smartes Thermostat Fußbodenheizung

Showcase Projekt zur Erprobung des Zusammenspiels von ioT, Cloud und Business Intelligence

# Der Plan

1. IoT device ersetzt herkömmliches Thermostat zur Steuerung der Fußbodenheizung
2. Sensordaten werden an eine Anwendung in der Cloud geschickt und verarbeitet
3. Steuerungsfunktion übermittelt Anweisungen an iot-Thermostat

# Die Komponenten

1. IoT device:  
   Bestandteile: Microcontroller, Klima-Sensor, Display, Potentiometer, Relais, Netzteil.  
   Der Microcontroller kommuniziert mit dem Backend um:  
   Das Gerät am Service anzumelden und zu authentifizieren,  
   Instruktionen vom Service abzuholen,  
   Messdaten an den Service zu senden.
2. Web Service Backend:  
   Stellt die notwendigen REST Schnittstellen bereit um:  
   Iot Geräte zu verwalten und zu verifizieren,  
   Instruktionen wie z.B. ein geändertes Heizprogramm an das IoT device zu senden,  
   Sensordaten zu empfangen um diese dann weiter zu verarbeiten – beispielsweise charakteristische Kennzahlen des Raumes zu erheben.
3. Statistik-Engine:  
   Führt die Auswertung von Sensordaten durch mit dem Ziel:  
   Relevante Kennzahlen des Raumes und der Wohnung ermitteln, die vom Heizprogramm des Thermostats verwendet werden können.

# Dokumentation IoT device

# Dokumentation Backend

## Architektur Schema

## Schnittstellendefinition

### Anmeldeservice

REST Schnittstelle zum Anmelden eines Geräts.

URL: /api/Register?id=#device\_Guid#&pin=#secure\_pin#  
http Methode: GET

Input: parameter id und pin

Rückgabe:

http status 200 o.k. wenn id und pin verifiziert werden konnten.   
 json Result {„servertime“:#secondsSince1970}  
http 400 bad request wenn id und pin ungültig sind.

### Status update Service

REST Schnittstelle zum Empfang von Statusinfos eines Geräts

URL: /api/Status  
http Methode: POST  
Content: json Objekt vom Typ deviceEvent  
Beispiel:   
{“deviceId”=”#id”,  
“eventkey”: “string”,  
“eventvalue”: “string”}

Input: json Objekt

Rückgabe:

http status 200 o.k. wenn Objekt empfangen und verarbeitet werden konnten.   
http 400 bad request wenn nicht.

### GetProgram Service

REST Schnittstelle zum Empfang von Statusinfos eines Geräts

URL: /api/Program?id=#device\_Guid  
http Methode: GET  
Content: json Objekt vom Typ ‘Program’

Datentyp Program:

String sensorId  
List<programItem>

Datentyp programItem:

String progItemId  
int seconds  
float targetValue

### Ambiences Service

REST Schnittstelle zum Empfang von Sensordaten

URL: /api/PostAmbience/  
http Methode: POST  
Content: json Objekt vom Typ Ambience  
Beispiel:  
{"AmbienceId": 0, "captureTime": "1970-01-01T00:00:00.000", "temperature":"21.5", "humidity":"55,9", "SensorId": "00000000-0000-0000-0000-000000000005"}