Test Provider Manual REST

It is advised to read the eHealthBox, consent, therapeutic link or therapeutic exclusion user manual first to gain a better understanding of the framework.

Note

As already discussed, the architecture and execution work of the REST test automation framework is heavy. The main reason is related to the security of IAM Connect (which cannot be bypassed). Even with that security, we could imagine a solution to request to the software vendors to have a specific client configuration (in acceptance) in order to facilitate the whole process. The requisites would be to have 2 different redirect URIs in the client configuration, with at least one that is **localhost**. It could be 2 times **localhost**, but with **2 differents ports**, or 1 URI referencing a **hosted back-end** and 1 **localhost**. Having that configuration, the login still have to be performed 2 times: 1 for authenticating in the app, and one for the Token interceptor, to be able to perform service calls. But other than that, everything can run in one part instead of two. We could even get rid of the **ResponseRepository** and temporarily save the needed data/object in java objects.

Introduction

This manual is meant for everyone that works on the development of the REST testing framework.

Target audience

- Developers of the eHealth REST services

Main project structure (submodules)

```
Template
citrus-gui
citrus-main
    AppiumRequest
    Assertion
    ResponseRepository
    ResponseRepositoryModel
    SeleniumRequest
    tokengenerator
```

Add a new service

Resources

Create the necessary resources in **ResponseRepositoryModel.** The classes created there have the sole purpose to save the data under test in the **ResponseRepository**. The classes are simple POJOs to access the data. The classes **MUST** extend **BaseResources**.

Data Access Objects

Create **DAO** classes in the response repository to be able to access the previously created **Models**.

Assertions

Create the test scripts in **Assertion** that will verify the fetched data from the application against an actual service call. The assertions are made with Citrus Framework (Detailed documentation: https://citrusframework.org/citrus/reference/2.8.0/html/index.html).

The framework allows the usage of JPath, which is helping to fetch the needed data from the REST service response.

SeleniumRequest / AppiumRequest

In that part of the project, it's the Software vendor that has to fill in the support code to fetch the data from their SUT.