ASInt – MEEC
2016/2017
Project

In this project students should develop a web application used to manage/control room occupancy.
In an organization such as IST, some rooms are available for public use by the students. In order to understand the occupancy of such rooms it is necessary to register when a student is studding on such rooms.
The system will be operated by two different class of users: admins and students. Admins manage rooms by making them available for use and observe current occupancy. Students check in/out in rooms and search for friends location.

1 Web application

The room occupancy system will be implemented as a web application, accessible using a browser. A suitable REST API should be implemented to guarantee that the system could be accessed from other clients (mobile applications or desktop applications).

1.1 Administrator

The administrator can add rooms to the system. Before a room is added to the system (made available) student can not used it. The administrator can also list all rooms available, in this listing presents all available rooms name at its current occupancy (number of registerd users).

The FENIX system provides a REST API for search and browse of available rooms in the various *campi* (Campus do Tagus, Campus da Alameda, e Campus Tecnológico e Nuclear). This system should use this FENIX API to provide the administrator with a list of possible rooms. The administrator can browse the various IST buildings and campus to select the appropriate rooms to become available to students

1.2 Students

The students should be able to browse available rooms and "check in" to one of those rooms when entering and starting to use it. Students should be able to checkout, when leaving the room.

Students can only checkout from a room he last checked in. The student can only be in one room at a time: checking in without first checking out will implicitly perform a checkout.

Students can see who is currently checked in at a room.

1.3 User management

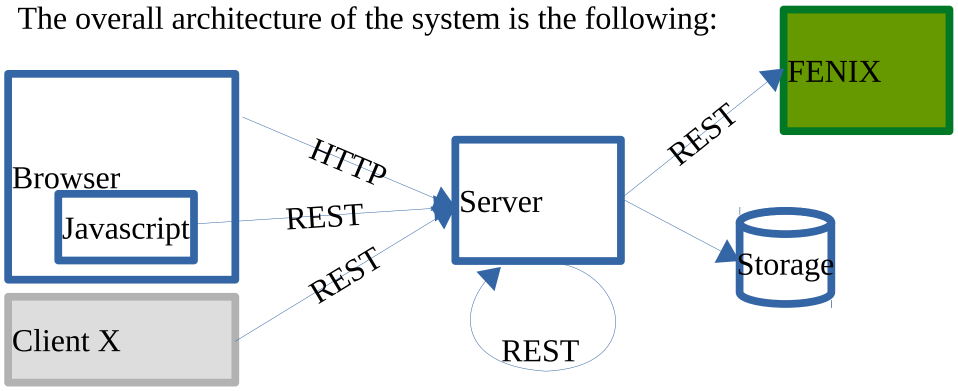
Before using the system, student should register, providing in this step a user-name user selected unique username, in this step the system should return a numerical identifier. The numerical identifier should be used in all interactions with the system.

The administrator username is **admin** and it identifier is **0.**

It is not necessary to implement authentication mechanism, but all interactions with the system should be accompanied by a user identifier.

2 Architecture

The overall architecture of the system is the following:



Client X should not be implemented, but the REST endpoints implemented in the system (Server) should be defined as if used by this future client.
The browser should implement part of the user interface using JavaScript (and interact with the server using REST).
When answering to HTTP requests, the server can also call REST points reimplemented by it. This will guarantee modularity of the system.

3 Implementation

Students can use any technology and programming language: python, php, node.js. The selection the language will not affect the grade.

The server should be deployed in the Cloud (for instance in the Google App Engine infrastructure)

The storage should be in the Cloud.

4 Data persistence

In order to simplify implementation, define one class for each of the data type necessary (Users, available rooms, Check-ins/outs) and store them in different arrays.

This will help the deployment in Google app engine.

5 Evaluation

Student should implement the maximum described functionality and architecture requirements. Alternative implementations will influence the final grade:

- Use of JavaScript on the browser.
- Implementation of a suitable rest API on the server.
- Interaction with the FENIX system
- Deployment on the cloud

5.1 Report

The students should write a report describing:

- The system architecture
- Implemented REST API
- Used technologies/libraries
- User interface of the system

5.2 Submission date

The final date for the submission of the work should be defined and agreed with the teacher. This date can be scheduled to after the Christmas holidays.

6 Further information

Jquery

- https://jquery.com/

Google app engine

- https://cloud.google.com/appengine/ (browse the page for free limits)
- Python
  - https://cloud.google.com/appengine/docs/python/
  - https://cloud.google.com/appengine/docs/python/download
- Cloud datastore
  - https://cloud.google.com/appengine/docs/python/datastore/
- Node.js
  - https://cloud.google.com/nodejs/
- php
  - https://cloud.google.com/appengine/docs/php/
-