**LibraryAdmin**

**könyvkölcsönző rendszer és webes felület**

**Fejleszői dokumentáció**

**1.0-ás verzió: 2022. március**

**Készítette: Frantal Attila - Csányi Marianna**

Tartalomjegyzék

[**A LibraryAdmin projekt indításának a célja:** 4](#_Toc98421060)

[**A szoftver licence:** 4](#_Toc98421061)

[**A szoftver kialakításának és céljának fő irányelvei:** 4](#_Toc98421062)

[**Stage RUP 1 ( Követelmények )** 5](#_Toc98421063)

[**Követelményfeltárás:** 5](#_Toc98421064)

[**Nem funkcionális követelmények:** 5](#_Toc98421065)

[**Funkcionális követelmények:** 5](#_Toc98421066)

[**Megszorítások:** 5](#_Toc98421067)

[**Stage RUP 2 ( Tervezés )** 6](#_Toc98421068)

[**UML tervek:** 6](#_Toc98421069)

[**MVC model:** 6](#_Toc98421070)

[**Control csomag:** 6](#_Toc98421071)

[**Model csomag:** 7](#_Toc98421072)

[**Adatbázis:** 7](#_Toc98421073)

[**View csomag:** 8](#_Toc98421074)

[**GUI terv:** 8](#_Toc98421075)

[**Stage RUP 3 ( Implementálás )** 9](#_Toc98421076)

[**ConnectDatabase osztály** 9](#_Toc98421077)

[**LoginFrame osztály (extends javax.swing.JFrame)** 10](#_Toc98421078)

[**MainFrame osztály (extends javax.swing.JFrame)** 10](#_Toc98421079)

[**SqlRunner osztály** 11](#_Toc98421080)

[**Client osztály** 13](#_Toc98421081)

[**Book osztály** 13](#_Toc98421082)

[**Tran osztály** 14](#_Toc98421083)

[**DatabaseControl osztály** 14](#_Toc98421084)

[**GuiControl** 16](#_Toc98421085)

[**MainControl oszály** 17](#_Toc98421086)

[**Stage RUP 4** 17](#_Toc98421087)

[**Stage RUP5** 17](#_Toc98421088)

[**Stage RUP6** 17](#_Toc98421089)

[**A LibraryUser webes felület létrehozásának célja:** 18](#_Toc98421090)

[**Stage RUP 1 ( Követelmények )** 18](#_Toc98421091)

[**Követelményfeltárás:** 18](#_Toc98421092)

[**Nem funkcionális követelmények:** 18](#_Toc98421093)

[**Funkcionális követelmények:** 18](#_Toc98421094)

[**Stage RUP 2 ( Tervezés )** 18](#_Toc98421095)

[**UML tervek:** 18](#_Toc98421096)

[**MVC model:** 18](#_Toc98421097)

[**Adatbázis:** 19](#_Toc98421098)

[**GUI terv:** 19](#_Toc98421099)

[**Stage RUP 4** 25](#_Toc98421100)

[**Stage RUP5** 25](#_Toc98421101)

[**Stage RUP6** 25](#_Toc98421102)

**LibraryAdmin**

**könyvkölcsönző rendszer**

# **A LibraryAdmin projekt indításának a célja:**

Egy könyvtárban a könyvek kölcsönzésének (és visszavételének) folyamatát hivatott modellezni a program. Nyilvántartjuk a könyvtártagok adatait és könyv tartozásait, a könyvek adatait és a könyvtárban lévő mennyiségüket. Tároljuk a kölcsönzés és visszavétel tényét időbélyeggel, és a tranzakcióhoz tartozó tagi és könyv adatokkal. Külön táblában megjelenítjük a könyvtartozással rendelkezőket.

# **A szoftver licence:**

- A program kizárólag a készítő tulajdona, tovább értékesítés nem engedélyezett.

- A program nem megvásárolható.

- Kizárólag a készítő iskolai órán elsajátított tudásának ellenőrzésére szolgál.

- Próbaverzió, javaslat esetén a készítő módosíthatja.

# **A szoftver kialakításának és céljának fő irányelvei:**

1. A szoftver kizárólag szabad felhasználású programokkal készült. (JAVA JDK, NetBeans, XAMPP, MariaDB, DIA)
2. A program futtatható a következő operációs rendszereken: Windows 7,8,XP,10, MAC IOS, Linux.
3. A program JAVA 14.0.2 verzióval készült.
4. Az adatbázis kezelő MariaDB 2.7.2.
5. A program RUP fejlesztési módszer szerint készült, a fejlesztés iteratív és inkrementális.
6. Alapvető funkciók:
7. Könyvtártagok adatainak nyilvántartása (tagi azonosító, név, telefon, e-mail, könyvtartozás).
8. Új tag felvétele, illetve meglévő tag adatainak módosítása.
9. Könyvek adatainak nyilvántartása (könyv azonosító, szerző, cím, készlet).
10. Új könyv felvétele, illetve meglévő könyv adatainak módosítása.
11. Tranzakciók (kölcsönzés/visszavétel) rögzítése időbélyeggel, tag és könyv adatokkal.
12. A könyvtartozással rendelkező tagok külön megjelenítése.

# **Stage RUP 1 ( Követelmények )**

## **Követelményfeltárás:**

### **Nem funkcionális követelmények:**

A programnak Windows operációs rendszer alatt futnia kell.

Gépigény:

Windows 10

4 GB RAM

2 GHz processzor

Háttérprogramok:

Java futtatókörnyezet, MariaDB adatbáziskezelő.

A fejlesztés NetBeans IDE 12.2 használatával történik, Java 14.0.2 és MariaDB 2.7.2 verziószámú komponensekkel.

A program Java nyelven készült.

### **Funkcionális követelmények:**

A program használatához autentikáció szükséges.

Az adatokat SQL adatbázis tárolja.

Az adatokat grafikus felületen kell megjeleníteni.

Adatok nyilvántartása:

-könyvtártagok azonosítója

-könyvtártagok neve

-telefonszáma

-E-mail címe

-egyenleg (könyvtartozás)

-könyvek adatainak tárolása (kód, szerző, cím, készlet)

-tranzakciók nyilvántartása

új adatok felvétele

meglévő adatok módosítása

## **Megszorítások:**

Autentikáció → adatbázisból, jelszó nem olvasható formátumban tárolva

A fő ablak vezérlő gombjai inaktívak míg az autentikáció nem sikeres.

Felvett tagot és könyvet nem törlünk az adatbázisból.

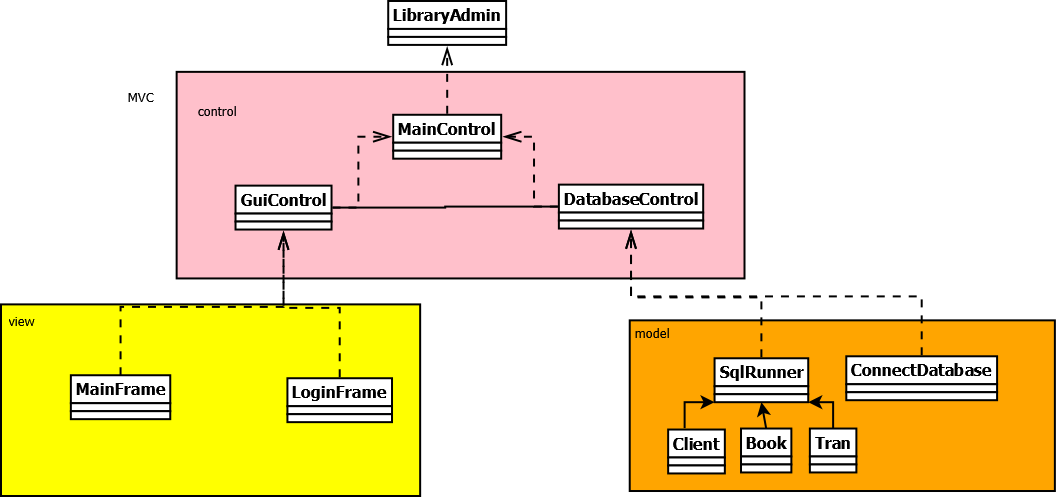
Adatbázis neve: library.

Autentikációs jelszót titkosítva kell tárolni.

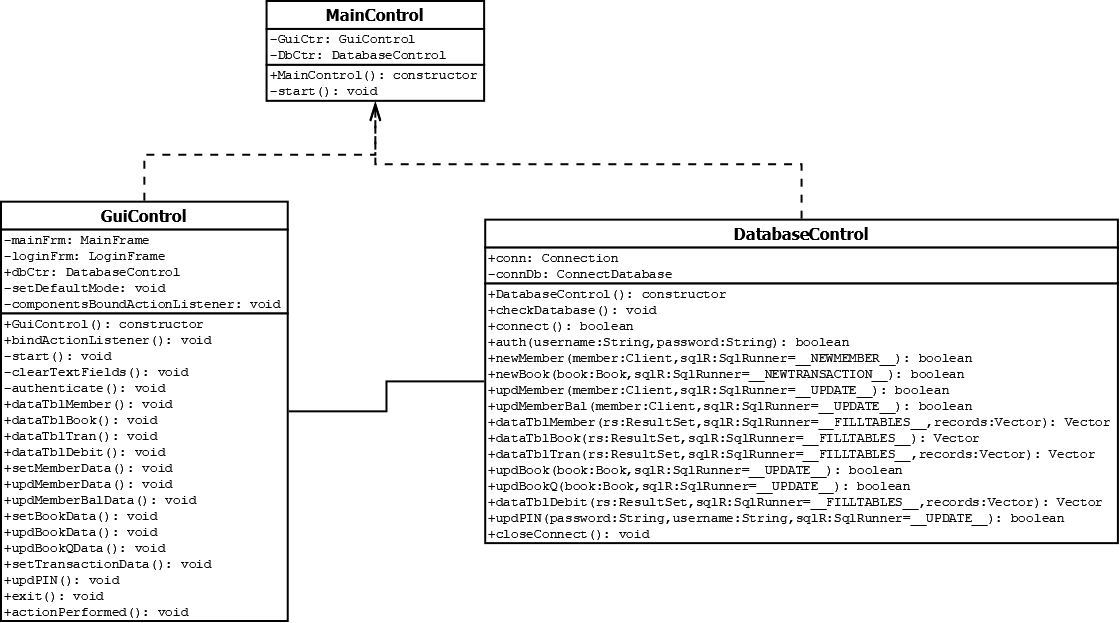
# **Stage RUP 2 ( Tervezés )**

## **UML tervek:**

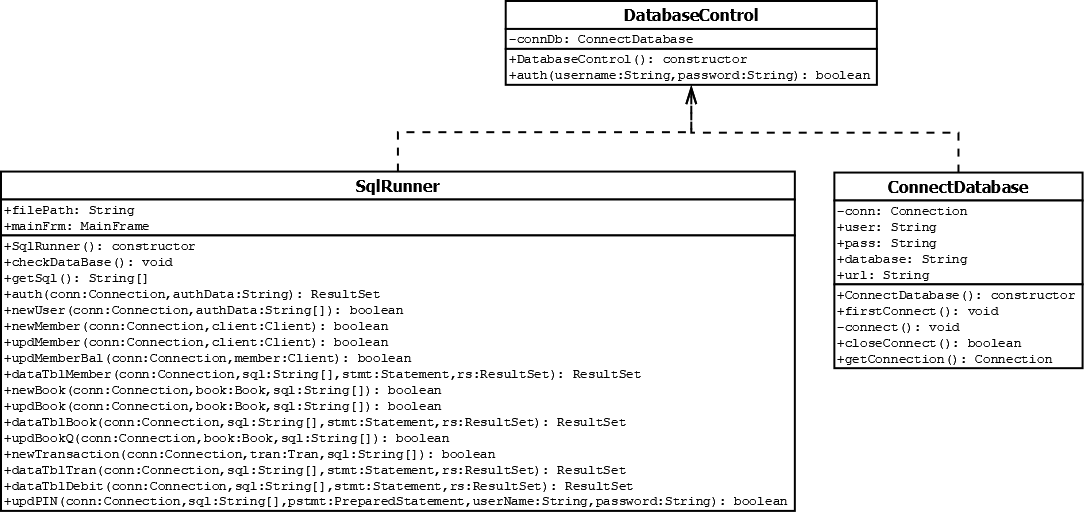
### **MVC model:**



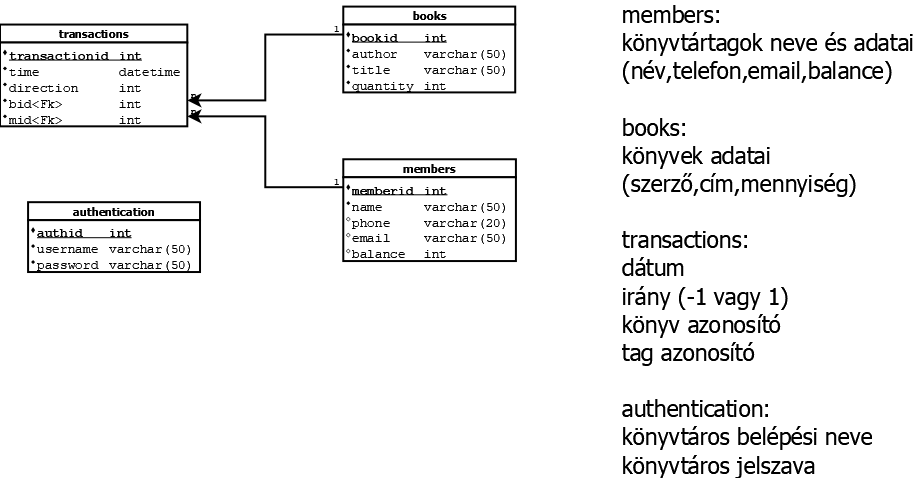
### **Control csomag:**



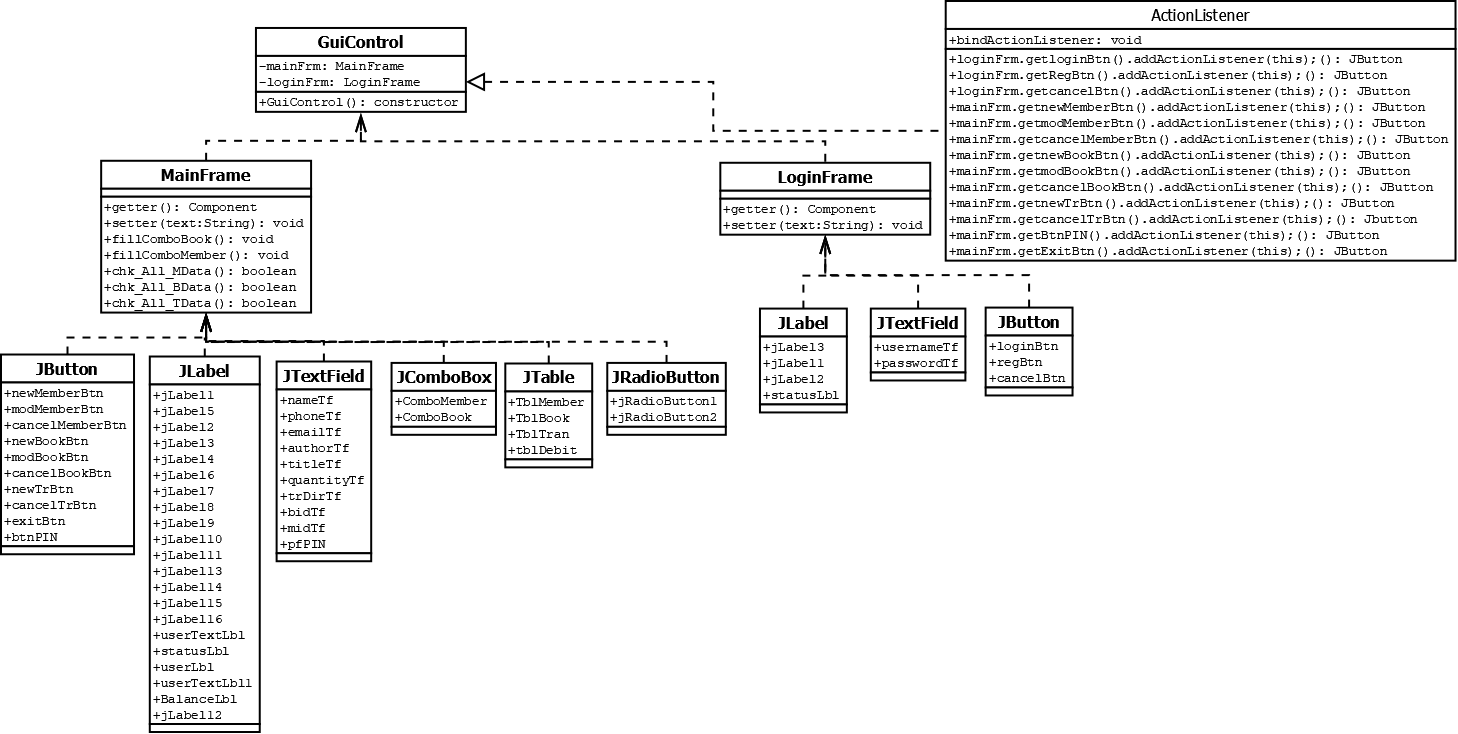
### **Model csomag:**



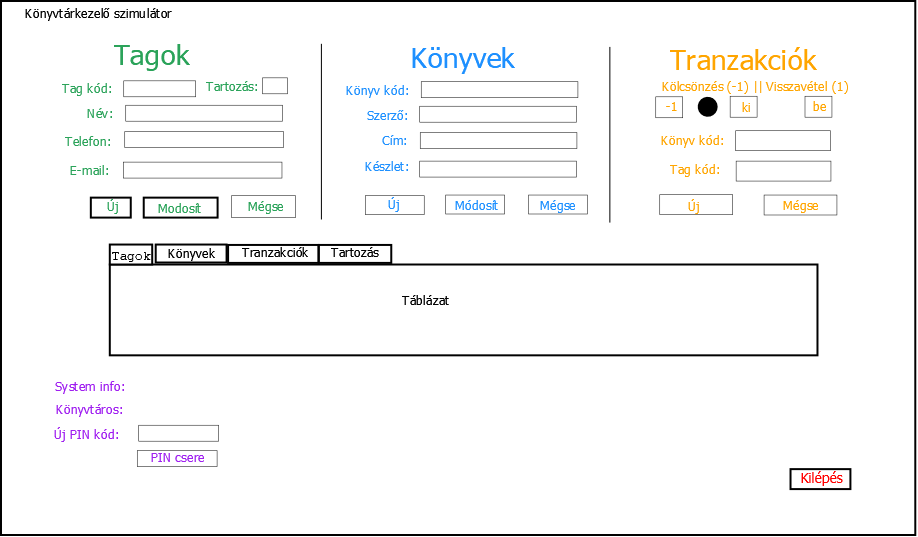
### **Adatbázis:**



### **View csomag:**



## **GUI terv:**



# **Stage RUP 3 ( Implementálás )**

## **ConnectDatabase osztály**

Importált csomagok:

java.sql.Connection, java.sql.DriverManager, java.sql.SQLException

Osztályváltozók:

private Connection conn ( Adatbázis kapcsolat )

private String user ( Felhasználónév az adatbázis kapcsolathoz )

private String pass ( Jelszó az adatbázis kapcsolathoz )

private String database ( Adatbázis neve a kapcsolódáshoz )

private String url ( Adatbázis kapcsolat url -je )

Metódusok:

ConnectDatabase() (konstruktor)

Bejövő paraméter: nincs

Beállítja az osztályváltozók alapértelmezett értékét.

firstConnect()

Attributum: public

Bejövő paraméter: nincs

Felépít egy általános kapcsolatot az adatbázis szerverrel az url, az user és a pass változók értéke szerint. Kivételkezelés: SQLException, catch ágon java hibaüzenet kiíratása debug módban.

Visszatérési érték: nincs

connect()

Attributum: public

Bejövő paraméter: nincs

Felépít egy adatbázishoz kötődő kapcsolatot az url, a database, az user, és a pass változók értéke szerint.

A kapcsolatot a conn változóban tárolja. Kivételkezelés: SQLException, catch ágon java hibaüzenet kiíratása debug módban.

Visszatérési érték: nincs

closeConnect()

Attributum: public

Bejövő paraméter: nincs

Lezárja az aktuálisan felépített és tárolt adatbázis kapcsolatot. Kivételkezelés: SQLException, catch ágon java hibaüzenet kiíratása debug módban.

Visszatérési érték: boolean

getConnection()

Attributum: public

Bejövő paraméter: nincs

Visszaadja a conn változó értékét.

Visszatérési érték: Connection

## **LoginFrame osztály (extends javax.swing.JFrame)**

Importált csomagok:

import javax.swing.JButton;

import javax.swing.JLabel;

import javax.swing.JPasswordField;

import javax.swing.JTextField;

Konstruktor: initComponents()

A bejelentkezési ablakot/űrlapot jeleníti meg.

Metódusok:

Az űrlap elemeinek getterei-setterei

## **MainFrame osztály (extends javax.swing.JFrame)**

Importált csomagok:

import javax.swing.JButton;

import javax.swing.JLabel;

import javax.swing.JTextField;

import java.sql.ResultSet;

import java.sql.PreparedStatement;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import javax.swing.JPasswordField;

import javax.swing.JTable;

Konstrunktor:

initComponents();

fillComboBook();

fillComboMember();

A program fő ablaka/űrlapja.

Metódusok:

Az űrlap elemeinek getterei-setterei

fillComboMember()

Attribútum: public void

Bemenő paraméterek: connection, sql

Visszatérési érték: combobox feltöltése memberid-kkal

SQLException, catch ágon java hibaüzenet kiíratása debug módban

fillComboBook()

Attribútum: public void

Bemenő paraméterek: connection, sql

Visszatérési érték: combobox feltöltése bookid-kkal

SQLException, catch ágon java hibaüzenet kiíratása debug módban

## **SqlRunner osztály**

Importált csomagok:

import java.io.IOException;

import java.nio.file.Files;

import java.nio.file.Path;

import java.sql.ResultSet;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.sql.Connection;

import java.sql.Statement;

import java.sql.DatabaseMetaData;

Osztályváltozók:

private String filePath; (futtatandó SQL file-ok elérési útvonala)

Konstruktor: this.filePath = "sql/" + filename + ".sql";

Bejövő paraméter: filename (az SQL file neve)

Visszatérési érték: az SQL file az elérési útvonalával együtt

Metódusok:

checkDataBase(Connection conn)

Bejövő paraméter: Connection, a ConnectDatabase osztály getConnection() metódusából

Ellenőrzi az adatbázis létezését a metaadatokból, hogy létezik-e a books tábla, ha nem létezik, akkor létrehozza az egész adatbázist.

Kivételkezelés: SQLException, catch ágon java hibaüzenet kiíratása debug módban.

Visszatérési érték: nincs

getSql()

Attributum: public

Bejövő paraméter: nincs

SQL file-t ;-kel tagolva feldarabolja

Visszatérési érték: a darabolás után az i-dik indexen lévő SQL utasítás

IOException, catch ágon java hibaüzenet kiíratása debug módban, és hibaüzenet a felhasználónak „Hiba a beolvasás során.”

auth(Connection conn, String[] authData)

Attributum: public ResultSet

Bejövő paraméter: adatbázis kapcsolat, azonosítási adatok

Azonosítási adatokat kéri le (felhasználónév, jelszó)

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Visszatérési érték: ResultSet (felhasználónév, jelszó)

newUser(Connection conn, String[] authData)

Attributum: public boolean

Bejövő paraméter: adatbázis kapcsolat, azonosítási adatok

Az új felhasználó bejelentkezési adatait kéri le (felhasználónév, jelszó)

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Visszatérési érték: boolean

newMember(Connection conn, Client client)

Attributum: public boolean

Bejövő paraméter: adatbáziskapcsolat, Client típusú objektum

Új könyvtártag adait szúrja be a members táblába

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Attributum: public boolean

updMember(Connection conn, Client client)

Attributum: public boolean

Bejövő paraméter: adatbáziskapcsolat, Client típusú objektum

Már létező könyvtártag adait módosítja a members táblában

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Attributum: public boolean

updMemberBal(Connection conn, Client member)

Attributum: public boolean

Bejövő paraméter: adatbáziskapcsolat, Client típusú objektum

Már létező könyvtártag tartozását módosítja a rögzített tranzakció alapján a members táblában

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Attributum: public boolean

dataTblMember (Connection conn)

Attributum: public ResultSet

Bejövő paraméter: adatbázis kapcsolat

A könyvtártag adatbázis táblát jeleníti meg az űrlapon

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Visszatérési érték: ResultSet (members tábla adatai)

newBook(Connection conn, Book book)

Attributum: public boolean

Bejövő paraméter: adatbáziskapcsolat, Book típusú objektum

Új könyv adait szúrja be a books táblába

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Attributum: public boolean

updBook(Connection conn, Book book)

Attributum: public boolean

Bejövő paraméter: adatbáziskapcsolat, Book típusú objektum

Már létező könyv adatait módosítja a books táblában

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Attributum: public boolean

dataTblBook (Connection conn)

Attributum: public ResultSet

Bejövő paraméter: adatbázis kapcsolat

A books adatbázis táblát jeleníti meg az űrlapon

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Visszatérési érték: ResultSet (books tábla adatai)

updBookQ(Connection conn, Book book)

Attributum: public boolean

Bejövő paraméter: adatbáziskapcsolat, Book típusú objektum

Már létező könyv mennyiségi adatát módosítja a books táblában

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Attributum: public boolean

newTransaction(Connection conn, Tran tran)

Attributum: public boolean

Bejövő paraméter: adatbáziskapcsolat, Tran típusú objektum

Egy könyv kiadását illetve visszvételét rögzíti mint tranzakciót

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Attributum: public boolean

dataTblTran (Connection conn)

Attributum: public ResultSet

Bejövő paraméter: adatbázis kapcsolat

A tranzakciók táblát jeleníti meg az űrlapon

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Visszatérési érték: ResultSet (transactions tábla adatai)

dataTblDebit (Connection conn)

Attributum: public ResultSet

Bejövő paraméter: adatbáziskapcsolat

A könyvtártagok tartozásait jeleníti meg táblázatban az űrlapon

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Visszatérési érték: ResultSet (tartozások adatai)

updPIN(Connection conn, String[] authData)

Attributum: public boolean

Bejövő paraméter: adatbáziskapcsolat, azonosítási adatok

A bejelentkezett könyvtáros jelszavát változtatja meg

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Attributum: public boolean

## **Client osztály**

Osztályváltozók:

private Integer memberid;

private String name;

private String phone;

private String email;

private Integer balance;

Metódusok:

Az osztályváltozók getterei és setterei

Attributum: public

## **Book osztály**

Osztályváltozók:

private Integer bookid;

private String author;

private String title;

private int quantity;

Metódusok:

Az osztályváltozók getterei és setterei

Attributum: public

## **Tran osztály**

Osztályváltozók:

private int direction;

private int bid;

private int mid;

Metódusok:

Az osztályváltozók getterei és setterei

Attributum: public

## **DatabaseControl osztály**

Importált csomagok:

import java.sql.Connection;

import model.ConnectDatabase;

import java.sql.ResultSet;

import model.SqlRunner;

import java.sql.SQLException;

import java.util.Vector;

import model.Book;

import model.Client;

import model.Tran;

osztályváltozók:

private Connection conn;

private ConnectDatabase connDb;

konstruktor: checkDatabase()

metódusok:

checkDatabase()

Attributum: public, void

Bejövő paraméter: nincs

Meghívja az SqlRunner osztály checkDataBase() metódusát

Visszatérési érték: nincs

connect()

Attributum: protected boolean

Bejövő paraméter: nincs

Meghívja a ConnectDatabase osztály getConnection() metódusát

Visszatérési érték: boolean

auth(String[] authData)

Attributum: protected int

Bejövő paraméter: felhasználónév, jelszó

Meghívja az SqlRunner osztály auth() metódusát

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Visszatérési érték: egész szám

newUser(String[] authData)

Attributum: protected boolean

Meghívja az SqlRunner osztály newUser() metódusát

Bejövő paraméter: felhasználónév, jelszó

Visszatérési érték: boolean

newMember(Client member)

Attributum: protected boolean

Meghívja az SqlRunner osztály newMember() metódusát

Bejövő paraméter: Client típusú objektum

Visszatérési érték: boolean

newBook(Book book)

Attributum: protected boolean

Meghívja az SqlRunner osztály newBook() metódusát

Bejövő paraméter: Book típusú objektum

Visszatérési érték: boolean

newTransaction(Tran tran)

Attributum: protected boolean

Meghívja az SqlRunner osztály newTransaction() metódusát

Bejövő paraméter: Tran típusú objektum

Visszatérési érték: boolean

updMember(Client member)

Attributum: protected boolean

Meghívja az SqlRunner osztály updMember() metódusát

Bejövő paraméter: Client típusú objektum

Visszatérési érték: boolean

updMemberBal(Client member)

Attributum: protected boolean

Meghívja az SqlRunner osztály updMemberBal() metódusát

Bejövő paraméter: Client típusú objektum

Visszatérési érték: boolean

updBook(Book book)

Attributum: protected boolean

Meghívja az SqlRunner osztály updBook() metódusát

Bejövő paraméter: Client típusú objektum

Visszatérési érték: boolean

updBookQ(Book book)

Attributum: protected boolean

Meghívja az SqlRunner osztály updBookQ() metódusát

Bejövő paraméter: Client típusú objektum

Visszatérési érték: boolean

dataTblMember()

Attributum: protected Vector<Vector<String>>

Meghívja az SqlRunner dataTblMember() metódusát

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Visszatérési érték: Vector<Vector<String>>

dataTblBook()

Attributum: protected Vector<Vector<String>>

Meghívja az SqlRunner dataTblBook() metódusát

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Visszatérési érték: Vector<Vector<String>>

dataTblTran()

Attributum: protected Vector<Vector<String>>

Meghívja az SqlRunner dataTblDebit() metódusát

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Visszatérési érték: Vector<Vector<String>>

dataTblDebit()

Attributum: protected Vector<Vector<String>>

Meghívja az SqlRunner dataTblTran() metódusát

SQLException, catch ágon java hibaüzenet kiíratása debug módban

Visszatérési érték: Vector<Vector<String>>

updPIN(String[] authData)

Attributum: protected boolean

Meghívja az SqlRunner osztály updPIN() metódusát

Bejövő paraméter: Client típusú objektum

Visszatérési érték: boolean

closeConnect()

Attributum: protected void

Bejövő paraméter: nincs

Visszatérési érték: nincs

Bazárja az élő adatbázis kapcsolatot

## **GuiControl**

Implementálja az ActionListener interfészt, az eseménykezelés helyben történik.

Mezők:

- private mainFrm ( Példányosított MainFrame objektum )

- private loginFrm ( Példányosított LoginFrame objektum )

- private dbCtr ( Példányosított DatabaseControl objektum )

Változók:

MainFrame mainFrm – a fő panel/ablak osztályának példányosítása

LoginFrame loginFrm – a bejelentkező ablak osztályának példányosítása

Konstruktor:

Bejövő paraméter: DatabaseControl dbCtr tárolva a dbCtr mezőben.

ComponentsBoundActionListener() metódus hívása.

ComponentsBoundActionListener:

Nincs bejövő paraméter.

A formokon elhelyezett komponensek kötése az ActionListener interfészhez.

A komponensekhez getter és setter metódusokkal lehet hozzáférni.

SetDefaultMode:

Nincs bejövő paraméter.

Példányosítja a MainFrame osztályt és láthatóvá teszi. Példányosítja a LoginFrame osztályt és

láthatóvá teszi a MainFrame felett.

Elindítja az adatbázis kapcsolat felépítését a DatabaseControl connect() metódusának meghívásával.

A connect() metódus visszatér egy boolean típussal melyet kiértékel és az eredmény függvényében

a MainForm felületén a statusLbl komponensen megjeleníti a kapcsolódás eredményét.

A newBtn, modifyBtn, cancelBtn, pinBtn gombok inaktívvá tétele.

## **MainControl oszály**

Osztályváltozók:

private DatabaseControl dbCtr;

private GuiControl guiCtr;

konstruktor: start()

metódusok:

start()

Attributum: private void

Nincs bejövő paraméter.

Példányosítja a DatabaseControl és a GuiControl osztályt.

# **Stage RUP 4**

Tesztelés

# **Stage RUP5**

Átadás

# **Stage RUP6**

Karbantartás

**LibraryUser**

**Webes felület**

# **A LibraryUser webes felület létrehozásának célja:**

A LibraryAdmin könyvkölcsönző program adatbázisához kapcsolódva regisztrált tagoknak lehetősége legyen a LibraryUser webes felületen távoli eléréssel lekérdezni a könyvtartozásaik adatait és ellenőrizni a kölcsönzési, illetve visszaviteli határidőt.

# **Stage RUP 1 ( Követelmények )**

## **Követelményfeltárás:**

### **Nem funkcionális követelmények:**

A programnak Windows operációs rendszer alatt futnia kell.

Gépigény:

Windows 10

4 GB RAM

2 GHz processzor

Háttérprogramok:

Java futtatókörnyezet, MariaDB adatbáziskezelő.

Molbiltelefon

A fejlesztés NetBeans IDE 12.2 használatával történik, PHP, CSS és MariaDB 2.7.2 verziószámú komponensekkel.

### **Funkcionális követelmények:**

A program a LibraryAdmin programhoz kapcsolódik.

Az adatokat a LibraryAdmin program SQL adatbázisa tárolja.

Az adatokat grafikus felületen kell megjeleníteni.

Adatok nyilvántartása:

-könyvtártagok azonosítója

-könyvtártagok neve

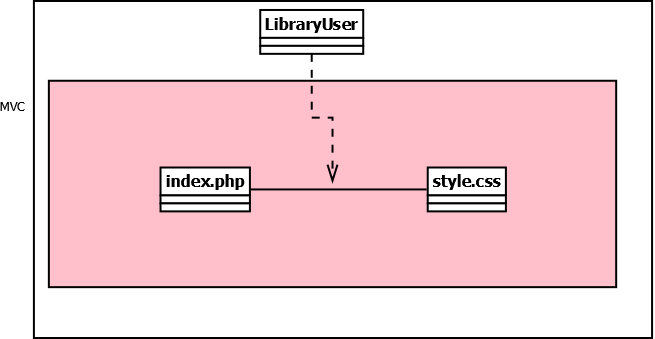
-egyenleg (könyvtartozás)

-könyvek adatainak tárolása (kód, szerző, cím)

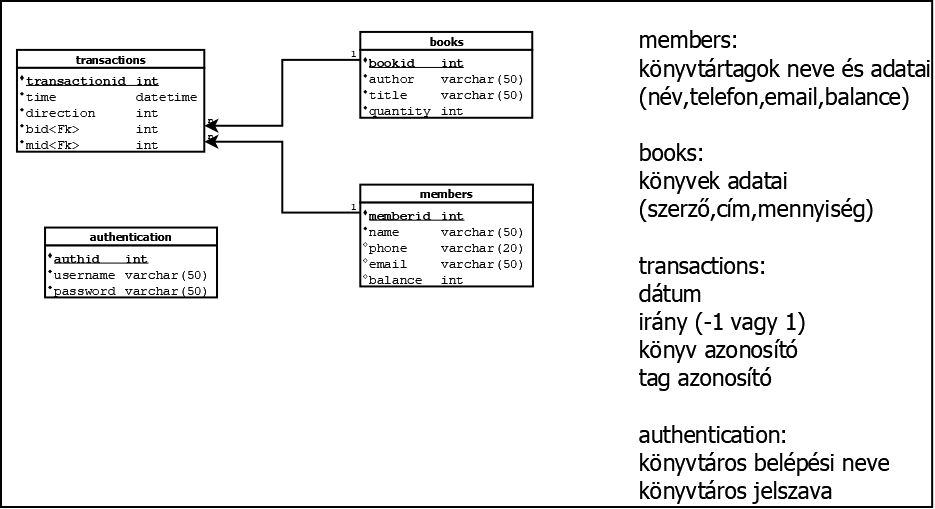
# **Stage RUP 2 ( Tervezés )**

## **UML tervek:**

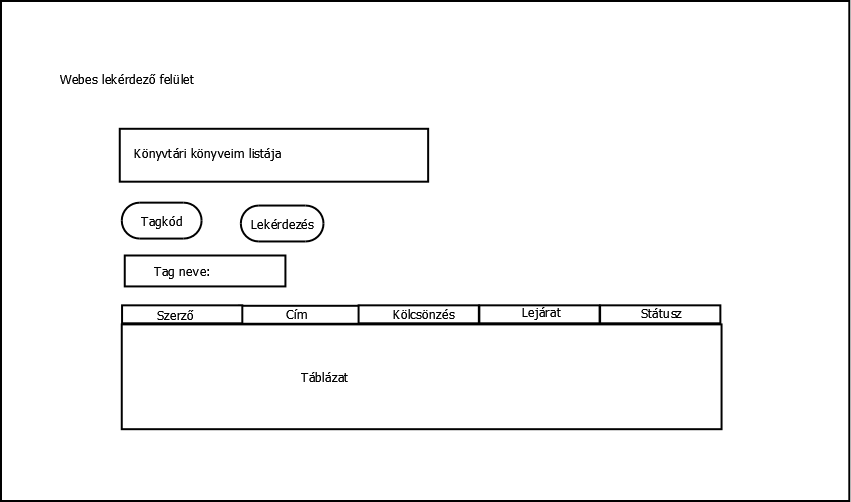
### **MVC model:**



### **Adatbázis:**



## **GUI terv:**



**Stage RUP 3 ( Implementálás )**

**index.php**

<?php

// http://localhost/php/LibraryUser/index.php

echo <<<EOT

<link rel="stylesheet" href="style.css">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<body>

<div id="container">

EOT;

echo <<<EOT

<form method="get">

<label id="cim">Könyvtári könyveim listája</label><br><br>

<input type='integer' name='member' value='' placeholder='Tag kód'>

<button id='gombQ' type='submit' name='btnQuery'><span>Lekérdezés</span></button>

</form>

EOT;

if(isset($\_GET['btnQuery'])) {

resTable();

}

function resTable(){

// mysqli\_connect(host, username, password, dbname, port, socket)

$conn = mysqli\_connect('localhost','root','','library');

if(!$conn){

echo "Hiba! A kapcsolódás sikertelen!";

}

mysqli\_set\_charset($conn, "utf-8");

$mid = $\_GET['member'];

$sql = "

select

b.author

,b.title

,MAX(DATE(t.time)) as takeout

,DATE\_ADD(MAX(DATE(t.time)),INTERVAL 30 day) as due

,CASE WHEN CURRENT\_DATE()>DATE\_ADD(MAX(DATE(t.time)),INTERVAL 30 day) THEN 'Lejárt!' ELSE 'Élő' END as status

from

transactions as t

join members as m

on t.mid = m.memberid

join books as b

on t.bid = b.bookid

where

t.mid = $mid

group by

1,2

having

sum(t.direction) < 0

order by

4

;

";

$sql2 = "

select

m.name

from

members as m

where

m.memberid = $mid

;

";

$res2 = mysqli\_query($conn, $sql2);

if($mid==''){} //if member\_id field is not filled then do nothing

else if (mysqli\_num\_rows($res2)>0) {

while($sor2 = mysqli\_fetch\_assoc($res2)) {

echo "<label>" . "Tag neve: " . $sor2['name'] . "</label>";

}

}

$res = mysqli\_query($conn, $sql);

if($mid==''){ //if member\_id field is not filled then write this message

echo "<div id='uzenet'>Adj meg tag kódot!</div>";

}

else if (mysqli\_num\_rows($res2)==0) { //if member\_id is not exists then write this message

echo "<div id='uzenet'>Nincs ilyen tag!</div>";

}

else if (mysqli\_num\_rows($res)>0) {

echo "<table border='1'>\n";

echo "<th>Szerző</th><th>Cím</th><th>Kölcsönzés</th><th>Lejárat</th><th>Státusz</th>\n";

while($sor = mysqli\_fetch\_assoc($res)) {

echo "\n<tr>\n";

echo "<td>" . $sor['author'] . "</td>";

echo "<td>" . $sor['title'] . "</td>";

echo "<td>" . $sor['takeout'] . "</td>";

echo "<td>" . $sor['due'] . "</td>";

echo "<td>" . $sor['status'] . "</td>";

echo "\n</tr>\n";

}

echo "</table>";

} else {

echo "<div id='uzenet'>Nincs könyvtartozásod.</div>";

}

mysqli\_close($conn);

}

echo "</div>";

echo "</body>";

**style.css**

body {

padding: 0;

margin: 0;

display: flex;

/\* align-items: center;\*/

justify-content: center;

min-height: 100vh;

background-image: linear-gradient(135deg, #0042D9 0%, #0093FF 60%, #0078FF 100%);

}

#container{

margin-top: 40px;

}

#cim{

font-size: 50px;

font-weight: 600;

color: #ADD8E6;

text-align: center;

}

table{

border-collapse: collapse;

background-color: #FFBC00;

margin: 10px;

}

table td{

padding: 5px;

font-size: 30px;

/\*background-image: linear-gradient(135deg, #0042D9 20%, #0093FF 60%, #0078FF 100%);\*/

background-color: #0093FF;

border-color: blue;

color: #ADD8E6;

font-family: 'Trebuchet MS', sans-serif;

}

table th{

padding: 5px;

background-color: blue; /\*#FFE900;\*/

color: #ADD8E6;

font-size: 30px;

font-family: 'Trebuchet MS', sans-serif;

}

button {

cursor: pointer;

position: relative;

padding: 1rem 4rem;

border-radius: 3.75rem;

line-height: 2.5rem;

font-size: 2rem;

font-weight: 600;

border: 1px solid #012880;

background-image: linear-gradient(-180deg, #FF89D6 0%, #C01F9E 100%);

box-shadow: 0 1rem 1.25rem 0 rgba(22,75,195,0.50),

0 -0.25rem 1.5rem rgba(110, 15, 155, 1) inset,

0 0.75rem 0.5rem rgba(255,255,255, 0.4) inset,

0 0.25rem 0.5rem 0 rgba(180, 70, 207, 1) inset;

margin-left: 50px;

}

button span {

color: transparent;

background-image: linear-gradient(0deg, #EE82DA 0%, #FEFAFD 100%);

-webkit-background-clip: text;

background-clip: text;

filter: drop-shadow(0 2px 2px hsla(290, 100%, 20%, 1));

}

button:active {

line-height: 1.5rem;

font-size: 1.5rem;

}

button::before {

content: "";

display: block;

height: 0.25rem;

position: absolute;

top: 0.5rem;

left: 50%;

transform: translateX(-50%);

width: calc(100% - 7.5rem);

background: #fff;

border-radius: 100%;

opacity: 0.7;

background-image: linear-gradient(-270deg, rgba(255,255,255,0.00) 0%, #FFFFFF 20%, #FFFFFF 80%, rgba(255,255,255,0.00) 100%);

}

button::after {

content: "";

display: block;

height: 0.25rem;

position: absolute;

bottom: 0.75rem;

left: 50%;

transform: translateX(-50%);

width: calc(100% - 7.5rem);

background: #fff;

border-radius: 100%;

filter: blur(1px);

opacity: 0.05;

background-image: linear-gradient(-270deg, rgba(255,255,255,0.00) 0%, #FFFFFF 20%, #FFFFFF 80%, rgba(255,255,255,0.00) 100%);

}

input{

height: 4rem;

width: 14rem;

font-size: 2rem;

font-weight: 600;

border-radius: 3.75rem;

background-image: linear-gradient(-180deg, #FF89D6 0%, #C01F9E 100%);

border: 1px solid #012880;

padding: 2.2rem 1rem;

color: black;

/\*background-image: linear-gradient(0deg, #EE82DA 0%, #FEFAFD 100%);\*/

-webkit-background-clip: text;

background-clip: text;

filter: drop-shadow(0 2px 2px hsla(290, 100%, 20%, 1));

}

::placeholder {

color: #ADD8E6;

font-family: 'Trebuchet MS', sans-serif;

background-image: linear-gradient(0deg, #EE82DA 0%, #FEFAFD 100%);

-webkit-background-clip: text;

background-clip: text;

filter: drop-shadow(0 2px 2px hsla(290, 100%, 20%, 1));

opacity: 1; /\* Firefox \*/

}

label{

padding: 5px;

margin: 10px;

font-size: 30px;

font-weight: 600;

color: #ADD8E6;

font-family: 'Trebuchet MS', sans-serif;

}

#uzenet{

margin: 10px;

padding: 5px;

font-size: 30px;

background-color: #0093FF;

border-color: blue;

color: #ADD8E6;

font-family: 'Trebuchet MS', sans-serif;

}

# **Stage RUP 4**

Tesztelés

# **Stage RUP5**

Átadás

# **Stage RUP6**

Karbantartás