**ŽILINSKÁ UNIVERZITA V ŽILINE**

FAKULTA RIADENIA A INFORMATIKY

Prenos dát

*Semestrálna práca*

Tomáš Klein

**Webová aplikácia na spracovanie IOT dát**

Žilina, 2018

1. **Zadanie semestrálnej práce**

Zadanie semestrálnej práce spočívalo vo vytvorení webovej aplikácie, ktorá by spracovávala IOT dáta. Preferované nástroje pre vytvorenie tejto semestrálnej práce bolo využitie jazyka Java, webová platforma DropWizard, databáza MySQL a použitie ORM Hibernate. Na tejto semestrálnej práci môže pracovať tím maximálne troch študentov.

Semestrálna práca má pozostávať z troch komponentov:

1. REST Web servis

2. Prezentačná časť

3. Koncové zariadenie

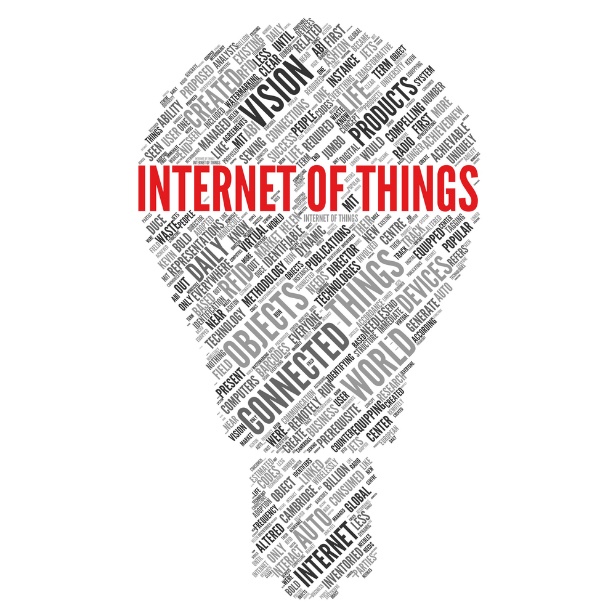
**1.1. REST Web servis**

Tento servis bude prijímať a poskytovať dáta. Daný web servis bude pozostávať z minimálne troch koncových bodov. (užívatelia, zariadenia a dáta pre zariadenia). Nad jednotlivými koncovými bodmi má byť možné vykonať CRUD operácie.

**1.2. Prezentačná časť**

Webové rozhranie, bude zobrazovať namerané údaje, zoznam zariadení atď. Pre lepšie hodnotenie je potrebné implementovať prihlásenie sa do systému pomocou užívateľského mena a hesla.

**1.3. Koncové zariadenie**

****Buď reálne zariadenie postavené na ľubovoľnom MCU, Raspberry Pi alebo emulované zariadenie, ktoré bude spúšťané na PC. Toto koncové zariadenie bude predstavovať senzor(napr. teplota, tlak, svetelnosť, rýchlosť vetra atď. ), ktorý bude odosielať dáta cez REST rozhranie na server.

1. **Teoretický rozbor**
   1. **Webové služby**

Webovú službu (web service) (WS) môžeme definovať ako funkcionalitu jednej aplikácie, ktorú využíva iná aplikácia, za účelom ich vzájomnej interakcie. Jedná sa o strojovú komunikáciu, tzn. že táto komunikácia je napísaná v strojovo spracovateľnom formáte. Bežne priebeha napríklad pomocou posielania správ vo formáte XML a protokolu HTTP(s).

Existuje niekoľko spôsobov a rôznych prístupov ako implementovať WS. Tri najčastejšie spôsoby použitia sú:

* RPC
* SOA
* REST
  1. **REST**

REST (Representational State Transfer) kladie na rozdiel od XML – RPC alebo SOAP dôraz na interakciu so stavovými zdrojmi, nie so správami alebo procedúrami. Je teda orientovaná dátovo. Rozdiel je tiež v tom, že sa nejedná o protokol, ale o architektúru rozhrania.

Medzi základné princípy RESTu patrí:

* každý zdroj musí mať svoj vlastný identifikátor (napr. URL);
* stav aplikácie sa určuje pomocou URL;
* pre získanie a manipuláciu so zdrojmi existuje jednotný prístup, ktorý zahŕňa štyri základné operácie CRUD
* zdroj môže mať rôzne reprezentácie (napr. XML, JSON, SVG), klient následne pracuje s touto reprezentáciou, nie priamo so zdrojom.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Zdroj** | **GET** | **PUT** | **POST** | **DELETE** |
| **URI kolekcia, napríklad http://example.com/resources/** | **Zoznam** (List) URI a prípadne ďalší detaily čle-nov kolekcie. | **Vymeniť** (Repla-ce) celú kolekciu za inú. | **Vytvoriť** (Create) nový záznam do kolekcie. Jeho ID je automaticky pridelené a väčši-nou vrátené touto operáciou. | **Zmazať** (Delete) celú kolekciu. |
| **URI prvku,**  **napríklad http://example.com/resources/142** | **Vrátiť** (Retrieve) reprezentáciu adresovaného člena v kolekcii, vyjadreného vhod-ným internetovým typom média. | **Upraviť** (Update) adresovaný člen kolekcie, alebo – pokiaľ neexistuje – **vytvoriť** (create) ho. | Jednať s adresova-ným členom ako s kolekciou a **vytvo-riť** v ňom novú položku. | **Zmazať** (Delete) adresovaný prvok z kolekcie. |

* 1. **RESTová architektúra**

Aby mohla byť architektúra považovaná za RESTovú, musí spĺňať šesť základách bodov, ktoré definoval Fielding vo svojej práci Architectural Styles and the Design of Network-based Software Architectures:

1. Architektúra klient-server

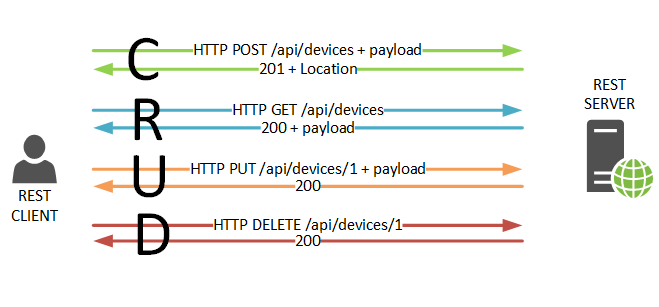
2. Bezstavovosť

3. Cache

4. Jednotné rozhranie

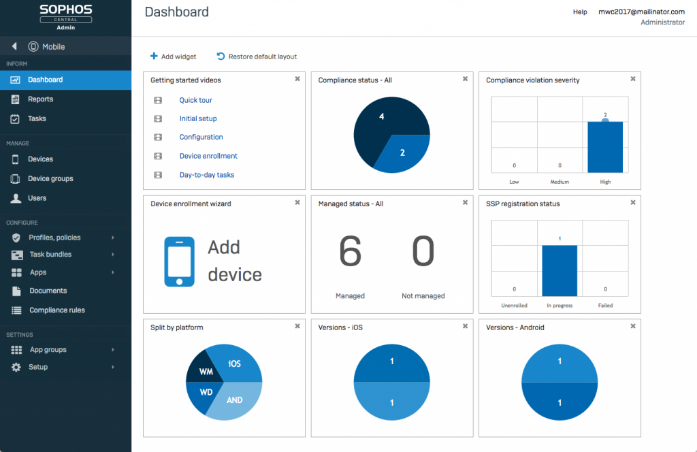
5. Vrstvenie systému

6. Code on Demand

* 1.  **CRUD operácie**
  2. **Porovnanie SOAPu a RESTu**

|  |  |  |
| --- | --- | --- |
| **#** | **SOAP** | **REST** |
| **1** | Protokol založený na posielaní správ vo forme XML | Architektúra rozhrania |
| **2** | Na komunikácia medzi klientom a serverom využíva WSDL | Na odosielanie a prijímanie dát používa XML alebo JSON |
| **3** | Volanie servisov prostredníctvom RPC metód | Jednoduché volanie servisov pomocou URL |
| **4** | Vracia výsledok, ktorý nie je pre človeka čitateľný | Výsledok je čitateľný, nakoľko sa jedná o čistý XML alebo JSON |
| **5** | Tranfer prostredníctvom HTTP, dokáže ale využiť aj iné protokoly, ako napríklad SMTP alebo FTP. | Transfer dát cez protokol, ktorý poskytuje URI. |
| **6** | JavaSript dokáže volať SOAP, ale je to zložité na implementáciu | Jednoduché volania prostredníctvom JavaScrip-tu |
| **7** | Pomalšie spracovanie oproti RESTu. | V porovnaní so SOAPom rýchlejšie |

1. **Návrh aplikácie**

********

***Užívateľské rozhranie***

*umožňujúce ľahkú orientáciu v aplikácií*

***SQL databáza***

*Pre uchovávanie dát*

*Odosielanie   
dát zo* ***senzora***

***Web servis*** *komunikácia pomocou REST API*

**Aplikácia bude spracúvať dáta vytvorené virtuálne emulovaným senzorom. Dáta budú niesť informáciu o teplote a vlhkosti. Dáta sú spracované ako objekty a následne uložené do vzdialenej databázy. Pre ľahšiu orientáciu v nameraných údajoch bude server poskytovať webové stránky s tabuľkou alebo grafom. Užívateľ bude mať možnosť dáta ako upravovať, mazať ale aj pridávať. Takisto bude servis poskytovať základnú autentifikáciu pre možnosť rozdelenia užívateľov do viacerých skupín.**

1. **Popis implementácie**
   1. **Emulovaný senzor – Retrofit**

**Ako koncové zariadenie sme si vybrali emulované zariadenie, ktoré bude náhodne generovať dáta (teplota a vlhkosť) a zasielať ich na webový servis pomocou REST API. Pre zasielanie dát na koncový bod webového servisu sme si vybrali Retrofit. Retrofit je typovo-bezpečný HTTP klient pre Android a Java. Na ich hlavnej stránke môžeme nájsť veľmi prehľadnú dokumentáciu a základný príklady ako s týmto klientom pracovať. Dáta z nášho emulovaného senzoru budú odosielané intervale 5 sekúnd.**

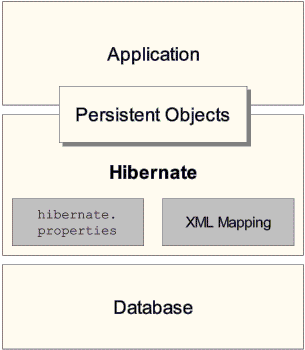
**Príklad: *Request body – pridanie nového užívateľa***

@POST("users/new")

Call<User> createUser(@Body User user);

* 1. **Web servis**

**Pre vytvorenie webového servisu sme si vybrali Java framework Dropwizard, ktorý poskytuje vysoko-výkonné RESTful webové rozhranie**. Dropwizard spája stabilné, kvalitné knižnice z ekosystému Java do jednoduchého, ľahkého balíčka, ktorý vám umožní zamerať sa vývoj kvalitných aplikácií. Dokumentácia na oficiálne stránke je veľmi rozsiahla a poskytuje vývojárovi kvalitné zázemie pre vytváranie ako jednoduchých tak aj zložitejších aplikácií na rôznorodé využitie. Náš webový servis bude spracovávať dáta zo senzora a tieto dáta bude uchovávať v SQL databáze. Takisto bude poskytovať autorizáciu a autentifikáciu pre základné zabezpečenie aplikácie. Aplikácia využíva aj serializáciu dát z JSON do JAVA objektov.

* + 1. **Hibernate**

Hibernate framework je open-source tool pre Java-based object/relational mapovanie. Umoňuje vyvíjať databázové aplikácie pomocou persistentných POJO (Plain Old Java Object) objektov, ktoré sú mapované na tabuľky v databáze. Umo6nuje takisto aj základné JAVA (resp. objektovo orientované) vlastnosti ako dedičnosť, kompozície a podobne.

Hibernate sa nestará len o samotné mapovanie objektov na databázové schémy. Poskytuje aj dotazovací mechanizmus s jednoduchým objektovo-orientovaný HQL jazykom, čo zjednodušuje a zrýchľuje vývoj aplikácií

* 1. **Prezentačná časť**

**Pre korektné využívanie aplikácie sme takisto implementovali grafické rozhranie.   
Ako základ sme si zvolili framework Bootstrap a pre prídavnú fukcionalitu sme využili knižnicu JQuery. Webové stránky sú generované pomocou FreeMarker engine. Využíva sa najmä na generovanie HTML stránok s meniacimi sa dátami.**



* + 1. **Bootstrap**

Front-end framework určený pre dizajn webových stránok a aplikácií s voľnou licenciou (open-source). Využíva CSS preprocesor Less, ktorý generuje CSS súbor zo zdrojového kódu napísaného JavaScript. Všetky informácie pre túto časť práce sme čerpali z oficiálnej stránky, kde je spracovaná kvalitná dokumentácia s množstvami príkladov a riešení.

Jednou z najväčších výhod je responzívny dizajn. Mobilné telefóny a tabletové počítače sa stávajú častejšie využívanými prostriedkami na prezeranie obsahu internetu. ootstrap tento problém rieši hlavne pomocou systému mriežky, ktorá delí obrazovku vertikálne na dvanásť častí - stĺpcov. Pri horizontálnom delení používame označenie rady. Kontajneru, ktorý vkladáme na stránku tak môžeme určiť šírku, respektíve koľko stĺpcov bude zabrať. Pri zobrazení obsahu na obrazovke, ktorá nie je vhodná na zobrazenie všetkých dvanástich stĺpcov v jednom rade, by bol obsah nečitateľný a v tomto sa kontajnery rozdelia do viacerých radov pod seba. Táto funkcia zabezpečí vždy kvalitne zobrazený obsah. Nastaviť môžeme aj odlišnú šírku kontajneru pre rôzne rozlíšenia obrazoviek, tak aby sme vždy dosiahli požadovaný dizajn.

* + 1. **JQuery**

Je JavaScript knižnica so širokou podporou internetových prehliadačov. Kladie dôraz najmä na interakciu medzi JavaScript a HTML. Jej účelom je uľahčenie používania JavaScript na webovej stránke. V našej práci túto knižnicu budeme využívať najmä na dynamickú grafickú zmenu prvkov, kontrolovanie zadaných hodnôt, vytváranie grafov a tabulky. Zjednodušuje tiež manipuláciu s DOM (Document Object Model - objektovo orientovaná reprezentácia HTML). DOM sa využíva pri zvolení prvku v HTLM kóde pre následnú zmenu jeho vlastností, čiže dynamickosť stránky. Pre zvolenie prvku musí byť zadefinovaný jeho identifikátor.

Knižnica jQuery obsahuje funkcie:

* Manipulácia s HTML/DOM
* Manipulácia s CSS
* Rôzne efekty a animácie
* AJAX
* Rôzne podporné programy

1. **GUI**
2. **Zdroje**

* <http://www.baeldung.com/retrofit>

|  |
| --- |
| * Diplomová práca – Univerzita Tomáša Bati v Zlíne – Fakulta aplikované informatiky – Bc, Mgr. Zita Bajúszová - Návrh na vytvoření REST / SOAP služby pre přístup do databáze – 2017 * <https://www.dropwizard.io> * http://neuron.tuke.sk/kostelni/tutorialy/hibernate/index.html |

* Bakalárska práca – Žilinská unverzita v Žiline – Fakulta riadenia a informatiky– Tomáš Klein - Webové služby pre vstavané systémy – 2017