

Slovenská technická univerzita

Fakulta informatiky a informačných technológií

Ilkovičova 3, 812 19 Bratislava

Databázové systémy

Zadanie č. 1 a č. 2

Jozef Melicherčík

Cvičiaci: Ing. Ondrej Kachman

Študijný odbor: Informatika

Ročník: 2. Bc

Obsah

Obsah.....	2
1. Znenie zadania.....	4
1. Špecifikácia scenárov	5
1.1 Vytvorenie nového záznamu	5
1.1.1 Registrácia nového zákazníka	5
1.2 Aktualizácia existujúceho záznamu	5
1.2.1 Zmena hesla používateľa.....	5
1.3 Vymazanie záznamu.....	6
1.3.1 Vymazať používateľa systému	6
1.4 Zobrazenie prehľadu viacerých záznamov	6
1.4.1 Zobrazenie vozidiel	7
1.5 Zobrazenie konkrétneho záznamu	7
1.5.1 Pozrieť profil	7
1.6 Filtrovanie záznamov spĺňajúcich určité kritéria.....	7
1.6.1 Zobrazenie podľa stavu objednávok.....	7
2. Diagramy	8
2.1 Logický dátový model.....	8
2.2 Fyzický dátový model	9
3. Implementácia	10
3.1 Pripojenie k databáze.....	10
3.2 Implementácia scenárov	11
3.2.1 Zmena hesla používateľa.....	11
3.2.2 Zobrazenie vozidiel	11
3.3 Neošetrené chybové stavy	11

4.	II. Iterácia	12
4.1	Špecifikácia scenárov	12
4.1.1	Zobrazenie úkonov vykonaných používateľom v systéme.....	12
4.1.2	Implementácia scenáru „Zobrazenie úkonov“	12
5.	Zhodnotenie	14
6.	Zhodnotenie II. Iterácie	14

1. Znenie zadania

Vo vami zvolenom prostredí vytvorte databázovú aplikáciu, **ktorá komplexne rieši minimálne 6 scenárov** vo vami zvolenej doméne. Presný rozsah a konkretizáciu scenárov si dohodnete s Vaším cvičiacim na cvičení. Aplikáciu vytvoríte v dvoch iteráciách. V prvej iterácii, postavenej nad relačnou databázou, musí aplikácia realizovať tieto všeobecné scenáre:

- Vytvorenie nového záznamu,
- Aktualizácia existujúceho záznamu,
- Vymazanie záznamu,
- Zobrazenie prehľadu viacerých záznamov (spolu vybranou základnou štatistikou),
- Zobrazenie konkrétneho záznamu,
- Filtrovanie záznamov spĺňajúcich určité kritériá zadané používateľom.

Aplikácia môže mať konzolové alebo grafické rozhranie. Je dôležité aby scenáre boli realizované realisticky - teda aby aplikácia (a teda aj jej používateľské rozhranie) naozaj poskytovala časť funkcionality tak, ako by ju očakával zákazník v danej doméne.

Scenáre, ktoré menia dáta musia byť realizované **s použitím transakcií** a aspoň jeden z nich musí zahŕňať **prácu s viacerými tabuľkami** (typicky vytvorenie záznamu a naviazanie cudzieho kľúča).

V druhej iterácii do aplikácie pridáte min. 1 scenár postavený na nerelačnej databáze Redis alebo Elasticsearch (dohoda s cvičiacim na inom type nerelačnej db je samozrejme možná). Konkrétny scenár si dohodnete s vaším cvičiacim v závislosti od použitej databázy a domény vašej aplikácie (napr. štatistiky o interakciách s jednotlivými záznamami aplikácie v Redise alebo vyhľadavávanie záznamov cez Elasticsearch).

Bez odovzdanej (teda cvičiacim akceptovanej) prvej iterácie nie je možné odovzdať druhú.

Pre získanie zápočtu je potrebné odovzdať (a cvičiaci musí akceptovať minimálnu úroveň kvality) obidve iterácie projektu.

1. Špecifikácia scenárov

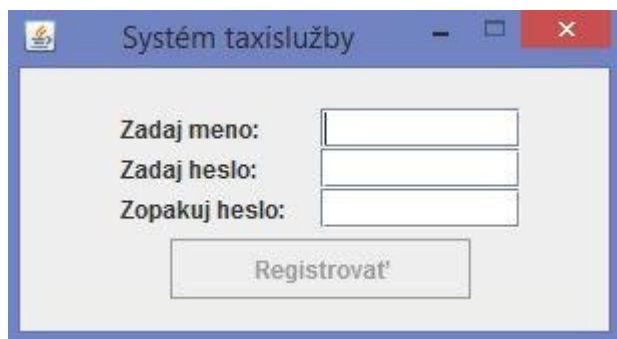
Projekt spĺňa minimálny počet implementovaných scenárov, pričom obsahuje aspoň jeden scenár z každého všeobecného scenáru.

1.1 Vytvorenie nového záznamu

V rámci všeobecného scenára “Vytvorenie nového záznamu” program implementuje tri samostatné podscenáre. Konkrétne je to “registrácia nového zákazníka”, “pridanie nového vozidla” a “pridanie nového používateľa”.

1.1.1 Registrácia nového zákazníka

Scenár registrácia umožňuje používateľovi systému vytvoriť nový zákaznícky profil. Pracuje s tabuľkou “osoba”, do ktorej vloží nový záznam. Po vytvorení nového záznamu v tabuľke “osoba”, vyberie jeho atribút “ID”, v tabuľke “adresa” vytvorí nový prázdny záznam a priradí “ID” atribútu “osobaID”.

The image shows a screenshot of a software window titled "Systém taxislužby". Inside the window, there is a registration form. It consists of three text input fields stacked vertically. The first field is labeled "Zadaj meno:", the second "Zadaj heslo:", and the third "Zopakuj heslo:". Below these fields is a button labeled "Registrovať". The window has a standard Windows-style title bar with minimize, maximize, and close buttons.

Obr. 1 Registrácia používateľa

1.2 Aktualizácia existujúceho záznamu

V rámci tohto všeobecného scenára program implementuje nasledovné podscenáre spĺňajúce jeho charakteristiku: „zmena hesla používateľa“ a „aktualizácia stavu pridelenej objednávky“.

1.2.1 Zmena hesla používateľa

Používateľ môže využitím scenára zmena hesla upraviť svoje aktuálne prihlasovacie heslo. Po aktivácii scenára sa používateľovi zobrazí nové okno, v ktorom vyplní požadované údaje (zadat' staré heslo, zadať nové heslo, zopakovať nové heslo). Pre úspešnú zmenu aktuálneho hesla sa musia zhodovať údaje „zadat' staré heslo“ s heslom uloženým v danom zázname tabuľky „osoba“ a rovnako aj údaje „zadat' nové heslo“ a „zopakovať nové heslo“.

Pokiaľ kontrolované údaje súhlasia, program upraví aktuálny záznam v tabuľke osoba prepísaním atribútu „pass“ novou inštanciou.

1.3 Vymazanie záznamu

Podscenár „vymazať používateľa systému“ je jediným scenárom umožňujúcim vymazanie záznamu z tabuľky.

1.3.1 Vymazať používateľa systému

Vymazanie používateľa je možné iba z pohľadu osoby, ktorá je prihlásená ako dispečer systému. Scenár umožňuje odstránenie záznamu z tabuľky osoba na základe výberu atribútu „login“ dispečerom. Po zvolení konkrétneho atribútu a stlačení tlačidla „Vymazať“ program odstráni záznamy vybraného používateľa v jednotlivých tabuľkách nasledovným postupom:

- na základe atribútu „login“ vyhľadá „id“ spracovávaného záznamu v tabuľke „osoba“
- na základe atribútu „id“ vyhľadá „taxikID“ spracovávaného záznamu v tabuľke „taxik“
- následne odstráni záznamy z tabuliek: „adresa“, „objednavkaosoby“, „servis“, „poistenie“ a „osoba“

Systém pracuje s predpokladom, že každý zamestnaný taxikár, ktorý je zaregistrovaný v databáze, je majiteľom auta, ktoré šofériuje. Preto sa pri vymazaní taxikára odstránia všetky záznamy jednotlivých tabuliek, ktoré s ním boli spojené (vrátane vozidla v tabuľke „taxik“). Neodstráni sa iba záznam v tabuľke „objednavka“.

The screenshot shows a window titled "Používatelia" (Users) with a table of user data. The table has columns: Meno, Priezvisko, Telefon, Email, Login, Číslo účtu, and Typ. Below the table, there are two controls: a dropdown menu labeled "Zobraziť podľa typu používateľa:" with "všetci" selected, and a button "Zobraziť". To the right, there is a label "Vymazať označeného taxikára:" and a button "Vymazať".

Meno	Priezvisko	Telefon	Email	Login	Číslo účtu	Typ
Andrej	Stotlmajer	0915 125 455	adko@zvnet.sk	Andrej01	2547 515236	taxikar
Rudoslav	Stromokocúr	0915 245 001	rudko@azet.sk	Rudo01		zakaznik
Adam			adam.velky@gmail...	Adam		zakaznik
Daniel	Pišťala		danko54@centru...	Daniel		taxikar
Adrián	Kubányi	0958 585 565	kubko@cubicon.org	Adrian		taxikar
		0915 587 328		Lajos	2547 589645	zakaznik
Stanislava	Drobná	0915 854 743		Stanka		taxikar
Ján	Nevelký	0915 854 893	j.nevelky@zvnet.sk	Jan01	2985 842153	dispecer
				Bibiana		zakaznik

Obr. 2 Vymazanie používateľa

1.4 Zobrazenie prehľadu viacerých záznamov

V programe sú implementované tri podscenáre daného všeobecného scenára. A to konkrétne: „zobrazenie objednávok“, „zobrazenie vozidiel“, „zobrazenie používateľov“

1.4.1 Zobrazenie vozidiel

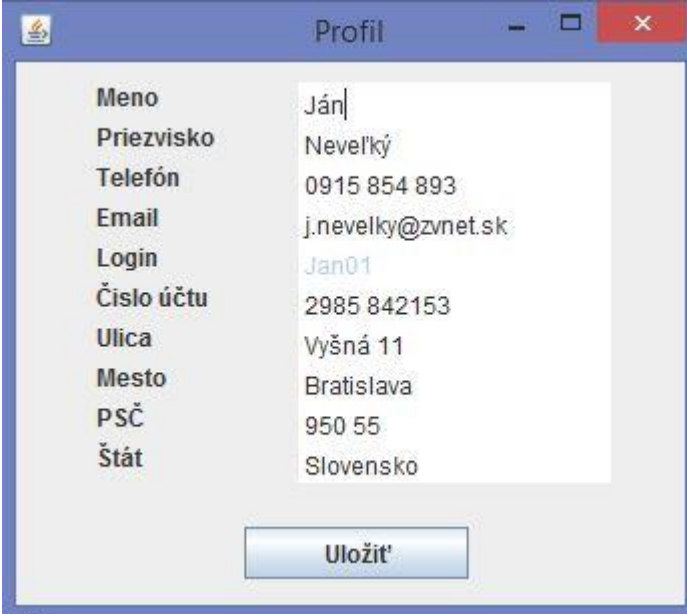
Zobrazenie vozidiel je umožnené iba osobám, ktoré sú prihlásené ako dispečer alebo taxikár v systéme. V rámci tohto scenára sa používateľom zobrazí prehľad aktuálneho vozového parku. V prehľade sa zobrazujú atribúty z tabuliek „taxik“, „osoba“, „poistenie“ a „servis“.

1.5 Zobrazenie konkrétneho záznamu

Podscenár „Pozrieť profil“ zahŕňa tento všeobecný scenár.

1.5.1 Pozrieť profil

Pozrieť profil si môže každý používateľ systému. Po zvolení tohto scenára sa zobrazia jednotlivé atribúty tabuliek „osoba“ a „adresa“ pre konkrétneho prihláseného používateľa.



The screenshot shows a window titled "Profil" with a list of attributes and their corresponding values. The attributes are listed on the left, and the values are on the right. At the bottom of the window is a button labeled "Uložiť".

Meno	Ján
Priezvisko	Nevelký
Telefón	0915 854 893
Email	j.nevelky@zvnet.sk
Login	Jan01
Číslo účtu	2985 842153
Ulica	Vyšná 11
Mesto	Bratislava
PŠČ	950 55
Štát	Slovensko

Uložiť

Obr. 3 Pozrieť profil

1.6 Filtrovanie záznamov spĺňajúcich určité kritéria

Všeobecný scenár je v programe implementovaný pomocou podscenárov „zobrazenie podľa stavu objednávok“ a „zobrazenie podľa typu používateľa“.

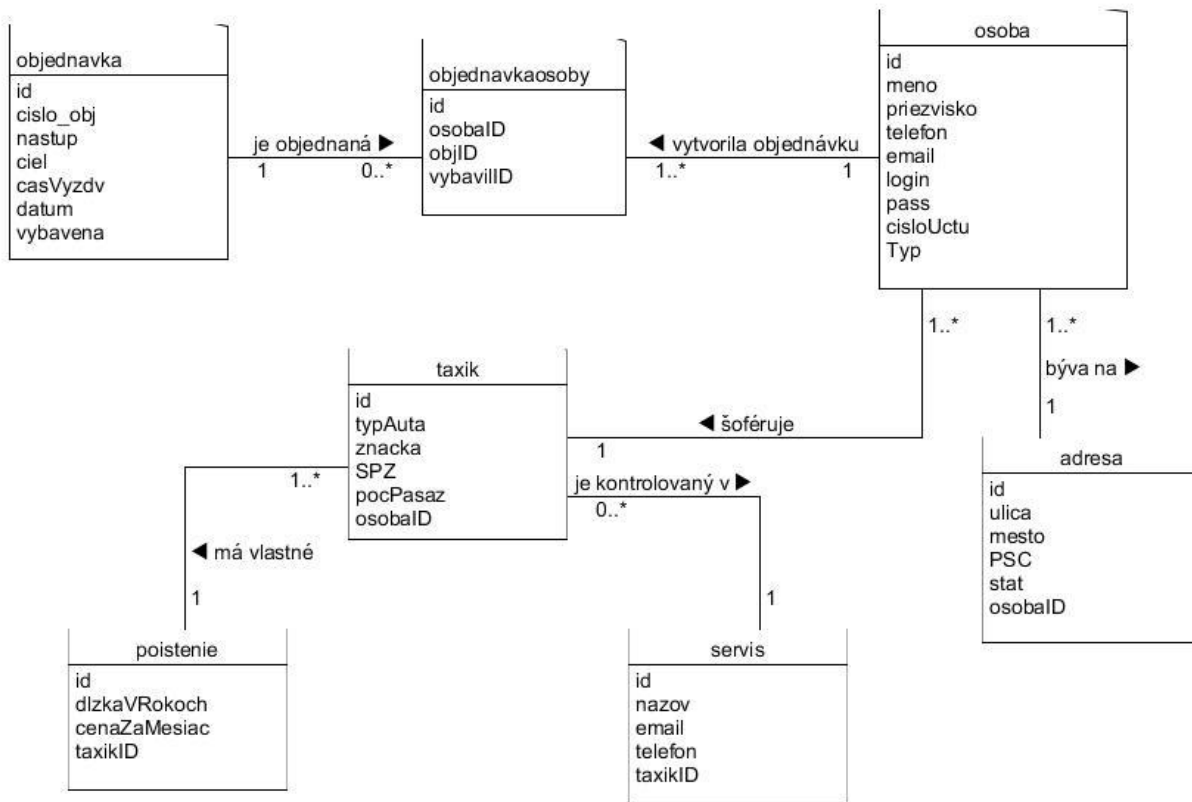
1.6.1 Zobrazenie podľa stavu objednávok

Scenár poskytuje používateľom taxikár a dispečer filtrovanie objednávok podľa vybraného kritéria stavu objednávky. Filtrovanie je umožnené pre vybrané hodnoty atribútu „vybavena“ v tabuľke „objednavka“. Používateľ môže vybrať jednu z možností inštancií daného atribútu „vybavena“, nevybavena“ alebo „všetky“.

2. Diagramy

2.1 Logický dátový model

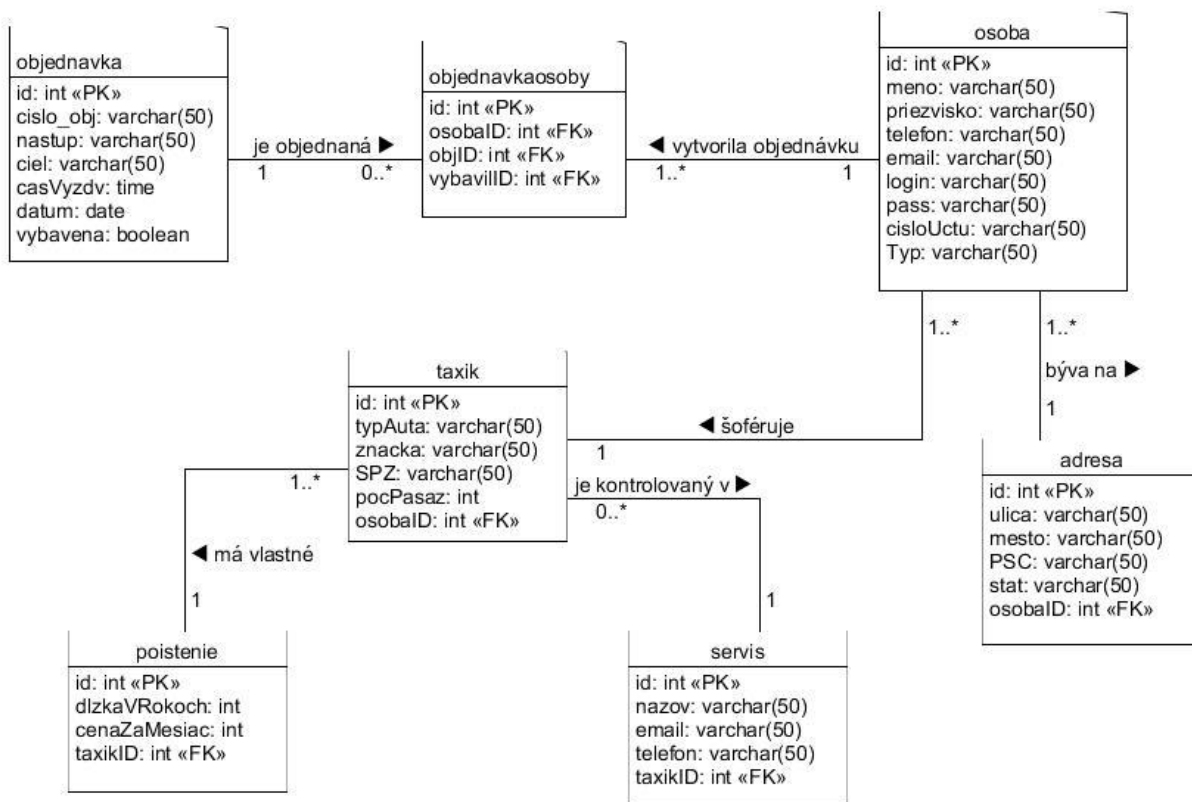
Databáza, ktorú využíva program obsahuje sedem tabuliek, pričom obsahuje aj jednu väzobnú tabuľku(objednavkaosoby).



Obr. 4 Logický dátový model

2.2 Fyzický dátový model

Fyzický dátový model pre databázu, ktorú využívam v programe.



Obr. 5 Fyzický dátový model

3. Implementácia

Zadanie som programoval v rozhraní MySQL jazyka SQL a rozhraní Eclipse jazyka Java. V programe využívam transakcie na ošetrovanie chybových stavov. V rámci práce s databázou využívam funkciu JOIN. Po prihlásení sa používateľovi otvorí nové okno, kde sa mu načíta tabuľka. Túto tabuľku získam využitím funkcie JOIN.

3.1 Pripojenie k databáze

Pripojenie k databáze MySQL z rozhrania Eclipse (jazyka Java) som vyriešil nasledovným spôsobom.

```
static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
static final String DB_URL = "jdbc:mysql://localhost/DBS";

static final String USER = "root";
static final String PASS = "";
static String sql;
static ResultSet rs ;
static Connection conn = null;
static Statement stmt = null;
private PreparedStatement stmt2;

public static void main(String[] args) throws IOException {

    gui.initialize();
    try{
        Class.forName("com.mysql.jdbc.Driver");

        conn = DriverManager.getConnection(DB_URL, USER, PASS);

    }catch(SQLException se){
        se.printStackTrace();
    }catch(Exception e){
        e.printStackTrace();
    }

}
```

Obr. 6 Ukážka kódu funkcie na pripojenie k databáze

3.2 Implementácia scenárov

3.2.1 Zmena hesla používateľa

Funkciu na zmenu hesla som implementoval nasledovným spôsobom. Program v databáze vyhľadá atribút „pass“ tabuľky „osoba“ a to na základe atribútu „login“ práve prihláseného používateľa. Následne uloží nové heslo do atribútu „pass“.

```
public boolean zmenaHesla(String passSt,String passNo,String passNo1) throws SQLException{
    PreparedStatement stmt = conn.prepareStatement("SELECT pass, typ FROM Osoba WHERE login = ?");
    stmt.setString(1, pouzivatel.getLogin());
    rs = stmt.executeQuery();
    if(rs.next()){
        if (rs.getString("pass").equals(passSt)){
            stmt = conn.prepareStatement("UPDATE Osoba SET pass = ? WHERE login = ?;");
            stmt.setString(1, passNo);
            stmt.setString(2, pouzivatel.getLogin());
            stmt.executeUpdate();
            return true;
        }
        else
            return false;
    }
    return false;
}
```

Obr. 7 Ukážka kódu funkcie na zmenu hesla

3.2.2 Zobrazenie vozidiel

Funkcia zobrazenie vozidiel umožňuje vybrať niektoré atribúty z tabuliek „osoba“, „taxik“, „servis“ a „poistenie“.

```
public ResultSet zobrVozidla() throws SQLException{
    PreparedStatement stmt = conn.prepareStatement(
        "SELECT t.typAuta, t.znacka, t.SPZ, t.pocPasaz, o.login, p.dlzskaVRokoch, p.cenaZaMesiac, "
        + "s.nazov, s.email, s.telefon FROM taxik t,osoba o, servis s, poistenie p "
        + "WHERE "
        + "t.osobaID = o.id "
        + "AND "
        + "t.id = s.taxikID "
        + "AND "
        + "t.id = p.taxikID");
    rs = stmt.executeQuery();
    return rs;
}
```

Obr. 8 Ukážka kódu funkcie na zobrazenie stavu vozového parku

3.3 Neošetrené chybové stavy

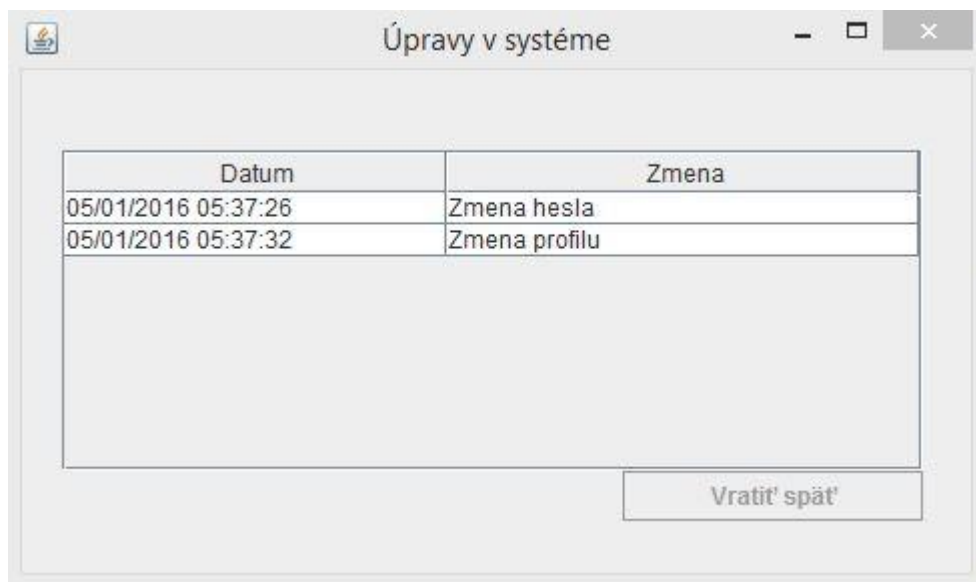
Zákazník má po prihlásení do systému možnosť vytvoriť novú objednávku. Pri vytváraní objednávky však musí čas nástupu a dátum, na ktorý sa vzťahuje objednávka, zadať v správnom formáte. Správny formát pre čas je hh:mm:ss a pre deň dd-mm-yy.

4. II. Iterácia

4.1 Špecifikácia scenárov

4.1.1 Zobrazenie úkonov vykonaných používateľom v systéme

Prostredníctvom tohto scenára si môže používateľ zobraziť zoznam posledných operácií, ktoré vykonal v systéme od prihlásenia. Po zobrazení všetkých predošlých operácií má možnosť vrátiť späť niektoré z nich, ktoré si vyberie. Následne sa vybrané operácie vrátia do stavu pred zmenami, ktoré vykonal používateľ po prihlásení do systému.



Datum	Zmena
05/01/2016 05:37:26	Zmena hesla
05/01/2016 05:37:32	Zmena profilu

Vratiť späť

Obr. 9 Formulár pre scenár „Zobrazenie úkonov“

4.1.2 Implementácia scenáru „Zobrazenie úkonov“

Scenár som implementoval v nerelačnej databáze Redis. Po spustení programu sa vykoná potrebné pripojenie nerelačnej databázy na server. Následne sa priebežne pri vykonávaní operácií v systéme používateľom vykonávajú potrebné zápisy do databázy. Pri volaní scenára sa používateľovi zobrazia záznamy uložené v databáze. Pokiaľ si používateľ zvolí možnosť spätného kroku, systém vráti vykonané operácie do stavu, v akom sa nachádzali pred vykonanými zmenami.

V prípade, že používateľ systém uzavrie a teda ukončí jeho chod, údaje sa z databázy stratia, pretože nie je zabezpečená perzistencia. Je to z dôvodu, že uvedené dáta nie je potrebné uchovávať dlhodobo. Pokiaľ sa používateľ v niektorej z operácií pomýlil a chce vykonať spätný krok, bude tak chcieť urobiť počas aktuálneho prihlásenia, kedy zmenu vykonal.

```

public void vytvorObjednavku(String datum, String cisloObj){
    jedis.select(0);
    jedis.sadd(Long.toString(jedis.incr(kluc)), datum, "Vytvorenie objednávky");
    jedis.set("cislo", cisloObj);
}

public void zmazObj() throws SQLException{
    vymazObjednavku(jedis.get("cislo"));
}

public void zmenHeslo(String datum, String stare, String nove){
    jedis.select(0);
    jedis.sadd(Long.toString(jedis.incr(kluc)), datum, "Zmena hesla");
    jedis.set("stare", stare);
    jedis.set("nove", nove);
}

public void zmeneneHesla() throws SQLException{
    zmenaHesla(jedis.get("nove"), jedis.get("stare"), jedis.get("stare"));
}

public void zmenProfil(String datum){
    jedis.select(0);

    jedis.sadd(Long.toString(jedis.incr(kluc)), datum, "Zmena profilu");
}

```

Obr. 10 Príklad implementácie vkladania a výberu dát z Redisu

```

public List<String> vratDatabazu(){
    jedis.select(0);
    List<String> vysledok = new LinkedList<String>();
    List<String> kluce = new LinkedList<String>();

    Set<String> names = jedis.keys("*");

    java.util.Iterator<String> it = names.iterator();
    while(it.hasNext()) {
        key = it.next();
        if (key.matches("[1-9]+")){
            kluce.add(key);
        }
    }

    for (int i = 0; i < kluce.size(); i++){
        vysledok.addAll(jedis.smembers(kluce.get(i)));
    }

    return vysledok;
}

```

Obr. 11 Príklad implementácie výberu dát z databázy Redis

5. Zhodnotenie

Zadanie spĺňa minimálne požiadavky na akceptovanie. V programe som implementoval každý zo všeobecných scenárov, pričom niektoré sú zastúpené viacerými podscenármi. Zároveň som využil aj agregáčnú funkciu COUNT() na výpis aktuálneho počtu nevybavených objednávok. V databáze som vytvoril sedem samostatných tabuliek a využívam jednu väzobnú tabuľku. V niektorých funkciách, kde sa upravujú určité záznamy v databáze, som ošetril chybové stavy využitím transakcií.

Program by som mohol zlepšiť využívaním viacerých JOINov, či doplnením niektorých scenárov.

6. Zhodnotenie II. Iterácie

V rámci druhej iterácie som program pripojil na nerelačné databázy Elasticsearch a Redis. Nakoniec som sa rozhodol použiť databázu Redis a to konkrétne na implementovanie scenára „Zobrazenie úkonov“. V rámci programu dokážem pomerne rýchlym spôsobom vrátiť operácie vykonané používateľom do pôvodného stavu.

Program by som mohol vylepšiť doplnením niektorých scenárov.