In this project students should develop a web application used to manage dissemination of messages to users working in particular physical locations.

In an organization such as IST some buildings are accessed by multiple users. In case of emergency some notifications should be forwarded through digital means to users in specific locations/buildings. Users nearby can also exchange messages between them.

The system will be operated by three different class of users: admins, regular users, and bots:

- Admins can access the logs of activities, and manage static information.
- Regulars users send their location to the server, send messages to nearby users, check who is nearby and receive messages (from other users or bots)
- bots are programms that are assigned to a specific building and periodically send messages that are in that builing.

1 Web application

The system will be implemented as a web application, accessible using a browser. A suitable API should be implemented to guarantee that the system could be accessed from remote clients (web, mobile, the administrator desktop application, and the bots).

1.1 Administrator

The administrator will access the system trough a desktop application that interacts with the server using a defined API.

The administrator has the following functionalities:

- define builds and their locations (latitude, longitude)

- list all users that are logged-in into the system

- list all users that are inside a certain buiding

- list the history of all the user movements and exchanged messages this list can be configured with a simple query to select the user or building.

1.2 Regular users

The regulars users access the system trough a browser.

The downloaded application/page will retrieve the location of the user/browser and send it periodically to the server.

The user application will have the following functionalities:

- login using the FENIX Authentication (istID and password).

- send messages to users nearby (in a defined distance range)

- define the distance range that will include the nearby users

- see who is neerby: on the range of the messages and on the same buiding.
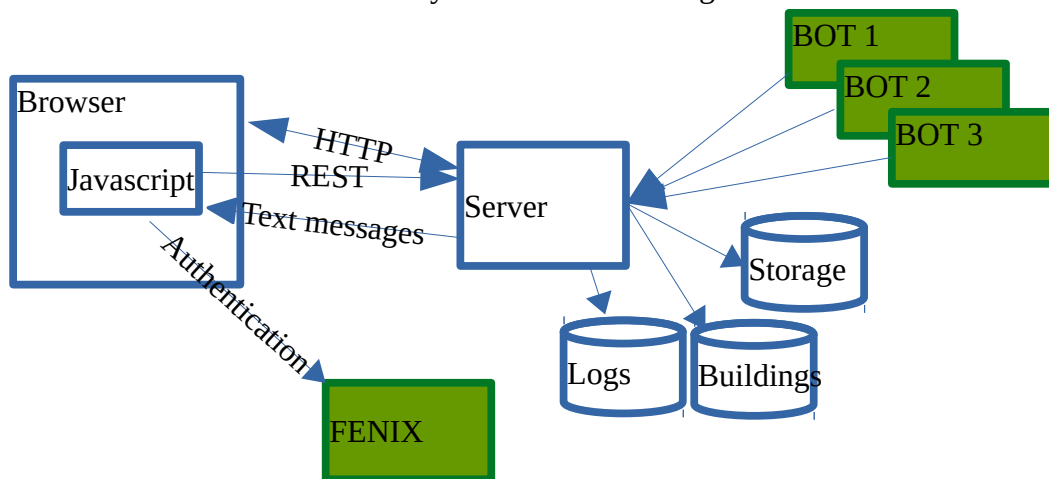
- Receive messages sent by other users and bots

1.3 Bots

Bots are programs that periodically send messages to users in a specific room.

At registration, bots are assigned a specific building. All messages sent by a specific bot will only be delivered to the users in that building.

2 Architecture

The overall architecture of the system is the following:

2.1 Buildings

The administrator should provide a list of buildings with their coordinates (latitude, longitude) in order to configure the system. This setup should be done through a specialized menu, with upload of a file with the following information:

- building ID

- Buiding Name

- latitude

- Longitude

The FENIX API provides a mechanism to list and browse all the buildings in the various *campi* (Campus do Tagus, Campus da Alameda, e Campus Tecnológico e Nuclear):

- https://fenix.tecnico.ulisboa.pt/api/fenix/v1/spaces/2448131360898

- https://fenix.tecnico.ulisboa.pt/api/fenix/v1/spaces/2448131360897

- https://fenix.tecnico.ulisboa.pt/api/fenix/v1/spaces/2448131392438

User should prepare a file with some of the buildings and retrieve the coordinates of those buildings

using google maps. In order to ease the implementation just use the center of the buildings and decide if a user is inside it if it is closer than a certain distance.

2.2 Logs

All user movements and with exchanged messages should be recorded on a Log.

This log can be queries by the administrator.

2.3 Bots

Bots are assigned to a certain building, and can send messages to users that are inside of it. Multiple bots can concurrently interact with the systems.

Students should find a way to register the various bots and assign them a specific building

2.4 Authenticatios

The login of the regular users should be done using the FENIX API:

- http://fenixedu.org/dev/tutorials/use-fenixedu-api-in-your-application

Each regular user should be identified with its ISTID.

The administrator should do a local login to the system (username: **admin** / password: **123**).

3 Implementation

Students can use any technology and programming language (python, php, java, node.js) to develop the system. The selection the programming language will not affect the final 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 and future deployment in the cloud, students should define one class for each of the data type necessary (Users, rooms, check-ins/check-outs) and store them in different dictionaries.

In the final version data should be stored in a Cloud database.

5 Multisite

In order to add scalability the system can be composed of several servers.

Im every instance every user is only interacting with a single server, but all the messages (generated by users or bots) are relayed to the various servers.