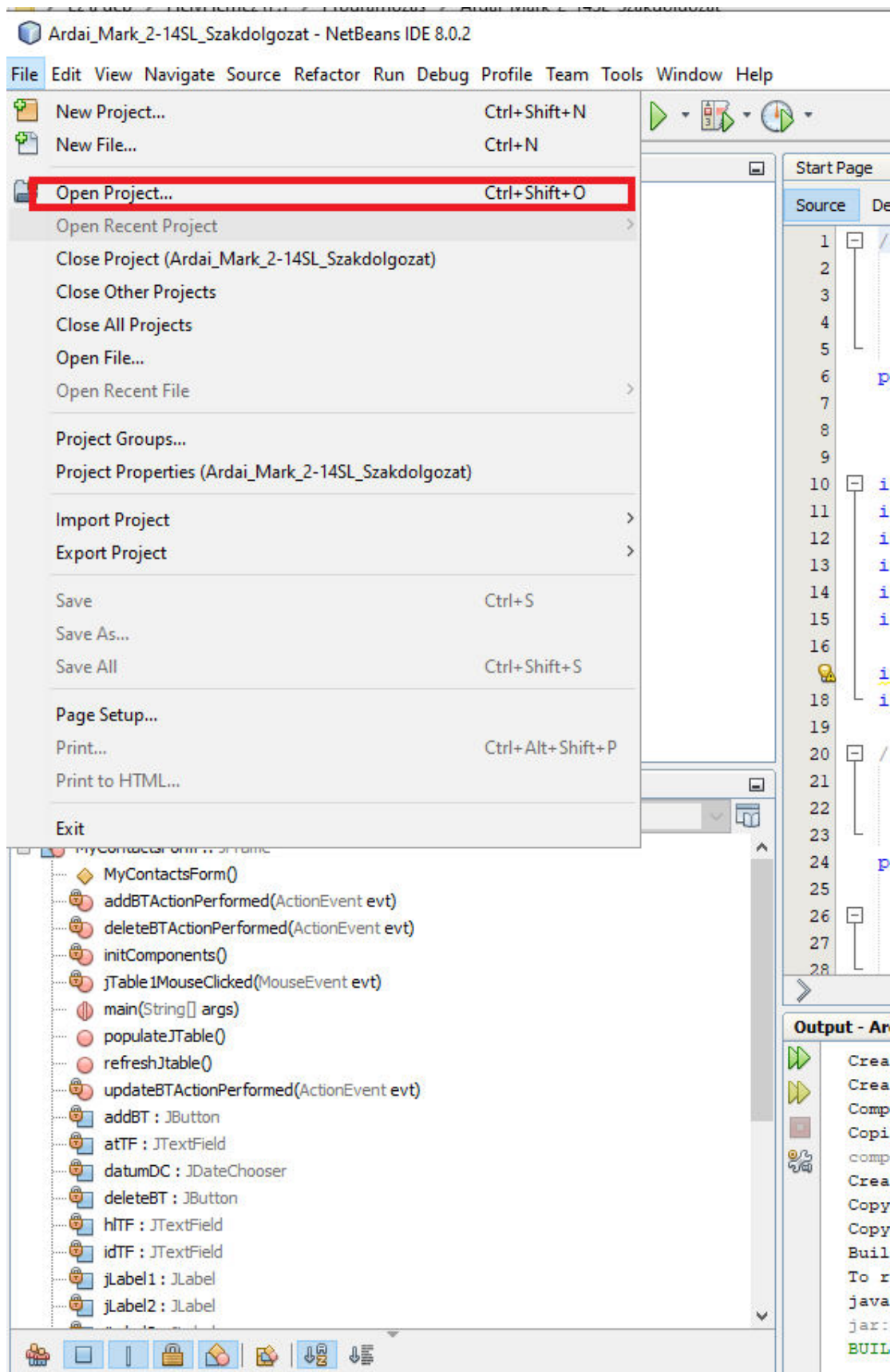




A mappa tartalmazza nekünk mysql-connector nevezetű java fájlt és a jcalendar fájlt, amit hozzá kell adnunk a projektünkhez (8.0.2 Netbeans alatt ezzel nem volt probléma). Amit úgy tudunk, hogy van egy olyan könyvtár a Projectnél, hogy Libraries erre szintén jobb klikk Add jar/folder és itt bele megyünk a szakdolgozatom mappába ahova be másoltam ezt a két jar fájlt, ezeket hozzá adjuk és nem fog hibát írni.

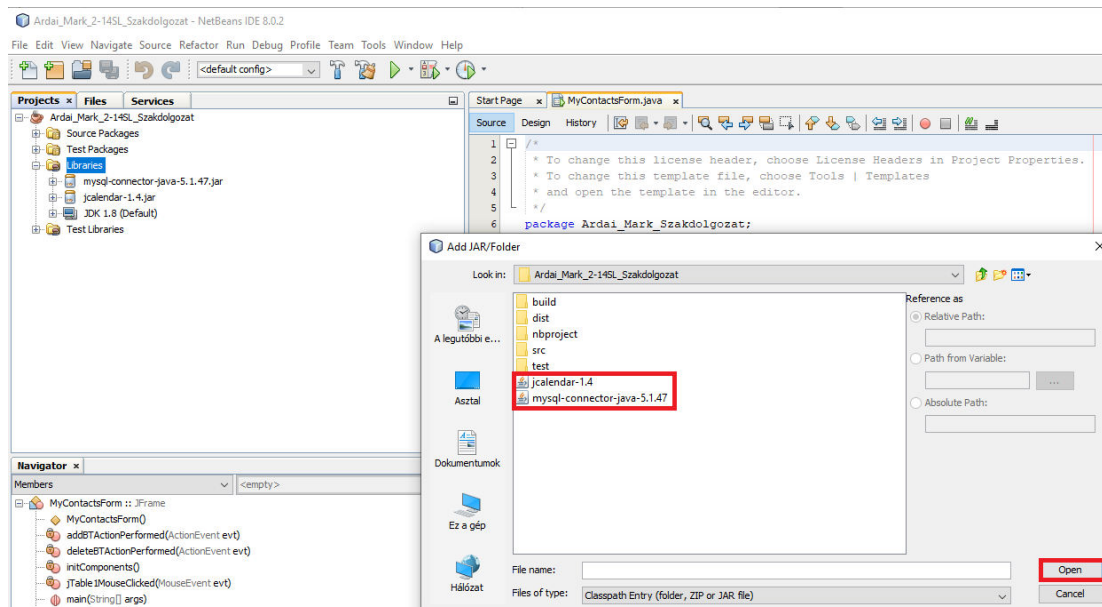
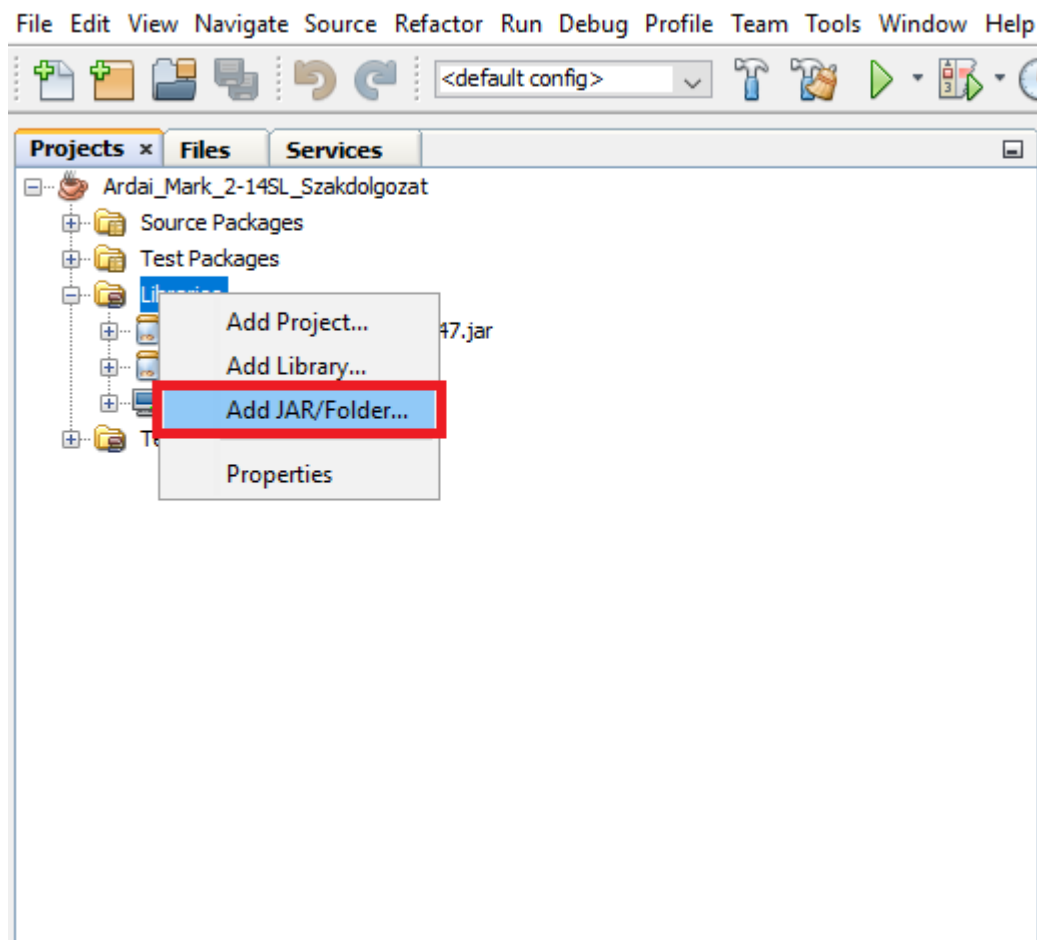
build	2019. 04. 05. 11:07	Fájlmappa	
dist	2019. 04. 05. 11:07	Fájlmappa	
nbproject	2019. 03. 28. 17:12	Fájlmappa	
src	2019. 04. 04. 19:52	Fájlmappa	
test	2018. 11. 04. 16:10	Fájlmappa	
build	2019. 04. 04. 19:52	XML dokumentum	4 KB
jcalendar-1.4	2013. 03. 21. 13:32	Executable Jar File	162 KB
manifest.mf	2018. 11. 04. 16:07	MF fájl	1 KB
mysql-connector-java-5.1.47	2018. 08. 07. 8:59	Executable Jar File	984 KB
szakdogasql	2019. 03. 04. 18:47	SQL fájl	3 KB

## Ardai Márk 2/14SL Szak Dolgozat Dokumentáció



# Ardai Márk 2/14SL Szak Dolgozat Dokumentáció

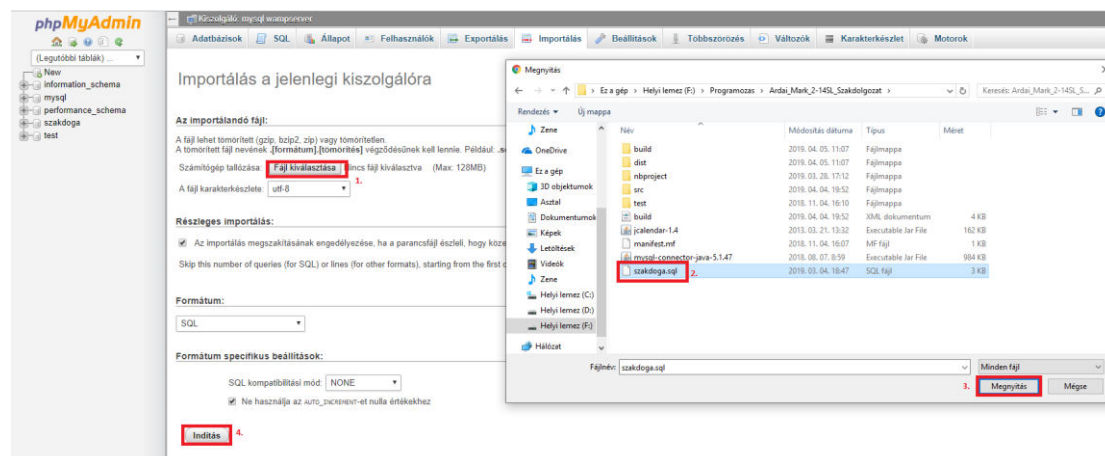
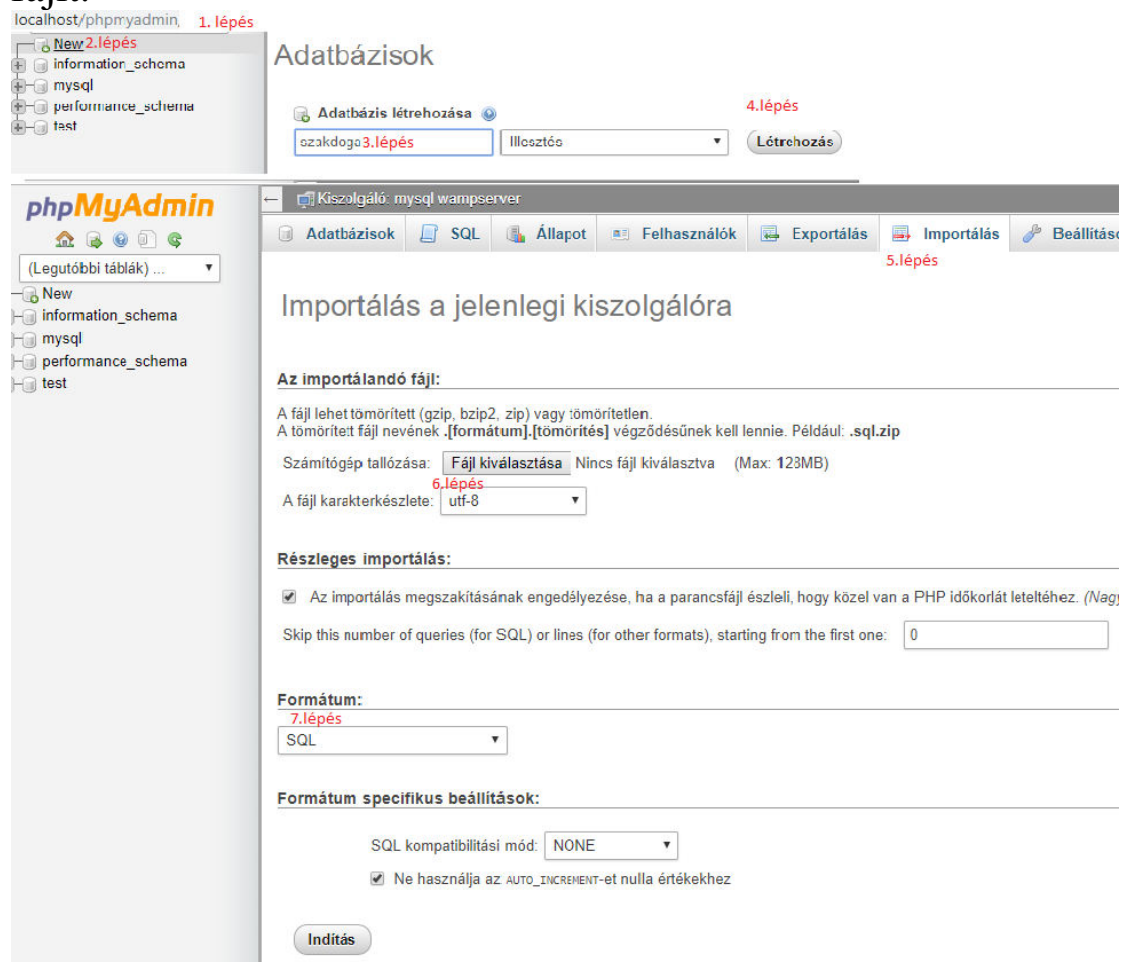
Ardai\_Mark\_2-14SL\_Szakdolgozat - NetBeans IDE 8.0.2



Ezen kívül szükségünk van a WampServer nevezetű programra amit Google segítségével le lehet tölteni én a 2.5 verziót használok, de igazából teljesen mindegy melyiket töltjük le. Ha letöltöttük a Netbeanst és a WampServer és benne van a programunk, nincs más teendőnk, mint felkeresnünk a

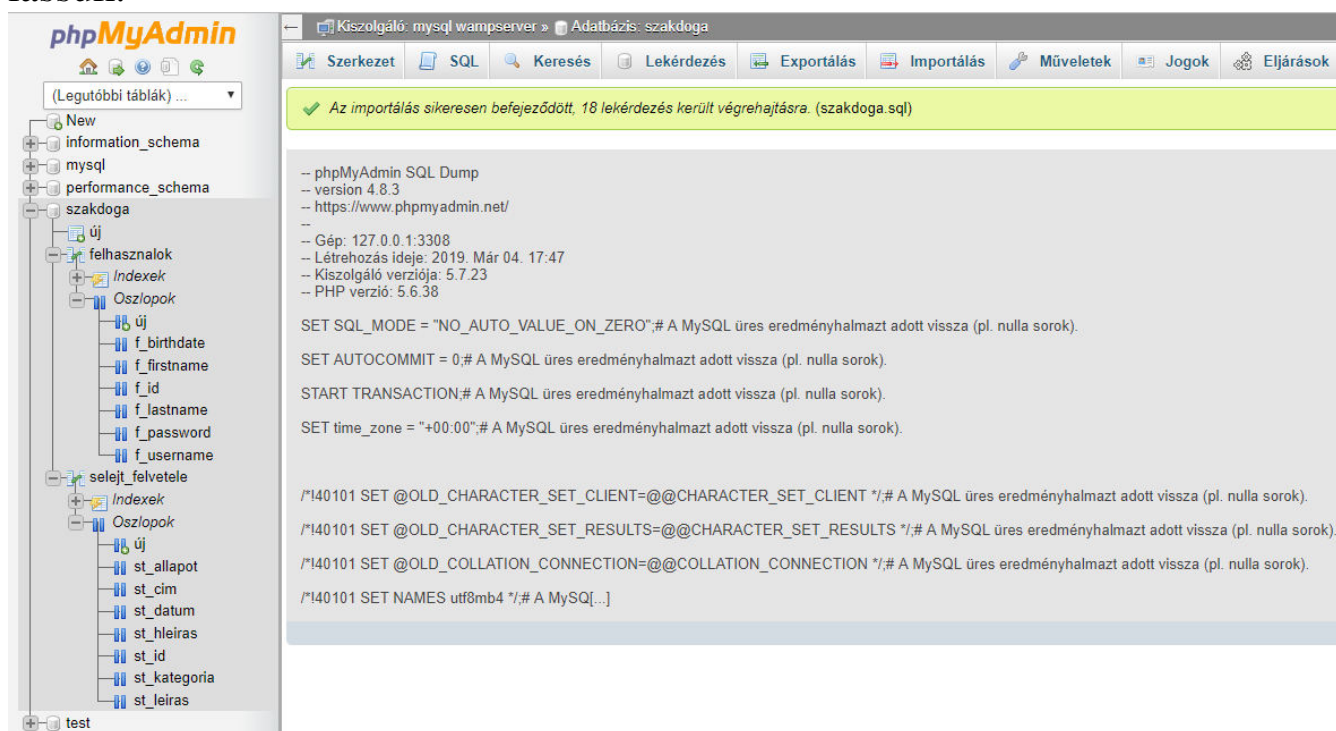
# Ardai Márk 2/14SL Szak Dolgozat Dokumentáció

Localhost/phpmyadmin nevezetű weboldalt ott létre hozni egy üres szakdoga nevezetű táblát, majd erre kattintva be importálni, szintén a szakdolgozatom mappába talált mentet szakdoga.sql fájlt.



## Ardai Márk 2/14SL Szak Dolgozat Dokumentáció

Ha mindent jól csináltunk a fájl be importálásakor ezt kell, lássuk:



### 4.4. A program használatának a részletes leírása

A programot egy exe fájlnak csináltam, meg ami a tömörített változatot bontja ki mappába. Ehhez szükségünk van egy Netbeans program én a 8.0.2-őt használom, de szerintem mindegyik megfelel hozzá. Továbbá szükségünk van egy WampServer programhoz, hogy a Localhost/phpmyadmin MySQL adatbázist tudjuk használni, mert szükségünk van rá a program használatakor. Ide menti fel az adatok, ezalatt a be regisztrált felhasználónevet értem a jelszót és az adatok, amire használjuk később a programot selejt lista vagy esetleg raktár kezelés céljából.

Az adatbázisban szerepel két tábla, ami tartozik a programhoz, egy felhasználó tábla, amit id-val egy vezetéknév, keresztnév, felhasználónév, jelszó születési dátum adatokat vár. A selejt táblába egy id-t, egy címet (megnevezést), leírást, hosszú leírást, kategóriát, állapotot, és egy dátumot. Ezt az adatbázist kötöttem össze a programommal, egy bizonyos mysql-connector java fájl segítségével. A dátumhoz szükségem volt egy jcalendar

nevezetű Java fájlra, hogy egy naptár kinézetű, dizájnt kapjak, amikor rányomok a dátum bevitelre.

Az első ablak, ami megjelenik, ha elindítjuk a programot egy bejelentkező felület. Itt egy felhasználó nevet és egy jelszót vár a felhasználó táblába felvett adatokból. Ezt, ha tovább megyünk alul, amit csináltam feliratot a Regisztrációs felületre, ott tudjuk meg adni keresztnév, vezetéknév, felhasználónév, jelszó és dátum mező által. Ezt a felhasználók táblából kéri, le mikor be szeretnénk jelentkezni a Login felületen. Az adat egyeztetés írtam különböző ellenőrzéseket, a Bejelentkezésnél, ha nincs, beregisztrálva ilyen felhasználó nem tud belépni, vagy ha helytelen a jelszó szintén fel dob egy hiba ablakot. A Regisztrálásnál szintén írtam egy ellenőrzést, ha a vezetéknév vagy a keresztnév közül valamelyik hiányos, nem tud regisztrálni, ha a két jelszó nem egyezik, szintén nem tud be regisztrálni. Ha mindent rendben talál csináltam egy üzenet dobozt, hogy Sikeres Regisztráció. Ezek után van lent szintén egy írás a Register felület alján, ami megkönnyíti, a bejelentkezés felületre való vissza lépést. Ha minden sikerült jól be írunk, amit regisztrálásnál adtunk, akkor tovább visz minket a Selejt Lista felületre, ha nem akkor szintén egy hiba üzenet doboz jelenik meg Valami Hibás írással.

A selejt lista felületen már a másik adatbázissal dolgozik, itt vár egy megnevezést, egy leírást, hosszú leírást, kategóriát, állapotot, és egy dátumot. Erre a részre csináltam egy contact nevezetű osztályt, amibe le példányosítottam egy selejtet, ami vár egy id-t egy megnevezést, egy kategóriát, leírást, hosszú leírást, állapotot, és egy dátumot, ahogy az adatbázisban is szerepel. Ezt az osztályt meghívtam egy másik osztályba contactQuery-be ahol létre hoztam egy listát, ami az adatbázisban lévő adatokat össze köti a programomban lévőekkel, hogy tudjak ki válogatni (select eljárás), tudjak feltölteni adatokat (insert eljárás), tudjak módosítani a selejteken (update eljárás) és, hogy tudjak törölni selejtet (delete eljárás). Ezeket az eljárásokat a gombokra hívtam meg, hogy amit a be írt mezőkre meg adtunk azt ő jegyezze meg és a különböző eljárásokkal dolgozzon vele. Ha módosítjuk az adatot, akkor,



amit be írtunk arra a helyre változzon, ha új selejtet veszünk fel, akkor töltse fel az adatbázisba, ha törölni akarunk valamit, akkor azt az adatbázisból ki töröli (id alapján). A tábla, amibe a selejt megjelenik szintén egy osztályra alapul a myModelre. Ebbe az osztályba meg adtam, hogy hány oszlop legyen az adatbázisban lévő adatok által, és hogy hány sor ezt természetesen ahány adat van az adatbázisban. A magasságát, szélességét már a selejt lista felületen írtam meg a kód részén, és hogy mit tartalmaz adott oszlop azt is én adtam meg, üressé tettem a táblát, mert alaphoz meg ad nekünk egy stílust, ha hozzá adjuk a dizájn részhez. Rá tudunk kattintani a selejtekre amelyiket, szeretnénk módosítani, és ha rá kattintunk automatikusan minden értéket, amit adtunk behelyezi a beírni kívánt mezőkre mindent az alapján, amit adtunk. Ez csupán némi segítség arra, hogy mit is szeretnénk át írni, esetleg törölni.

## V. Összegzés

Összegezve az egészet nagyon, jó tanuló példa volt ez a program, sokkal többet tudtam meg a Java programozás nyelvről, a jövőben is szeretnék ezzel foglalkozni vagy esetleg hasonló projekten. A program írás befejeztével elég sok dolgot értettem meg hamarabb a tanulásba, sokkal több dolognak néztem utána a programozás kapcsán, remélem sikerült teljesítenem az elvártnak megfelelően, én úgy érzem magamban igen, bár van, mit még írni a programon van még mit csinálni rajta, de ettől szép ez a szakma, hogy jönnek a jobbnál jobb gondolatok, legalábbis szerintem ez így van. Van még hova fejlődöm, és remélem a jövőben lesz is rá alkalmam.

Sok segítséget adott nekem a Youtube és innen is 1BestCsharp felhasználó.

<https://www.youtube.com/channel/UCS3W5vFugqi6QcsoAIHcMpw>

<https://stackoverflow.com/questions/tagged/java>