EKONOMICKÁ UNIVERZITAObrázok, na ktorom je text, hodiny

Automaticky generovaný popis

Fakulta hospodárskej informatiky

Katedra aplikovanej informatiky

Distribuované spracovanie dát

Filmová databáza

Seminárna práca

**Meno študentov:** Bc. Martin Jankech, Bc .Samuel Veštúr

**Stupeň/ročník/krúžok:** druhý/1.ročník/3 a 4 krúžok

**Školský rok:** 2021/2022

**Študijný program:** Informačný manažment

**Meno vyučujúceho:** SCHMIDT, Peter, Ing. Mgr., PhD.

# Použité technológie

VPN- Hamachi

Server a Databáza – cez Xamp – Apache + phpMyAdmin (MySQL)

Frontend – HTML, CSS, JavaScript, jQuery + Ajax – pre single Page aplikáciu, Bootstrap.

Backend – PHP

# ÚVOD

V tejto práci sme vytvorili distribuovaný informačný systém pre pre jednoduchú filmovú databázu. Naša webová aplikácia obsahuje úvodnú stránku, kde sa používateľ musí zaregistrovať. Po úspešnej registrácii sa môže prihlásiť do svojho účtu, kde si môže prezerať záznamy o filmoch, ako aj nové záznamy pridávať, editovať a zmazať. V práci sme pomocou hamachi prepojili 4 uzly, na ktoré sú dané dáta replikované(automaticky). Taktiež sme zabezpečili automatickú synchronizáciu dát na jednotlivých uzloch a to pomocou pomocného textového súboru, na ktorý sa zapisujú IP adresy a SQL príkazy uzlov, pre ktoré nastala chyba spojenia. Po obnovení spojenia sú záznamy na dané uzly(pri ktorých takto vznikla nekonzistencia) replikované. Po výpadku môže vypadnutý uzol upravovať len tie záznamy, ktoré boli ním vytvorené a taktiež ostatné uzly nemôžu upravovať záznamy vypadnutého uzla. V priebehu prace si naše riešenie podrobnejšie popíšeme.

# Databáza

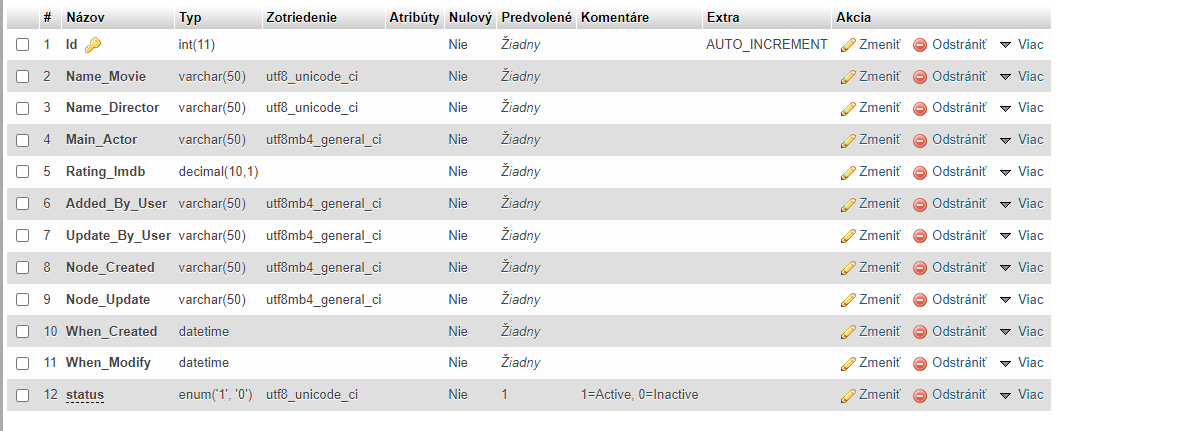
Na začiatku sme si premysleli, ako bude vyzerať databáza pre našu webovú aplikáciu. Po dôkladnom premyslení sme prišli nato, že budeme potrebovať 2 tabuľky. Jednu pre ukladanie záznamov o registrovaných používateľoch a druhú na ukladanie záznamov o filmoch.

Pri jednotlivých filmoch bude vždy jasné, ktorý používateľ ich vytvoril a na akom uzle boli vytvorené. V databáze taktiež zaznamenávame, ktorý používateľ film naposledy upravil a taktiež na akom uzle boli naposledy upravené. Pre náš program je kľúčový atribút Node\_Created ktorý uchováva na akom uzle daný záznam vznikol, pretože len ten uzol môže pri výpadku na daných záznamoch robiť zmeny, ktorý dané záznamy vytvoril.

## Ukážka štruktúry tabuliek

Obrázok, na ktorom je text

Automaticky generovaný popis



## Ukážka dát v databázach



Obrázok, na ktorom je text

Automaticky generovaný popis

## Konfigurácia prepojenia uzlov

Obrázok, na ktorom je text

Automaticky generovaný popisPo vytvorení databáz sme dané databázy exportovali na všetky uzly. Ďalej sme si dané uzly prepojili pomocou programu Hamachi.

Okrem domáceho uzla, ktorý je v tomto prípade na adrese 25.69.87.199 sme si ku nemu vytvorili pripojenie na ďalšie 3 uzly.

Aby sa vzdialené uzly mohli pripájať na náš server, tak sme museli v xampe v httpd.conf nastaviť prístup pomocou prepísania Require Local na Require all granted. Taktiež by to fungovalo, keby sme použili príkaz require ip <ip adresa uzla>, ale v našej jednoduchej aplikácii sme povolili prístup všetkým. Aj keď určite bezpečnejší je práve druhý príkaz.

Obrázok, na ktorom je text

Automaticky generovaný popis

Taktiež, ak sme sa chceli z localhostu pripojiť do databáz vzdialených uzlov, tak sme museli v httpd.xamp.conf prepísať nasledovný riadok znova na Require all granted.

Obrázok, na ktorom je text

Automaticky generovaný popis

Ďalším krokom bolo vytvorenie účtov pre vzdialené uzly v phpmyadmine



V lokánom phpmyadminovi boli vytvorené účty vzdialených uzloch uzlov notebook1, notebook2 a účet samuel. Všetkým sme nastavili globálne práva. Obdobným spôsobom sme museli vytvoriť účty aj pri ostatných uzloch.

## Opis dôležitých častí zdrojového kódu

Po všetkých týchto nastaveniach sme sa mohli pustiť do kódenia danej aplikácie, ktorú si teraz predstavíme. Na nasledujúcom obrázku môžeme vidieť všetky súbory, ktoré sa nachádzajú v adresáry našej aplikácie.

Obrázok, na ktorom je text, monitor, snímka obrazovky

Automaticky generovaný popis

Základný súbor, v ktorom sme prepájali uzly a vytvorili samotnú logiku distribúcie dát je súbor config.php a db.class.php

V config.php sme si definovali menné konštanty, ktoré využijeme pri pripojení na databázy všetkých uzlov.

Obrázok, na ktorom je text

Automaticky generovaný popis

V súbore connection.php sme vytvorili funkciu pomocou ktorej sa pripájame do databázy.

Obrázok, na ktorom je text

Automaticky generovaný popis

Táto funkcia nám vracia buď objekt mysqli, a to pokiaľ je dane pripojenie úspešne, ak nie je, tak nám vracia ipadresu uzlu ktorý sa nepripojil. Tak isto sme si tu nastavili Timeout na pripojenie na 1 sekundu, keďže v prípade výpadku uzlov trval pokus o pripojenie na odpojené uzly dlho a aplikácia bola pomalá.

V triede config.php si pomocou tejto funkcie nastavíme privátne členské premenné buď na objekt mysqli alebo string s IP adresou nepripojeného uzla.

Obrázok, na ktorom je text

Automaticky generovaný popis

Tieto premenné si vložíme do asociatívneho poľa connection

Obrázok, na ktorom je text

Automaticky generovaný popis

Toto pole ďalej preiterujeme a do poľa aviableconnection si vložíme len pripojené uzly (teda mysqli objekty) a do poľa notaviableconnection si vložíme tie uzly ktorých pripojenie bolo neúspešné.

Obrázok, na ktorom je text

Automaticky generovaný popis

Tieto polia si pomocou geterov vieme ďalej získať v iných častiach aplikácie. Napr. pri vykonávaní CRUD operácií na databáze.

Obrázok, na ktorom je text

Automaticky generovaný popis

Súbor db.php

V tejto triede sa nachádzajú 4 metódy pomocou ktorých vieme získavať, vkladať editovať a mazať záznamy z/do databázy. Metódy sú napísané tak, aby sa dali použiť na viacerých miestach v aplikácií. (teda SQL string sa vytvára na základe vložených parametrov a pri rozšírení programu nemusíme písať nove SQL stringy ale len použiť tieto metódy s inými parametrami). Metódy majú buď 2 alebo 3 parametre a to meno tabuľky a dáta + podmienky ktoré sa nachládajú v asociatívnom poli. Po vložení týchto 2 až 3 parametrov sa nám postupne vytvorí SQL string, ktorý sa vykoná len na uzloch, ktoré sa nachádzajú v poli aviableconnection(teda uzly ktoré sú pripojené ). Pre tie uzly, ktoré sú v poli notaviableconnetion sa daný SQL string nevykoná, ale zapíše sa do pomocného súboru notaviablenodes.txt.

Textový súbor notaviablenodes.txt číta metóda z config.php s názvom sychronize, ktorá sa pokúša pripojiť na uzly ktorých IP adresa a priradený SQL príkaz sa nachádza v tomto textovom súbore. Pokiaľ je pripojenie obnovené, tak je daný SQL príkaz pre daný uzol vykonaný a dáta sú tak synchronizované. Všetky príkazy, ktoré už boli vykonané sa z daného textového súboru vymažú.

Obrázok, na ktorom je text

Automaticky generovaný popis

Ako príklad si popíšeme len metódu Insert, ale podobne to funguje pre všetky metódy. (update,delete a getRows). Táto funkcia dostane ako parameter meno tabuľky a dáta v asociatívnom poli (pole key=>value). Najprv si overí, či používateľ vložil atribúty when\_crated,\_when modify a Node created a ak nie tak ich naplní. Ďalej z vloženého asociativneho poľa($data) extrahuje mená stĺpcov (key)a konkrétne dáta(value), ktoré sa majú vložiť do databázy. Takto sa postupne vytvorí SQL string uložený v premennej $querry. Tento string sa vykoná na uzloch, ktoré sú v poli aviableconnection(pripojené uzly) . Pre nepripojené sa SQL príkaz zapíše do textového súboru notaviablenodes.txt

Obrázok, na ktorom je text

Automaticky generovaný popis

Obrázok, na ktorom je text

Automaticky generovaný popis

Príklad využitia funkcie insert sa nachádza v súbore movieAtion.php, kde sú dané dáta získané pomocou POST metódy z formulára a pomocou ajaxu ihneď zobrazene na stránke (teda nemusíme refreshovat stránku po každom vložení záznamu).

Obrázok, na ktorom je text

Automaticky generovaný popis



Tu bola využitá daná insert funkcia pre tabuľku movies a s dátami o jednom filme ktoré sa nachádzajú v asociatívnom poli movieData.

Obrázok, na ktorom je text

Automaticky generovaný popis

Obrázok, na ktorom je text

Automaticky generovaný popis

Pre potrebu spracovania daného výstupu javascritom sme výsledky vrátili v Json formáte. Tu sa teda vykoná to, že sa dáta zapíšu do databázy a taktiež sú poslané dáta o úspechu na frontned v Json formáte, ktorý ich zobrazí bez nutnosti refreshovania stánky. Dané Jquery funkcie si môžete pozrieť v assets/script.js (tu ich neuvádzame lebo nemajú vplyv na zadanie, ale skôr nato že webová aplikácia je dynamickejšia a tým je zadanie elegantnejšie ).

Ďalšie metódy tu uvádzam len na ukážku, ale princíp je podobný, ako v popísanej metóde insert. Podobne sú tieto metódy volané v Movieaction.php a add-register.php, kde do nich vkladáme meno tabuľky buď „user“ alebo „movie“ a dáta a podmienky podľa potreby.

Metoda getRows

Obrázok, na ktorom je text

Automaticky generovaný popis

Obrázok, na ktorom je text, obrazovka, snímka obrazovky

Automaticky generovaný popis

Metoda Update

Obrázok, na ktorom je text

Automaticky generovaný popis

Obrázok, na ktorom je text

Automaticky generovaný popis

Metóda Delete

Obrázok, na ktorom je text

Automaticky generovaný popis

V súbore index.php sa nachádza tabuľka s filmami ako aj formulár na vkladanie filmov.

V súbore login.php na nachádza hlavná stránka s dvoma modálnymi formulármi pre registráciu a prihlásenie. V týchto soboroch sa nachádzajú zväčša html zápisy a preto ich tu celé nebudeme uvádzať(keďže majú okolo 200 až 300 riadkov)

V soboroch add-login.php a add-register sa nachádza logika pre spracovanie údajov získaných o používateľoch pri registrácii a prihlásení.

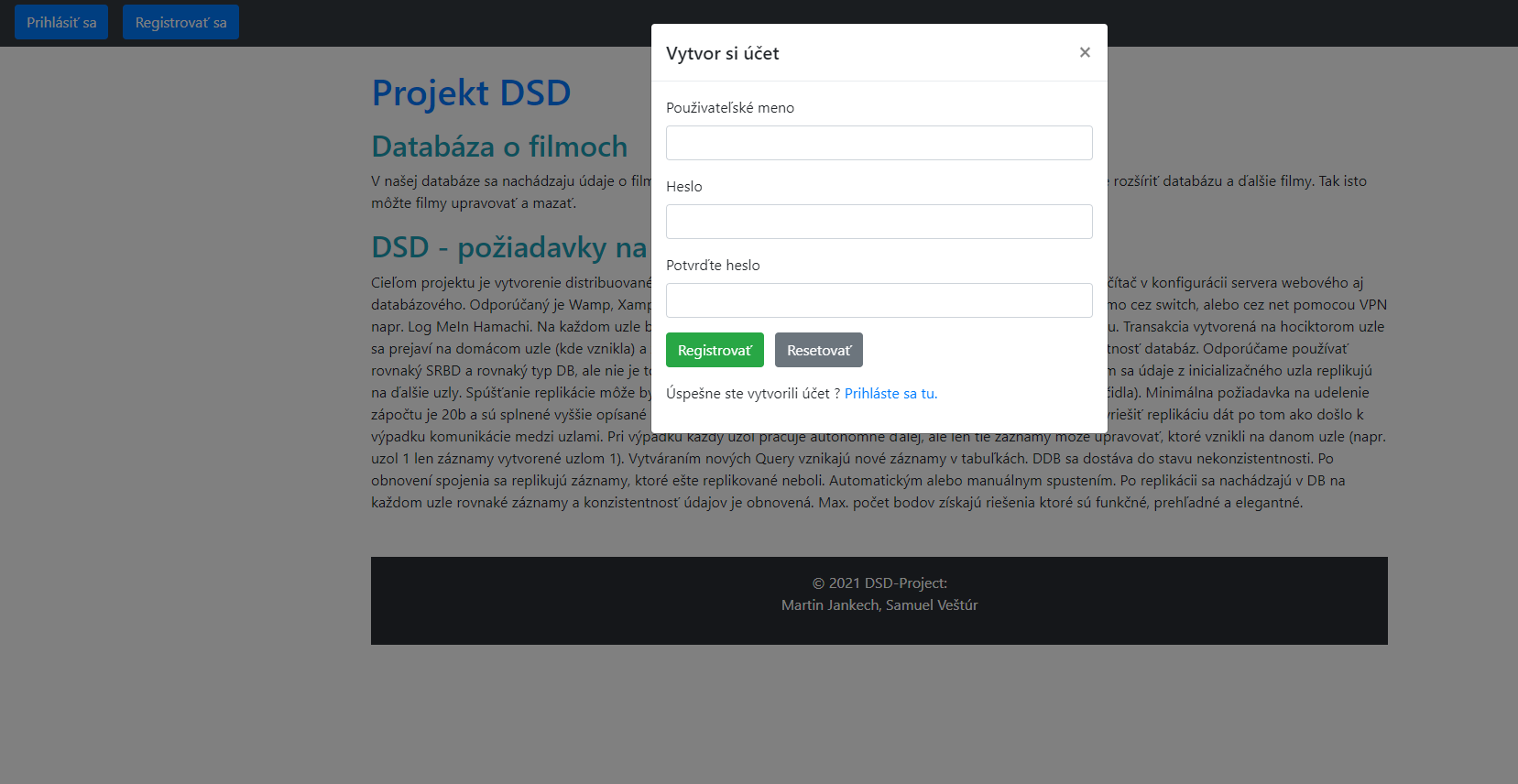
## Ukážka login.php

Obrázok, na ktorom je text

Automaticky generovaný popis

Obrázok, na ktorom je text

Automaticky generovaný popis

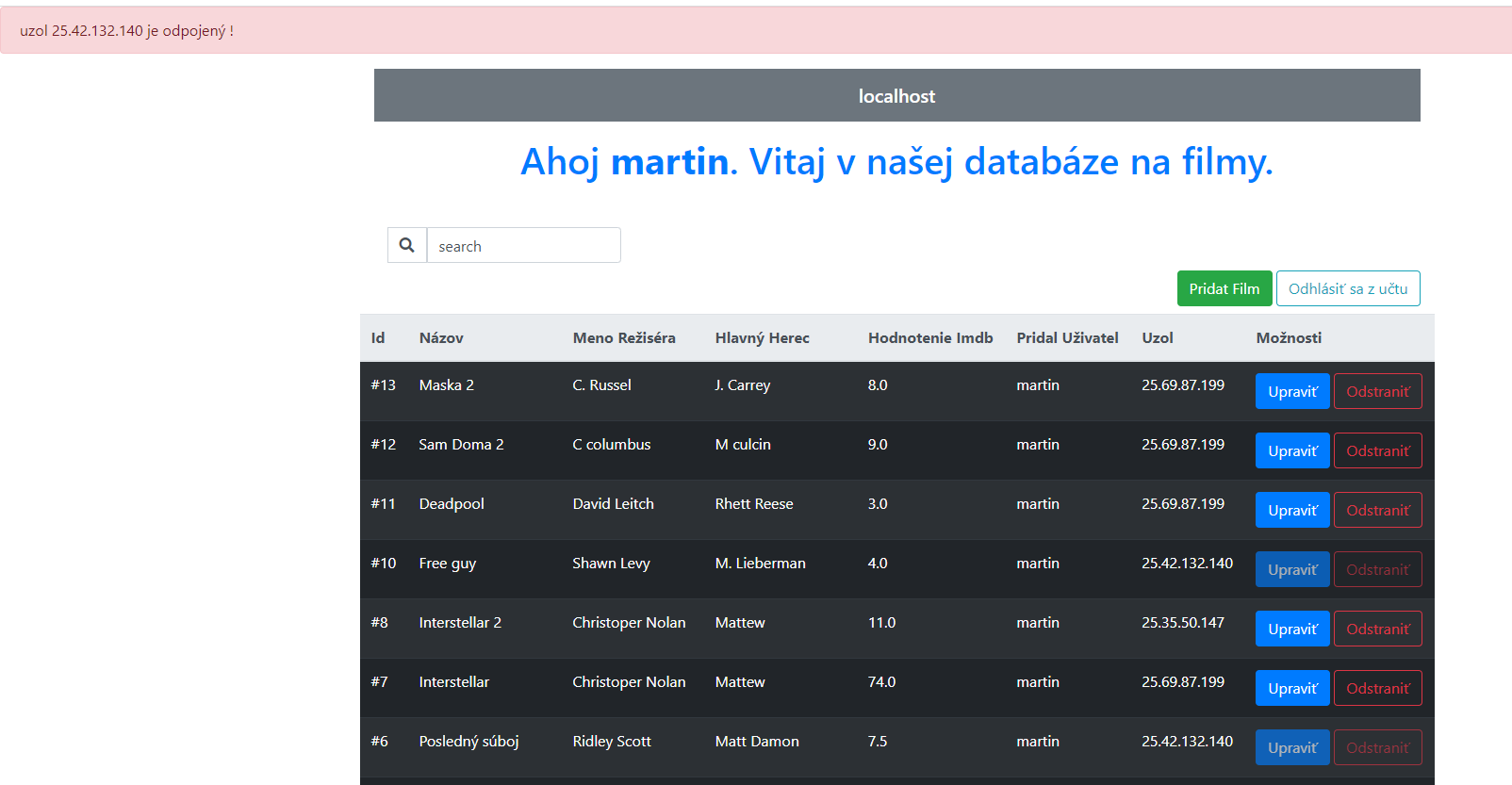


## Obrázok, na ktorom je text, snímka obrazovky, počítač, vnútri Automaticky generovaný popisUkážka index.php

# Ukážka login.php po výpadku 1 uzla (simulujeme odpojením uzla z hamachi)

Obrázok, na ktorom je text

Automaticky generovaný popis

Systém nás upozorní, že je odpojený jeden uzol(25.42.132.140) a taktiež pre záznamy, ktoré vznikli na vypadnutom uzle sa zablokujú tlačítka, aby sme ich na localhoste nemohli upravovať.

Na vypadnutom uzle(25.42.132.140) naopak môžeme upravovať len tie záznamy, ktoré vznikli na ňom a ostatné sú zablokované.

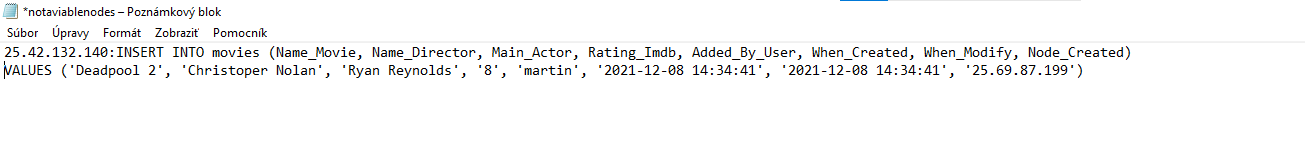
Obrázok, na ktorom je text, monitor, počítač, snímka obrazovky

Automaticky generovaný popis

Pokiaľ teraz niečo urobíme na localhoste, tak sa dané SQL príkazy pre vypadnutý uzol zapíšu do pomocného textového súboru.

Napr. aj teraz pridáme nový film Deadpool 2, tak sa daný film replikuje na všetky pripojene uzly, ale na ten čo je odpojený, sa SQL zapíše do textového súboru. (podobne to funguje aj na update, delete )

Textový súbor vyzerá nasledovne :



Po obnovení spojenia sa daný SQL príkaz vykoná a je z textového súboru zmazaný

Používateľ je alertom upozornený že uzly/uzol ktorý boli odpojené a teraz sa obnovilo spojenie, boli synchronizované s ostatnými uzlami.

Obrázok, na ktorom je text, snímka obrazovky, vnútri

Automaticky generovaný popis

Obrázok, na ktorom je text, snímka obrazovky, monitor, vnútri

Automaticky generovaný popis

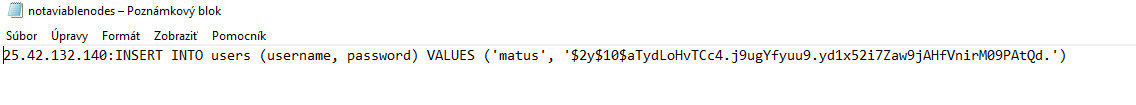
A dáta sa nachádzajú aj na uzle s ktorým sa obnovilo spojenie.

## Ukážka pri výpadku uzla a registrácii používateľa

Ak sa registrujeme a jeden uzol je odpojený tak je logika aplikácie rovnaká.

Obrázok, na ktorom je text

Automaticky generovaný popis



Obrázok, na ktorom je text

Automaticky generovaný popisPre odpojený uzol sa zapíše insert do pomocného textového súboru a po obnovení spojenia je používateľ opäť o tom oboznámený na login.php stránke plus užívateľ je zapísaný aj do tohto uzla.

Ukážka vyhľadávania

Na vyhľadávanie sme použili jednoduchý javascriptovy skript.

Obrázok, na ktorom je text

Automaticky generovaný popis

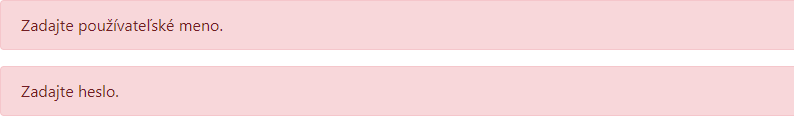
Obrázok, na ktorom je text

Automaticky generovaný popisvyhľadanie filmu interstellar

Obrázok, na ktorom je text

Automaticky generovaný popisvyhľadanie filmu maska

Ošetrili sme si taktiež chybné vstupy

Pokiaľ by užívateľ poslal prázdny registračný formulár

Obrázok, na ktorom je text

Automaticky generovaný popisPokiaľ by zadal heslo kratšie ako 6 znakov

Obrázok, na ktorom je text

Automaticky generovaný popisAlebo pokiaľ by používateľ zadal zle prihlasovacie údaje

# Záver

V tejto ukážke sme si popísali postup, akým sme vytvárali našu distribuovanú aplikáciu. Myslíme si, že sa nám zadanie podarilo splniť v plnom rozsahu, keďže sme zabezpečili ako aj automatickú replikáciu údajov na všetky uzly, tak aj synchronizáciu údajov po tom, ako došlo k výpadku a databázy odpojených uzlov sa dostali do nekonzistencie. Taktiež sme zabezpečili to, že keď nám vypadne spojenie medzi uzlami, tak tieto uzly môžu pracovať autonómne ďalej, ale upravovať môžu iba tie záznamy, ktoré vytvorili(pre ostatné majú zablokované tlačidlá) V tejto ukážke sme opísali hlavnú logiku a funkcie, ktorými náš program zadanú funkcionalitu zabezpečuje. Samozrejme, že zdrojový kód je oveľa rozsiahlejší, ale nevideli sme dôvod čitateľa zaťažovať opisom html css a jquery funkcií, ktoré len robia náš program dynamickejší a štýlovejší, ale neplnia samotnú podstatu zadania.