Unit tests documentation

Document’s description

Titre	Unit tests documentation
Objet	Unit tests documentation
Auteurs	Stanislas Hégron
E-mail	psynderFR@gmail.com
Mots-clés	Unit tests; Documentation
Promotion	2022
Date de mise à jour	19 June 2021
Version du modèle	1.3.0

Review table

Date	Version	Auteur(s)	Sections(s)	Commentaire
10/06/2020	1.0	Stanislas Hégron	All	Creation of document’s architecture
05/02/2021	1.1.1	Stanislas Hégron	API unit tests	Updating informations about using process
13/06/2021	1.2.0	Stanislas Hégron	Informations before start	Add information section at start of *API unit test*
19/06/2021	1.3.0	Stanislas Hégron	Environment configuration	Add environment configuration instructions

Table of content

API unit tests

Informations before start

Since the locking therapist's account has been implemented, unit tests can no longer be launched all at once. Because the therapist's accounts are now locked, many tests will fail because they cannot access the therapist's access token variable. To unlock them you'll have to ask an admin to do so.

Same thing for users that have to validate their account by mail confirmation.

To fix that, please follow the following instructions about environment configuration.

A - Environment configuration

As explained before, many tests will not work because therapists' accounts are locked at the creation. Only an admin can unlock them with the route `api_adress/admin/therapists/:therapistUid/unlock/`. But there is no route to unlock admin accounts.

To create one (**locally**) you have to use the following command on directory server/admin:

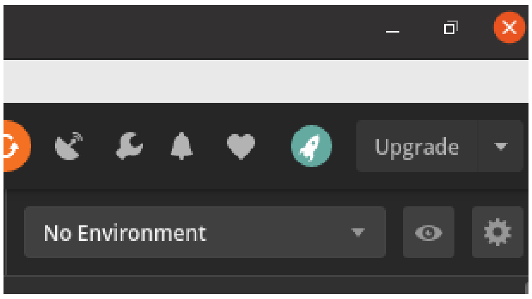
```
$ node adminAccount.js
```

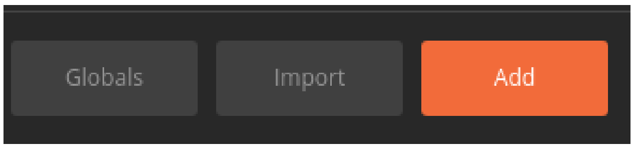
Then create an admin account and log it in. If you don't know how to use it, add *--help* at the end of the command.

This script allows you to create, log and delete a local admin account. If you want to launch remote tests, please contact someone with access to remote admin accounts to create you one or to give you access to one.

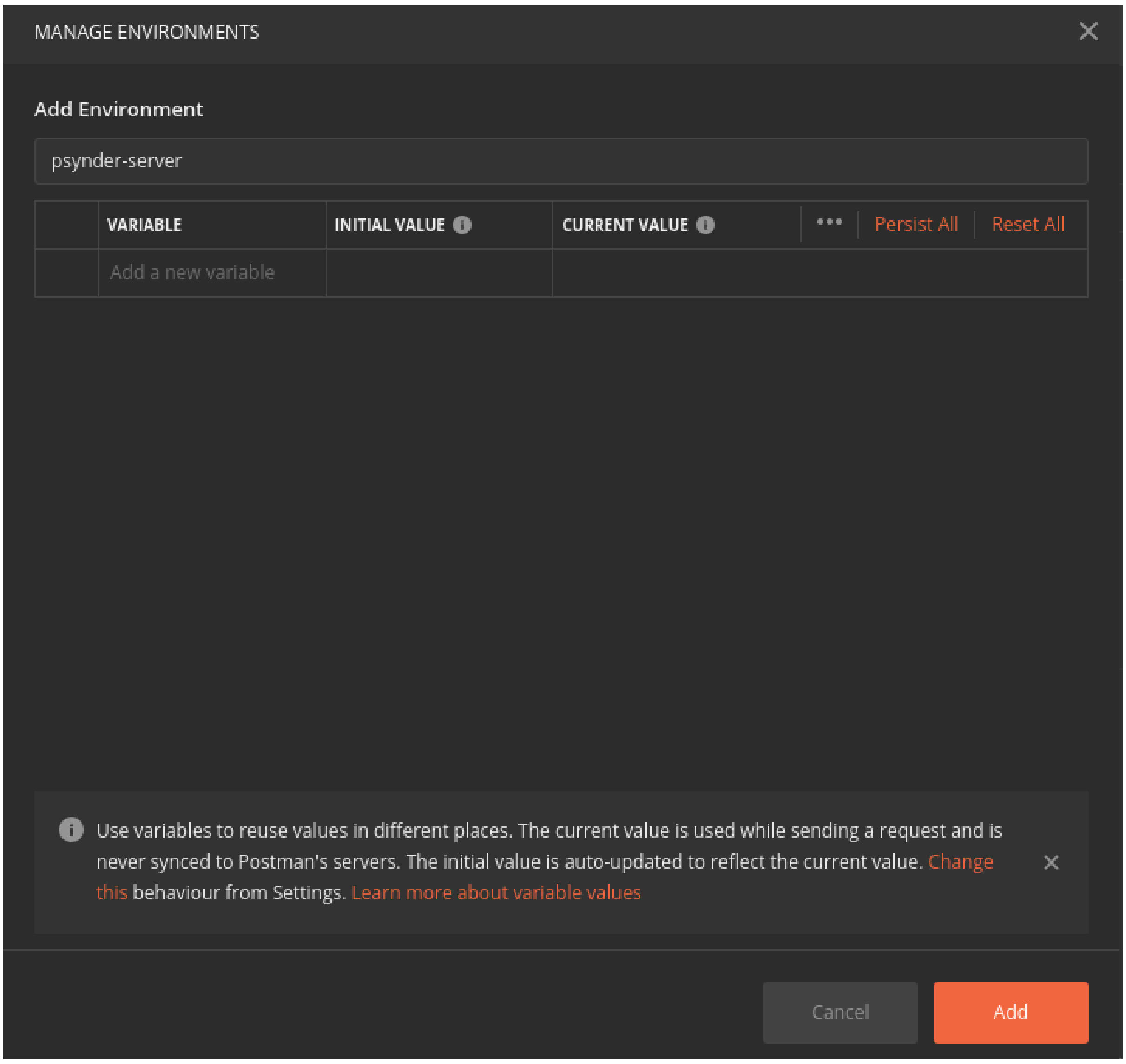
When the account is logged, it should give you an access token (don't copy simple quotes at beginning and end of the token) and a user id, keep them both !

For the next step, you'll have to install **Postman** and create an environment with the name you want. For that, go on your top-right corner and click on the little gear. Then click on the button *Add*.



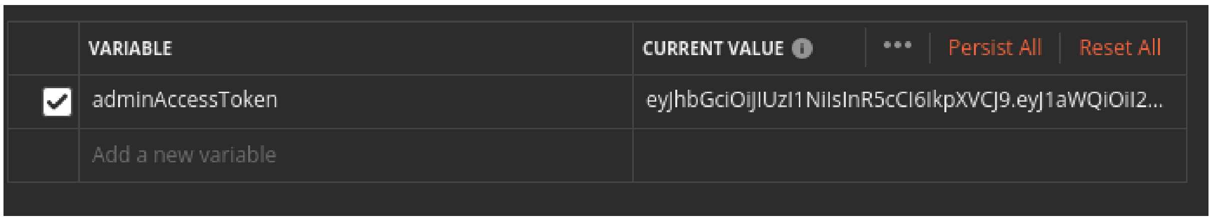


Now, on the new windows just opened, give a name to your environment but no need to add variables they will be created through tests. And finally, click on the *Add* button on the bottom.



Use variables to reuse values in different places. The current value is used while sending a request and is never synced to Postman's servers. The initial value is auto-updated to reflect the current value. Change this behaviour from Settings. Learn more about variable values

Then add a variable called *adminAccessToken* with the access token you get before.



	VARIABLE	CURRENT VALUE
✓	adminAccessToken	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1aWQiOiI2...
	Add a new variable	

B - Local server

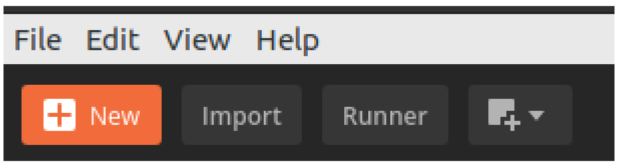
These unit tests are here to test if we can get, add, update and delete values in our database. First tests (these without user or therapist relations) were the firsts we did to test how routes work and if we succeeded in getting each type of call. They are also here to allow us to see if we could get parameters with bad values and return an error.

The first step to launch the API unit test is to build the docker container of the server. However, you need to install **Docker-compose** and **MongoDB** and run them to allow the server's docker container to build and up with following commands:

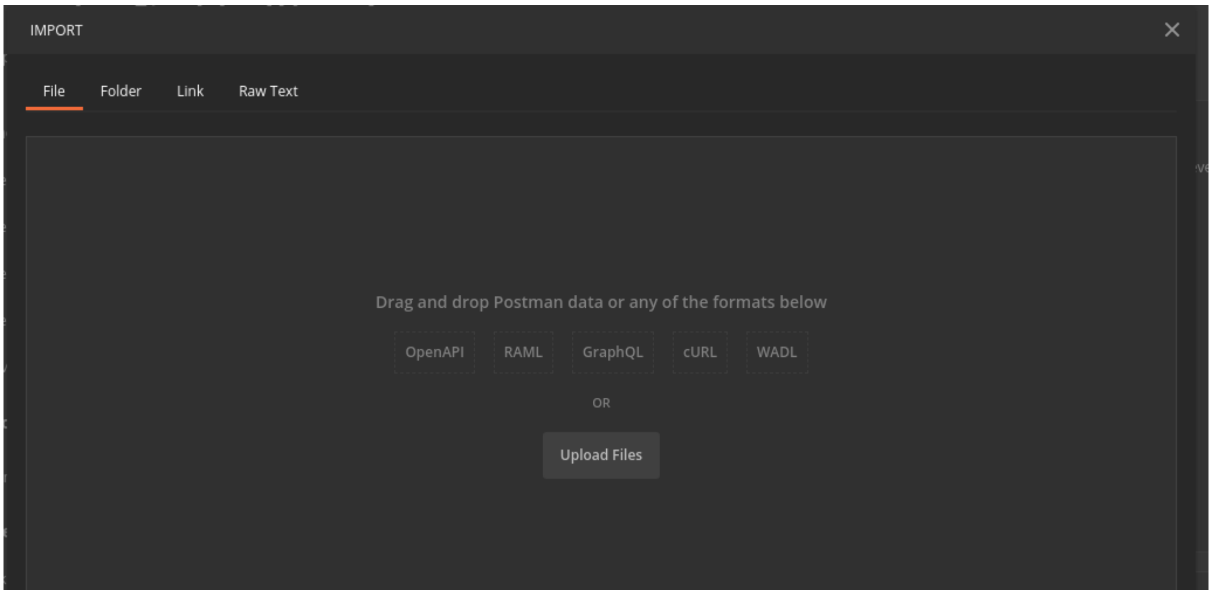
```
$ docker-compose build server
$ docker compose up server
```

Now you'll have to start Postman and import the unit tests collection named **EIP_tests_routes_local_server.postman_collection.json**. If you don't know how to import a collection, please follow the following steps.

To import a collection, you have to click on the button on the top-left corner with the name *Import*.



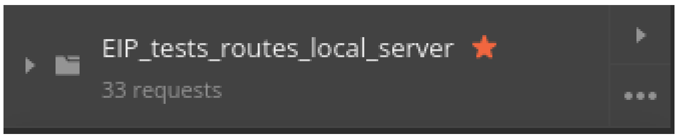
Then, click on the *Upload files* button.



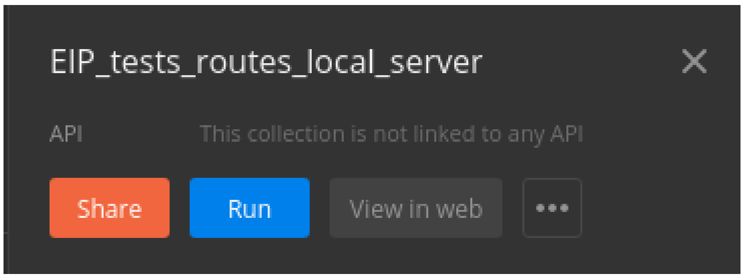
And now, you have to search the file on your computer. You can find it on the repository that you have cloned. Then go on the directory *tests* and there is the file *EIP_tests_routes_server.postman_collection.json*. Then validate and you should be able to now see the collection imported on your Postman application.

For the next step, you have to create an environment with the name you want. If you don't know how to, please follow the next steps on category A - Environment configuration.

Then to run the unit test, you have to click on the little arrow on the collection's name like that:



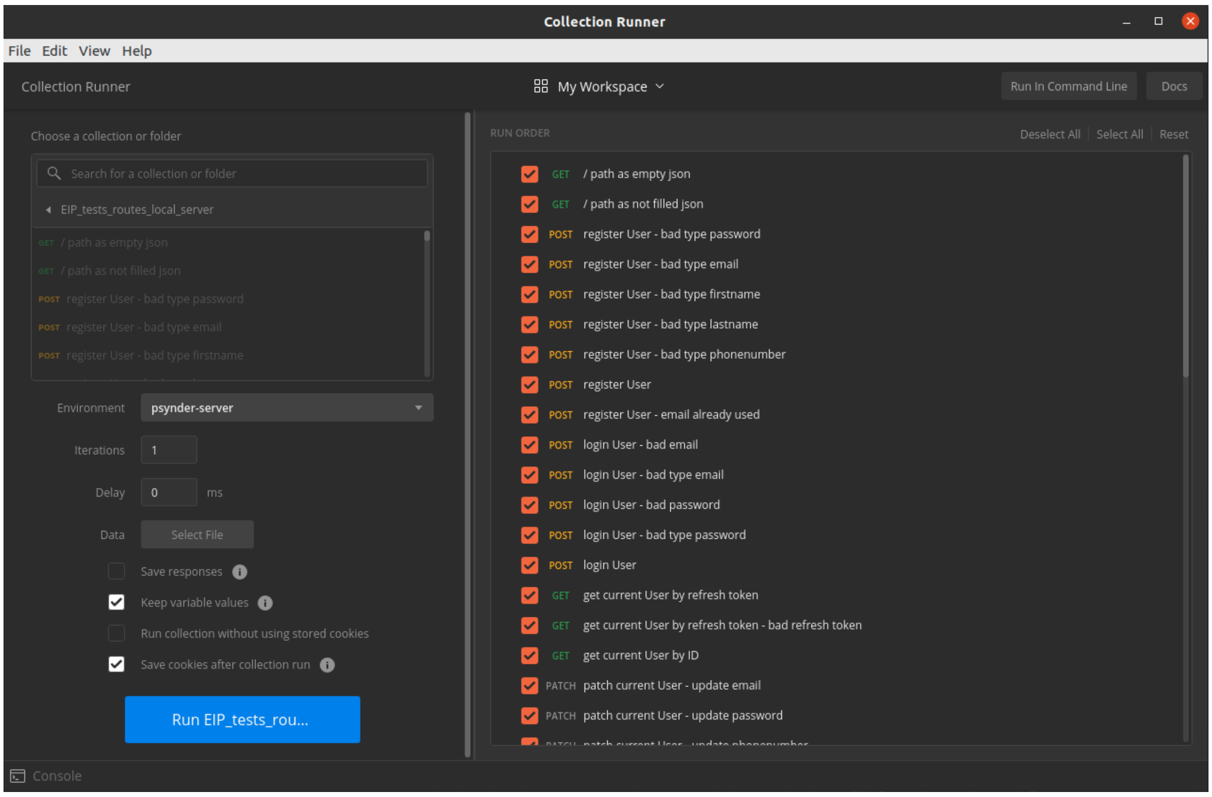
In the next step, you have to click on the button *Run*.



EIP_tests_routes_local_server

API This collection is not linked to any API

And finally, click on the button *Run EIP_tests_routes_local_server*.



EIP_tests_routes_local_server

Environment: psynder-server
Iterations: 1
Delay: 0 ms
Data: Select File

☐ Save responses
☑ Keep variable values
☐ Run collection without using stored cookies
☑ Save cookies after collection run

Run EIP_tests_rou...

RUN ORDER

```
☑ GET    / path as empty json
☑ GET    / path as not filled json
☑ POST   register User - bad type password
☑ POST   register User - bad type email
☑ POST   register User - bad type firstname
☑ POST   register User - bad type lastname
☑ POST   register User - bad type phonenumber
☑ POST   register User
☑ POST   register User - email already used
☑ POST   login User - bad email
☑ POST   login User - bad type email
☑ POST   login User - bad password
☑ POST   login User - bad type password
☑ POST   login User
☑ GET    get current User by refresh token
☑ GET    get current User by refresh token - bad refresh token
☑ GET    get current User by ID
☑ PATCH  patch current User - update email
☑ PATCH  patch current User - update password
☑ PATCH  patch current User - update phonenumber
```

If you did everything fine, it should be ok and all unit tests should be successful !

C - Remote server

Like for local server, you’ll have to import the collection of tests and to define an environment. But you have to select the file named **EIP_tests_routes_remote_server.postman_collection.json**.

These are so for the remote server where the API is at the address http://x2022psynder2988170462000.francecentral.cloudapp.azure.com:8080/.