ADInt – MEEC

2022/2023 - Project

IST Big Brother

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 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

When registering an activity students should also provide the time period corresponding (starting and ending date/time).

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.

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

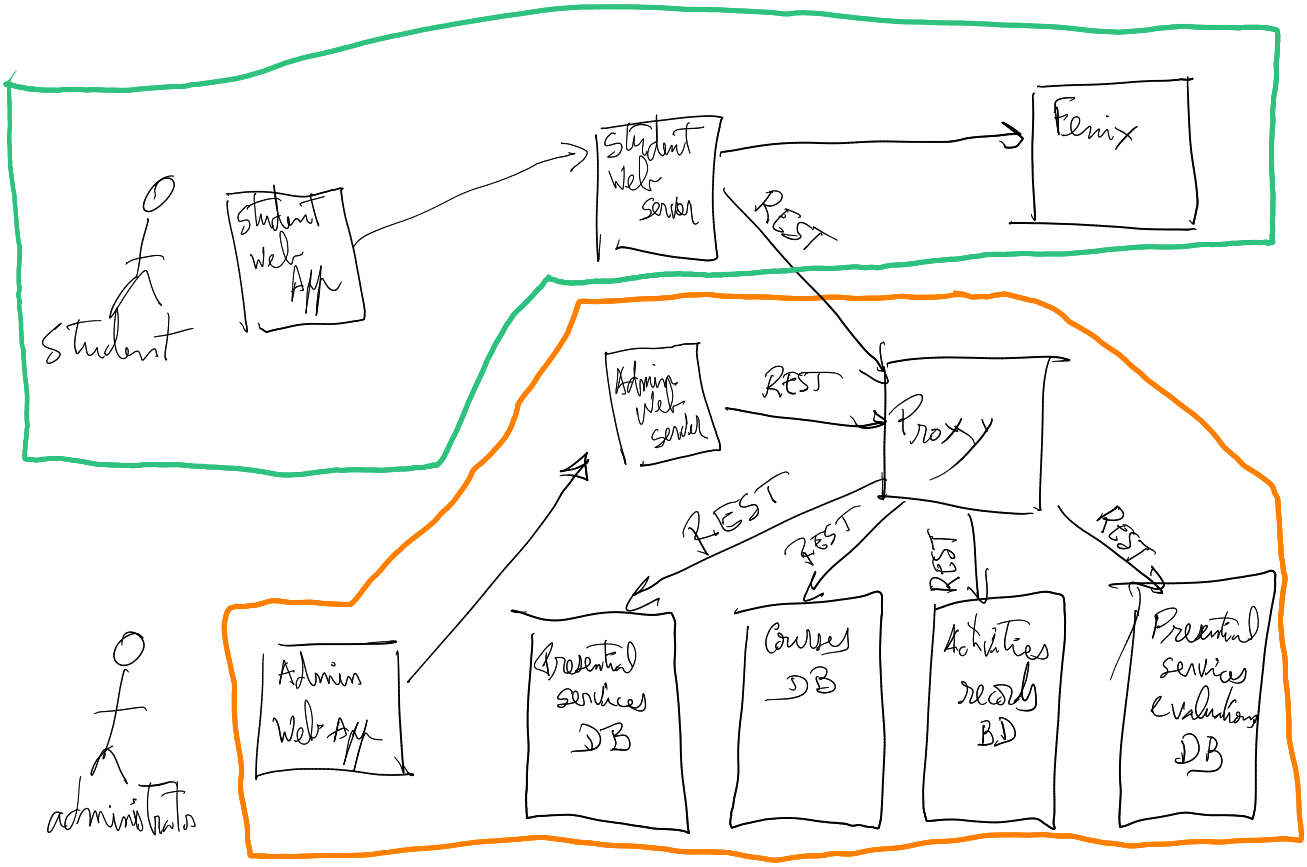
2 Architecture

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 implemented 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.

All of the endpoints developed in the intermediary project will be used in the completion of the project, but some additional endpoints may be necessary.

The Data models implemented in the intermediate project will also be reused, probably with no, or minimal, changes.

If student find out that the developed endpoints should be changed there is no problem with that.

4 Final Project

In the final project students will complement the intermediate project so that the general

functionalities are correctly implemented, marked in green in the first image.

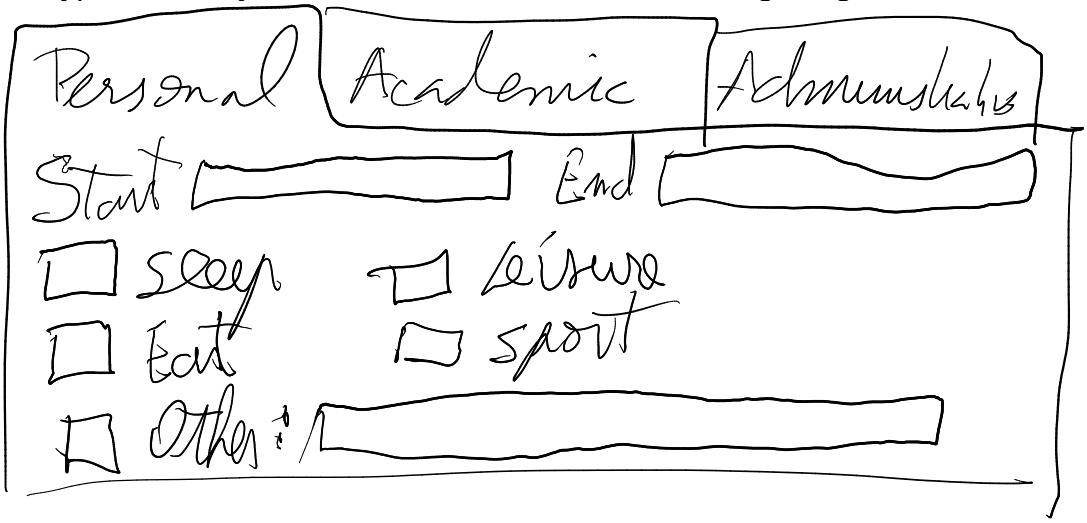
4.1 Students web applications

The **student web application** will simulate a mobile application and should only be executed in a smartphone. This application will have a single page, where all the operations will be performed. Any changes on the UI should be done using Javascrip, and (besides the download of HTML, CSS and Javascrip) all communication with the server should be done in REST/AJAX. So student should also define a set of REST endpoints that the Student Web server exports to the Application.

The **Student web application** will be divided in 2 areas:

- input area
- output area.

The input area will allow the student to register all the activities. This area should have 3 tabs where each type of activity is created, as shown in the following image:

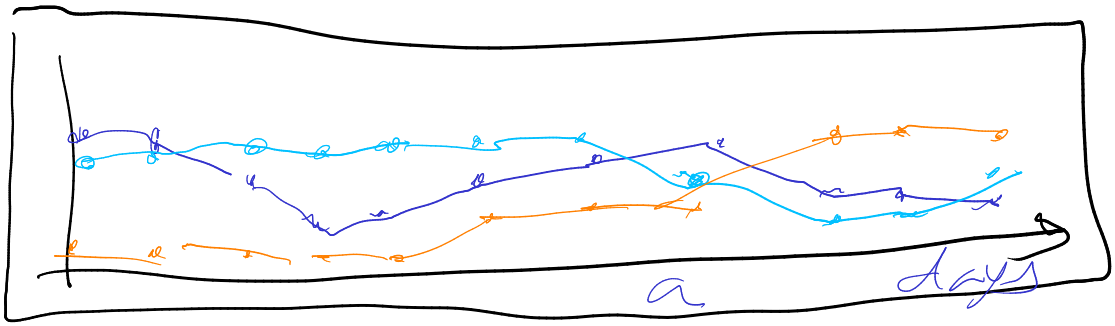


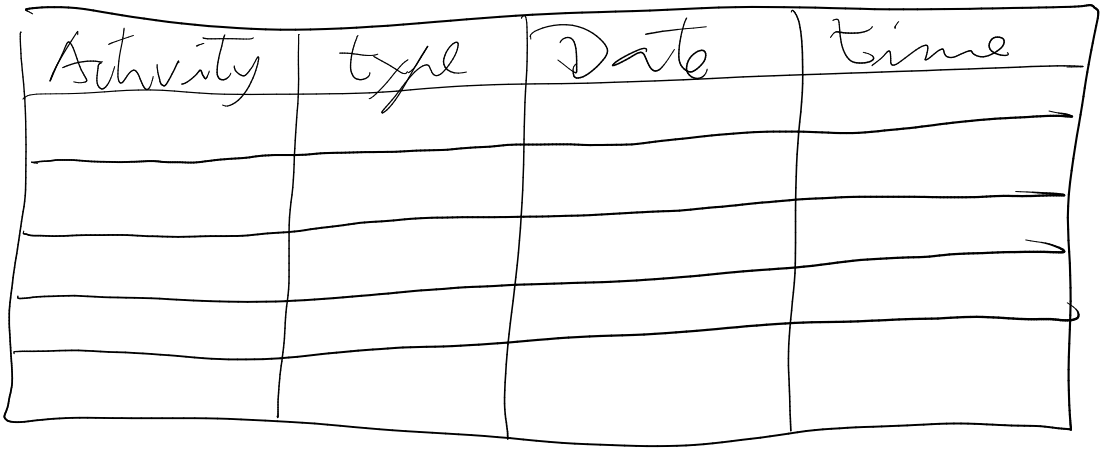
Student should define a set of HTML input fields that will allow users to create all the activities. In the case of Personal activities the data is static, but on the other two types of activities it depends:

- The list of courses that is presented to the student should only contain the coursed the user is enrolled at in FENIX.
- The list of presential services is retrieved from the Presential services DB.

After (or when) the user registers one administrative activity, the user should be able to evaluate such service.

The output area will present a list of all the activities and a plot. The plot will show for the past day, how many hours the user spent in each type of activity: academic, personal or administrative:





The table and plo should be updated every time a new activity is created.

4.2 Administrator web applications

The system should provide to the administrator the following functionalities:

- Create a **presential service**,
- List all presential services
- Delete a service, after which no user can register administrative activity there.
- 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.

4.3 User authentication

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

In the final version, all accesses to the Admin web application, are still considered done by an authenticated administrator

IN the final version, students should integrate OAuth authentication into the Student web app/server to allow users type their FENIX username/password. After this authentication

Student web server will also be abble to retrieve the courses the user is enrolled at

5 Final project development

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

1. Implement the student authentication.
2. Implement the creation of Personal activities (HTML, Javascrip, REST endpoint and pyhton code)
3. Implement the creation of a Academic activity
4. Implement the creation of a Administrative activity (also with the evaluation of the service)
5. Implement the list of activities in the student web application (populate in the start of the application and update whenever a activity is created
6. Implement the plot.
7. Modify the Admin Web app.

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

6 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

7 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.

8 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.