

## Tecnologie per IoT

### Lab Software Part 2

---

- Exercise 1. Develop a RESTful style Catalog of a distributed platform for general purpose services. Identify the most suitable HTTP methods (among GET, POST, PUT and DELETE) and develop the web services to:
1. Retrieve information about IP address and port of the message broker in the platform
  2. Add a new device with the following information
    - unique deviceID
    - end-points (i.e. Rest Web Services and/or MQTT topics)
    - available resources (e.g. Temperature, Humidity and Motion sensor)
    - “insert-timestamp” when this device was added  
(**SUGGESTION:** to avoid synchronization issues, this attribute is managed and updated only by the Catalog according to its system clock)
  3. Retrieve all the registered devices
  4. Retrieve a specific device with a deviceID
  5. Register a new user with the following information
    - unique userID
    - name
    - surname
    - email address(es)
  6. Retrieve all the registered users
  7. Retrieve a specific user with a certain userID
  8. Add a new service with the following information
    - unique serviceID
    - description of the service
    - end-points (i.e. Rest Web Services and/or MQTT topics)
    - “insert-timestamp” when this service was added  
(**SUGGESTION:** to avoid synchronization issues, this attribute is managed and updated only by the Catalog according to its system clock)
  9. Retrieve all the registered services
  10. Retrieve a specific service with a certain serviceID

**This information is stored in a JSON file and all the information among the actors in the platform must be exchanged in JSON**

Implement an additional feature of the Catalog to remove all the devices and services with “insert-timestamp” higher than two minutes. The Catalog has to take this action periodically (for example every 1 minute).

**Handle possible errors in invoking the web services (e.g. wrong command or wrong number of parameters).**

**Optional:** You can replace the JSON file with a database to save information. Anyway, information among the actors in the platform must be exchanged in JSON via REST.

- Exercise 2. Develop a **client** python application for invoking the RESTful Catalog developed in *Exercise\_1*. This application has to retrieve information about
- the message broker
  - all the registered devices
  - device with a specific deviceID given as input
  - all the registered users
  - device with a specific userID given as input
- Exercise 3. Develop a **client** python application, that emulates an IoT device, to invoke the RESTful Catalog developed in *Exercise\_1*. This application has to periodically (for example every 1 minute) either register a new device or refresh the old registration by updating its “insert-timestamp”. During the refresh of an old device registration, the Catalog has to update also the “insert-timestamp”.
- Exercise 4. Extend the functionalities of the last version of the temperature Web Service (refer to Exercise 3.2 in Lab Hardware – part 3) to invoke the RESTful Catalog developed in *Exercise\_1*. This new feature has to periodically (for example every 1 minute) either register its information or refresh the old registration by updating the corresponding “insert-timestamp”.
- Exercise 5. Extend the functionalities of Catalog developed in *Exercise\_1* to work as MQTT subscriber either to register a new device or to refresh the old registration by updating its “insert-timestamp”.  
The Catalog must subscribe to a specific topic used for this purpose only.
- Exercise 6. Develop a python MQTT publisher, that emulates an IoT device, to periodically (for example every 1 minute) either register a new device or refresh the old registration in the Catalog developed in *Exercise\_2*. During the refresh of an old device registration, the Catalog has to update also the “insert-timestamp”.