IoT project
Room selection decision support system
2024-2025

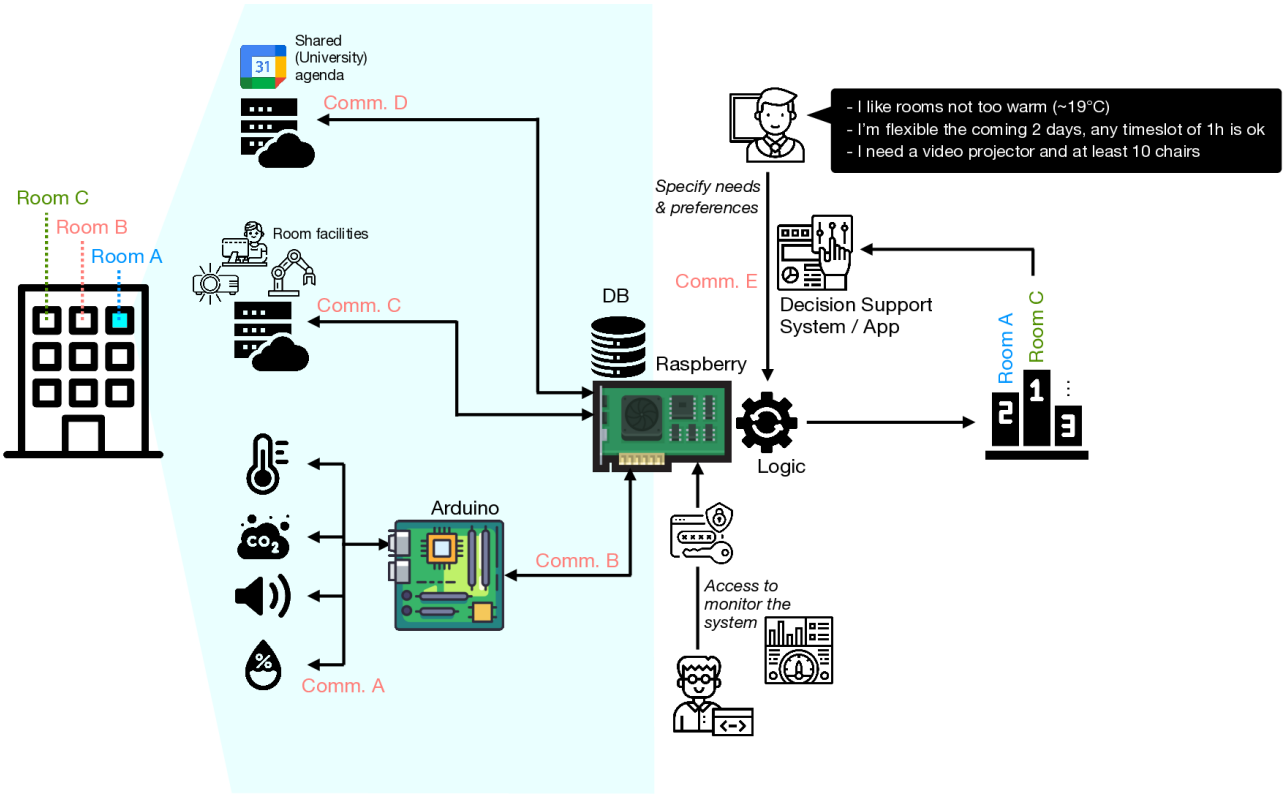

Figure 1 – Design of EV parking station service based on HTTP communications

1 System architecture requirements

A client ask you to design a decision support system to help students and teachers to select the most appropriate room of a public building based both on the characteristics of the rooms (incl., how healthy the room is considering its temperature, Co2 level, air quality, room facilities...) and the preferences expressed by the end-user (i.e., the student or teacher). Figure 1 provides an overview of the overall system architecture to be designed and implemented. Your job/task is the following :

1. Specify (and justify) what communication protocols you would recommend to use for communications denoted by Comm A, B, C, and D (Comm. E being specified as a REST API) ;
2. Specify and implement an appropriate database to store the sensor-related data and room facilities-related data. Your choice must be justified
3. Define the set of criteria of your decision tree, which must be motivated and, to the best possible extent, supported by regulations and/or standards at the European or National level.

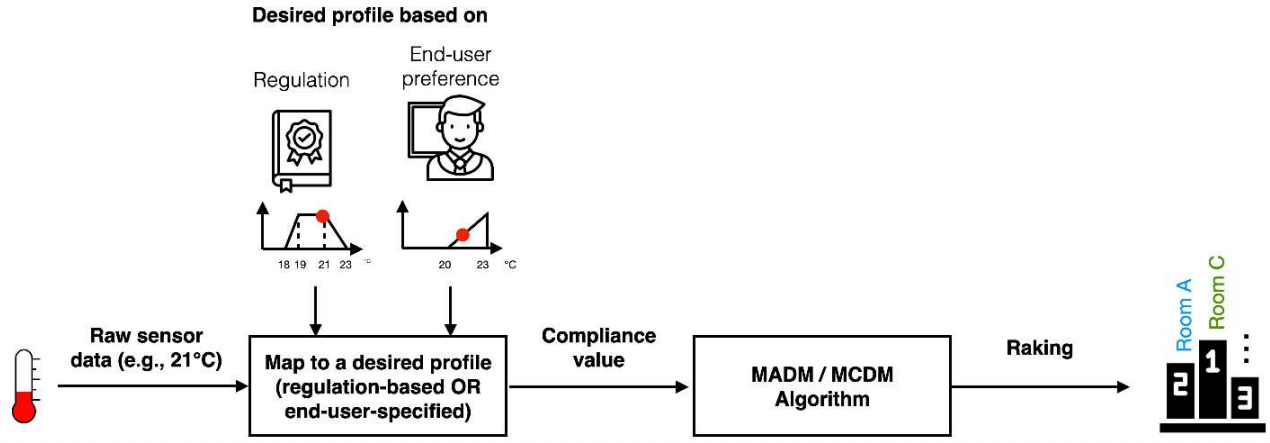

Figure 2 – Design of EV parking station service based on HTTP communications

4. After having specified the criteria, design, develop and implement an algorithm capable of combining and aggregating the different criteria and end-user preferences to end up with a final score for each room and thus a ranking of the rooms (from the most appropriate one to the less appropriate). Figure 2 provides an overview of the pre-processing step you need to do to convert (map) the raw sensor data to a number (value) compliant with the desired profile, which is by default specified based on recommendation (from EU, regulatory framework, etc.), and that **can be adapted** by the end-users depending on their preferences.
   — *Tip : this problem can be seen as a multi-criteria decision making (MCDM) problem. I suggest you to have a look at AHP (Analytic Hierarchy Process) or similar techniques (e.g., TOPSIS, MUAT...) as starting point for the design of your algorithm*
5. Specify the REST APIs of your Decision Support System (i.e., Comm. E), and use a widely adopted open-source tool such as Swagger (https://swagger.io) to document all your APIs to make third party developers able to use your system. Two sets of REST APIs should be designed and released, namely :

   REST API1 providing access to all input data of the Decision Support System, including (i) temperature, Co2, sound, humidity values over a specific period of time ; (ii) room facilities of a given room ; (iii) calendar events over a specific period of time ;

   REST API2 providing access to the results of the Decision Support System, where the end-user's inputs (preferences, requirements, ...) shall be specified via REST API parameters.

   As a first step, these APIs are made available for free (publicly available), and then – *if time permits (optional / bonus point)* – make your REST APIs secured using JSON Web Tokens (JWT) [1].
6. Design two distinct user interfaces, namely :

   UI1 providing access to end-users to outputs/results of your decision support algorithm (i.e., final scores and ranking between rooms). This UI interface must allow end-users to both specify their preferences between criteria using the Saaty scale(e.g., Temperature 5x more important than Luminance) and adjust the "Desired profile" as depicted/illustrated in Figure 2. UI1 must also provide access to the the Swagger documentation (as a second tab for developers).

   UI2 providing access to the system administrator to monitor the proper functioning of the Decision Support System. Among other functionalities this UI2 must allow the adminsitrator to visualize (via plots) the evolution of the sensor data over time (temperature, Co2, sound, humidity...), periods when sensors where disconnected, etc. ; room facilities of a given room ; calendar events over a specific period of time, etc. *Tip :* Graphana-like dashboarding solution can be used.

Each student group must write a report explaining / justifying the different architectural design choices made to address the above expectations / points (including screenshots showing the well functioning for each expectation / point is a "plus"). Each group needs to explain / detail how it organized itself (internally), meaning who was in charge of what in the project, the milestones, gantt chart (real vs. initial plan), etc.

1. see e.g. https://blog.logrocket.com/secure-rest-api-jwt-authentication/, https://jwt.io, or other blogs/ tutorials on the Web to help you in this stage.