**ŽILINSKÁ UNIVERZITA V ŽILINE**

FAKULTA RIADENIA A INFORMATIKY

**WEBOVÁ APLIKÁCIA PRE SPRACOVANIE IOT DÁT**

SEMESTRÁLNA PRÁCA

Predmet: Prenos dát

**Ján Gáfrik**

**Michal Badura**

**Žilina, 2017 Ľubomír Štec**

# Úvod

Na predmete Prenos dát sme dostali ako semestrálnu prácu navrhnúť webovú aplikáciu pre spracovanie IOT dát. Na tejto práci sme pracovali v tíme troch študentov. Podľa zadania táto aplikácia pozostáva z troch komponentov: Web servisu, prezentačnej časti a koncových zariadení(skutočné alebo simulované).

V tejto práci sú odosielané, simulované hodnoty senzorov pomocou navrhnutej aplikácie. Senzory posielajú informácie o stave ovzdušia pomocou REST servisu do databázy hodnôt senzorov. Tie sa prostredníctvom REST servisu zobrazujú vo webovom rozhraní. K týmto údajom vo webovom rozhraní sa dostanú len prihlásení užívatelia a administrátor. Užívateľ má možnosť zobraziť si históriu nameraných dát zo senzorov a upravovať vlastný profil. Administrátor môže pridávať a mazať nové senzory a mazať registrovaných užívateľov.

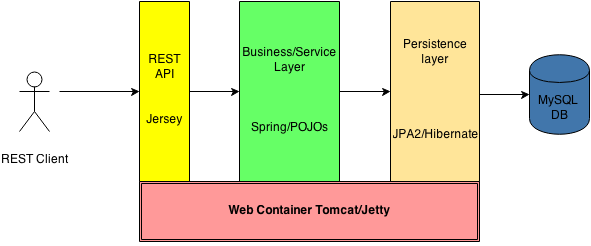
# Špecifikácia zadania

V úvode už bolo spomenuté, že navrhnutá aplikácia pozostáva z troch komponentov.

1. Web servis (REST): prijíma a poskytuje dáta medzi koncovými bodmi(simulované hodnoty pre senzory, samotné senzory a užívatelia). Pre každé koncové zariadenie sú implementované základné CRUD(create, read, update, delete) operácie.
2. Prezentačná časť: webové rozhranie, ktoré zobrazuje zoznam senzorov a umožňuje zobrazenie nameraných hodnôt. K týmto údajom majú prístup len registrovaní resp. prihlásení užívatelia.
3. Koncové zariadenia: môže byť reálne alebo simulované zariadenie(my sme sa rozhodli použiť simulované zariadenie). Toto simulované zariadenie predstavuje niekoľko senzorov na meranie: teploty, rýchlosti vetra, vlhkosti, tlaku ovzdušia a prašnosti.

# Analýza problému

## 2.1 REST servis

Hlavným prvkom webovej aplikácie je Web servis (REST). REST je architektúra rozhrania, navrhnutá pre distribuované prostredie. Jeho hlavnou úlohou je prenášať dáta medzi klientom a serverom pomocou jednoduchých HTTP volaní.

Veľkou výhodou REST-u je jednoduché pristupovanie k dátam a práce s nimi pomocou HTTP volaní. REST servis sa skladá z viacerých podservisov (resource). Každý „resource“ musí mať jedinečný identifikátor(user, sensor, value) a prístup k funkciám je definovaný pomocou CRUD operácii: create, read, update, delete. URL by mala byť v tvare ...servis/resource/metóda/parametre. Základne metódy pre prácu s dátami sú : GET (získanie dát), POST (vytvorenie dát), PUT (úprava dát) a DELETE (zmazanie dát). Výstupné dáta z REST-u môžu byť vo formáte: XML, JSON, HTML, PDF.

## 2.2 Hibernate

Hibernate je framework, ktorý umožňuje objektovo-relačné mapovanie. Slúži na prepojenie databázy s java objektami. Využíva mapovacie súbory v ktorých má definované akým spôsobom sa majú dáta z objektu transformovať do databázy a naopak. Ďalej má v sebe uložené prístupové údaje k databáze a mapovanie k jednotlivým tabuľkám a POJO súborom. POJO sú súbory kde sa definujú parametre tabuľky z databázy a príslušne funkcie(get-re, set-re) na prácu s nimi.

## 2.3 AngularJS

Pri tvorbe webové rozhrania sme použili AngularJS čo je javascriptový webový framework. Môže sa pridávať do HTML stránok cez tag <script>. Slúži na oddeľovanie aplikačnej logiky od zobrazovacej logiky. Je to knižnica napísaná v javascripte a pre jej použitie je nutné zavolať zdrojovú knižnicu. Tento framework sme použili na vytvorenie back-end vrstvy webového rozhrania.

# Implementácia

Ako server sme použili localhost, ktorý beží na Wampserveri na adrese: [*http://localhost:8080/*](http://localhost:8080/)

Prenos dát medzi databázou a webovým rozhraní zabezpečí REST servis. Webové rozhranie má funkcie: prihlásenia, registrácie, zobrazenia všetkých užívateľov a pridávanie nových senzorov(len administrátor), zobrazenie senzorov, ich hodnôt a históriu posledných desiatich nameraných hodnôt(v tabuľke aj v grafe), editovanie užívateľského profilu a odhlásenie užívateľa.

Na simulovania senzorov máme samostatný program, ktorý náhodne generuje hodnoty a odosiela ich cez REST do databázy.

## 3.1 Popis REST rozhrania

Implementovaný REST servis pozostáva z troch „resource-ov“(pre tabuľku senzorov, užívateľov, hodnôt), každý pre jednu tabuľku našej databázy. V resource sú použité základné metódy pre prácu s dátami GET a POST. Ďalej sú implementované CRUD operácie ako napríklad: *getSensorList, saveDataSensor, updateSensor, deleteSensor.* K týmto funkciám sa pristupuje pomocou URL adresy:

[*http://localhost:8080/TestRestful/webresources/user/getSensorList*](http://localhost:8080/TestRestful/webresources/user/getSensorList)*? +* parametre

Výstupom sú dáta vo formáte json:

*{ "sensor": [ {"idSensor": 12, "name": "FR25", "type": "teplotny", "value": 12 ] }*

Každý „resource“ využíva metódy java objektu ,ktorý je tiež pre každú tabuľku vlastný. V ňom sú implementované funkcie na prácu s dátami v databáze pomocou select-ov.

Na prepojenie databázy a java objektov sme použili hibernate a pojos súbory, ktoré nám vygeneroval automaticky Netbeans. V hibernate bolo potrebné nastaviť cestu k databáze a prihlasovacie údaje. V pojos sú uložené parametre každej tabuľky (mapovanie) v databáze a príslušne metódy na prístup k hodnotám tabuľky.

## 3.2 AngularJS implementácia

Angular sa v html používa pomocou direktív, ktoré začínajú prefixom ng-.

Inicializovanie aplikácie sa vkladá tagom ng-app=" " do html objektu. Pomocou scriptu sa v aplikácii vytvorí modul angular.module(), ktorým sa nakonfiguruje úvodná stránka. Do modulu sa postupne vkladajú kontroléry, ktoré slúžia na ovládanie skriptov bežiacich na stránke. Kontrolér sa pripája k HTML dokumentu prostredníctvom direktívy ng-controller=" ". Každý kontrolér v sebe definuje $scope, ktorý drží referencie na aktuálne premenné.

**function** MyController($scope) {

$scope.message = "Hello, world";

}

Komunikácia a prenos hodnoty parametrov medzi kontrolérmi zabezpečuje $rootScope, ktorý je rodičom všetkých $scope.

## 3.3 Java Generátor hodnôt

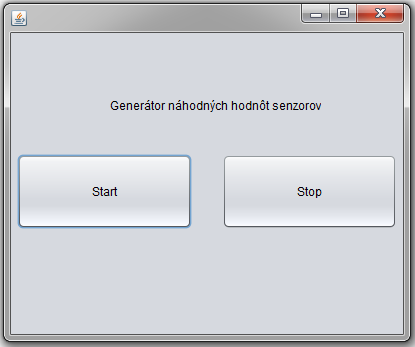
Pre simulovanie náhodných hodnôt sme spravili jednoduchú JAVA aplikáciu, ktorá pomocou RESTu komunikuje so serverom.

Aplikácia obsahuje tri jednoduché triedy. Trieda *RandomNumers, FakeSenzorGenerator,* a *SenzorSimulator.*

V triede *RandomNumers* je metóda *getRandomNumbers.* Vstupnými parametrami sú pole hodnôt, v ktorom všetky prvky posunie a na prvé miesto vloží náhodnú hodnotu. Druhým a tretím parametrom je spodná a horná hranica čísel, z ktorých sa vyberá náhodná hodnota.

Trieda *FakeSenzorGenerator* plní úlohu komunikačného modulu. Metóda *SendPost* má ako vstupný parameter pole, ktoré odosiela a typ senzora pre ktorý sú hodnoty určené. V metóde sa vyskladá URL adresa vo formáte: *"http://localhost:8080/TestRestful/webresources/value/updateValueByType?type="*

+ typ(ktorý príde ako vstupný parameter) + samotné hodnoty

 *SenzorSimulator* implementuje predchádzajúce dve metódy spolu s GUI rozhraním pre užívateľa. Trieda spúšťa dve vlákna. Vlákno v ktorom beží *swing* grafické rozhranie a mouseListener. V druhom vlákne prebieha engine aplikácie spolu s komunikáciou pomocou RESTu so serverom.

# Záver

Cieľom semestrálnej práce bolo navrhnúť webovú aplikáciu pre spracovanie IOT dát zo senzorov pomocou REST servis-u. Bolo potrebné naučiť sa ako pracovať s REST servisom a pri tvorbe webového rozhrania s AngularJS. Toto sa nám úspešne podarilo a vytvorili sme danú aplikáciu, ktorá pracuje spoločne so simulátorom senzorov.