Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológií

Ilkovičova 2, 842 16 Bratislava 4

Databázové systémy

Druhá iterácia

Jakub Mičo

Študijný program: Informatika

Ročník: 2

Krúžok: Streda. 10:00

Predmet: Databázové systémy

Vedúci projektu: Ing. Eduard Kuric

Ak. rok: 2015/16

**Zadanie:**

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 Vašim cvičiacim na cvičení. Aplikáciu vytvoríte v dvoch iteráciach. 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).

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

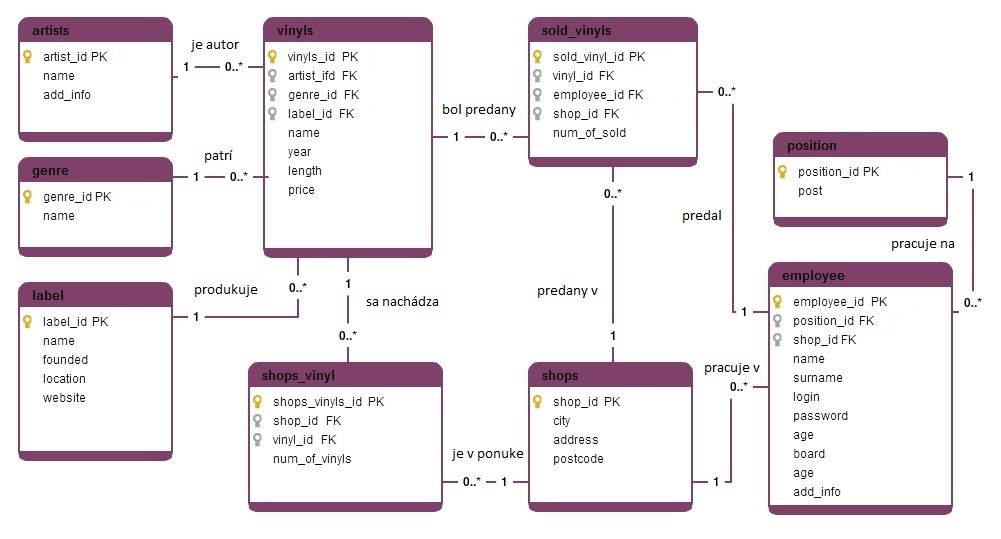
V druhej iterácii do aplikácie pridáte min. 1 scenár postavený na nerelačnej databáze [Redis](http://redis.io/" \t "_blank) alebo [Elasticsearch](https://www.elastic.co/products/elasticsearch) (dohoda s cvičiacim na inom type nerelačnej db je samozrejme možná). Konkrétny scenár si dohodnete s vašim cvičiacim v závislosti od použitej databázy a domény vašej aplikácie (napr. štatistiky o interakciách s jednotlivými záznamami aplikácie v Redise alebo vyhľadávanie záznamov cez Elasticsearch).

**Riešenie:**

Ako tému svojej práce som si zvolil obchod s LP platňami a hudobninami. Obchod, ako aj aplikácia, sa nazýva „StellarWind“. Aplikácia realizuje proces predaja jednotlivých albumov v jednotlivých predajniach. Jedná sa o fiktívny reťazec predajní, aplikácia teda musí riešiť prístup viacerých užívateľov v reálnom čase, zobrazovanie, filtrovanie dát, archiváciu, a samotnú správu dát.

**SQL časť (Prvá iterácia):**

**Logický model:**



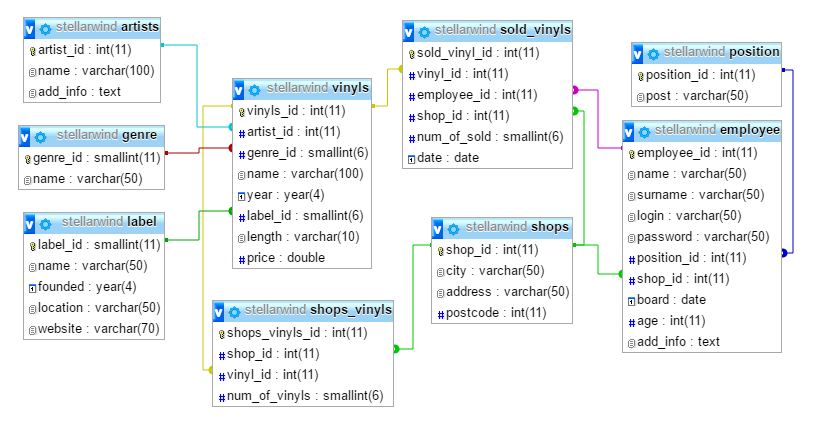
*Obr. 1 Logický dátový model SQL databázy*

Logický model databázy navrhnutý pomocou nástroja dostupní na stránke <http://creately.com/>. Podobne ako fyzicky model zobrazuje jednotlivé entity a vzťahy medzi nimi. Dve hlavné entitu sú tabuľka „sold\_vinyls“ a „shop\_vinyls“ .

Tabuľka „shop\_vinyls“ slúži na vedenie evidencie albumov a ich počet, pre jednotlivé obchody. V tabuľke „sold\_vinyls“ sú uložené záznamy o jednotlivých albumoch, kedy boli predané, koľko, v ktorom obchode a kým.

Entita vinyl nesie informácie o autorovi ktorí je uložení v tabuľke artist a podobne informácie o žánri a vydavateľstve. Ku každému pracovníkovi je pridelený jeho domáca predajňa ako aj jeho funkcia uložená v tabuľke „positiom“

**Dátový model (fyzický):**



*Obr. 2. fyzický dátový model SQL databázy*

Fyzický dátový model pozostáva s 9 entít, ktoré obsahujú nutné dáta pre funkčnosť programu. Jedná sa o úplný fyzický dátový model vygenerovaný prostredím PhpMyAdmin v ktorom bola databáza navrhnutá.

Vysvetlivky k dátovému modelu:

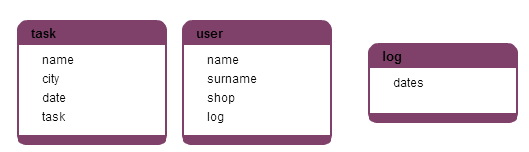
1. Entita „artist“ : obsahuje informácie o umelcoch, konkrétne hudobníkoch alebo kapelách, má vzťah s entitou vinyls (kardinalita 1 ku n)
2. Entita „genre“: má rovnaký vzťah podobne ako entita „artist“, nesie informácie o hudobnom žánri platne
3. Entita „label“: podobne ako „artist“ a „genre“, obsahuje aj dodatkové informácie v podobe atribút ako webstránka, rok založenia...
4. Entita „vinyls“: obsahuje informácie o jednotlivých albumoch, ich autora, žáner... každý album obsahuje práve jedného autora, žáner, vydavateľstvo. (Možnosť rozšírenia pre viac autorov pre jednu platňu pridaním väzobnej entity medzi artist a vinyl)
5. Entita „sold\_vinyls“ jedná sa o entitu, ktorá uchováva informácie o predaji
6. Entita „shop\_vinyls“ uchováva informácie typu: „koľko kusov daného albumu je v danej prevádzke“
7. Entita „shops“ predstavuje jednotlivé prevádzky, každý zamestnanec môže pracovať práve v jednom obchode.
8. Entita „employees“ udržiava informácie o pracovníkoch ich prístupové údaje do system...
9. Entita „post“ predstavuje jednotlivé pozície

**Špecifikácia scenárov:**

Aplikácia realizuje CRUD funkcionalitu pre entity artist, genre, label, vinyls, emplyoee, shops a post, CRU pre entitu shops\_vinyls a CR funkcionalitu pre sold\_vinyls. Pre entity employee, shops\_vinyls a sold vinyls je k dispozícií filter pre rýchle a pohodlné vyhľadanie informácií. Implementované sú scenáre pre zobrazenie rôznych štatistík, viac informácií v odseku špeciálne scenáre. K dispozícií je aj funkcionalita pre zobrazenie detailného profilu pracovníka.

Aplikácia je realizovaná pre vykonávanie predajného procesu a vedie si históriu predaja.

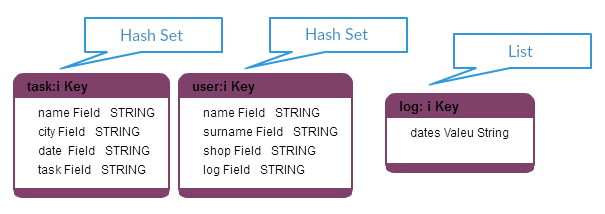
**No-SQL časť (Druhá iterácia):**



*Obr. 1 Logický dátový model NoSQL databázy*

Logický model zobrazuje dva entity:

* Task – obsahujúca informácie o  jednotlivých úlohách
* User - obsahujúca informácie o jednotlivých useroch, log obsahuje aj jednotlivé logy



*Obr. 2. Fyzický dátový model NoSQL databázy*

Databázový model je zostavený z dvoch HashSet-ov a jedného listu. Hash set task:i obsahuje informácie o jednotlivých úlohách v systéme. Hash set user:i obsahuje informácie o jednotlivých užívateľoch, obsahuje aj názov logového súboru ktorý je reprezentovaný ako List jednotlivých dátumov prihlásenia a odhlásenia.

**Špecifikácia scenárov:**

Do aplikácie bol pridaný Task Board, jedná sa o formulár kde pracovníci môžu naplánovať úlohy, respektíve vytvoriť zoznam úloh ktoré majú byť splnené. Aplikácia realizuje CRUD funkcionalitu nad jednotlivými úlohami. Úlohy sú ukladané do Redis-ovej databázy ako Hash Set-y s názvom „task:poradové číslo task-u“. Každý Hash Set obsahuje polia meno pracovníka, ktorý task vytvoril, kedy ho vytvoril, pre ktorý obchod je task určený a samotná správa.

Druhý scenár je zaznamenávanie doby prihlásenia pre každého užívateľa. Informácie o užívateľoch sú uložené v Hash set-e v tvare „user:poradové číslo“, obsahujú meno, priezvisko, obchod a meno Listu, v tvare access:poradové číslo“, v ktorom sú dátumy prihlásenia a odhlásenia.

Pri každom úspešnom prihlásení je uložený vstupný dátum do listu následne pri úspešnom odhlásení je uložený druhý dátum. V prípade straty spojenia zo serverom aplikácia vytvorí chybový log do ktorého uloží dátum a čas straty spojenia a dátum je pri opätovnom spustení nahratý.

**Use case predajného procesu:**

**Stručný popis:**

Proces predaja platne

**Primárny hráč:**

Pracovník hudobnín (predavač, vedúci, účtovník, riaditeľ)

**Vstupné predpoklady:**

Požiadavka zákazníka o kúpu tovaru

**Hlavný prúd:**

1. Pracovník vyplní textové polia filtra v tabe “Catalogue“ a stlačí „Find“
2. Systém vyberie z databázy údaje, ktoré prešli fitrom.
3. Pracovník zvolí položku v tabuľke a stlačí tlačidlo „Sell“
4. Systém odblokuje dve textové polia, v prvom sa zobrazí názov a interpret
5. Pracovník vyplní pole a zadá počet kusov ktoré chce predať
6. Ak počet ktorý pracovník zadal je menší alebo rovný ako počet ktorý je v tabuľke systém vytvorí záznam v databáze, konkrétne v tabuľke „sold\_vinyls“ a zníži počet kusov v tabuľke „shops\_vinyls“ ak podmienka nie je splnená, tak alternatívny prúd.
7. Proces predaja je ukončený

**Výstupy:**

Záznam v databáze o predaji

**Alternatívny prúd:**

1. Zobrazí sa chybová sprava a proces objednávky je ukončený

**Špeciálne scenáre:**

***SQL scenáre***

1. Tab „Catalagoue“ scenár vypisuje dáta z 3 tabuliek, vykonáva subquery, 3x JOIN – **Zobrazenie všetkých platní vo všetkých obchodoch spolu s autorom žánrom a dodatkovými informáciami**
2. Tab „History“ scenár vypisuje dáta zo 4 tabuliek, vykonáva subquery, 4x JOIN, funkcia ROUND, násobenie stĺpcov – **Zobrazenie všetkých predaných platní vo všetkých obchodov a osobu zodpovednú za jeho predaj, cenu predaja ...**
3. Koláčový graf a tabuľka v tabe „Statistics“, subquery, 2x JOIN, funkcie SUM a ROUND, použité GROUP BY - **Zobrazuje podiel predaja v jednotlivých obchodoch**
4. V okne „More Statistics“ tab Employee graf – násobenie stĺpcov, 2x JOIN, 2x subquery, GROUP BY – **Zobrazenie zisku pre jednotlivých pracovníkov**
5. V okne „More Statistics“ tab Montly graf – funkcie EXTRACT, MONTHNAME, STR\_TO\_DATE, SUM a GROUP BY spolu s HAVING - **Zobrazenie mesačného príjmu pre jednotlivé roky**
6. V okne „More Statistics“ tab Other graf + tabuľka – funkcia SUM, 2x JOIN, HAVING, LIMIT – **Zobrazí 10 top najpredávanejších interpretov**
7. V okne „More Vinyls“ tab Genre funkcia COUNT, LEFT JOIN a GROUP BY – **Zobrazí 10 najviac zastúpených žánrov vo všetkých predajniach**
8. V okne „More Vinyls“ tab Artist funkcia COUNT, LEFT JOIN a GROUP BY – **Zobrazí 10 najviac zastúpených umelcov vo všetkých predajniach**

***NoSQL scenáre – Redis***

1. Tab „Task Board“ scenár poskytuje funkcionalitu CRD, **je možné v systéme vytvoriť úlohu ktorá sa zobrazí všetkým pracovníkom v systéme**, úlohu je možno splniť a teda odstrániť zo zoznamu, čítať a vytvárať novú
2. V okne „Show Access“ v tab „Employee“, **zobrazuje jednotlivé prihlásenia užívateľov**, a ráta celkový čas prihlásenia, časy pre každého pracovníka sú vyhodnotené a zobrazené v grafe.

**Stručný opis návrhu a implementácie:**

**Programovacie prostredie:**

Aplikácia bola realizovaná v programovaciom jazyku JAVA JDK8 v IDE Netbeans na vytvorenie grafiky, okien, bola použitá technológia JAVAFX a jednotlivé okná predstavujú fxml súbory v package View. SQL Databáza bola vytvorená v jazyku MySQL, subjazyk rodiny SQL, a práca s ňou bola realizovaná pomocou nástroja PhpMyAdmin. Databáza ako ja PhpMyAdmin boli uložené na lokálnom uložisku ktoré bolo realizované pomocou nástroja XAMPP.

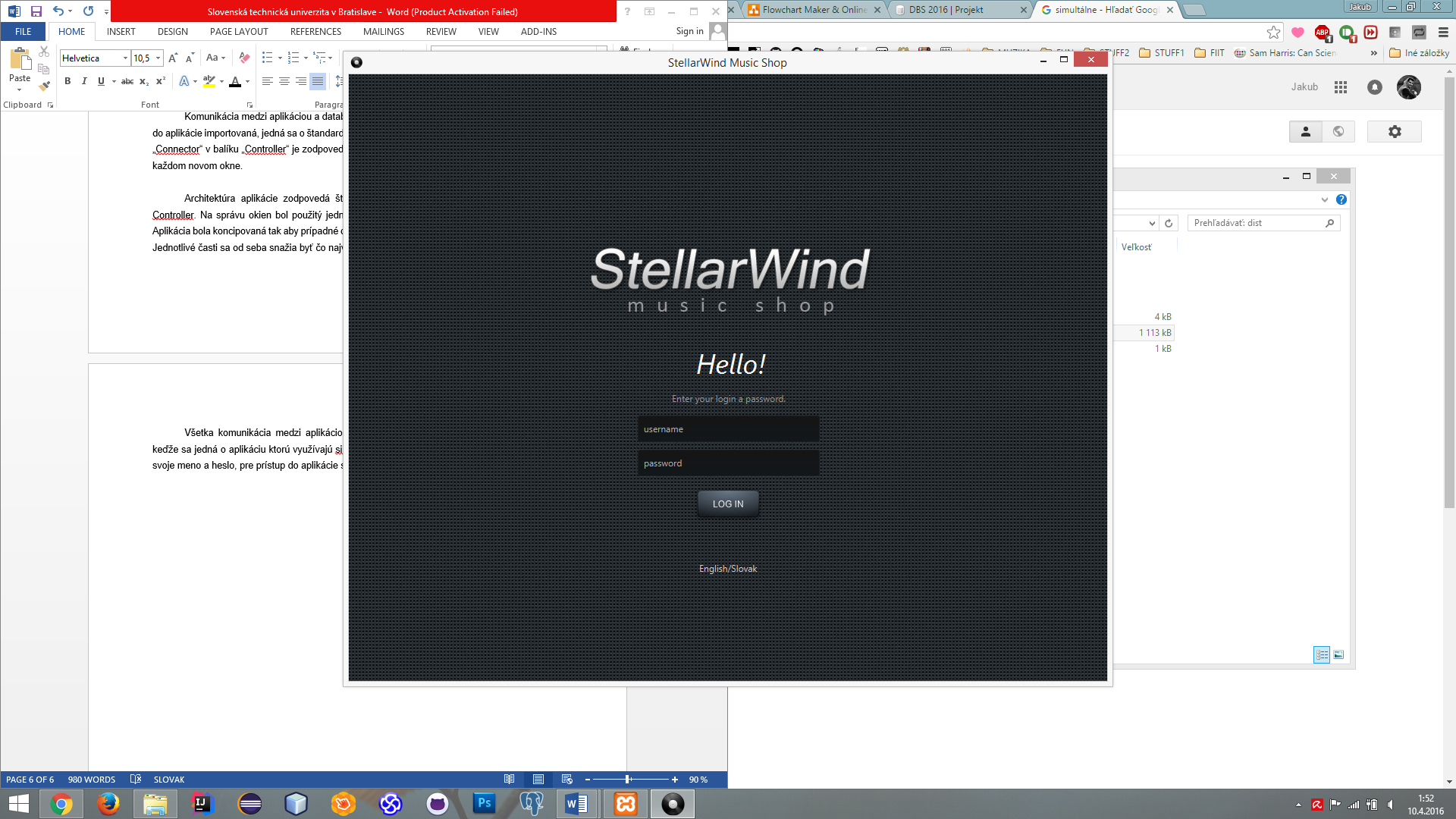
Na vytvorenie nerelačnej databázy bol použitý balík nástrojov pre prácu s key-value databazou Redis, konkrétne verzia „Redis-x64-2.8.2400“

Komunikácia medzi aplikáciou a databázou prebiehala pomocou JDBC knižnice ktorá bola do aplikácie importovaná, jedná sa o štandardizovaný balík pre prácu s databázou v JAVE. Metóda „Connector“ v balíku „Controller“ je zodpovedná za pripojenie k databáze, pripojenie sa otvára pri každom novom okne.

Architektúra aplikácie zodpovedá štandardnému architektonickému vzoru Mode-View-Controller. Na správu okien bol použitý jednoduchý framework pre výmenu StackPanu v okne. Aplikácia bola koncipovaná tak aby prípadné dorobenie inej funkcionality bolo jednoduché a rýchle. Jednotlivé časti sa od seba snažia byť čo najviac nezávisle.

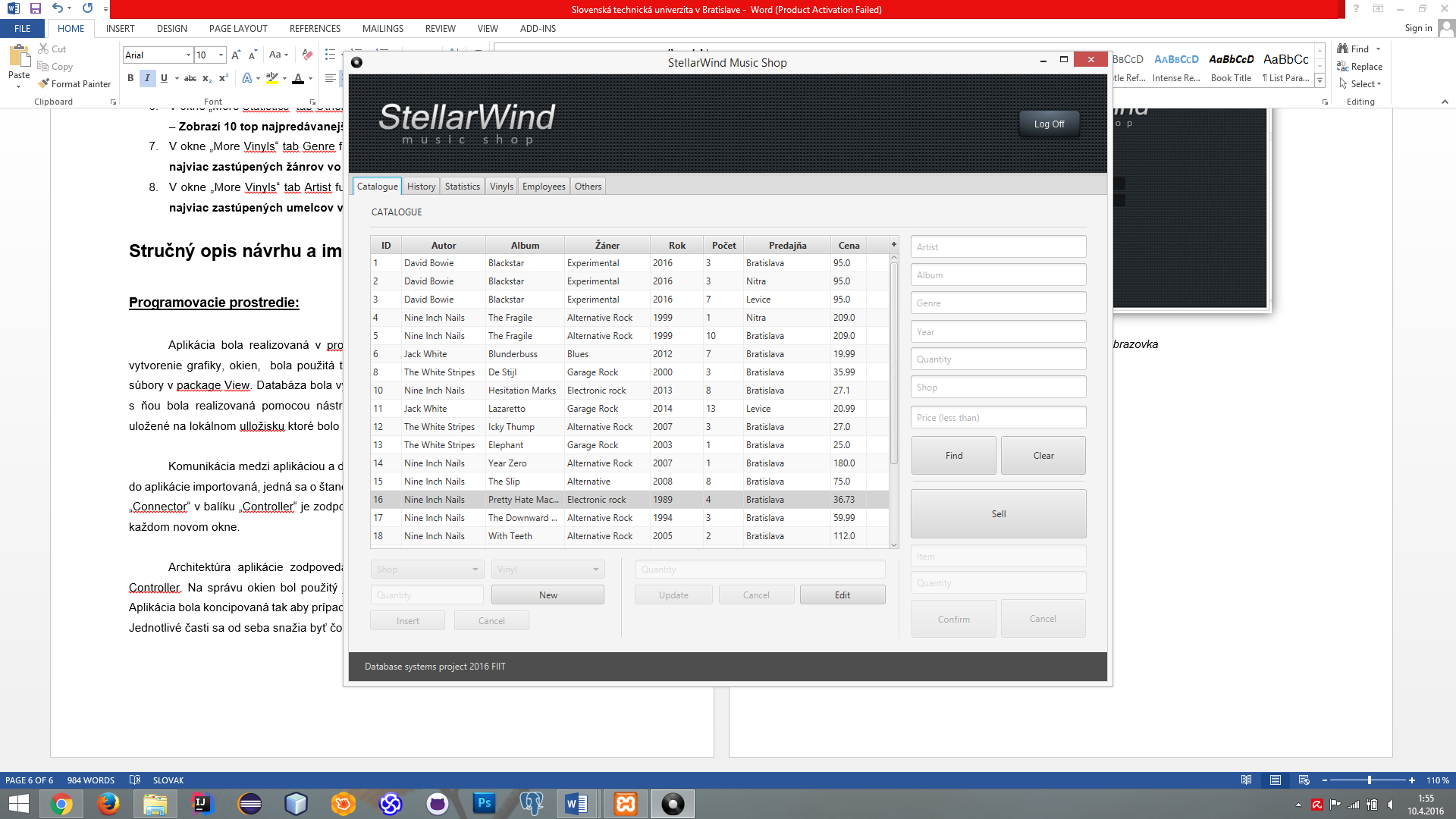
Všetka komunikácia medzi aplikáciou a databázou je realizovaná pomocou transkacií, keďže sa jedná o aplikáciu ktorú využívajú simultálnne viacerí používatelia. Každý používateľ ma svoje meno a heslo, pre prístup do aplikácie sa treba zvoľte meno a heslo „admin, admin“.

**Grafické rozhranie:**



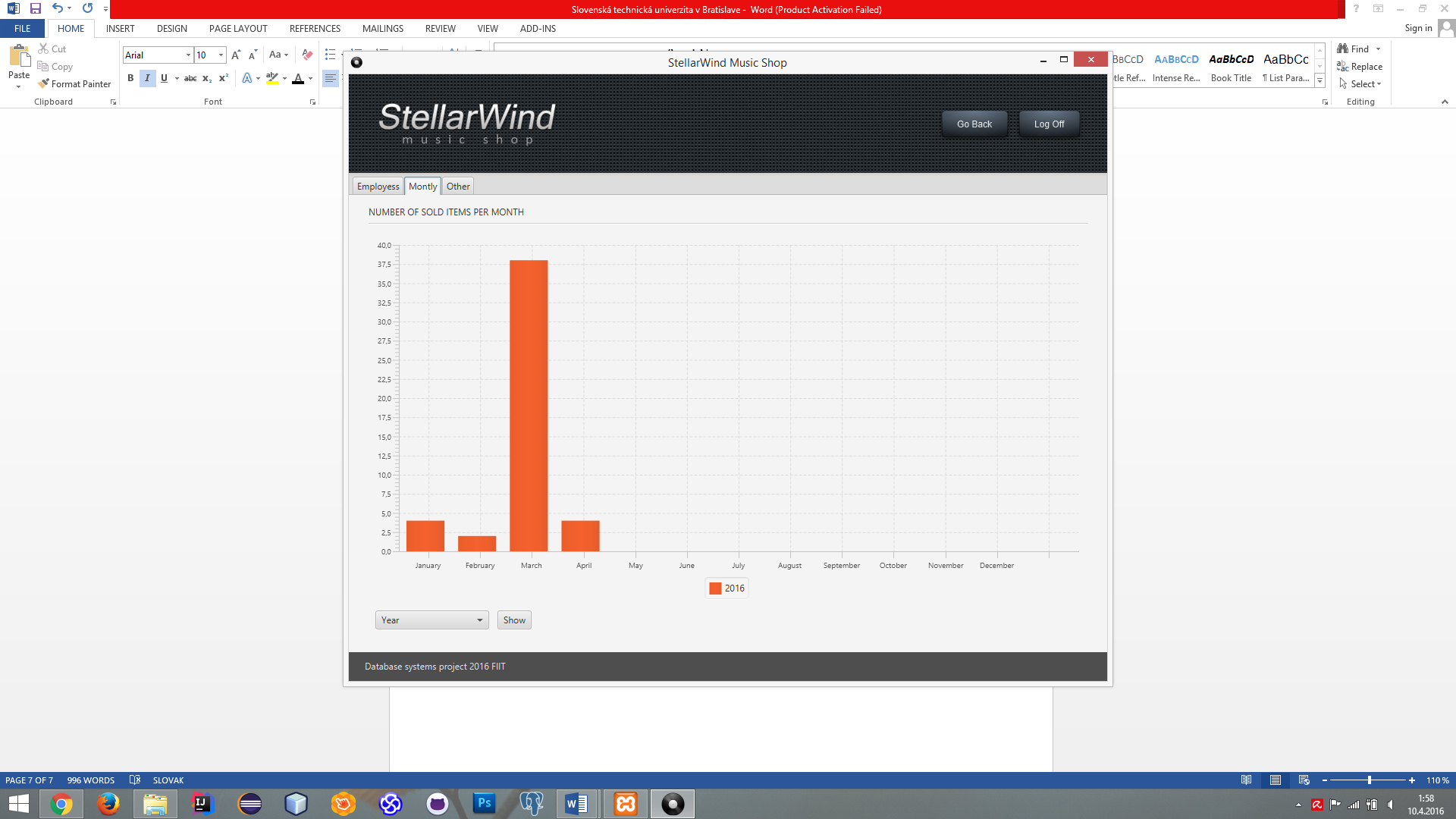
*Obr. 3 Prihlasovacia obrazovka*

**Vizualizácia dát**



*Obr. 4. Okno po prihlásení, katalóg platní pre jednotlivý obchod (hlavné okno)*

Dáta z databázy sú vizualizované viacerými spôsobmi, základný spôsob je pomocou tabuľky ako je možné vidieť na obrázku č. 4. no v aplikácií sú použite aj vizualizačné nástroje ktoré poskytuje JavaFx a to BarChart, obrázok č. 5 ako aj PieChart, obrázok č. 6.

*Obr. 5 Príklad štatistiky pomocou stĺpcového grafu*

*Obr. 6 pohľad na vizualizáciu dát z databázy pomocou Pie Chart grafu*

