SMART SHOP PROJECT

Aim

This software creates dynamic environment conditions to fit people moods in order to generate a more suitable place for purpose of the area.

It can be used to make a shop more endearing, to encourage the customers to buy our products, or it can be used in a garden to let the area more relaxing for visitors, or it can be used in a cinema to get more addictive the projections.

How it works

It works on sensorial environment conditions like lights, temperature, color and music to generate a more suitable place for the people inside. Linking this software with sensors and actuators let the system to control all these variables and modify them with a custom logic implemented in a decision service.

Structure

*This prototype software deploys some sample **RESTful microservices that communicate between HTTP protocol**. The microservices' net is used to collect all the data from sensors microservices, to process them in a decision service and to communicate to actuator microservices the action needed.*

What's a RESTful microservice?

A RESTful web microservice implements REST architecture.
REST means "Representational State Transfer" and it is an "architectural style". A REST architecture has a **complete independent logic for server and client** and the communication between client and server is **stateless**: every message sent by a client to the server must be like it is his first message, not related to previous communication.

Why choose REST architecture?

The most important advantages of REST are **reliability and scalability** due to the clear separation between client and server and their complete independent deployment. This let also to use **different code languages and specific technologies for each service**.

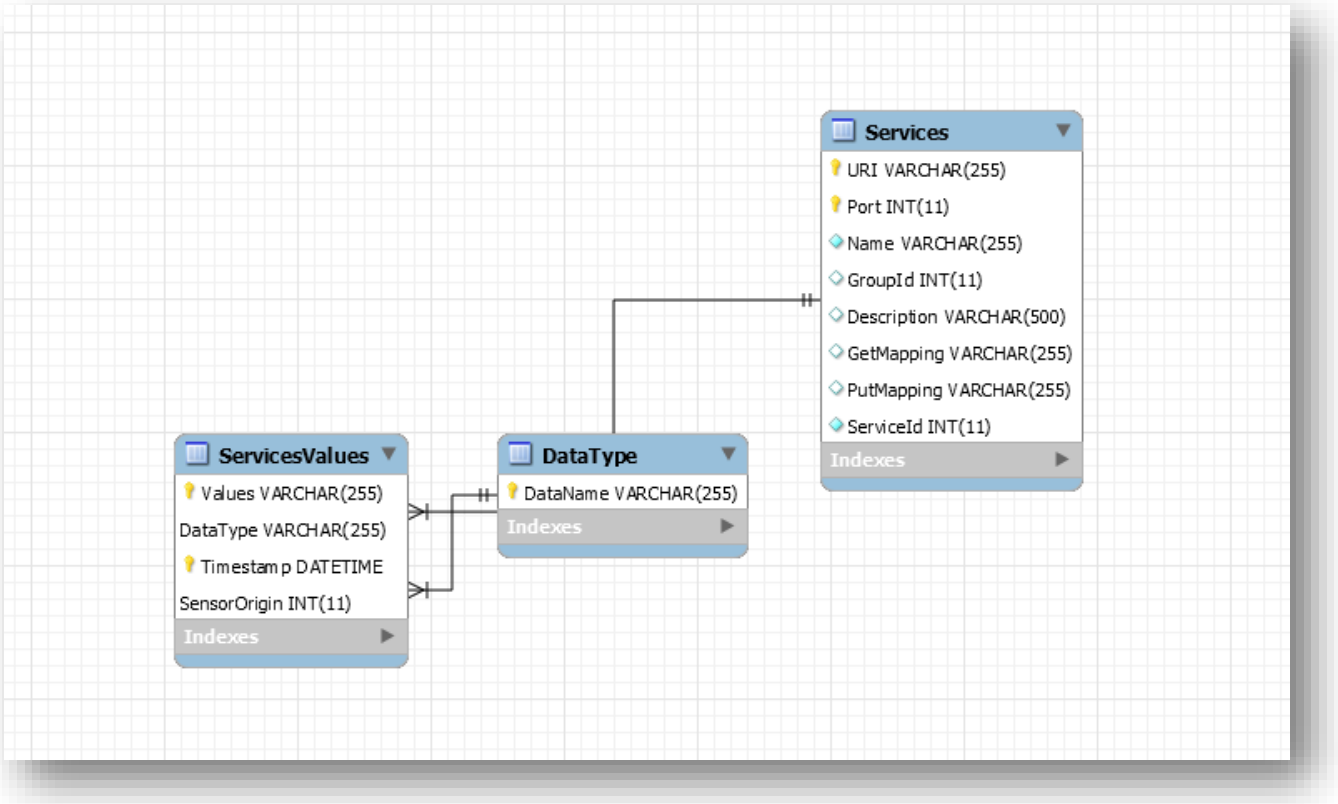
Project details

We used as code language **Java** using **version 8**.

We exploit **"Spring" framework to implement REST architecture** and deploy microservices with an embedded version of **"Apache Tomcat" (service deployer)** included in project's JAR package. We initialized the project with "Spring Boot" and managed compilation with **Maven.**

Database integration:

We saved collected data inside a Relational Database handled by **MySQL** RDBMS to retrieve old information that can be used to predict future conditions. Database tables logic is:



The access to the DB is managed by a Core Decision Service throw a **jdbc connector driver** obtainable from https://dev.mysql.com/downloads/connector/j/.

Core Decision Service

We deployed our core service on a **hosted VPS** (Virtual Private Server) with trusted network, 2-factor authentication and snapshot backup system.

This microservice run as system service and permits communications between all services and grants consistency of the ecosystem and process other services data inputs.