1. Création des utilisateurs + structure

créer X utilisateurs, un token et la structure de stream par utilisateur et voir l'impact sur core/register

Resources à monitorer: CPU, HDD, RAM core & register

1.1 Utilisateurs

HTTP POST https://reg.pryv-n4a.ch/user

headers:

- 'Content-type: application/json'

Body: données dans **pryv-n4a-users.json**

1.2 Tokens

HTTP POST https://{USERNAME}.pryv-n4a.ch/auth/login

headers:

- 'Origin: https://sw.pryv-n4a.ch'
- 'Content-type: application/json'

Body: données dans **pryv-n4a-token-creation.json**

Il faudra stocker le token généré par utilisateur pour le call de création de streams.

1.3 Structure de streams

HTTP POST https://{USERNAME}.pryv-n4a.ch/

headers:

- 'Content-type: application/json'
- 'Authorization: {TOKEN_FROM_1_2}'

Body: Le contenu dans **streamStructure.json**

2. Processing des mesures

Envoyer des mesures similaires à l'app oBPM riva sur 1 ou X utilisateurs.

Resources à monitorer: CPU, HDD, RAM core & hook, logs sur container pryv_queue_1 de la machine hook qui publie les temps de processing.

HTTP POST https://{USERNAME}.pryv-n4a.ch/obpminput

headers:

- 'Authorization: {TOKEN_FROM_1_2}'

- 'Content-type: multipart/form-data'
- 'X-Requested-With: XMLHttpRequest'

Fields:

- event: Le contenu dans **measurement.json**

File:

- name: 'file'
- content: **measurement.data**
- options: {"type":"application/octet-stream","filename":"pp.pack"}

Vous trouverez dans le fichier **uploadMeasurement.sh**, un exemple pour effectuer ce call en cURL.