1. Východiská

1.1 Úvod

Cieľom bakalárskej práce je vytvoriť interaktívnu webovú aplikáciu pomocou JavaScript, HTML5 a node.js servera. Jadrom bude databáza a jej administrácia, ktorá bude obsahovať materiál k jednotlivým mikrokurzom. Pre užívateľa bude výučbový materiál sprístupnený vo viacerých režimoch. Bude realizované prezentovanie slovnej zásoby a niekoľko mikroaktivít na precvičovanie. Ako užívatelia sú predpokladané deti vo veku šesť až desať rokov.

Vo východiskovej kapitole sa pozrieme na samostatnú teóriu a vysvetlenie pojmov potrebných na porozumenie mojej bakalárskej práce. V druhej časti vymenujem použité technológie a nástroje, ktoré je potreba využívať pri programovaní interaktívnej webovej stránky. V poslednej časti ukážem porovnanie podobných bakalárskych prác, ktoré som našiel.

1.2 Teória

1.2.1 Edukačný softvér

Edukačný softvér je softvér, ktorý slúži na podporu učenia sa, rozvoj informačnej gramotnosti a mal by poskytovať spätnú väzbu používateľovi. Mal by pomáhať učiteľovi pri výklade učiva a následné opakovanie. Väčšinou sa rozdeľuje podľa obsahu na jednoúčelové a viacúčelové. Jednoúčelové slúžia len pre jeden vybraný vyučovací predmet. Viacúčelové edukačné softvéry slúžia pre vyučovanie na viacerých predmetoch, napríklad biológii a chémii. Podľa funkcie ich vieme rozdeliť na monofunkčné programy, ktoré disponujú len jednou funkciou, teda jedným učivom. Multifunkčné edukačné programy obsahujú rôzne aktivity, ktoré slúžia používateľom na precvičovanie, simuláciu, alebo aj výklad. Musia spĺňať rôzne technické nároky, kde medzi potrebné parametre môže patriť aj pripojenie na internet. [1]

Moja práca spadá do kategórie jednoúčelový softvér s výukou len pre jeden vyučovací predmet. Bude mať primerané dynamické používateľské prostredie. Bude slúžiť pre

individuálnu prácu žiaka. Softvér bude multifunkčný. Bude obsahovať precvičovanie a výukovú zónu pre žiakov.

1.2.2 E-learning

E-learning je spôsob výuky pomocou elektronických prostriedkov, ktoré by mali uľahčiť dosiahnutie vzdelávacieho cieľa. Môžeme teda tvrdiť, že sa jedná o učenie, kde sú zaradené aj edukačný softvér, alebo rôzne multimediálne prvky ako napríklad prezentácie, alebo edukačné videá. [2]

Môj edukačný softvér je navrhnutý pre deti ktoré sa budú učiť na školských počítačoch, nakoľko ešte nie všetky deti majú prístup k vlastným zariadeniam. Výučba by mala ušetriť čas aj peniaze učiteľovi a deťom, aby si nemuseli nosiť knížky do školy a kupovať drahé učebnice. E-learning začína byť čím ďalej, tým viac rozšírenejší na školách či už v podobe tabletov, počítačových miestností, alebo dokonca interaktívnych tabúľ.

1.2.3 Mikrolearning

Mikrolearning je forma E-learningu populárna v dnešnej dobe. Mikrolearning označuje pokročilý postoj k učeniu. Cieľom je využiť zameranie na učenie veľmi malých častí učiva v krátkom čase neustálym opakovaním jednotlivých častí. Hlavné črty Mikrolearningu sú krátke časové úseky s nízkym úsilím, veľmi malé časti učiva rozdelené na fragmenty. [3]

Tieto črty by mala obsahovať aj moja web aplikácia určená pre deti. Mala by byť zameraná na precvičovanie slovnej zásoby v malých dávkach krátko a často za sebou. Momentálne existujú podobné aplikácie na mobil, ktoré precvičujú slovnú zásobu formou mikrolearningu.

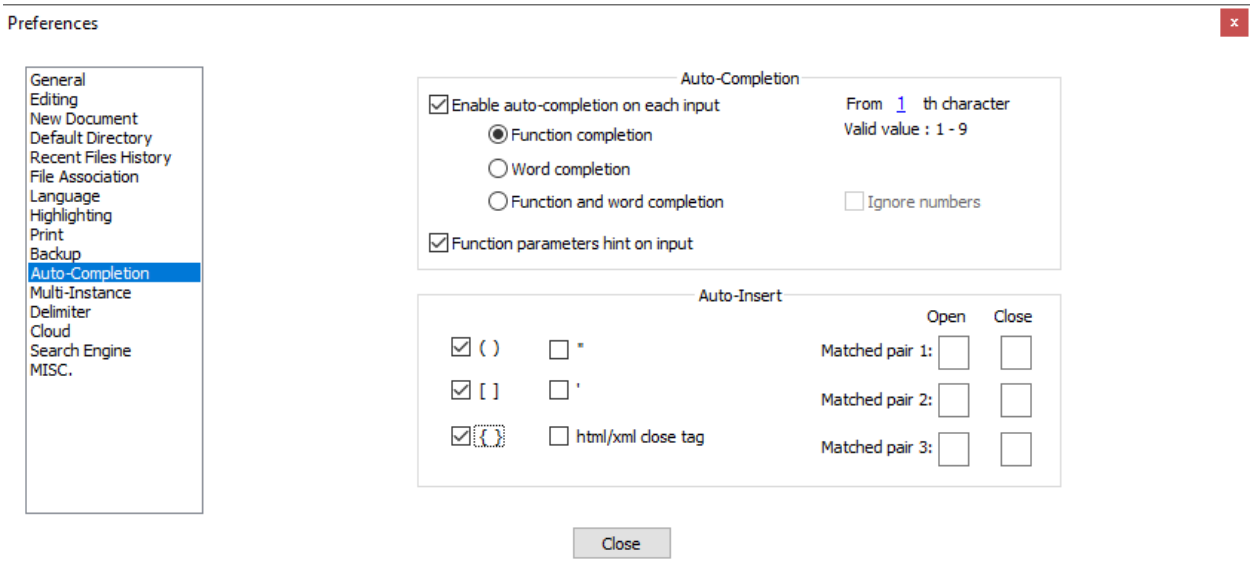
1.3 Využité technológie a nástroje

1.3.1 Notepad++

Notepad++ je textový editor a editor zdrojových kódov. Je to ‚open source' softvér, teda hocikto môže zobrat zdrojový kód editora a pozmeniť si ho podľa vlastných predstáv. Zatiaľ je podporovaný iba na operačnom systéme Microsoft Windows. Podporuje

väčšinu základných programovacích jazykov. Pre nás je dôležitá podpora JavaScriptu. Ak chýba podpora JavaScriptu, dá sa definovať pridaním plug-inu. [4]

Pre nás je dôležité nastaviť si správne zvýrazňovanie textu podľa nami zvoleného jazyka a prácu nám uľahčí aj automatické dopĺňanie funkcii v jazyku JavaScript a automatické dopĺňanie zátvoriek. V Notepade++ budeme editovať celú bakalársku prácu, teda šablóny HTML stránok a súbory s príponou .js.



1.3.2 HTML5

HTML5 je verzia značkovacieho jazyka HTML slúžiaceho na tvorbu webových stránok. Opisuje základnú štruktúru tvorby Web stránok. Elementy sú reprezentované popiskami. Klasický HTML dokument je rozdelený na hlavu a telo. [5]

V mojej práci budem používať HTML na tvorbu základnej web stránky, ktorú budem neskôr použitím EJS používať ako šablónu na vytvorenie efektu ‚single page' stránky, to znamená, že sa všetko deje na jedinej stránke bez potreby opätovného načítania.

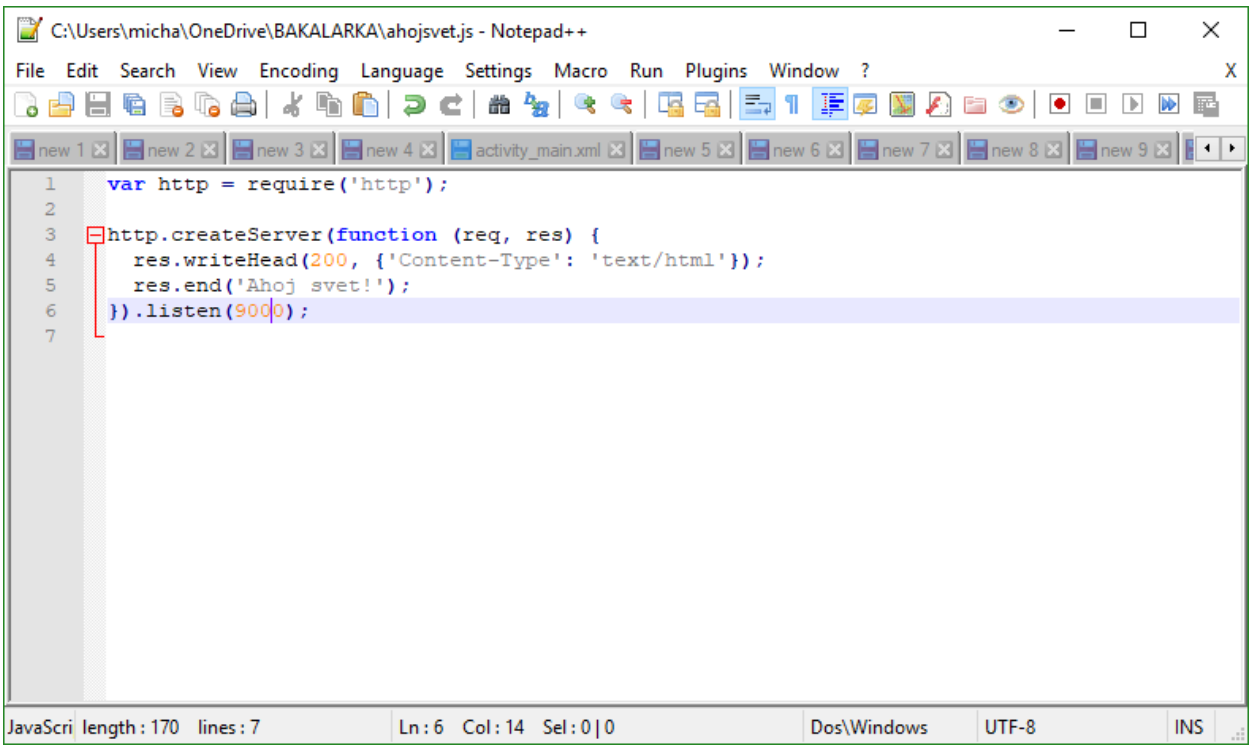
1.3.3 CSS

CSS je jazyk, ktorý popisuje štýl HTML dokumentu. Popisuje ako by mali byť elementy HTML dokumentu zobrazené. Pomocou CSS súboru upravíme vzhľad zobrazovanej stránky podľa štandardov jednoduchého zobrazovania používateľského rozhrania.

1.3.4 JavaScript

JavaScript je skriptovací programovací jazyk. Jazyk je používaný hlavne pri tvorbe webových stránok. Podporuje ho väčšina moderných prehliadačov. Používa sa často na klientovi, kde vykonáva príkazy. JavaScript funguje ako interpreter, teda vykonáva príkazy priamo bez nutnosti predchádzajúceho kompilovania do strojového jazyka. Príkazy sa vykonávajú tak, ako idú za sebou. [6]

Na obrázku vidno ukážku JavaScript kódu jednoduchej aplikácie ktorý zobrazí jednoduchý text na stránke.

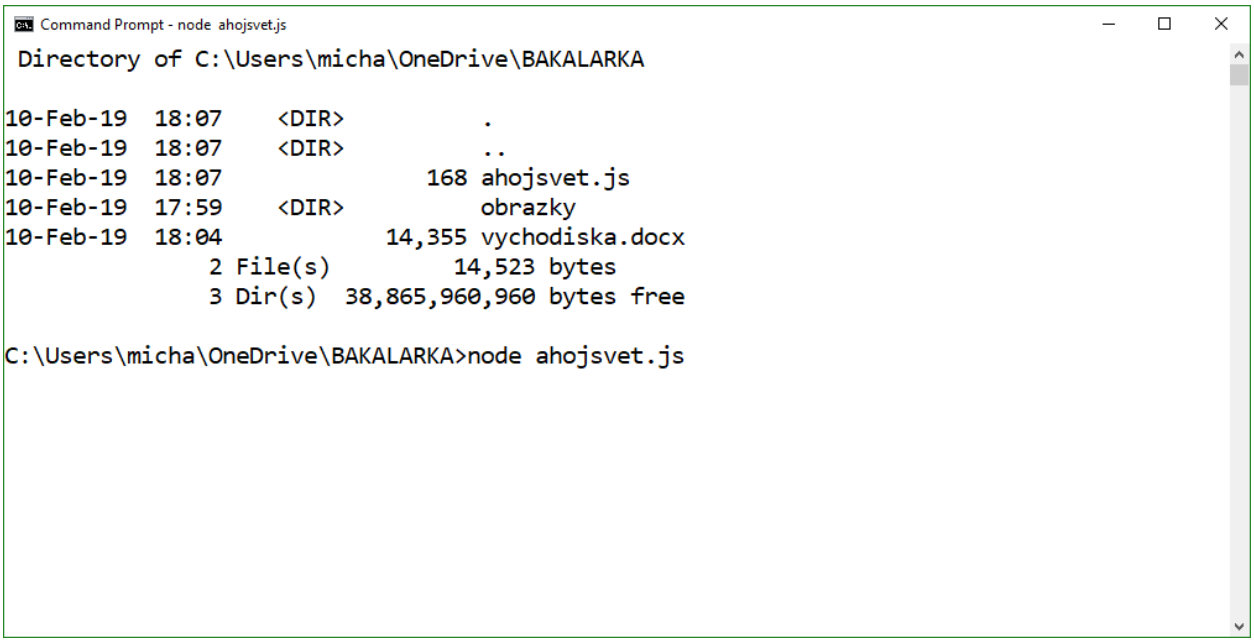


1.3.5 Node.JS

Node.JS je viacplatformové prostredie stále bežiace a vykonávajúce JavaScript kód na strane servera. Používa sa na bežanie skriptov na dynamické vytváranie web stránok pred tým, než je poslaná na klient stranu. Node.JS používa architektúru zloženú na udalostiach z klienta, čo umožňuje asynchrónnu komunikáciu medzi serverom a klientom. [7]

Na beh mojej práce používam Node.JS verziu 10.15.1 ako server na ktorom beží JavaScript kód. Umožňuje aj komunikáciu s databázou. Na obrázku je príkaz na

spustenie servera príkazom node ahojsvet.js a samotný JavaScript program zobrazí jednoduchý text na stránke. Server počúva na porte 9000.



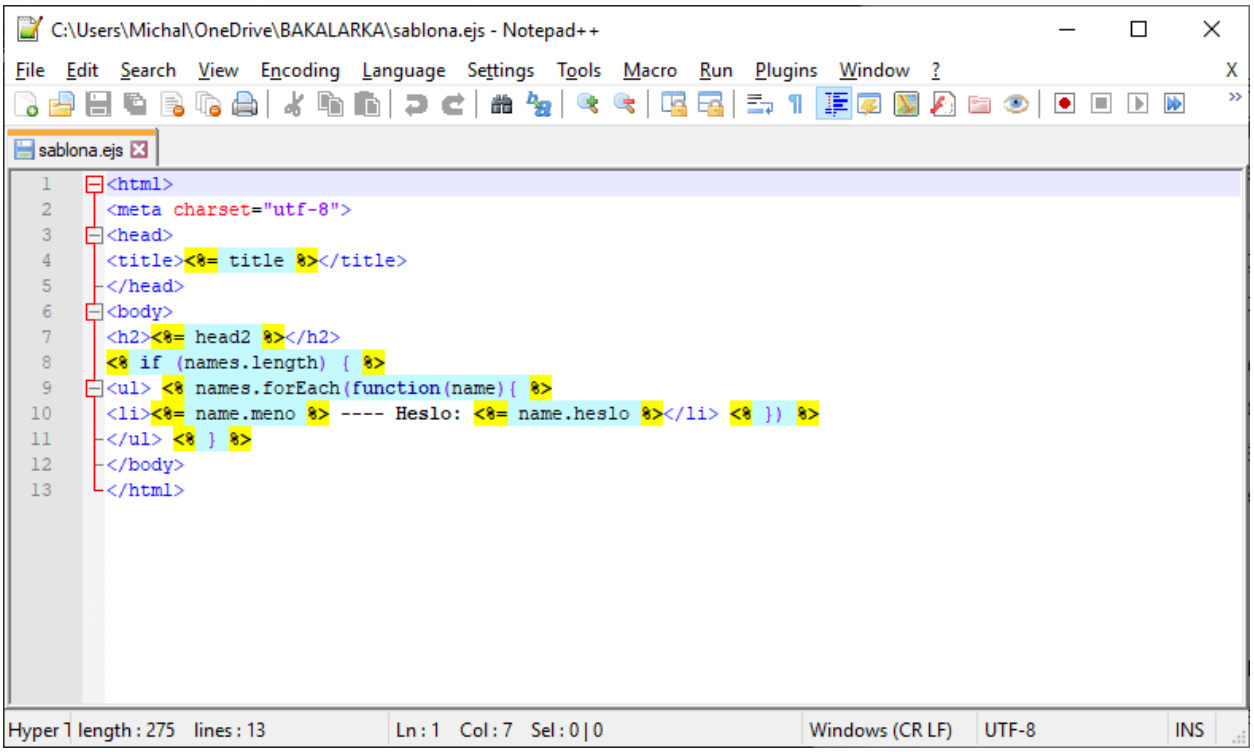
Na obrázku vidíme na klient strane zobrazenú stránku s jednoduchým textom.

1.3.6 Npm

Npm je skrátene Node.JS package manager. Spravuje všetky balíčky potrebné na beh JavaScript programu. Ak potrebujeme použiť nejakú funkciu z daného modulu, ktorý potrebujeme, treba v súbore s príponou .js napísať príkaz require(‘názov modulu‘) a ešte predtým v príkazovom riadku spustiť príkaz npm install názov modulu. Vtedy môžeme používať všetky funkcie, ktoré potrebujeme. Medzi najbežnejšie moduly používané pri tvorbe web stránok je modul s názvom http.

1.3.7 EJS

EJS je skratka pre embedded JavaScript templating, teda preložené vstavané JavaScript šablónovanie na klient strane. Slúži na dynamické vytváranie stránok. Samotná šablóna má formu HTML dokumentu, avšak obsahuje špeciálne popisky EJS, ktoré vyzerajú takto : <%= premenná %>. Celé je to uložené v súbore s príponou .ejs.



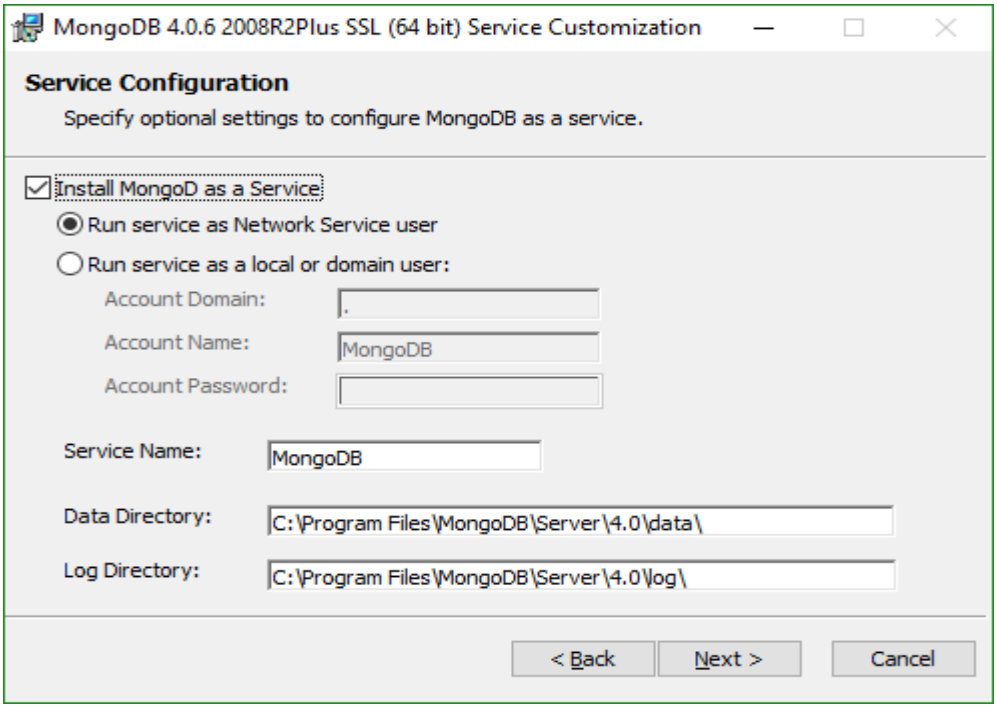
1.3.8 Express

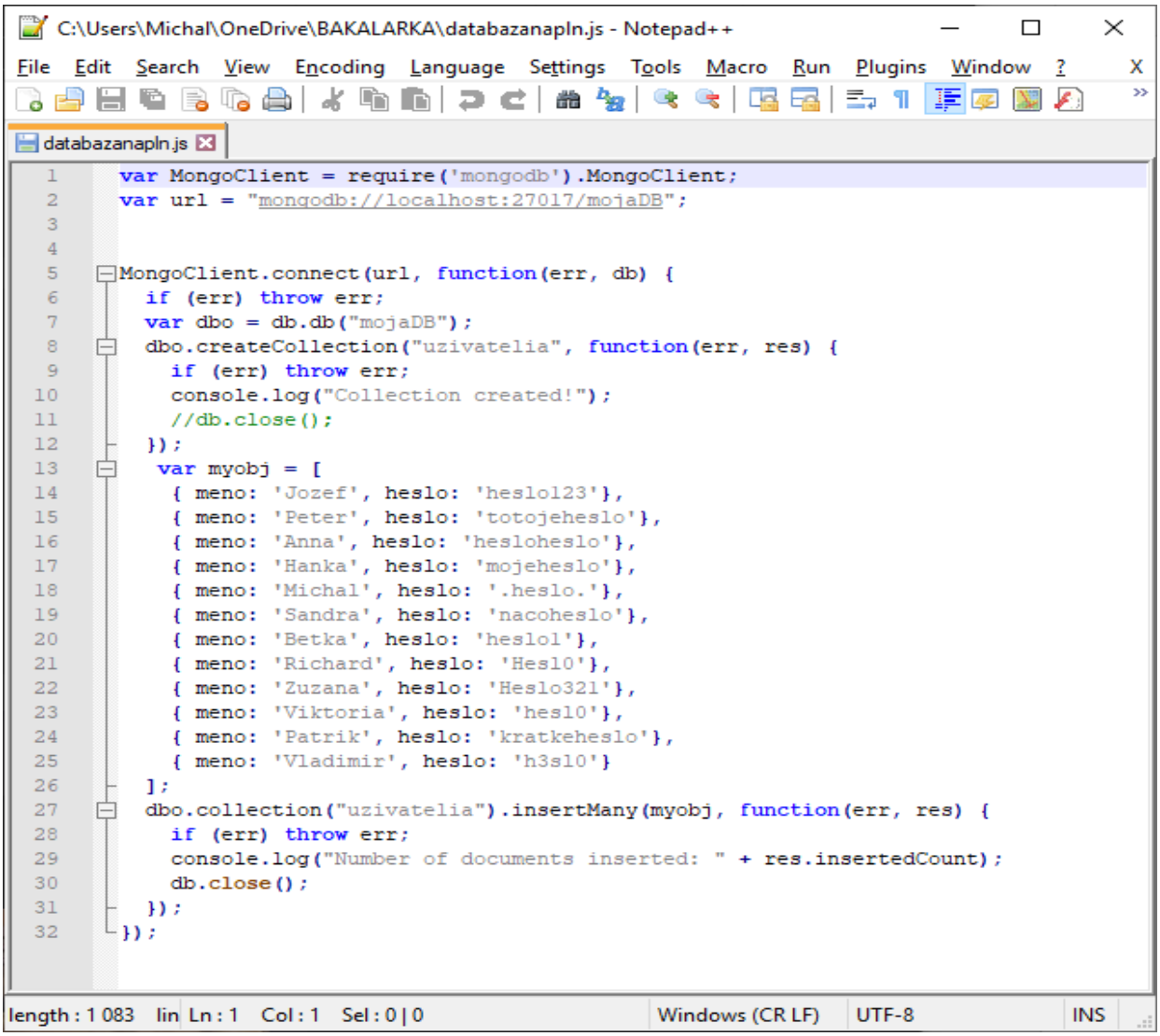
Express, alebo aj Express.JS je serverový JavaScript framework. Mení celú syntax programu, no znateľne zjednodušuje samotný kód.

Inštaluje sa príkazom npm install express –save.

1.3.9 MongoDB

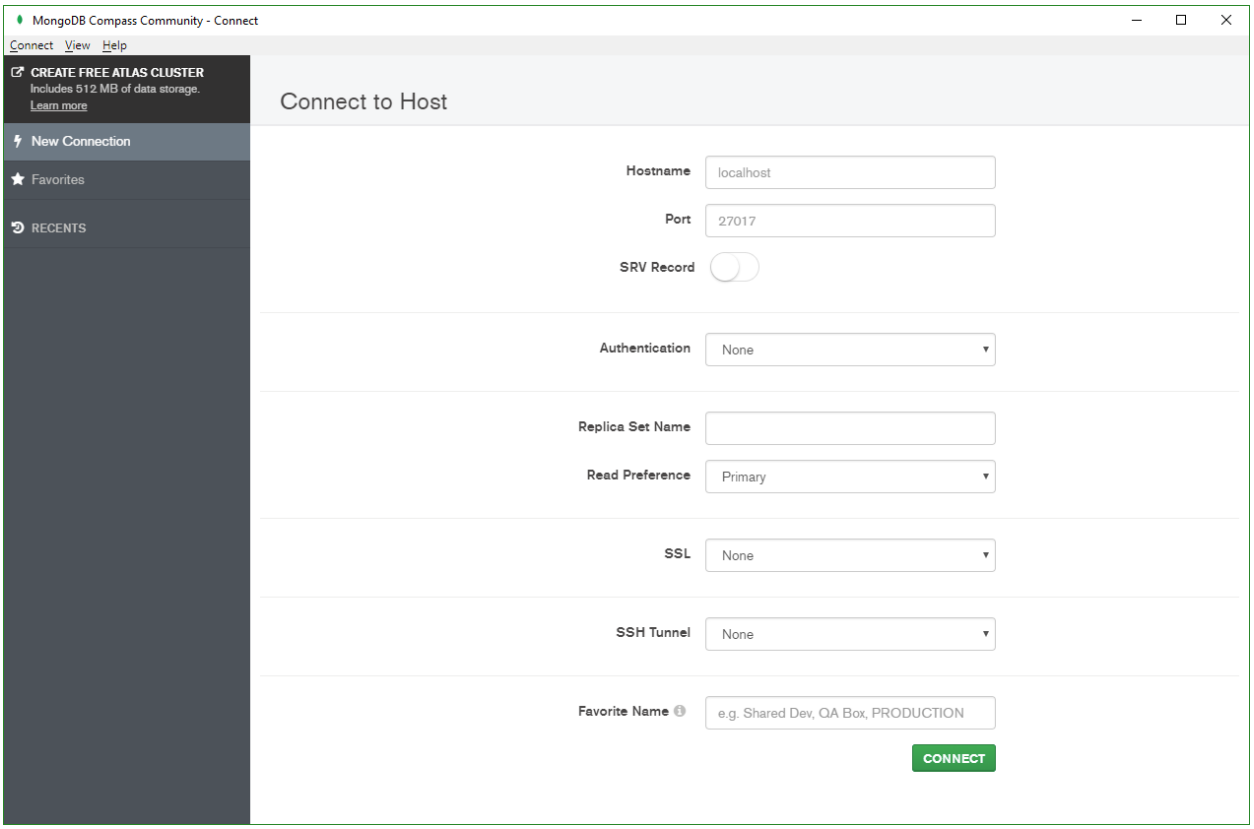
Na perzistenciu dát sa použije objektová NoSql databáza MongoDB. Treba nainštalovať databázový server ako network service user. V mojej aplikácii používam verziu 4.0.6.





1.3.10 MongoDB Compass

MongoDB Compass je používateľské rozhranie pre databázu MongoDB. Pomocou neho sa dajú ľahko vytvárať záznamy a tabuľky. Treba si vytvoriť vlastnú databázu. Budeme používať prednastavený port 27017.



1.4 Podobné bakalárske práce

Tomáš Michalík – Children's activities with cubes for training elementary skills

V práci používa rovnaké technológie ako ja. Rovnako je to edukačná aplikácia s cieľom E-learningu, lenže forma prevedenia je odlišná. Využíva odlišné moduly na okamžitú komunikáciu, pričom ja nič také nepoužívam. Ja som sa zameral skôr na komunikáciu s databázou a na dynamické templatovanie.

Expresný prehľad

Express je minimálna a flexibilná webová aplikácia Node.js, ktorá poskytuje robustný súbor funkcií na vývoj webových a mobilných aplikácií. Uľahčuje rýchly vývoj webových aplikácií založených na Node. Nižšie sú uvedené niektoré z hlavných prvkov rámca Express -

• Umožňuje nastaviť middlewares, aby odpovedali na požiadavky HTTP.

• Definuje smerovaciu tabuľku, ktorá sa používa na vykonávanie rôznych akcií na základe metódy HTTP a adresy URL.

• Umožňuje dynamicky vykresľovať HTML stránky na základe odovzdávania argumentov šablónam.

Inštalácia Express

Najprv nainštalujte rámec Express globálne pomocou NPM, aby ho bolo možné použiť na vytvorenie webovej aplikácie pomocou terminálu uzlov.

$ npm install express --save

Vyššie uvedený príkaz uloží inštaláciu lokálne do adresára node_modulesdirectory a vytvorí adresár express in node_modules. Mali by ste nainštalovať nasledujúce dôležité moduly spolu s Express -

• body-parser - Toto je middleware node.js na spracovanie dát formulárov kódovaných JSON, Raw, Text a URL.

• cookie-parser - Parse hlavičky súboru cookie a vyplní req.cookies objektom, ktorý je zadaný pomocou názvov súborov cookie.

• multer - Jedná sa o middleware node.js na spracovanie multipart / form-data.

$ npm install parser-body -save

$ npm install cookie-parser --save

$ npm nainštalovať multer --save

Ahoj svet Príklad

Nasleduje veľmi základná aplikácia Express, ktorá spúšťa server a počúva na porte 8081 pre pripojenie. Táto aplikácia reaguje s Hello World! pre žiadosti na domovskú stránku. Pre každú inú cestu bude odpovedať 404 Not Found.

```
var express = vyžadujú ('expres');

var app = express ();


app.get ('/', funkcia (req, res) {

  res.send ('Hello World');

})


var server = app.listen (8081, function () {

  var host = server.address () adresa

  var port = server.address ()


  console.log ("Príklad aplikácie počúvajúci na http: //% s:% s", hostiteľa, portu)

})
```

Uložte vyššie uvedený kód do súboru s názvom server.js a spustite ho nasledujúcim príkazom.

Pracovníci knižnice budú používať webovú stránku lokálnej knižnice na ukladanie informácií o knihách a dlžníkoch, zatiaľ čo členovia knižnice ju budú používať na

prezeranie a vyhľadávanie kníh, zisťovanie, či sú k dispozícii kópie, a potom ich rezervujú alebo požičiavajú. Aby sme mohli efektívne ukladať a získavať informácie, uložíme ich do databázy.

Aplikácie Express môžu používať mnoho rôznych databáz a existuje niekoľko prístupov, ktoré môžete použiť na vykonávanie operácií Create, Read, Update a Delete (CRUD). Tento tutoriál poskytuje stručný prehľad niektorých dostupných možností a potom podrobne ukazuje vybrané mechanizmy.

Aké databázy môžem používať?

Aplikácia Express môže používať akúkoľvek databázu podporovanú Node (Express sám nedefinuje žiadne špecifické dodatočné správanie / požiadavky na správu databáz). Existuje mnoho populárnych možností, vrátane PostgreSQL, MySQL, Redis, SQLite a MongoDB.

Pri výbere databázy by ste mali zvážiť veci, ako je časová produktivita / krivka učenia, výkonnosť, jednoduchosť replikácie / zálohovania, náklady, podpora komunity atď. Zatiaľ čo neexistuje jediná "najlepšia" databáza, takmer všetky populárne riešenia by mal byť viac ako prijateľný pre malé až stredne veľké stránky, ako je naša Miestna knižnica.

Ďalšie informácie o možnostiach nájdete v časti Integrácia databázy (Expresné dokumenty).

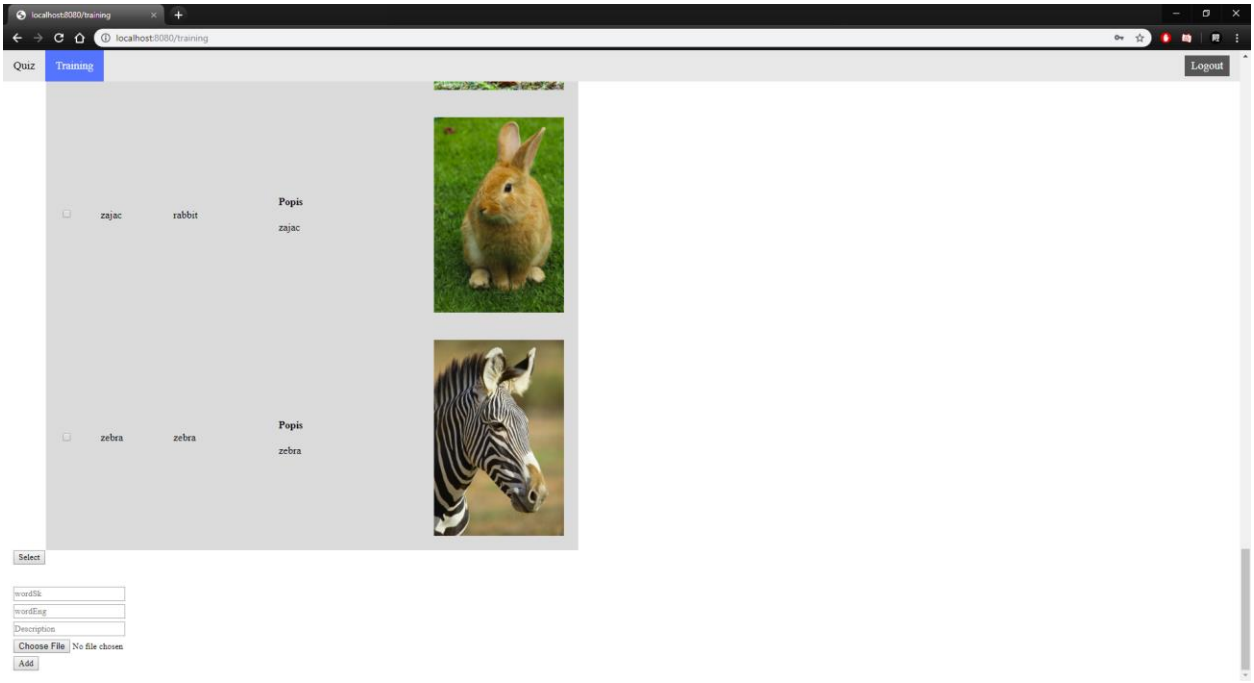
Aký je najlepší spôsob interakcie s databázou?

Existujú dva prístupy na interakciu s databázou:

• Používanie natívneho jazyka dotazov databáz (napr. SQL)

• Použitie objektového dátového modelu ("ODM") / objektového relačného modelu ("ORM"). ODM / ORM predstavuje údaje webovej stránky ako objekty jazyka JavaScript, ktoré sa potom mapujú na základnú databázu. Niektoré ORM sú viazané na konkrétnu databázu, zatiaľ čo iné poskytujú databázovo-agnostický backend.

Najlepší výkon je možné dosiahnuť pomocou SQL, alebo čokoľvek, čo databáza podporuje dotazovací jazyk. ODM sú často pomalšie, pretože používajú prekladový kód na mapovanie medzi objektmi a formátom databázy, ktorý nemusí používať najefektívnejšie databázové dotazy (to platí najmä v prípade, ak ODM podporuje rôzne databázové backendy a musí robiť väčšie kompromisy, pokiaľ ide o databázu) sú podporované).



Výhoda používania ORM spočíva v tom, že programátori môžu naďalej myslieť skôr na objekty JavaScriptu než na sémantiku databáz - to platí najmä vtedy, ak potrebujete pracovať s rôznymi databázami (na rovnakých alebo rôznych webových stránkach). Poskytujú tiež jasné miesto na vykonávanie validácie a kontroly údajov.

Tip: Použitie ODM / ORM často vedie k nižším nákladom na vývoj a údržbu! Ak nie ste oboznámení s natívnym jazykom dotazu alebo výkonom, je dôležité, aby ste zvážili použitie ODM.

Čo mám používať ORM / ODM?

Na stránke správcu balíkov NPM je k dispozícii mnoho riešení ODM / ORM (pozrite sa na značky odm a orm pre podmnožinu!).

Niekoľko riešení, ktoré boli v čase písania populárne:

• Mongoose: Mongoose je nástroj na modelovanie objektov MongoDB určený na prácu v asynchrónnom režimevar express = require('express');

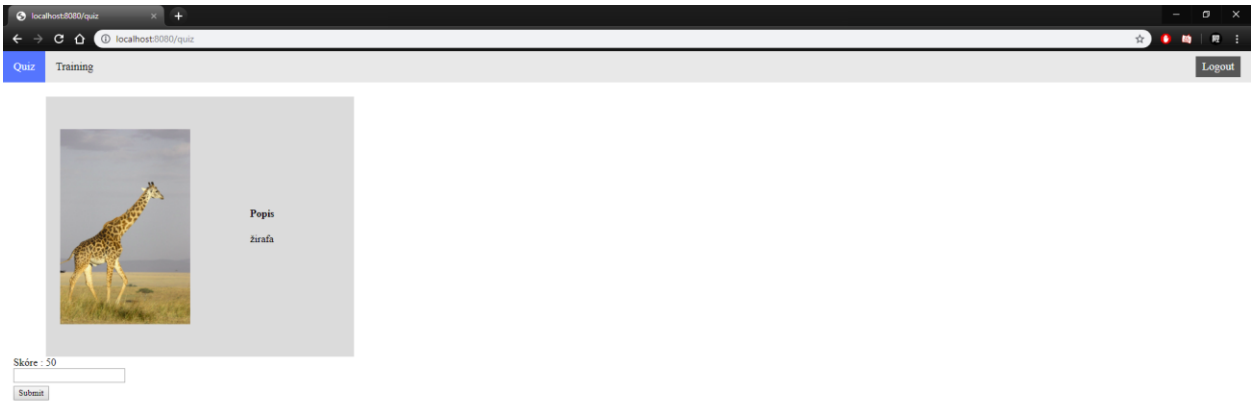
var app = express();

```
app.get('/', function (req, res) {
   res.send('Hello World');
})

var server = app.listen(8081, function () {
   var host = server.address().address
   var port = server.address().port

   console.log("Example app listening at http://%s:%s", host, port)
})
```

Save the above code in a file named server.js and run it with the following command.



Expresný prehľad

Express je minimálna a flexibilná webová aplikácia Node.js, ktorá poskytuje robustný súbor funkcií na vývoj webových a mobilných aplikácií. Uľahčuje rýchly vývoj webových aplikácií založených na Node. Nižšie sú uvedené niektoré z hlavných prvkov rámca Express -

• Umožňuje nastaviť middlewares, aby odpovedali na požiadavky HTTP.

• Definuje smerovaciu tabuľku, ktorá sa používa na vykonávanie rôznych akcií na základe metódy HTTP a adresy URL.

• Umožňuje dynamicky vykresľovať HTML stránky na základe odovzdávania argumentov šablónam.

Inštalácia Express

Najprv nainštalujte rámec Express globálne pomocou NPM, aby ho bolo možné použiť na vytvorenie webovej aplikácie pomocou terminálu uzlov.

$ npm install express --save

Vyššie uvedený príkaz uloží inštaláciu lokálne do adresára node_modulesdirectory a vytvorí adresár express in node_modules. Mali by ste nainštalovať nasledujúce dôležité moduly spolu s Express -

• body-parser - Toto je middleware node.js na spracovanie dát formulárov kódovaných JSON, Raw, Text a URL.

• cookie-parser - Parse hlavičky súboru cookie a vyplní req.cookies objektom, ktorý je zadaný pomocou názvov súborov cookie.

• multer - Jedná sa o middleware node.js na spracovanie multipart / form-data.

$ npm install parser-body -save

$ npm install cookie-parser --save

$ npm nainštalovať multer --save

Ahoj svet Príklad

Nasleduje veľmi základná aplikácia Express, ktorá spúšťa server a počúva na porte 8081 pre pripojenie. Táto aplikácia reaguje s Hello World! pre žiadosti na domovskú stránku. Pre každú inú cestu bude odpovedať 404 Not Found.

```
var express = vyžadujú ('expres');

var app = express ();


app.get ('/', funkcia (req, res) {

   res.send ('Hello World');

})


var server = app.listen (8081, function () {

   var host = server.address () adresa
```

```
  var port = server.address ()


  console.log ("Príklad aplikácie počúvajúci na http: //% s:% s", hostiteľa, portu)
})
```

Uložte vyššie uvedený kód do súboru s názvom server.js a spustite ho nasledujúcim príkazom.

Pracovníci knižnice budú používať webovú stránku lokálnej knižnice na ukladanie informácií o knihách a dlžníkoch, zatiaľ čo členovia knižnice ju budú používať na prezeranie a vyhľadávanie kníh, zisťovanie, či sú k dispozícii kópie, a potom ich rezervujú alebo požičiavajú. Aby sme mohli efektívne ukladať a získavať informácie, uložíme ich do databázy.

Aplikácie Express môžu používať mnoho rôznych databáz a existuje niekoľko prístupov, ktoré môžete použiť na vykonávanie operácií Create, Read, Update a Delete (CRUD).

Tento tutoriál poskytuje stručný prehľad niektorých dostupných možností a potom podrobne ukazuje vybrané mechanizmy.

Aké databázy môžem používať?

Aplikácia Express môže používať akúkoľvek databázu podporovanú Node (Express sám nedefinuje žiadne špecifické dodatočné správanie / požiadavky na správu databáz). Existuje mnoho populárnych možností, vrátane PostgreSQL, MySQL, Redis, SQLite a MongoDB.

Pri výbere databázy by ste mali zvážiť veci, ako je časová produktivita / krivka učenia, výkonnosť, jednoduchosť replikácie / zálohovania, náklady, podpora komunity atď. Zatiaľ čo neexistuje jediná "najlepšia" databáza, takmer všetky populárne riešenia by mal byť viac ako prijateľný pre malé až stredne veľké stránky, ako je naša Miestna knižnica.

Ďalšie informácie o možnostiach nájdete v časti Integrácia databázy (Expresné dokumenty).

Aký je najlepší spôsob interakcie s databázou?

Existujú dva prístupy na interakciu s databázou:

• Používanie natívneho jazyka dotazov databáz (napr. SQL)

• Použitie objektového dátového modelu ("ODM") / objektového relačného modelu ("ORM"). ODM / ORM predstavuje údaje webovej stránky ako objekty jazyka JavaScript, ktoré sa potom mapujú na základnú databázu. Niektoré ORM sú viazané na konkrétnu databázu, zatiaľ čo iné poskytujú databázovo-agnostický backend.

Najlepší výkon je možné dosiahnuť pomocou SQL, alebo čokoľvek, čo databáza podporuje dotazovací jazyk. ODM sú často pomalšie, pretože používajú prekladový kód na mapovanie medzi objektmi a formátom databázy, ktorý nemusí používať najefektívnejšie databázové dotazy (to platí najmä v prípade, ak ODM podporuje rôzne databázové backendy a musí robiť väčšie kompromisy, pokiaľ ide o databázu) sú podporované).

Výhoda používania ORM spočíva v tom, že programátori môžu naďalej myslieť skôr na objekty JavaScriptu než na sémantiku databáz - to platí najmä vtedy, ak potrebujete pracovať s rôznymi databázami (na rovnakých alebo rôznych webových stránkach). Poskytujú tiež jasné miesto na vykonávanie validácie a kontroly údajov.

Tip: Použitie ODM / ORM často vedie k nižším nákladom na vývoj a údržbu! Ak nie ste oboznámení s natívnym jazykom dotazu alebo výkonom, je dôležité, aby ste zvážili použitie ODM.

Čo mám používať ORM / ODM?

Na stránke správcu balíkov NPM je k dispozícii mnoho riešení ODM / ORM (pozrite sa na značky odm a orm pre podmnožinu!).

Niekoľko riešení, ktoré boli v čase písania populárne:

• Mongoose: Mongoose je nástroj na modelovanie objektov MongoDB určený na prácu v asynchrónnom režime

Čo je Node.js?

Node.js je open-source, multiplatformové runtime prostredie používané na vývoj webových aplikácií na strane servera. Aplikácie Node.js sú napísané v jazyku JavaScript a môžu byť spustené na širokej škále operačných systémov.

Node.js je založený na architektúre riadenej udalosťami a neblokujúcom vstupno-výstupnom rozhraní API, ktoré je navrhnuté na optimalizáciu priepustnosti a škálovateľnosti aplikácií pre webové aplikácie v reálnom čase.

Počas dlhého časového obdobia bol rámec pre vývoj webových aplikácií založený na modeli bez štátnej príslušnosti. Model bez štátnej príslušnosti je miesto, kde sa údaje generované v jednej relácii (napríklad informácie o užívateľských nastaveniach a

udalostiach, ktoré sa vyskytli) neudržiavajú na použitie v nasledujúcej relácii s týmto užívateľom.

Bolo potrebné vykonať veľa práce na udržanie informácií o relácii medzi požiadavkami pre používateľa. Ale s Node.js existuje konečne spôsob, ako môžu webové aplikácie mať obojsmerné spojenia v reálnom čase, kde klient aj server môžu iniciovať komunikáciu, čo im umožňuje voľne si vymieňať dáta.

Prečo používať Node.js?

V nasledujúcich kapitolách sa pozrieme na skutočnú hodnotu Node.js, čo však robí tento rámec tak slávnym. V priebehu rokov väčšina aplikácií vychádzala z rámca žiadosti o odpoveď bez štátnej príslušnosti. V týchto typoch aplikácií je na vývojárovi, aby zabezpečil, že bol zavedený správny kód, aby sa zabezpečilo, že stav webovej relácie bude zachovaný, kým používateľ pracuje so systémom.

Ale s webovými aplikáciami Node.js môžete teraz pracovať v reálnom čase a mať obojsmernú komunikáciu. Stav je zachovaný a klient alebo server môžu spustiť komunikáciu.

Vlastnosti Node.js

Pozrime sa na niektoré kľúčové vlastnosti Node.js

Asynchrónne udalosti riadené IO pomáha súbežné spracovanie žiadostí - To je pravdepodobne najväčší predajné miesta Node.js. Táto vlastnosť v podstate znamená, že ak je požiadavka prijatá uzlom pre niektorú operáciu vstupu / výstupu, vykoná operáciu na pozadí a pokračuje v spracovaní ďalších požiadaviek.

To je dosť odlišné od ostatných programovacích jazykov. Jednoduchý príklad je uvedený v nižšie uvedenom kóde

```
var fs = vyžadujú ('fs');
        fs.readFile ( "sample.txt", funkcia (chyba, dáta)
        {
            console.log ("Čítanie údajov dokončené");
    });
```

Vyššie uvedený útržok kódu sa zaoberá čítaním súboru s názvom Sample.txt. V iných programovacích jazykoch by sa nasledujúci riadok spracovania uskutočnil len po prečítaní celého súboru.

Ale v prípade Node.js je dôležitým zlomkom kódu, ktorý si treba všimnúť, deklarácia funkcie („funkcia (chyba, údaje)“). Toto je známe ako funkcia spätného volania.

Takže čo sa tu deje, je to, že operácia čítania súborov začne na pozadí. A iné spracovanie sa môže uskutočniť súčasne počas čítania súboru. Akonáhle je operácia čítania súboru dokončená, táto anonymná funkcia sa zavolá a text "Čítanie údajov dokončených" sa zapíše do denníka konzoly.

Uzol používa modul V8 JavaScript Runtime, ktorý používa prehliadač Google Chrome. Uzol má wrapper cez JavaScript engine, ktorý robí runtime engine oveľa rýchlejší a preto spracovanie požiadaviek v rámci Node sa tiež stáva rýchlejším.

Manipulácia so súbežnými požiadavkami - Ďalšou kľúčovou funkciou Node je schopnosť spracovať súbežné pripojenia s veľmi minimálnou réžiou na jeden proces.

Knižnica Node.js používa JavaScript - Toto je ďalší dôležitý aspekt vývoja v Node.js. Veľká časť vývojovej komunity je už dobre oboznámená s javascriptom, a preto sa vývoj v Node.js stáva ľahším pre vývojára, ktorý pozná javascript.

Pre rámec Node.js existuje aktívna a živá komunita. Kvôli aktívnej komunite sú vždy k dispozícii aktualizácie kľúčov. To pomáha udržať rámec vždy aktuálny s najnovšími trendmi vo vývoji webu.

Kto používa Node.js

Node.js používa široká škála veľkých spoločností. Nižšie je zoznam niektorých z nich.

Paypal - Mnoho stránok v Paypal tiež začali prechod na Node.js.

LinkedIn - LinkedIn používa Node.js na napájanie svojich mobilných serverov, ktoré poháňajú produkty iPhone, Android a Mobile Web.

Mozilla implementovala Node.js na podporu API pre prehliadače, ktoré majú pol miliardy inštalácií.

Ebay je hostiteľom služby HTTP API v službe Node.js

Kedy použiť Node.js

Node.js je najlepšie pre použitie v streamingových alebo event-based aplikáciách v reálnom čase

Chatové aplikácie

Herné servery - rýchle a vysoko výkonné servery, ktoré potrebujú spracovať tisíce požiadaviek naraz, potom je to ideálny rámec.

Vhodné pre prostredie spolupráce - Toto je vhodné pre prostredia, ktoré spravujú dokumenty. V prostredí správy dokumentov budete mať viac ľudí, ktorí uverejnia svoje dokumenty a urobia neustále zmeny pomocou kontroly a kontroly dokumentov. Takže Node.js je vhodný pre tieto prostredia, pretože slučka udalostí v Node.js môže byť spustená vždy, keď sa zmenia dokumenty v spravovanom prostredí.

Zdroje

[1] - © 2012 PaedDr. Stanislav Javorský
http://cec.truni.sk/javorsky/informacna-gramotnost/5_pedagogick_softvr.html

[2] – E-learning
https://cs.wikipedia.org/wiki/E-learning

[3] - Microlearning
https://en.wikipedia.org/wiki/Microlearning

[4] - Notepad++
https://en.wikipedia.org/wiki/Notepad%2B%2B

[5] – HTML5
https://en.wikipedia.org/wiki/HTML5

[6] – JavaScript
https://en.wikipedia.org/wiki/JavaScript

[7] – Node.JS
https://en.wikipedia.org/wiki/Node.js

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Michal Pázmány
Študijný program: aplikovaná informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: aplikovaná informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Edukačná aplikácia na precvičovanie slovnej zásoby
Education application for vocabulary traning

Anotácia: Využitie mikrolearningu sa stáva veľmi populárnym hlavne v oblasti nácviku slovnej zásoby. Či už ide o materinský jazyk alebo cudzí jazyk. Pomocou počítača pripojeného na internet je mikrolearning dostupný v každej chvíli a tak efektívne pomáha pri nácviku.

Cieľ: Vytvoriť interaktívnu webovú aplikáciu pomocou JavaScript, HTML5 a node.js servera. Jadrom bude databáza a jej administrácia, ktorá bude obsahovať materiál k jednotlivým mikrokurzom. Pre užívateľa bude výučbový materiál sprístupnený vo viacerých režimoch. Bude realizované prezentovanie slovnej zásoby a niekoľko mikroaktivít na precvičovanie. Ako užívatelia sú predpokladané deti veku 6 - 10 rokov.

Vedúci: RNDr. Marek Nagy, PhD.
Katedra: FMFI.KAI - Katedra aplikovanej informatiky
Vedúci katedry: prof. Ing. Igor Farkaš, Dr.

Dátum zadania: 15.10.2018

Dátum schválenia: 17.10.2018

doc. RNDr. Damas Gruska, PhD.
garant študijného programu

.....
študent

.....
vedúci práce