ADInt – MEEC

2022/2023 - Project

IST Big Brother – Intermediate project

In this project students should develop a system (**IST-BB**) that will allow the control of the effort every student will the during the day and semester. The system will also allow students to evaluate the various IST services (cantina, cafeteria, administrative services, that he interacts with.

The final system will have two types of human users: students and administrators.

Students will access the system using in order to register the various activities they perform during the day. After interacting with a service, students will also evaluate them. Administrators will see the statistics about every user, but also the evaluations of registered services.

All users will authenticate using the FENIX username and password.

In this work (1st and 2nd part of the project) students should

- Define the architecture of the systems (web services, web application, interaction with external services)
- Define the set of resources to be made available by the various components
- Define the relevant information (attributes) of such resources to be stored in a Database
- Define the interfaces (WEB and REST) to access such resources
- Implement simple prototypes (in python and JavaScript) the replicate the behavior of students mobile application
- Implement a simple web server for administrators access.

This document will present the overall architecture of the system, and will describe the functionalities to implement in the intermediate project. All decisions here made can be reused in the final project or changed if issues are found.

1 Final system

1.1 Users

The final system will be operated by two classes of users:

- Students that use a mobile application (simple web application) to register all activities done during the day. Student will also be able to evaluate an administrative services they interact with.
- Administrator will access students activities statistics, and services evaluations. Administrators will also be responsible to register in the system the possible **presential services**.

1.2 Student activities

During the day students can perform the following activities in three major areas:

- Personal
    - Sleep
    - Eat
    - Leisure
    - Sports
    - Other
- Academic
    - Attend classes
    - Study
- Administrative
    - **Presential services** services at IST

1.3 Data sources

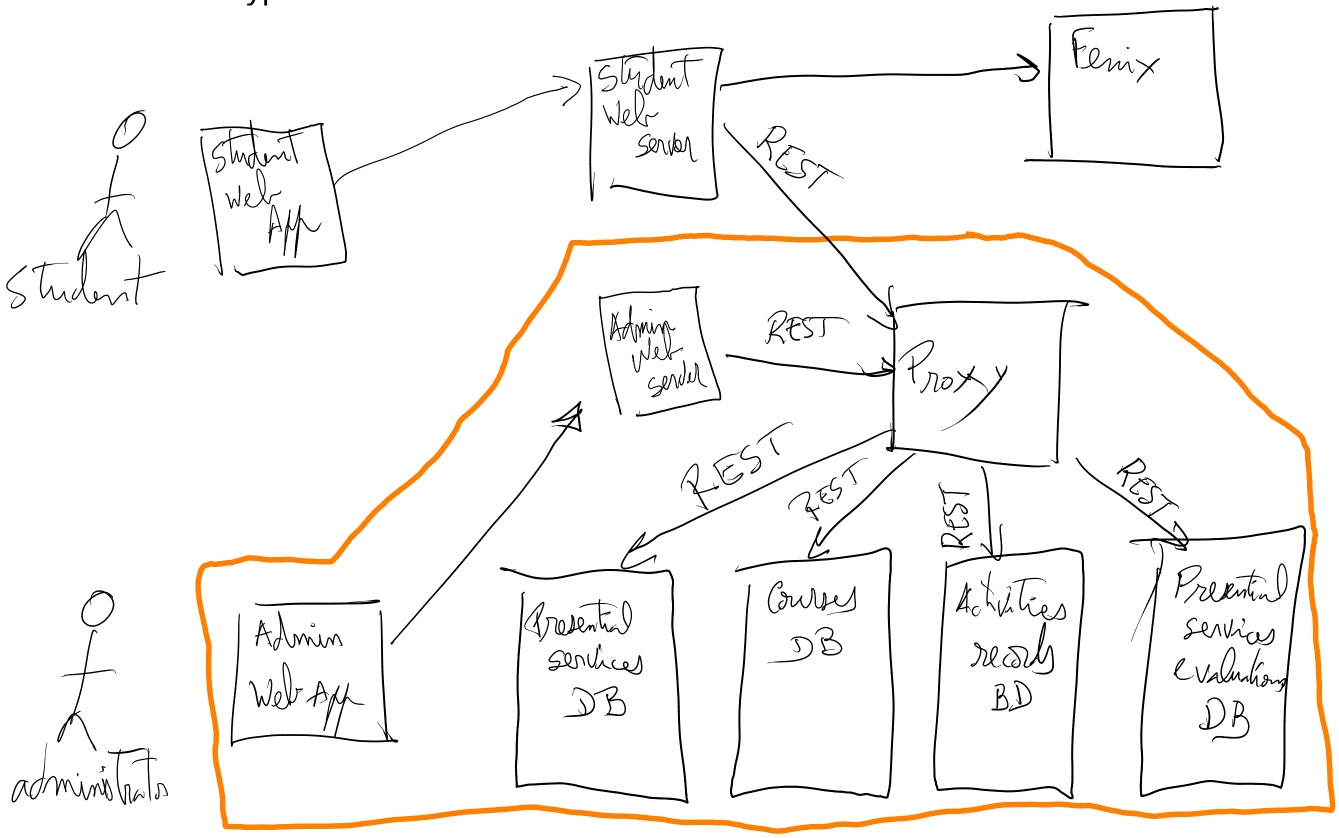
The previous categories are fixed and hard coded, but some more particular information (in the Academic and Administrative areas) is dynamic.

The classes and study information should be related to the Courses the student is attending, The course each student is attending should be retrieved from FENIX in the final version. In the intermediate version, any course can be registered.

The list of **presential services** should be updated by the administrator.

2 Architecture

The following picture shows the various components that make the **IST-BB** system, along with the various types of interactions.



The users (Administrators and Students) interact with the system using web browsers to access the **IST-BB** system. The **IST-BB** web server does not access directly any database, but only interacts using REST with the proxy. This proxy redirects all requests to the correct web-service:

- *Presential Services DB*
- *Courses DB*
- *Activities Records DB*
- *Presential Services Evaluation DB*

Depending on the operation performed by the user on the Web applications, different services should be contacted in sequence.

In the Final version, students will be authenticated using the FENIX OAUth service, and the courses a student is enrolled to will be retrieved also from FENIX.

3 Intermediate project

In the intermediate project, students should only implement the components inside the orange area: define the resources each DB component will manage, implement those components as REST web-services, implement the Proxy and a simple Admin Web Application.

3.1 Resources

The resources that should be defined are:

- Presential services
- Attended Courses
- Activities records
- Presential services evaluations

To each resource, students should define the corresponding SQLAlchemy model that will be used to persistently store such information. Each DB component/service will only manage a single resources.

3.2 Services

Each of the DB components should be implemented as a independent web-service (FLASK application) with a specific set of endpoints. These web-services should only be contacted by the proxy service.

The proxy service is another FLASK application that receives all requests from the Web applications and redirect them to the suitable DB component/service.

The proxy service should allow the following operations:

- Create a presential services
- List presential services
- Create a course
- List all courses
- Create an activity (of the types listed in Section 1.2)
- List all activities
- Create one presential service evaluations
- List all evaluations of a presential service

Each functionality should have one endpoint defined on the proxy and corresponding endpoints in the DB components.

Other functionalities and endpoints may be necessary.

The proxy and Dbs should only export REST services and endpoints.

3.3 Administrator web applications

The system should provide to the administrator the following functionalities:

- Create a **presential service**,
- List all evaluations of a service

- List all attendances for a course
- List all activities by a students

These functionalities should be implemented in a FLASK application (Admin web server) that presents a menu and 4 pages for the functionalities. The Admin Web server should contact the proxy to execute the corresponding functionalities.

3.4 User authentication

In the intermediate version of **IST-BB** there is no need to perform user authentication.

All accesses to the Admin web application, are considered done by an authenticated administrator, and all requests performed by students are done by a single one.

4 Intermediate project development

Students should follow the proposed steps in order to implement the intermediate project:

1. Define and implement the data-model of the database used by the each of the various DB components
2. Define the endpoints (URL and transferred data) for the various DB services
3. Implement the DB components (SQLAlchemy, endpoints and python code) and test each component independent
4. Implement the Proxy and verify the forwarding of requests to the correct DB component
5. Implement the Admin Web application

Students should make decisions about some things not covered in this assignment.

5 Error validation

All endpoints should guarantee that incorrect call will deliver and error and not do incorrect behavior.

In case of invalid input the endpoints should return a suitable error code

6 Delivery

Students should deliver the code of all the components along with a simple **pdf** document that presents the data models and implemented REST API, and any decision made during the development of the intermediate project.

7 Evaluation

The intermediate project will be evaluated taking into consideration the number of functionalities implemented, the correction of the API, error validations and the code structure.