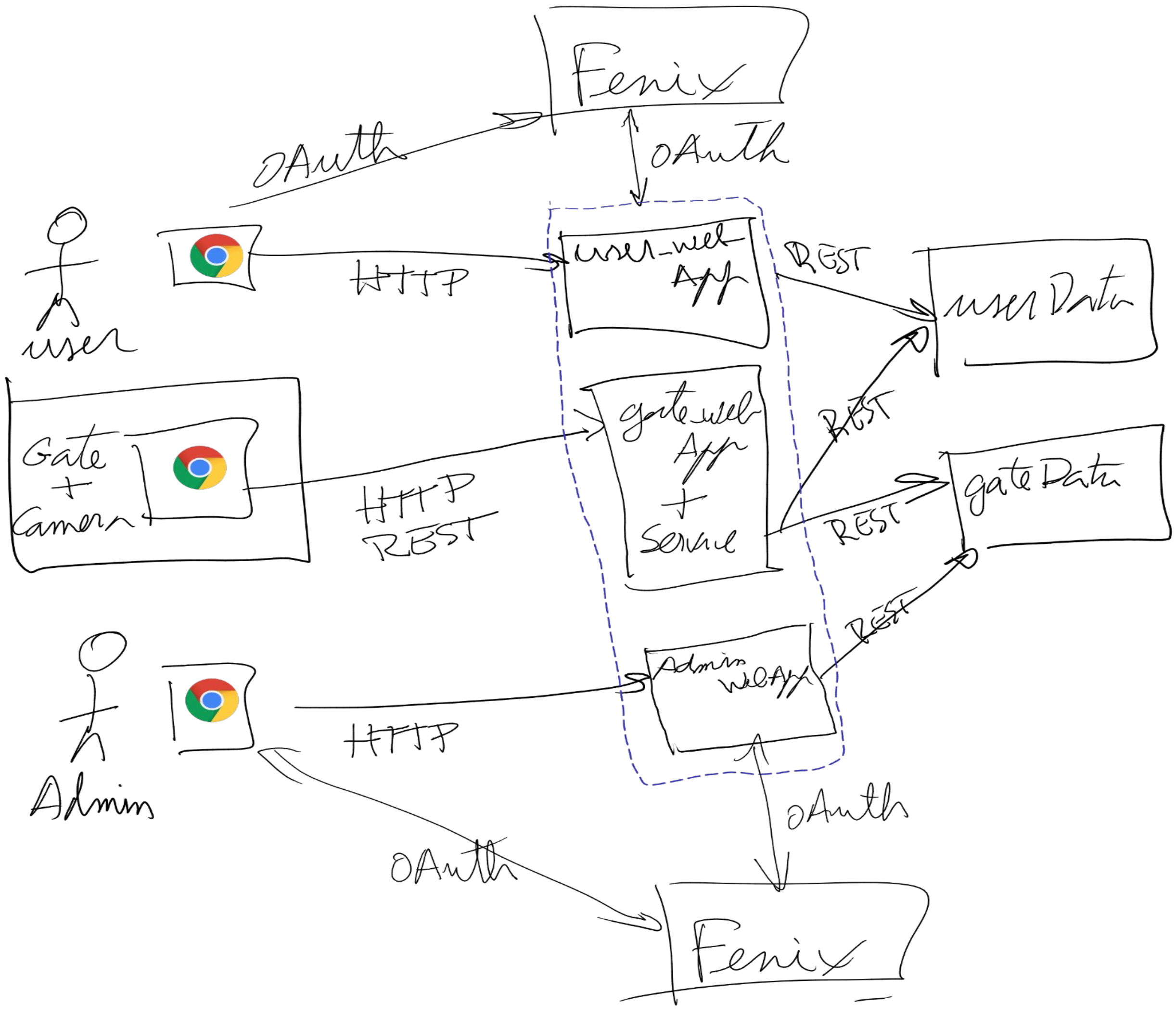
ADInt – MEEC
2021/2022 - Project
Access control system – Final project

The final version of the IST-GATE system will be an extended version of the preliminary project.

The main ideia for the project will be the same, but the gate-app and user_app will be implemented in different technologies and the system will store additional information.

1 Architecture

The new architecture is a bit differente from the one of the intermediate project, but some components remain similar:



1.1 User_web_app

The new version of the **user_app** will run on the smartphone inside a browser and is called **user_web_app**.

On the smartphone the user should access the **user_web_app**, authenticate and select one of the two available pages:

- one page will present a QR-code with the secret. Every time the user accesses such page a new code is generated and presented as a QRCode.
- The second page will show the whole history of accessed gates by the user.

1.2 Gate_web_app

The new version of the gates will have a camera attached that will read QR-codes presented by the users.

The **gate_web_app** will replace the old **gate_app** and is composed of a simple web page with a JavaScript component to read and decode QRCodes.

The JavaScript after decoding a QRCde will invoke a suitable web-service that will verify it.

If the code is valid, this page should show a green light that turns off when the gate closes.

1.3 Admin_web_app

This application allows administrators to register new gates (as in the intermediate project).

A new functionality is to see the statistics of each gate:

- Anonimized records of every attempt (successful or failed) to open the gate

2 Data storage

Students should create an additional service to store the relevant user data.

Students should pay attention to the new information that the system will store related to users and gates.

3 Authentication

The two applications that allow user to interact with the system (user_web_app and Admin_web_app) should perform authentication with the FENIX system using Oauth.

The gates should continue to be validated against gateID and corresponding secret.

4 Code organization

The various web apps can be agregated into the same flask applicatio for simplicity of development, but they should be distinguished by the endpoints

5 Implementation

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 **UserData** Service
2. Implement and test the REST API of the **UserData** so that new gates can be registers and listed.
3. Implement the **user_web_app** to allow user to get a QRCode. This application should perform authentication on FENIX.
4. Implement the gate_web_app
5. Change the **admin_web_app** to allow FENIX authentication of the user. This application can use a configuration file with the istID of the administrators
6. Implement the two pages that show the listings of gate accesses
   - **administrator_app** - list of all attempts to open the gate without user information
   - **user_web_app** - list of all the gate accesses by the current user

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

6 Fault Tolerance

Students should implement a simple fault tolerance mechanism for the **Gate_Data** service, with two replicas. Both replicas should be synchronized and as long as one is working other services should be able to access data.

7 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