ReDREAM
change your energy

D2.6 Co-Creative Connection with Third Parties and Devices

March 2022

Technical References

EU Initiative	Horizon 2020 | H2020-LC-SC3-2020-EC-ES-SCC
Grant Agreement Nr.	957837
Project Acronym	ReDREAM
Project Title	Real Consumer Engagement Through A New User-Centric Ecosystem Development for End-Users' assets In A Multi-Market Scenario
Project Coordinator	Universidad Pontificia Comillas
Project Duration	36 months

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

Deliverable No.	D2.6	
Dissemination level [1]	Public	
Work Package	WP2	
Task	D2.6: Co-creative connection with third parties and devices [18]	
Lead beneficiary	STEMY	
Responsible Authors	Carlos Álvarez, Carlos Becker	
Other authors/contributors	*STEMY*	*Carlos Rodriguez, Antonio Hernandez, Jaime Boal, Sergio Diaz, Alberto Castillo.*
	RIMOND	*Hamed Abbasi, Andreea Miroslav*
	SOULSIGHT	*Juan Martino*
	COMILLAS	*Francisco Martín, Badr Ghorbal, Alvaro Sanchez Miralles, Olga Rico, Ruben Rodríguez, Carmen Valor.*
	ENER	*Maria Regidor*
	ZeZ	*Lucija Nad, Mislav Kirac*
	BWCE	*Alison Turnbull*
	BIO	*Andrea Ferrante, Giacomo Nardoni*
	CIVI	*Martina Di Gallo, Angelo Giordano*
Contributing beneficiary(ies)	*Direct: Soulsight (Feedback from WP1)* *Indirect: RIMOND (Requirements), ZEZ, BWCE, BIO, ENER (Translations)*	
Due date of deliverable	*31 March 2022*	
Actual submission date	*31 March 2022*	

[1] PU = Public
PP = Restricted to other programme participants (including the Commission Services)
RE = Restricted to a group specified by the consortium (including the Commission Services)
CO = Confidential, only for members of the consortium (including the Commission Services)

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

Review

Reviewers	Stephane Galland (UBFC), Javier Rodrigo Gutierriez (OMIE)
Reviewing period	March 2022
Approved by reviewers	YES

Document History

Issue	Date	Author	Comments
V0.0	20/12/21	*Carlos Álvarez*	*Begin of document*
V1.1	03/01/22	*Carlos Álvarez*	*Added detailed sequence diagrams for Energy API*
V1.2	20/01/22	*Carlos Álvarez*	*Added Annex with Energy API usage examples*
V1.3	24/01/22	*Carlos Álvarez*	*Added Mobility API section*
V1.4	28/02/22	*Carlos Álvarez*	*Submitted for review*
V1.5	24/03/22	*Carlos Álvarez*	*Modified document according to reviewers' feedback*

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

Summary

ReDREAM Project

The energy market is rapidly transforming, and so is the role of the Consumer. Yesterday's passive consumers are central actors in today's energy markets. As new prosumers, energy markets can benefit from their generation, consumption and storage capabilities. The EU-funded ReDREAM project will enable the effective participation of consumers and prosumers in the energy market. The project will develop a strategy for creating a value generation chain based on a revolutionary service-dominant logic in which services are exchanged. The project will foster the demand response tools and energy/non-energy services that enable consumers to participate in the energy market. This will lead to the establishment of a new concept: a connected user-centred energy ecosystem.

Deliverable Summary

How does the final cloud architecture differ from the one proposed in Deliverable D1.6?
How are the cloud logins handled?
How are cloud-to-cloud communications handled?
How do Front-end interfaces and ReDREAM partners' services interact with the Energy API?
Energy API documentation
Mobility API documentation
Comfort API documentation
Energy API usage examples and common use cases

This deliverable will define the final details of the Cloud Architecture for the project, how logins are implemented and how cloud-to-cloud communications are handled. The Architecture diagram remains the same as the one provided in Deliverable D1.6 (Figure 1).

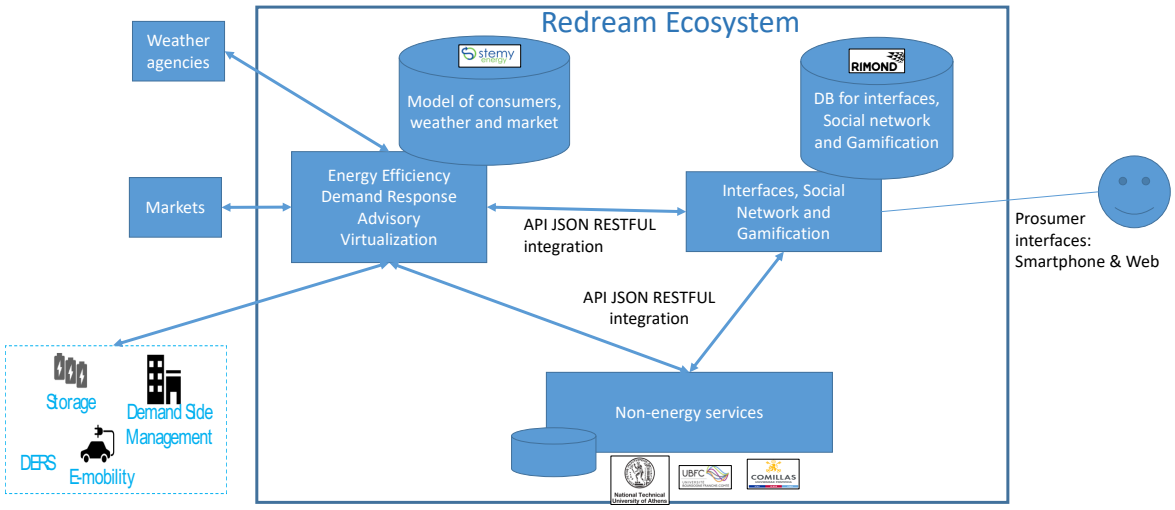


Figure 1: Interoperability of different systems in REDREAM

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

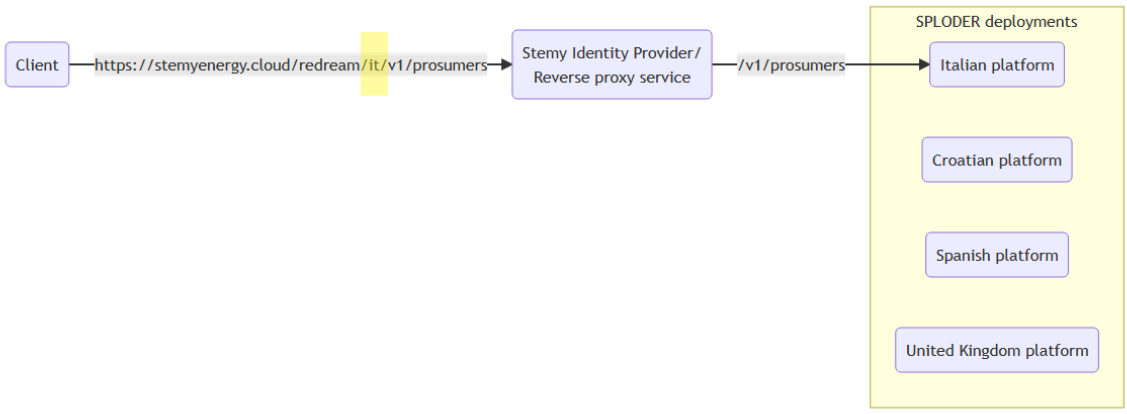


Figure 2: Stemy Energy API deployments

- Communications between services will be done using REST APIs.
- There will only be one Identity Provider Service for the project, the one implemented by STEMY. This Identity Provider Service will serve all users interacting with User Interfaces, such as web or mobile front-ends. ReDREAM partners' services will not be behind STEMY's reverse proxy.
- Authentication use cases will be performed using said Identity Provider Service, obtaining an access token.
- The front-end developed by RIMOND will store that token and will put it in the HTTP request headers made to the Energy API.
- If the front-end needs to request data from the Mobility API, for example, they will forward the valid token in the body of the HTTP request. If the Mobility API needs data from the Energy API, they will use that token to make the request.

This deliverable also documents the APIs for the Energy service, the Mobility service and the Comfort service, it provides a usage guide for the Energy API with examples and common use cases, and it defines multiple implementation details of the cloud architecture outlined in Deliverable D1.6, as a result of the collaboration of multiple ReDREAM partners such as RIMOND, NTUA, UBFC and STEMY. The Energy API is open to any user, not only ReDREAM partners but credentials must be requested to STEMY to access it. To request user credentials, please write to info@stemyenergy.com.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

Table of Acronyms

Acronyms	Description
ACS	Air Conditioning System
AP(s)	Action Point(s)
API	Application Programming Interface
CA	Consortium Agreement
EC	European Commission
ECP	Electric Charging Post
ECR	European Commission Reporting
EPOV	Energy Poverty Observatory
EV	Electric Vehicle
GA	Grant Agreement
GDPR	General Data Protection Regulation
H2020	Horizon 2020 programme
HBS	Household Budget Survey
DSO	Distribution System Operator
LTP	Linked Third Party
OAuth 2	Open Authorization
PC	Project Coordinator
PMB	Project Management Board
PTC	Project Technical Committee
RES	Renewable Energy Sources
TPR	Ten Percent Rule
TSO	Transmission System Operator
T&C	Terms & Conditions
UC	Use Case
UML	Unified Modelling Language
SILC	Survey on Income and Living Conditions
SOC	State of Charge
WP(s)	Work Package(s)

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

Disclaimer

This publication reflects only the author's view. The Agency and the European Commission are not responsible for any use that may be made of the information it contains.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

Table of Contents

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

List of Tables

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

List of Figures

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

1 Introduction

The document will be structured in the following way. In Section 2 we will start working based on the proposal written in Deliverable D1.6 and the final architecture will be outlined, as a result of working closely with several ReDREAM partners such as RIMOND, NTUA or UBFC.

Once we have defined the definitive cloud architecture, we will be able to describe how logins and sessions are handled in the system. After that, several common use cases will be described in detail with sequence diagrams to ensure that every developer knows how to handle the communication of several services APIs.

Three service APIs will be documented:

- Energy API
- Mobility API
- Comfort API

All the contributions of this deliverable will be gathered in the Conclusions chapter. This context of this deliverable is found in Layer 2 (Open co-creation) and Layer 5 (Open services pools), as seen in Figure 3. Exhaustive documentation of common use cases for the Energy API has been included in Annex I.

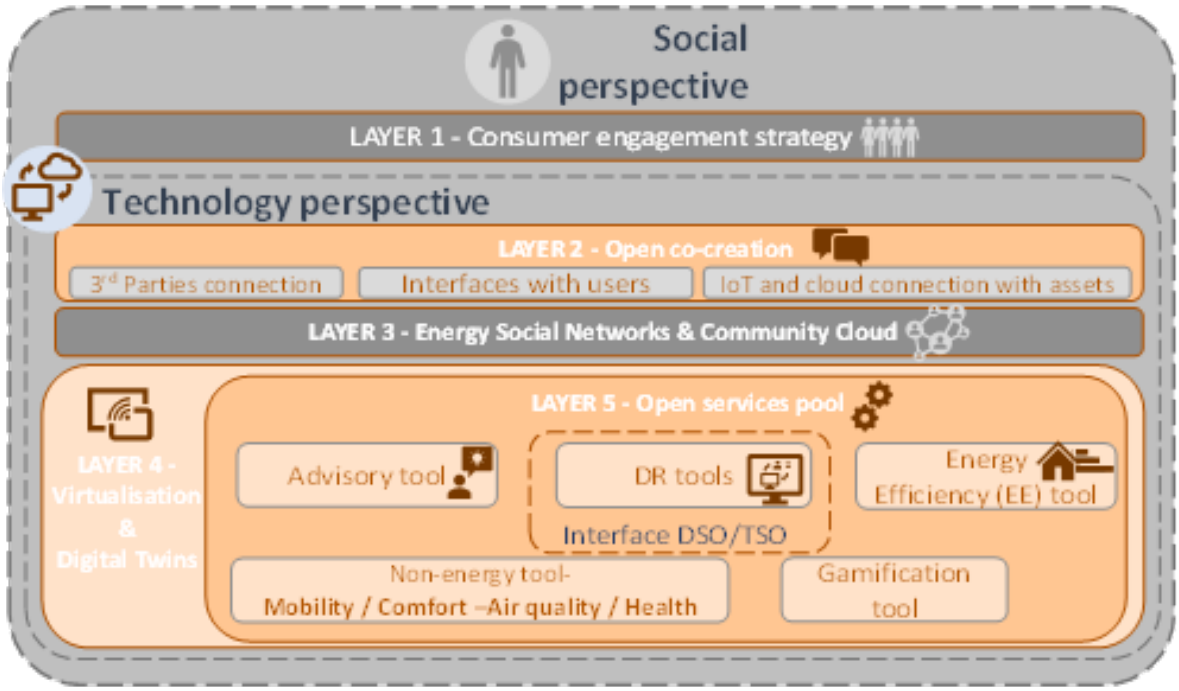


Figure 3: ReDREAM Ecosystem Layers

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

1.1 Energy API

The Energy API has several deployments, one per demo site or country. All the different deployment URLs will be identified. If one partner needs data from several demo sites, they will need to make the same request with different parameters for each Energy API deployment.

The documentation of the Energy API will be shared using the OpenAPI 3.0 format and will be accessible for REDREAM partners. A shortlist of the most common requests will also be included, and examples of common use cases will be added in the Annex.

In the context of the Energy API, details about how the anonymization process works will be described. The Energy API is open to any user, not only ReDREAM partners but credentials must be requested to STEMY to access it. To request user credentials, please write to info@stemyenergy.com.

1.2 Mobility API

The mobility service is compound by multiple unique services. Those services will be classified into different categories according to their relevance. This classification will be described in detail. Also, a brief description of the implementation details will be included, such as the Docker architecture and the use of a Docker compose file to get a productive environment quickly. The documentation will be added in Deliverable D3.5

1.3 Comfort API

The comfort service will have an API to expose its data. This API will allow users or services to retrieve data related to thermal comfort (e.g. air temperature, humidity, air quality, etc). The Comfort API will be documented using the OpenAPI standard. The comfort service will be documented thoroughly in Deliverable D3.6.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

2 Sign up and sign out from clouds – Authentication procedures.

The requirements for the authentication procedures were outlined in Deliverable D1.6. After working closely with Partners such as RIMOND, UBFC and NTUA; the final cloud architecture and cloud-to-cloud communications were defined.

2.1 Clouds' Architecture

STEMY has one instance of the Identity Provider/Reverse Proxy in the cloud. The purpose of this deployment is to handle authentication, issue access tokens, validate tokens and forward requests to the correct Energy API deployment based on URL path parameters rules. This service will be referred to in the document as Identity Provider, Reverse Proxy or Authentication service interchangeably.

STEMY has one deployment of the Energy API platform per country/demo-site, listed in Table 1:

Deployments	
Country	**URL path parameter**
Spain	es
United Kingdom	uk
Italy	it
Croatia	hr

Table 1: Energy API deployments

These Energy API deployments are completely independent and do not share information between them. STEMY's reverse proxy service forwards the request to the correct server based on a path parameter in the request's URL, as seen in Figure 4 and Figure 5.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

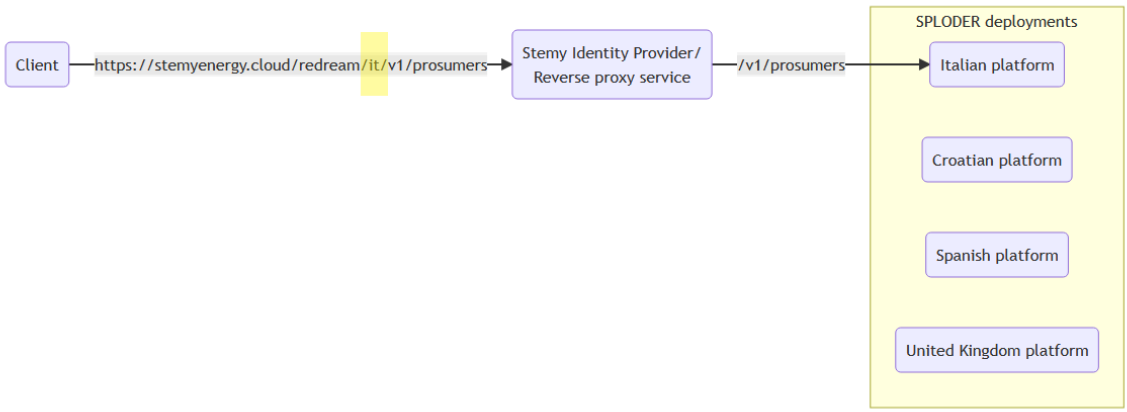


Figure 4: Request to the Italian Energy API deployment

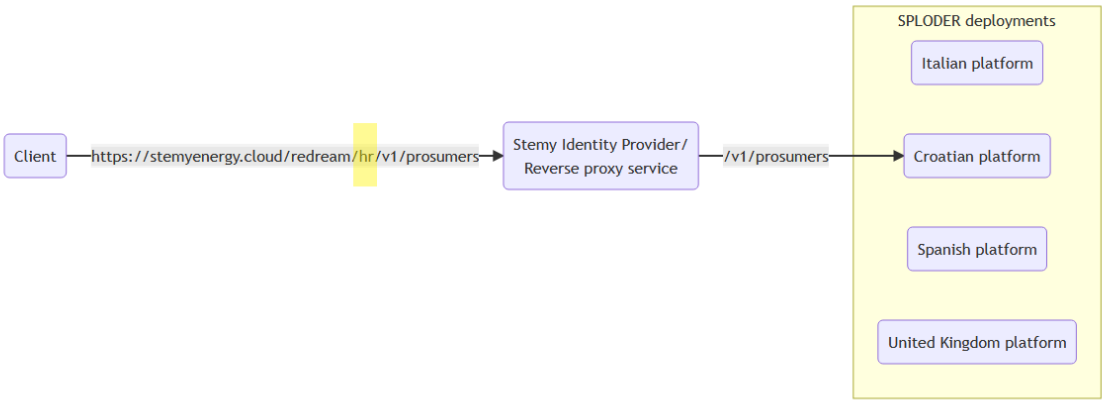


Figure 5: Request to the Croatian Energy API deployment

Even though it was mentioned in Deliverable D1.6, the use of STEMY's authentication server as a central reverse proxy to forward the requests to the correct partners API will not be needed. Therefore, only STEMY services will be behind STEMY's authentication service. As we will see in the following section, the agreed approach is for the front-end to request a token and forward it to any partner that might need it.

Additional techniques such as IP address whitelisting could be implemented to provide an additional layer of security. That is, for example, that NTUA will only accept incoming connections from RIMOND's backend server.

No additional changes are required in the previously outlined architecture.

2.2 Clouds' Logins

2.2.1 Previously proposed approach

In Deliverable D1.6, the approach to handling user logins was the following:

- STEMY has the master table of IDs since the users make the initial registration on their ecosystem.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

- Once a prosumer registration has happened (user and the password), STEMY will send that info to an endpoint of RIMOND (or any other developer that could need it) so they may store it.
- Then once the user makes the login through RIMOND Interface, it will request a token to the API of STEMY (or any other developer) to start making requests.

This approach has several disadvantages. Firstly, we are replicating sensible information such as user credentials in multiple instances, one per partner. This could lead to an inconsistent application state, where not all the partners have the same information. For example, a user could sign up in the STEMY Ecosystem. At that moment the user's credentials would have to be forwarded to all partners. If one of the partner's servers is down and cannot receive this information, the state of our application would be inconsistent. (See Figure 6).

This approach also complicates common use cases such as Password Reset requests, since the information would have to be replicated in every partner's database.

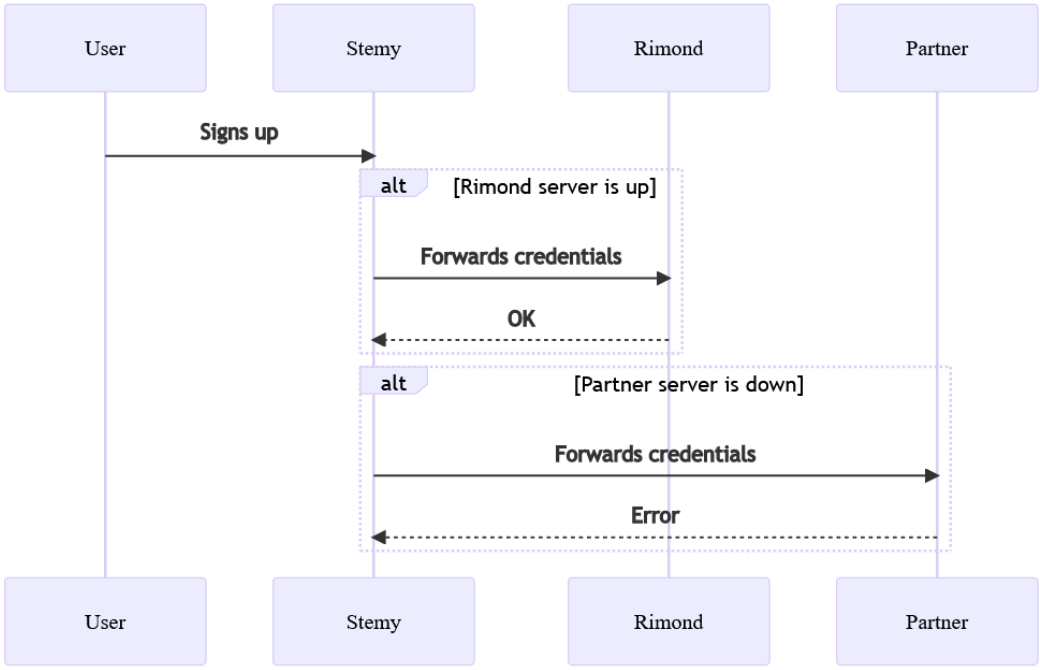


Figure 6: Disadvantages of the proposed approach in Deliverable D1.6

2.2.2 New agreed approach

The newly agreed approach greatly simplifies the previous one. Instead of replicating the user credentials multiple times, STEMY will be the only identity provider in the project, but ReDREAM partners' services will not be behind STEMY's reverse proxy.

- New users will sign up using the STEMY Ecosystem.
- Existing users will be able to sign in using the STEMY Ecosystem.
- Existing users will be able to restore their passwords using the STEMY Ecosystem.
- Existing users will be able to sign in to interfaces developed by ReDREAM partners like RIMOND using the credentials they used to sign up into the STEMY Ecosystem.
    - RIMOND will request an access token with the user credentials to STEMY's Identity Provider Service. This eliminates the need for RIMOND to store the user's credentials.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

- RIMOND's interface will use the provided access token to authenticate with STEMY's Energy REST API and make requests.
- If RIMOND's interface needs to communicate with another ReDREAM partner, they will forward the access token in the request. This way, the partner will be able to make requests to the STEMY API too.

The following section describes the integration between services in more detail.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

3 Integration between services

3.1 How a Front-end interface interacts with the Energy API

The Energy API is protected behind STEMY's authentication server/reverse proxy, as described in Deliverable D1.6. To access the Energy API, users must have a valid access token issued by STEMY's authentication server. The first step is to communicate with the Energy API is to request a valid token.

The process is described in Figure 7. The front-end must perform the following request (Table 2):

Method	*POST*
URL	https://stemyenergy.cloud/oauth2/token
Headers	Authorization: Basic base64(email:password) Content-Type: application/x-www-urlencoded
Body	grant_type=client_credentials&scope=DESIRED_SCOPES

Table 2: Request get a valid access token

The email and password must be concatenated using the colon character ':' and encoded using the base64 algorithm.

A crucial part of the request is to specify the token scopes. "*Scopes are a mechanism in OAuth 2.0 to limit an application's access to a user's account. An application can request one or more scopes […] and the access token issued to the application will be limited to the scopes granted*".[1] The STEMY authentication server supports two scopes:

- prosumers:read – Gives access to prosumers data that does not identify them, such as variable values. E.g. indoor temperature measurement.
- prosumers:personal_data – Gives access to sensitive data that may identify the user, such as name, surname, phone, address, etc.

If STEMY's Identity Provider server responds with a valid token, that would mean that the credentials entered by the user are correct. If STEMY's Identity Provider server responds with an Unauthorized, the credentials are invalid. If this happens, users could be redirected to the STEMY Ecosystem to restore their passwords if needed, as shown in Figure 7.

[1] OAuth 2.0 Scopes - https://oauth.net/2/scope/

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

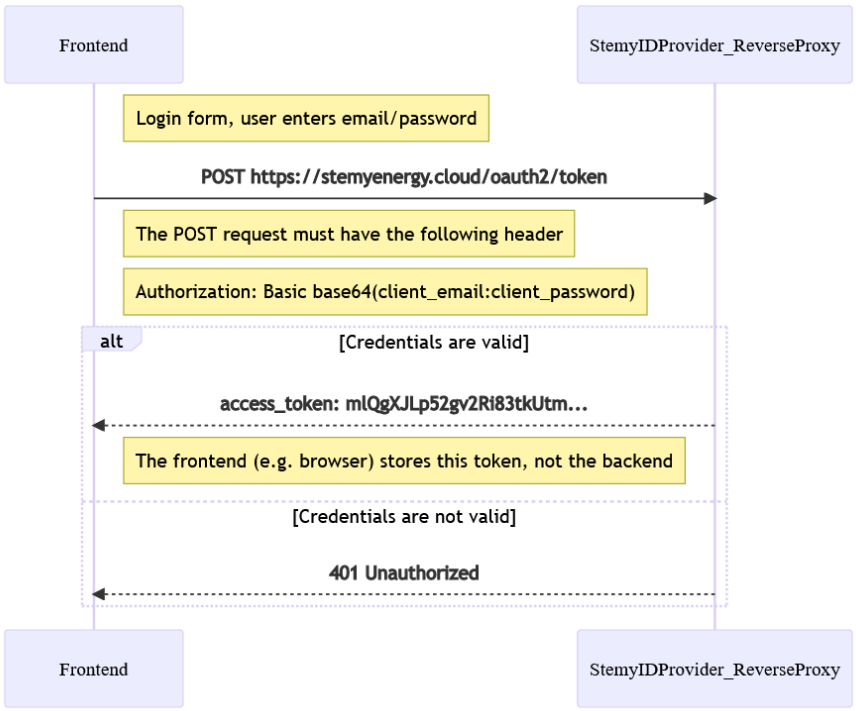


Figure 7: Requesting a valid token to STEMY's ID Provider

In the project's use case, the front-end application may require access to the user's personal data, but third-party services should not be able to identify the prosumer by name, surname or address or any other personal information; they will only know their unique ID. If the front-end application needs access to personal data, but also needs to communicate with other ReDREAM partners' services, the front-end will need to request two tokens:

1. One with access to personal data, that is, using the scope: "prosumers:read prosumers:personal_data". This will be the token used by the front-end to display the users' name, for example.
2. Another without the prosumers' personal data scope: "prosumers:read". ***This token will be the one forwarded to ReDREAM partners***, as explained in the following sections.

For example, if we wanted to get a token using the cURL tool, the command would look like this:

```
curl --location --request POST 'https://stemyenergy.cloud/oauth2/token' \
--header 'Authorization: Basic base64(email:password)' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'grant_type=client_credentials' \
--data-urlencode 'scope=prosumers:read prosumers:personal_data'
```

The valid response to a token request would be the following:

```
{
   "access_token": "7VzRfHAvbHAKDkvJTWvtL0qtKbDQirMlcdTM5vSnQWk.BniLIK9oOaWptBUr_c1jmZg23Nx9wvZUmNA1uaEuFDM",
   "expires_in": 86399,
   "scope": "prosumers:read prosumers:personal_data",
   "token_type": "bearer"
}
```

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

Once the front-end gets a valid token, it will be its responsibility to store it. Ideally, it should not be stored in the RIMOND backend, only in the front-end. Every request that the front-end makes to the Energy API must have the token in the HTTP request header like so:

Authorization: Bearer <TOKEN>

This token has an expiration time of 24 hours. If the token expires, users must be prompted to log in again with their credentials.

All requests related to a specific prosumer must have the *prosumerId* as a path parameter. However, the front-end client does not have that *prosumerId* at first. If a client logs in and makes a GET request to the */prosumers* endpoint, they will get the *prosumerId* in the response body. This request will serve two purposes:

1. Validate the token. If the token is expired or invalid, the request will throw a 401 Unauthorized response
2. Act as a *whoami* function, it will tell you the *prosumerId* associated with that token in STEMY's database.

This use case is described in Figure 8:

- We assume that the front-end already has a valid token.
- The front-end makes a *GET* request to */prosumers* to find out which *prosumerId* correspond to the token
- The Energy API responds with a *prosumerId*: *100*
- The front-end may request the KPIs of the prosumer by making a *GET* request to */prosumers/100/kpis-summary/*
- It might be the case that the token is expired, so the front-end will receive an *HTTP* code *of 401 Unauthorized*. In that case, the front-end must prompt the user to enter its credentials again to request a new valid token.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

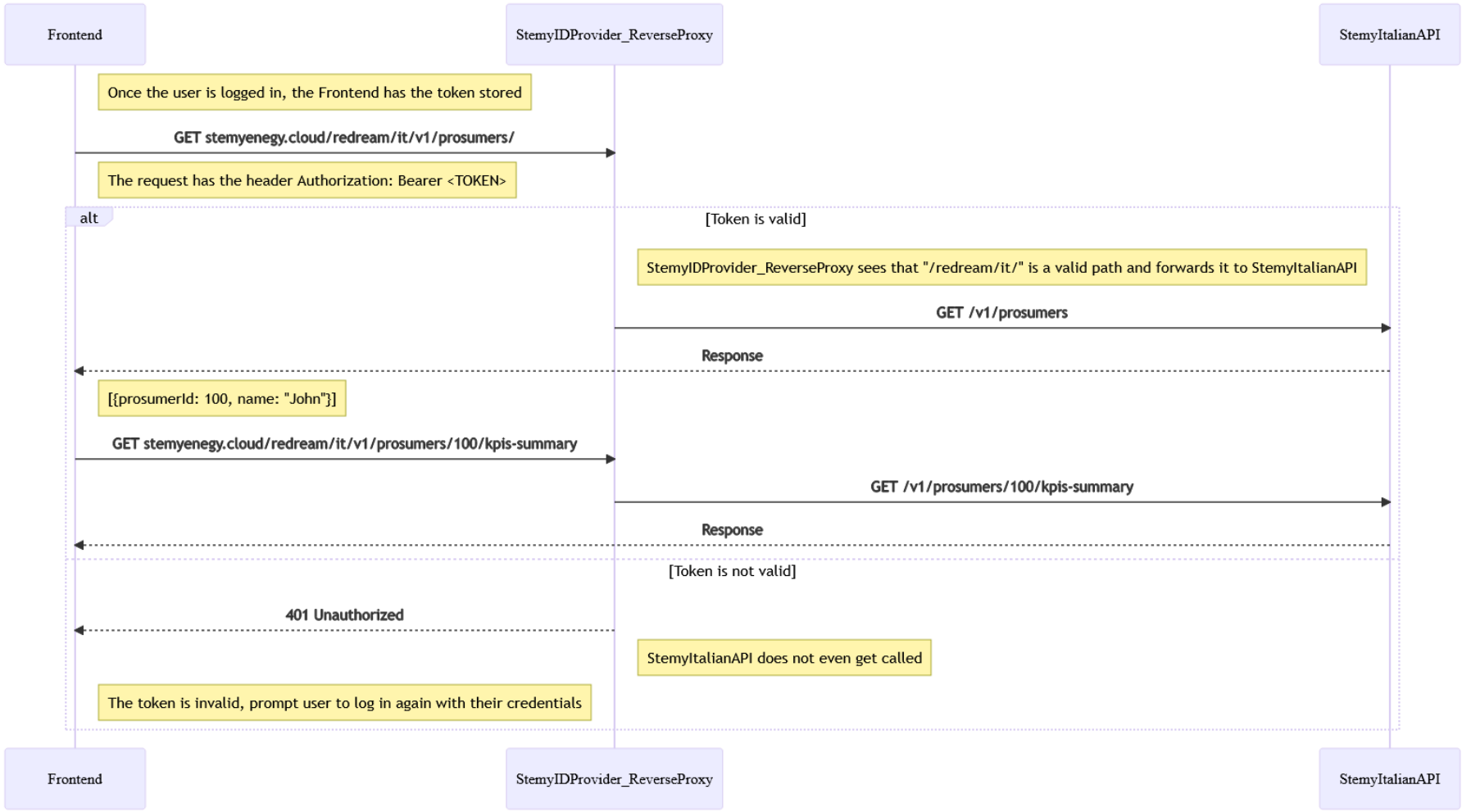


Figure 8: Front-end interacting with the Energy API

3.2 How a Front-end interface interacts with ReDREAM partners

3.2.1 Interaction with just one ReDREAM partner

As seen in Figure 9, there are the following actors:

- Front-end: interface, social network and gamification. E.g., RIMOND.
- Redream partner: comfort service, mobility service. E.g., NTUA, UBFC, etc.
- Energy API.

In this case, the front-end will request data from a ReDREAM partner. This partner might have to request data from the Energy API. To do that, the front-end will make an HTTPS request with the user token in the request body, not in the request URL for security reasons (read more about it here):

"URLs are stored in web server logs - typically the whole URL of each request is stored in a server log. This means that any sensitive data in the URL (e.g. a password) is being saved in clear text on the server"

Redream Partner's platform will use that token to perform the *whoami* request *(/prosumers*) to get the *prosumerId* and make the necessary requests to fetch the data.

This use case is described step by step in Figure 9:

- We assume that the front-end already has a valid token.
- The front-end will request a ReDREAM partner, forwarding the valid token issued by STEMY.
- The ReDREAM partner's service will perform a *GET* request to */prosumers* to find out which *prosumerId* correspond to the token.
- The Energy API responds with a *prosumerId*: *100.*
- The ReDREAM partner's service may request the KPIs of the prosumer by making a *GET* request to */prosumers/100/kpis-ev/*.
- It might be the case that the token is expired, so the front-end will receive an *HTTP* code *of 401 Unauthorized*. In that case, the ReDREAM partner's service will have to forward that code to the front-end, which must prompt the user to enter its credentials again to request a new valid token.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

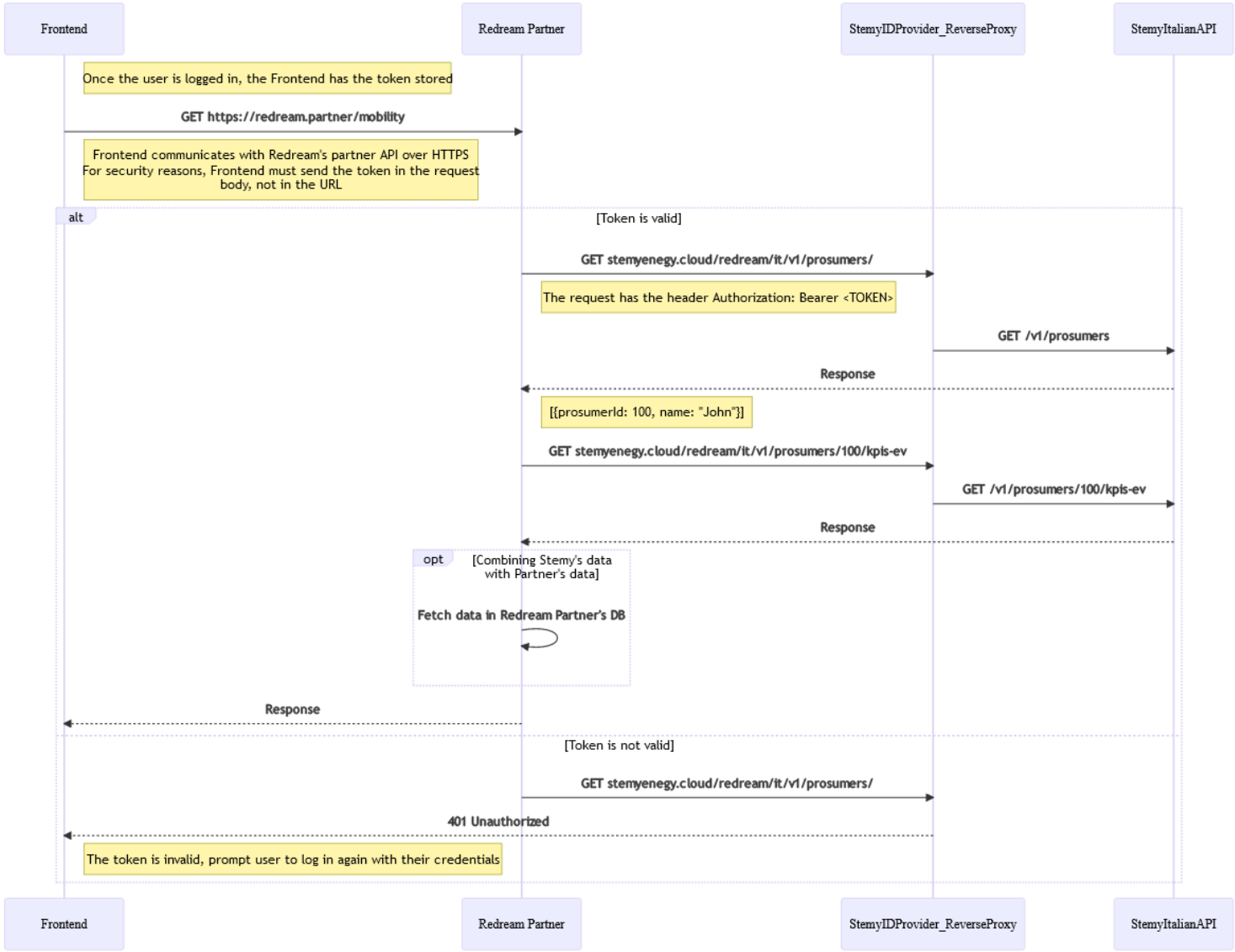


Figure 9: Front-end and one ReDREAM partner interacting with the Energy API

3.2.2 Interaction with multiple ReDREAM partners

This use case (Figure 10) is almost identical to the previous one. If one of the partner's services needs to communicate with another partner, they will forward the user token at the request of the body too. Then, the process should be the same as the one described in the previous section. Interaction with just one ReDREAM partner

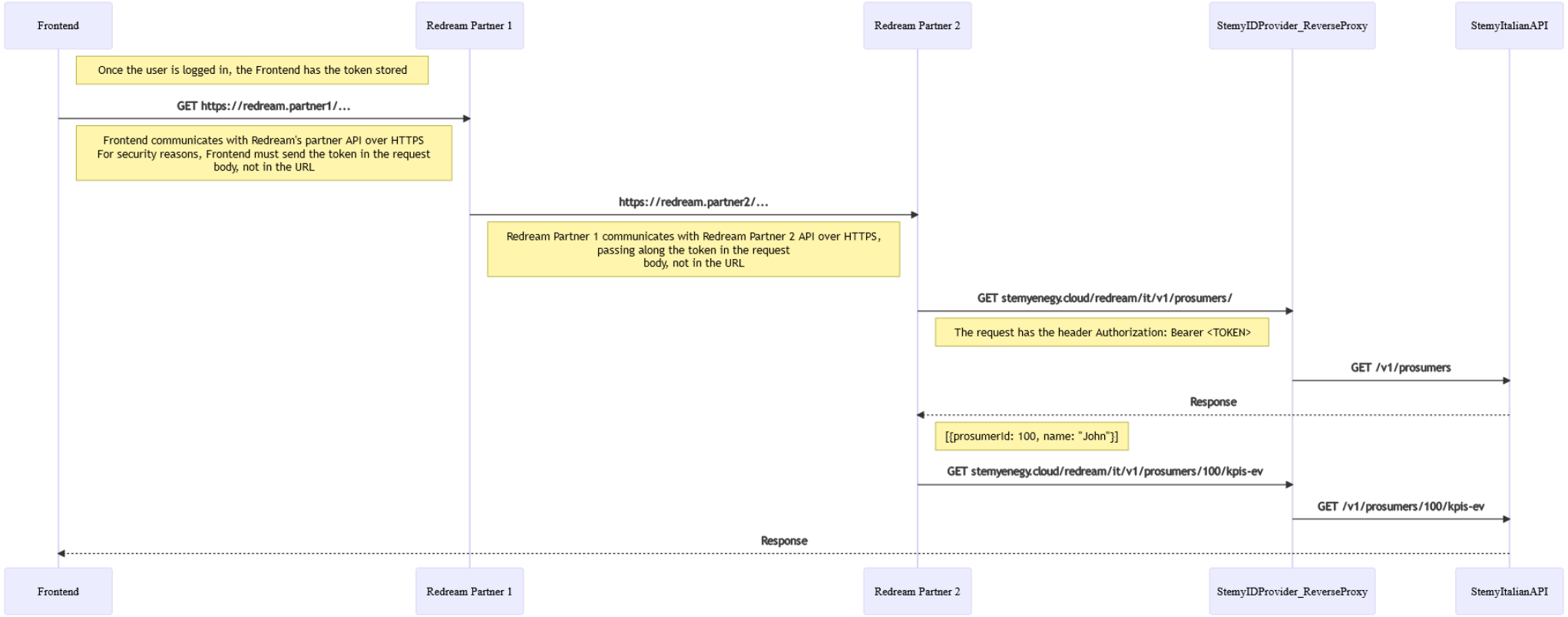


Figure 10: Front-end and multiple partners interacting with the Energy API

4 Energy API documentation

4.1 OpenAPI full documentation

The Energy API developed by STEMY Energy is documented using the OpenAPI specification.

"The OpenAPI Specification (OAS) defines a standard, language-agnostic interface to RESTful APIs which allows both humans and computers to discover and understand the capabilities of the service without access to source code, documentation, or through network traffic inspection. When properly defined, a consumer can understand and interact with the remote service with a minimal amount of implementation logic.

An OpenAPI definition can then be used by documentation generation tools to display the API, code generation tools to generate servers and clients in various programming languages, testing tools, and many other use cases."

As mentioned in previous Deliverables, there are four deployments of the SPLODER platform, one per demo site. Each deployment has its unique URL and the deployments do not share information between them. The details are explained in Table 3:

Energy API deployments	
Demo Site	**URL**
United Kingdom	https://stemyenergy.cloud/redream/uk/v2/
Spain	https://stemyenergy.cloud/redream/es/v2/
Italy	https://stemyenergy.cloud/redream/it/v2/
Croatia	https://stemyenergy.cloud/redream/hr/v2/

Table 3: Energy API deployment URLs

The documentation file of the Energy API may be accessed using the following link:

https://stemyenergy.com/documentation/apis/redream

The password for the zip file is the following:

Word-Bagpipe-Canteen-Panic-Sterling1-Keg

The zip includes the following files:

- index.html – File with the interactive documentation.
- Stemy Energy.postman_collection.json - Postman collection with all the requests, please refer to Annex I: Energy API usage example for more details.
- stemy-energy-platform-api.yml – OpenAPI source file, to be used with OpenAPI visor/editors.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

At the time of writing this document, the Energy API version is v2. If the Energy API evolves, changes will be reflected in the previous link. However, below is a summary of the most common requests in the Energy API.

The Energy API is open to any user, not only ReDREAM partners but credentials must be requested to STEMY to access it. To request user credentials, please write to info@stemyenergy.com.

4.2 Anonymization

Data anonymization is crucial for the project. This will allow accessing customers' data without being able to identify them. To work with anonymized data, Energy API clients will work only with resource ids. The mechanism to anonymize data is using token scopes, as detailed in How a Front-end interface interacts with the Energy API. All the requests that return personal data performed with tokens requested without the "prosumers:personal_data" scope will return empty strings in sensible fields such as name, surname, address, coordinates, etc. It is the responsibility of the front-end application to request a token without said scope and forward it correctly to other ReDREAM partners' services.

4.3 Energy API user roles

The Energy API has the following user roles, ordered from lower to higher access level:

- **Prosumer**. A prosumer would be an individual consumer.
- **Manager**. A manager manages a portfolio of assets (prosumers) and helps them with the onboarding process. Every prosumer has a manager assigned.
- **Aggregator**. They participate in the electricity market. Every manager and prosumer has an aggregator assigned.
- **Stemy**. Account role for sensible data.

4.4 API Requests: Aggregators

Requests to get information regarding the aggregators. Aggregators participate in the electricity markets and manage some devices from certain consumers to help the system to be balanced. Avoiding the thermal plants to be turned on and reducing drastically CO2 and other gases emissions to the atmosphere. Only aggregator credentials will be able to access this endpoint.

Aggregators	
Request	**Description**
GET /aggregators/{aggregator_id}	Get the details of an aggregator instance
GET /aggregators/{aggregator_id}/der-electric-consumption-values	Enumerates the consumption values of the aggregator's DERs
GET /aggregators/{aggregator_id}/market-variables-values	Enumerates the values of the aggregator's market variables
GET /aggregators/{aggregator_id}/market-variables	Enumerates the aggregator's market variables
GET /aggregators/{aggregator_id}/prosumers	Enumerates the aggregator's prosumers

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

GET /aggregators/{aggregator_id}/prosumers-base-demand-values	Enumerates the aggregator's prosumers base demand values
GET /aggregators/{aggregator_id}/prosumers-electric-from-grid-values	Enumerates the aggregator's prosumers electricity from grid values
GET /aggregators	Enumerates the aggregators

Table 4: Aggregators API requests

4.5 API Requests: Regions-CO2

Requests to get information regarding the Regions CO2. These regions are used in the Energy API to track the CO2 emissions caused by the energy mix of each region. Only aggregator credentials will be able to access this endpoint, regular prosumer credentials will not have access.

Regions CO2	
Request	**Description**
GET /regions-co2	List all CO2 regions
GET /regions-co2/{region_id}	Gets CO2 region data
GET /regions-co2/{region_id}/emissions	Gets CO2 region emissions by region id

Table 5: Regions-CO2 API requests

4.6 API Requests: Devices

Requests related to device management. Examples of devices are:

- Electric vehicle charge point
- Heat pump
- Thermostat
- Electric radiator
- Domestic hot water tank

Every device is associated with a specific prosumer, and every device has one or several variables (e.g. temperature measurement, operation mode, active power measurement, etc.). Prosumer credentials will only be authorized to access devices with ids that are associated with their account. For example:

- Alice has two devices with the following ids: 1, 2
- Bob has one device with the following id: 3
- If Alice tries to access Bob's device, the Energy API will respond with a 401 Unauthorized response.

Managers and aggregators will be only authorized to access devices associated with their Prosumers.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

Devices	
Request	***Description***
POST /devices	Creates a new instance of a device
PUT /devices/{device_id}	Updates an existing device
GET /devices/{device_id}	Obtains a specific device
GET /devices/{device_id}/variables	Enumerates the device's variables

Table 6: Devices API requests

4.7 API Requests: Locations

Requests related to locations management. Locations are a way to organize variables. Locations may contain other locations, so you may have a hierarchy. Locations are a way for prosumers to organize the user experience. For example:

- **Home**: Main location
  - **Upper floor**: child location
    - **Temperature Measurement**: Variable belonging to a Thermostat device on the upper floor
  - **Lower floor**: child location
    - **Temperature Measurement**: Variable belonging to a Thermostat device on the lower floor

Indirectly, locations also organize devices. Since every variable belongs to a device, that device will be in the location of its variables.

Prosumers will only be able to access their locations, managers and aggregators will only be able to access their prosumers' locations.

Locations	
Request	***Description***
POST /locations/	Create new location
GET /locations/{location_id}	Get location
PUT /locations/{location_id}	Update location

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

Locations	
Request	***Description***
DELETE /locations/{location_id}	Delete location
GET /locations/{location_id}/devices	List location devices
GET /locations/{location_id}/value	Get the latest location average value of a magnitude
GET /locations/{location_id}/values-aggregated	List location average values of a magnitude aggregated by time slice
GET /locations/{location_id}/variable	List location variables

Table 7: Locations API requests

4.8 API Requests: Managers

Requests related to managers. Managers manage a portfolio of assets (prosumers) and are in charge of the onboarding process of new prosumers. However, they do not participate in the market by themselves, that is the role of aggregators.

Each manager will only be granted access to himself. Aggregators will have access to the managers associated with them.

Managers	
Request	**Description**
GET /managers/{manager_id}	Get the details of a manager instance
GET /managers/{manager_id}/der-electric-consumption-values	Enumerates the consumption values of the manager's DERs
GET /managers/{manager_id}/prosumers	Enumerates the manager's prosumers
GET /managers/{manager_id}/prosumers-base-demand-values	Enumerates the manager's prosumers base demand values
GET /managers/{manager_id}/prosumers-electric-from-grid-values	Enumerates the manager's prosumers electricity from grid values
GET /managers	Enumerates the managers

Table 8: Managers API request

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

4.9 API Requests: Markets

Requests related to the electricity markets. Markets are accessible to prosumers only if their electric tariff (which is a market variable) belongs to the said market. Aggregators and managers have full access to all markets.

Markets	
Request	***Description***
GET /markets/{market_id}	Obtains a specific market
GET /markets	Enumerates all markets.
GET /markets/{market_id}/market-variables	Obtain all market variables related to a specific market.

Table 9: Markets API Requests

4.10 API Requests: Market variables

Requests that deal with market variable and their values. Electricity markets have several data stored in variables. Prosumers only have access to their tariff market variable. Aggregators and managers have full access to all market variables.

Market variables	
Request	**Description**
GET /aggregators/{aggregator_id}/market-variables-values	Enumerates an aggregator's market variables values.
GET /aggregators/{aggregator_id}/market-variables	Enumerates an aggregator's market variables.
GET /market-variables/{market_variable_id}	Returns the selected market variable.
GET /market-variables/{market_variable_id}/values	Enumerates a market variable's values.

Table 10: Market variables API requests

4.11 API Requests: Optimizations

Requests that deal with optimizations algorithms and their outputs.

Only aggregator accounts have access to optimizations.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

Optimizations	
Request	**Description**
GET /optimizations	Enumerates an aggregator's market variables values.
GET /optimizations/{optimization_id}/prosumer-profiles-out	Get an optimization's prosumer profiles output data
GET /optimizations/{optimization_id}/prosumer-der-profiles-out	Get an optimization's prosumer der profiles output data

Table 11: Optimization API requests

4.12 API Requests: Products

Requests that deal with STEMY and third-party products. Prosumers, managers and aggregators have access to products.

Products	
Request	**Description**
GET /products	Obtain all existing products
GET /products/{product_id}	Get product details

Table 12: Products API requests

4.13 API Requests: Prosumer Advisor service

Requests related to the Prosumer Advisor service. The objective of this service is to analyse each prosumer energy consumption profile to create an estimated power margin curve with a tag.

The power margin is calculated considering the predictions of electric consumption and generation taking into consideration the maximum contracted power.

The tag code is calculated considering the different electric prices along the day.

Prosumer Advisor service tags	
Tag	**Description**
1. Consumption strongly recommended	More generation than consumption expected
2. Consumption recommended	Consumption recommended
3. Consumption warning	Average prices of the day

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

4. Consumption not recommended	Most expensive prices of the day
5. Consumption forbidden	Consumption too close to the maximum contracted power

Table 13: Prosumer Advisor service tags

Since these endpoints use the prosumer id, only prosumers may access their data, and the associated manager and aggregators.

Prosumers	
Request	**Description**
GET /prosumers/{prosumer_id}/events	Get prosumer events related to the Energy Advisor Service and their tags
GET /prosumers/{prosumer_id}/power-margin-pred	List all consumption recommendations related to the Energy Advisor Service

Table 14: Prosumer Advisor service API requests

4.14 API Requests: Prosumers

Requests that deal with prosumers. Prosumers will only be able to interact with their prosumer id. This prevents a prosumer from accessing other prosumers' data.

Prosumers	
Request	**Description**
GET /prosumers	List all prosumers associated with the token
GET /prosumers/{prosumer_id}/devices	List prosumer devices
GET /prosumers/{prosumer_id}/base-demand-values	List prosumer base demand
GET /prosumers/{prosumer_id}/electric-from-grid-values	List prosumer electric energy from grid values
GET /prosumers/{prosumer_id}/kpis-summary	List of all general KPIs of the prosumer
GET /prosumers/{prosumer_id}/kpis-summary-day	List of all daily general KPIs of the prosumer
GET /prosumers/{prosumer_id}/kpis-flexibility	List of all flexibility KPIs of the prosumer
GET /prosumers/{prosumer_id}/indoor-temperature-values	List prosumer indoor temperature values
GET /prosumers/{prosumer_id}/outdoor-temperature-values	List prosumer outdoor temperature values

Table 15: Prosumers API requests

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

4.15 API Requests: Users

Requests that deal with users (email addresses).

Users	
Request	**Description**
GET /users	List all users associated with the token

Table 16: Users API requests

4.16 API Requests: Variables

Requests that deal with variables.

Variables	
Request	**Description**
GET /variables/{variable_id}/value	Get last variable value
GET /variables/{variable_id}/values	List variable values
GET /variables/{variable_id}/values-aggregated	List variable values aggregated by time slice

Table 17: Variables API requests

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

5 Mobility API documentation

5.1 General information

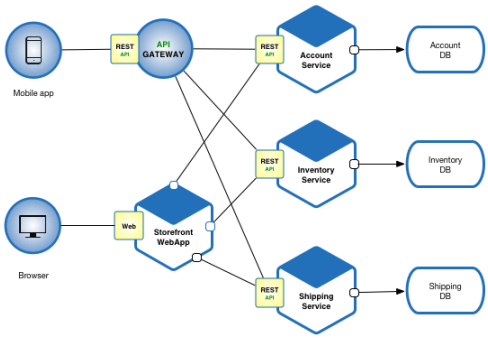
The mobility service is compound by 16 different unique services. These services can be spitted in 4 different categories:

- 3 services are compulsory and a part of the whole architecture management
- 2 services are compulsory and needed for data and authentication management
- 5 services are in charge of the web requests management and computation core of the whole project
- 6 services oversee others and could be optional in an unsupervised production environment

The services are all containerized in a Docker environment. A single docker compose file is able to start the whole Mobility service in a single command line.

The overall architecture is based on microservices architecture, an architectural style that structures an application as a collection of services that are:

- Highly maintainable and testable
- Loosely coupled
- Independently deployable
- Organized around business capabilities
- Owned by a small team



The microservice architecture enables the rapid, frequent and reliable delivery of large, complex applications. It also enables an organization to evolve its technology stack.

5.2 API documentation

The full documentation of the Mobility API is planned to be released in Deliverable D3.5, please refer to that document.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

6 Comfort API documentation

6.1 General information

The comfort API allows users or services to retrieve data related to thermal comfort (e.g. air temperature, humidity, air quality, etc). You may find a summary of the most common request in Table 18.

Thermal Comfort	
Request	**Description**
GET /	List thermal comfort data
GET /ClothingEnsembles	List all clothing ensembles by code and description
GET /MetabolicRates	List all possible metabolic rates such as “sleep”, “relax”, “dance”, etc.
GET /Cities	List all registered cities in the service with coordinates
GET /SensorData	Retrieve sensor data by sensor id

Table 18: Thermal Comfort API endpoints

6.2 API documentation

The Comfort API is documented using the OpenAPI standard. The documentation is accessible using the following link:

http://morpheus.thermo.mech.ntua.gr/swagger/index.html

The project leading to this application has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 957837.

7 Conclusion

The contributions of this deliverable (Table 19) are the following:

1. Define the implementation details of the Cloud Architecture for the project. There will only be one Identity Provider Service in the project, the one implemented by STEMY. All use cases related to Identity and Authentication will have to go through this service, such as sign in, sign up, password resets or access token requests.
2. Define how cloud-to-cloud communications will be handled, with special detail on the authentication side. Access tokens will be requested to STEMY's Identity Provider Service and will be passed around between the Services developed by ReDREAM partners. This will be done using the requests' body for additional security. Additional safety measures such as IP whitelisting will be adopted to ensure that only IPs from ReDREAM servers can talk to each other.
3. Document the following APIs:
    a. Energy API
    b. Mobility API
    c. Comfort API
4. Add exhaustive documentation for common use cases related to the Energy API.

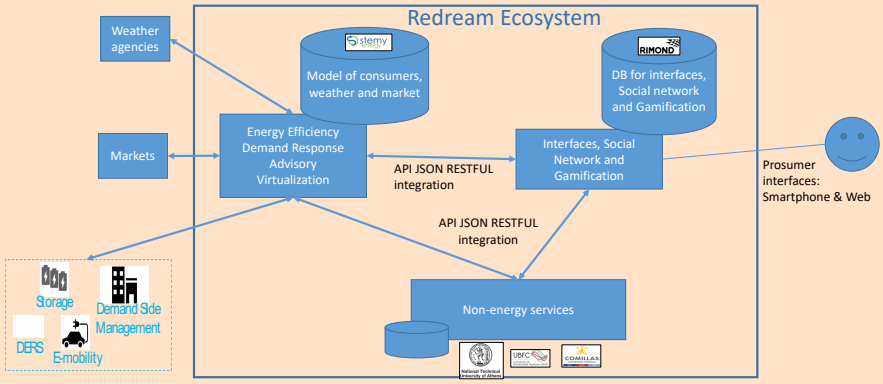
Contribution	Summary
Define the Cloud Architecture	
Handling authentication in cloud-to-cloud communications	STEMY will be the only Identity Provider in the project, but ReDREAM partners' services will not be behind STEMY's reverse proxy. Partners will request a valid token to STEMY's Identity Provider Service and will forward it if they need to communicate with other Partners' services
API Documentation	Documentation for the following REST APIs: 1. Energy API. The Energy API is open to any user, not only ReDREAM partners but credentials must be requested to STEMY to access it. To request user credentials, please write to info@stemyenergy.com. 2. Mobility API 3. Comfort API
Energy API usage examples	Exhaustive documentation of common use cases for the Energy API that may prove useful for ReDREAM service developers.

Table 19: Summary of conclusions

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

Annex 1: Energy API usage example

This section will explain in detail how to get a working development environment with Postman and document common use cases for the Energy API.

a) Download and install Postman

Postman is a REST API client app that will be used to illustrate all the examples in this annexe. It can be downloaded here.

b) Requesting access to the Postman API request collection and environment

To download the Postman API request collection, please refer to section 4.1 OpenAPI full documentation to download the files.

c) Import the Postman API request collection and environment

Firstly, download the documentation files to your development machine. To get a working Postman environment we must import the following file from the repository:

- *Stemy Energy.postman_collection.json*

To do that, click on the *Import* button as shown in Figure 11.

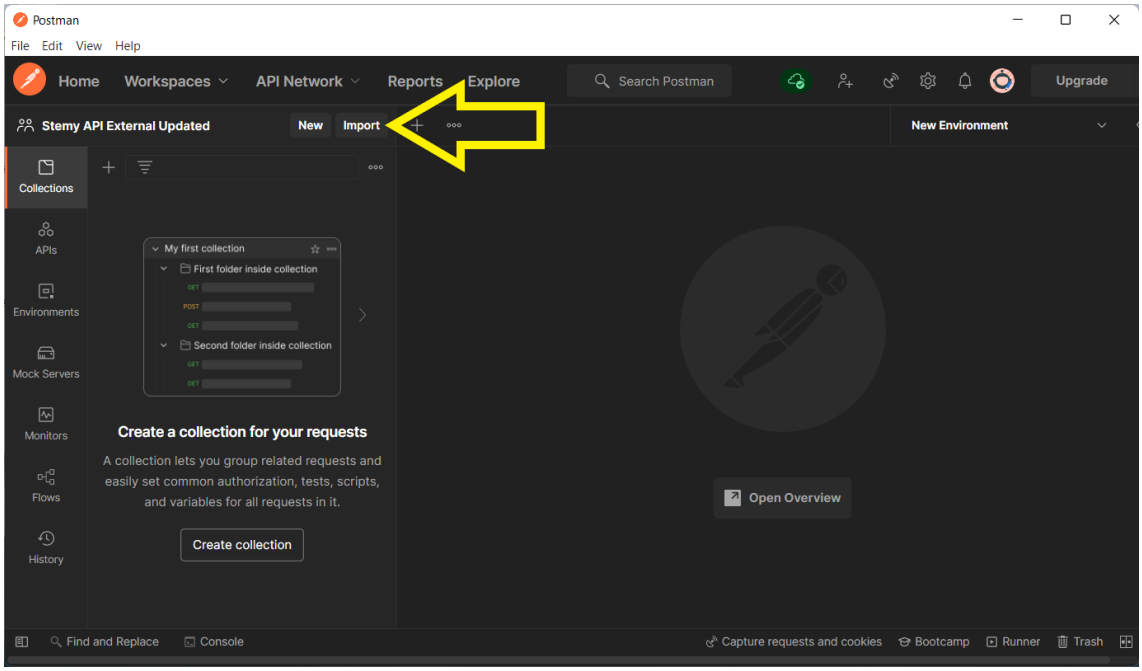


Figure 11: Import files in Postman

Next, click on the *Upload Files* button (Figure 12). Then, select the postman collection file.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

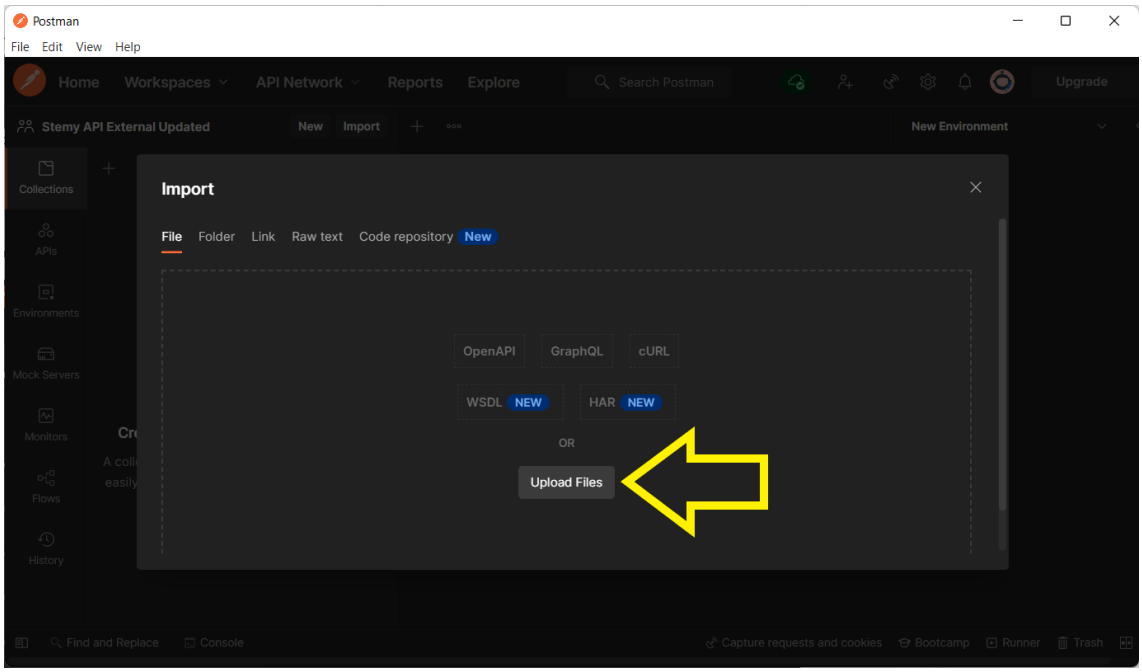
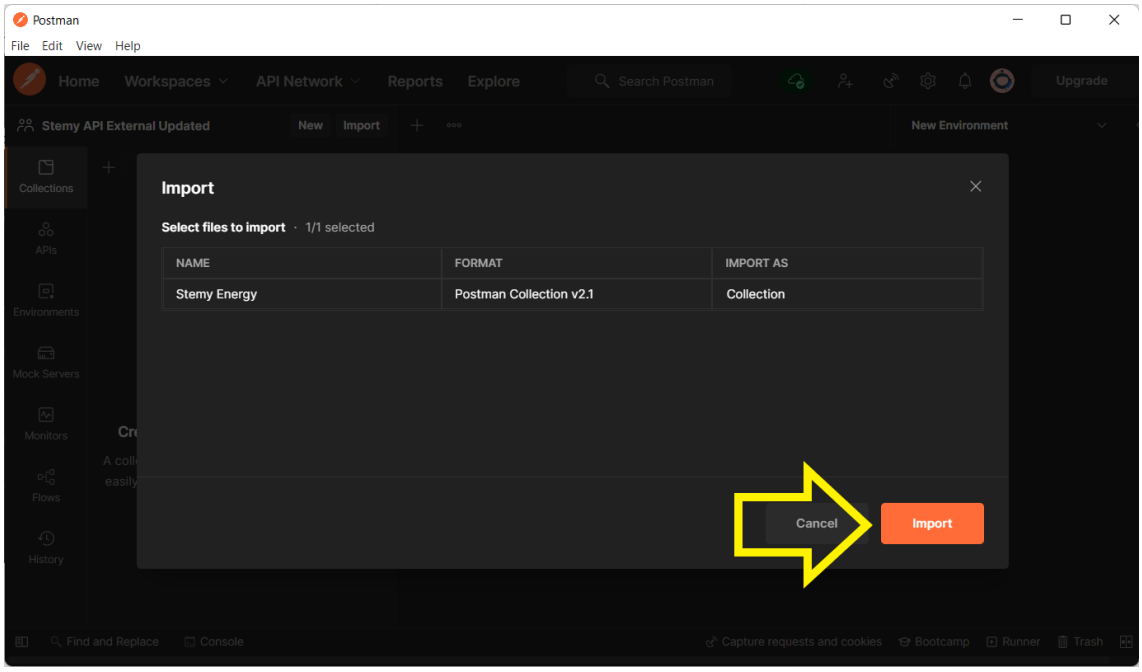


Figure 12: Upload files in Postman

Finally, click on the Import button to save the request collection.



The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

d) Requesting a token

First, ensure that you have the imported collection active. Click on the name of the collection (Stemy Energy), and then click on the Authorization tab (Figure 13).

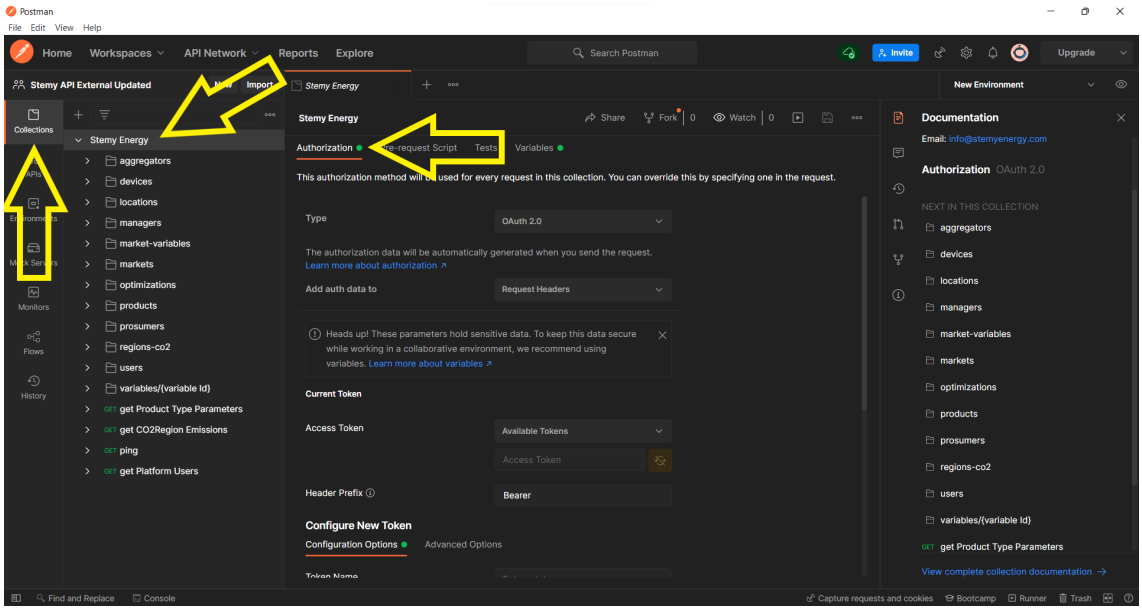


Figure 13: Access Authorization menu in Postman

To get an access token, scroll down until you see the Client ID and Client Secret input boxes. In there, you will have to input the following data:

- Grant Type: Client Credentials
- Access Token URL: https://stemyenergy.cloud/oauth2/token
- Client ID: the username that you want to log in with
- Client Secret: the password of said username.

Finally, click on Get New Access Token (Figure 14).

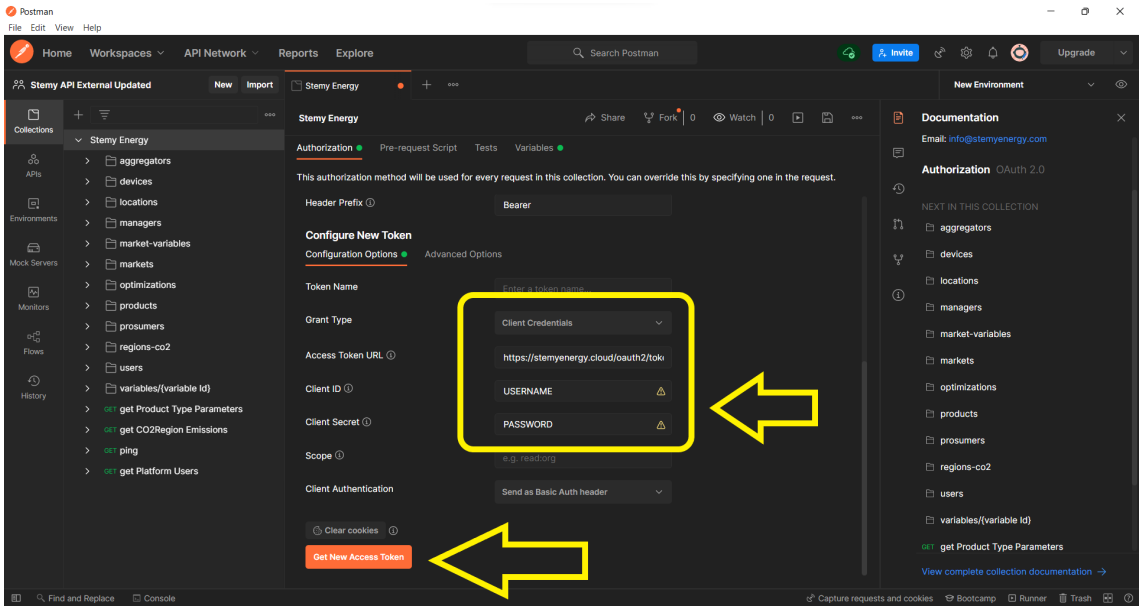


Figure 14: Getting a new Access Token

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

If the credentials are correct, you will see the Authentication complete dialogue (see Figure 15). If not, please review that you have completed all the previous steps correctly, in order and that the username and password you used are correct.

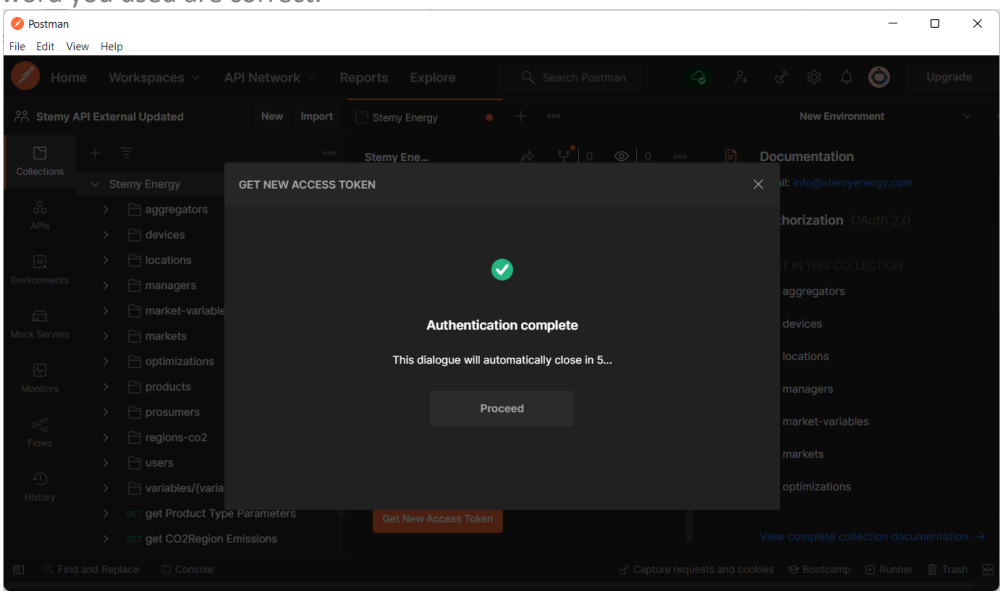


Figure 15: Success authenticating

You may give the token a name to be able to identify it using the pencil icon. When you are ready, press the Use Token (Figure 16).

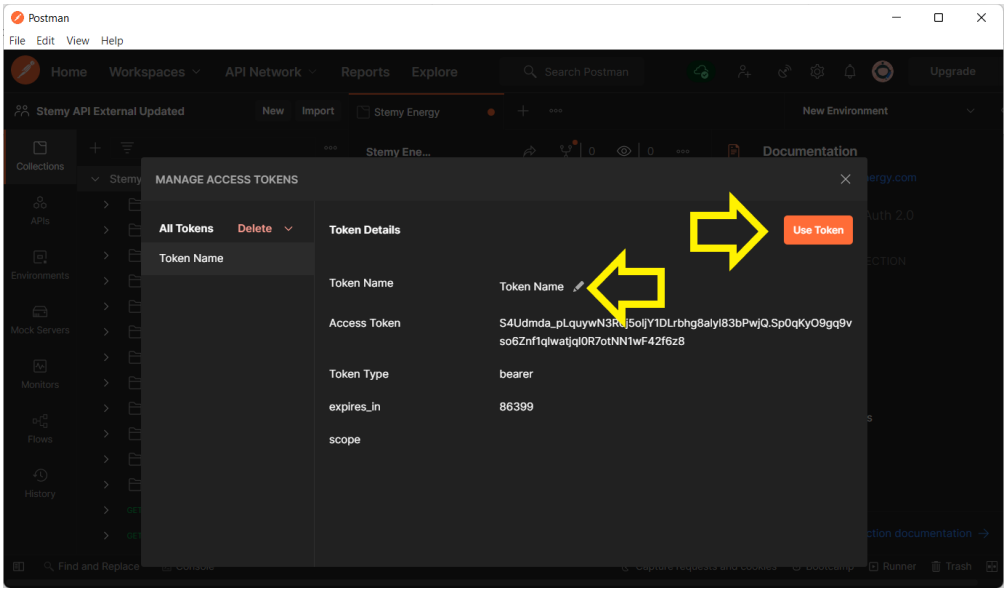


Figure 16: Renaming and saving the token

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

If you need to work with multiple tokens, you will be able to change them using the Access Token dropdown in the Authorization tab of the Stemy Energy Collection (Figure 17).

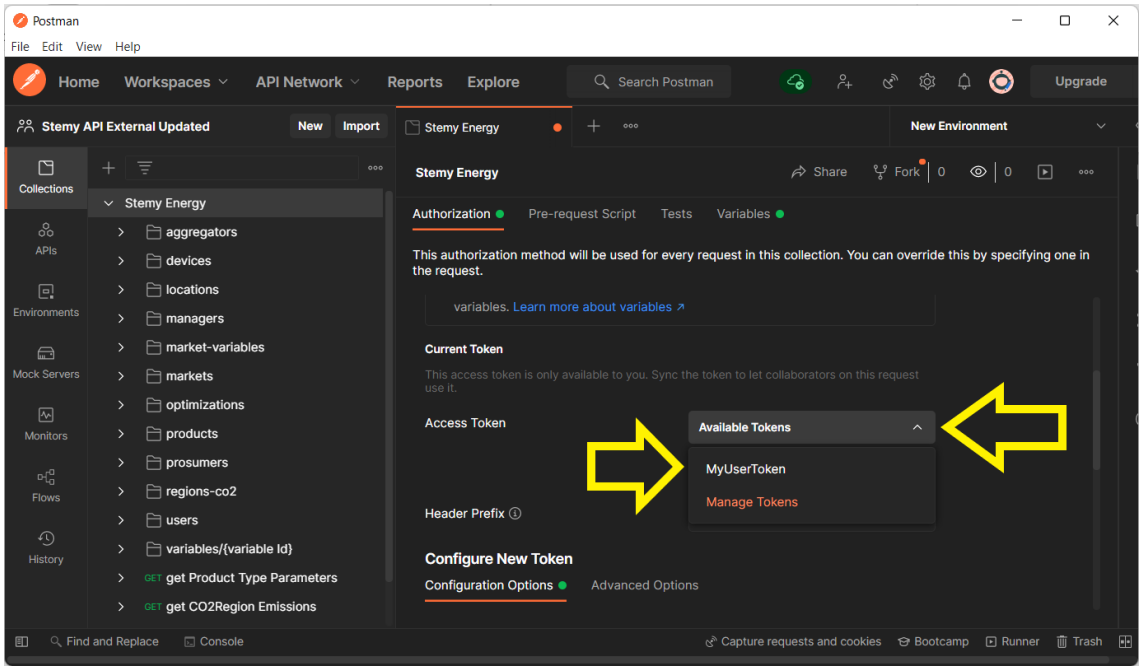


Figure 17: Managing multiple tokens

e) Changing the target Energy API deployment

You may change the target Energy API deployment by editing the baseUrl variable, under the Variables menu in the Stemy Energy collection (Figure 18).

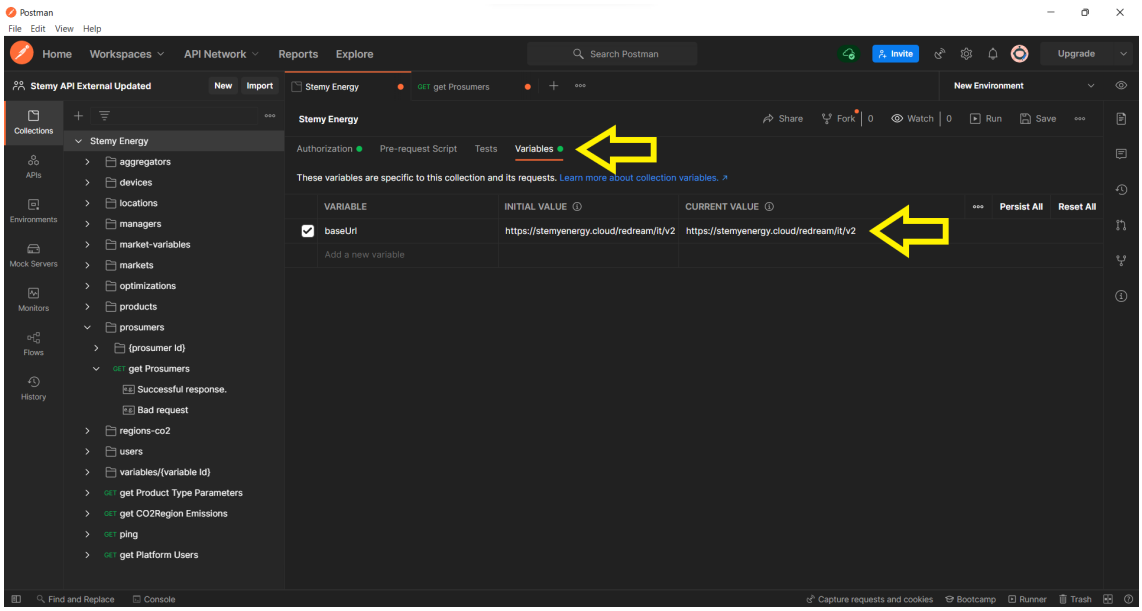


Figure 18: Changing target API deployment

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

f) Finding out the prosumerId associated with a token

All requests related to a specific prosumer must have the *prosumerId* as a path parameter. However, the front-end client does not have that *prosumerId* at first. If a client logs in and makes a GET request to the */prosumers* endpoint, they will get the *prosumerId* in the response body (Figure 19). This request will serve two purposes:

1. Validate the token. If the token is expired or invalid, the request will throw a 401 Unauthorized response
2. Act as a whoami function, it will tell you the *prosumerId* associated with that token in STEMY's database.

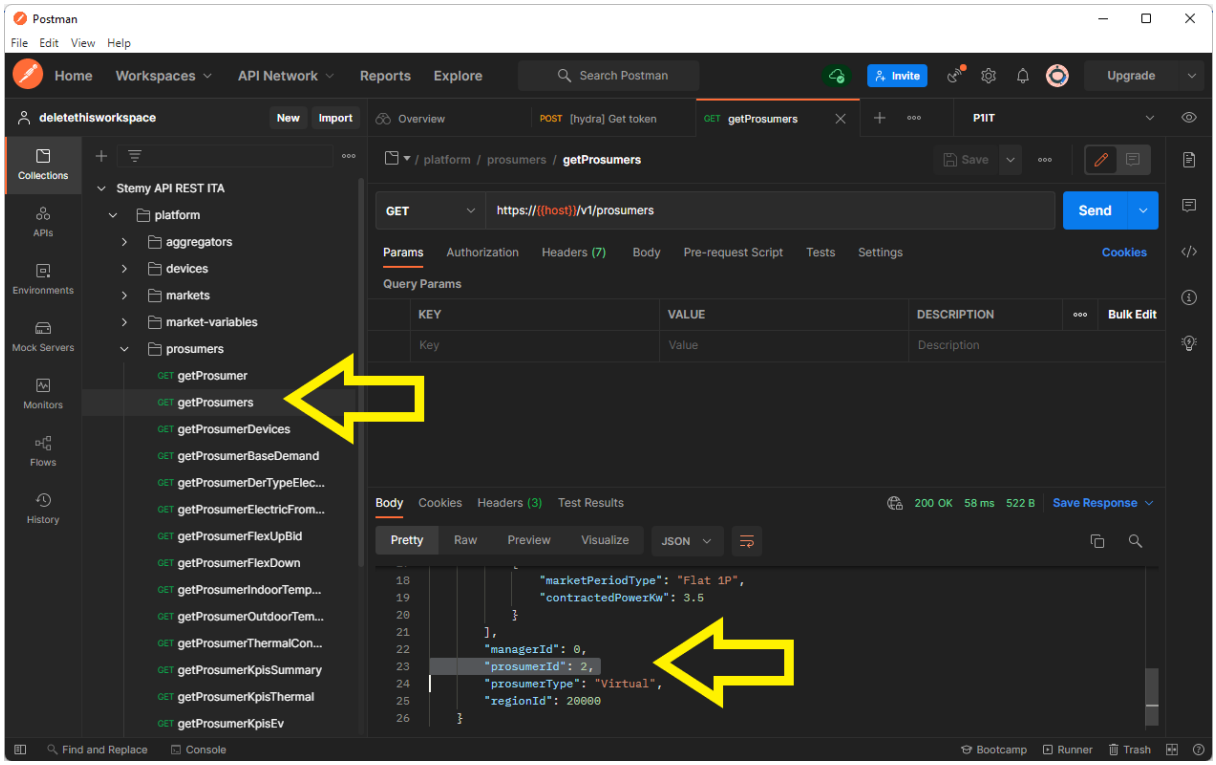


Figure 19: How to find out the prosumerId associated with a valid token

It is important to know that this works only for tokens associated with prosumers. If the token is associated with an aggregator, for example, the client must perform the /aggregators request to find their aggregatorId.

g) Getting information about the prosumer's energy tariff

The prosumer's energy tariff is a market variable. You may find the *marketVariableId* field in the /prosumers response body, as shown in Figure 20.

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

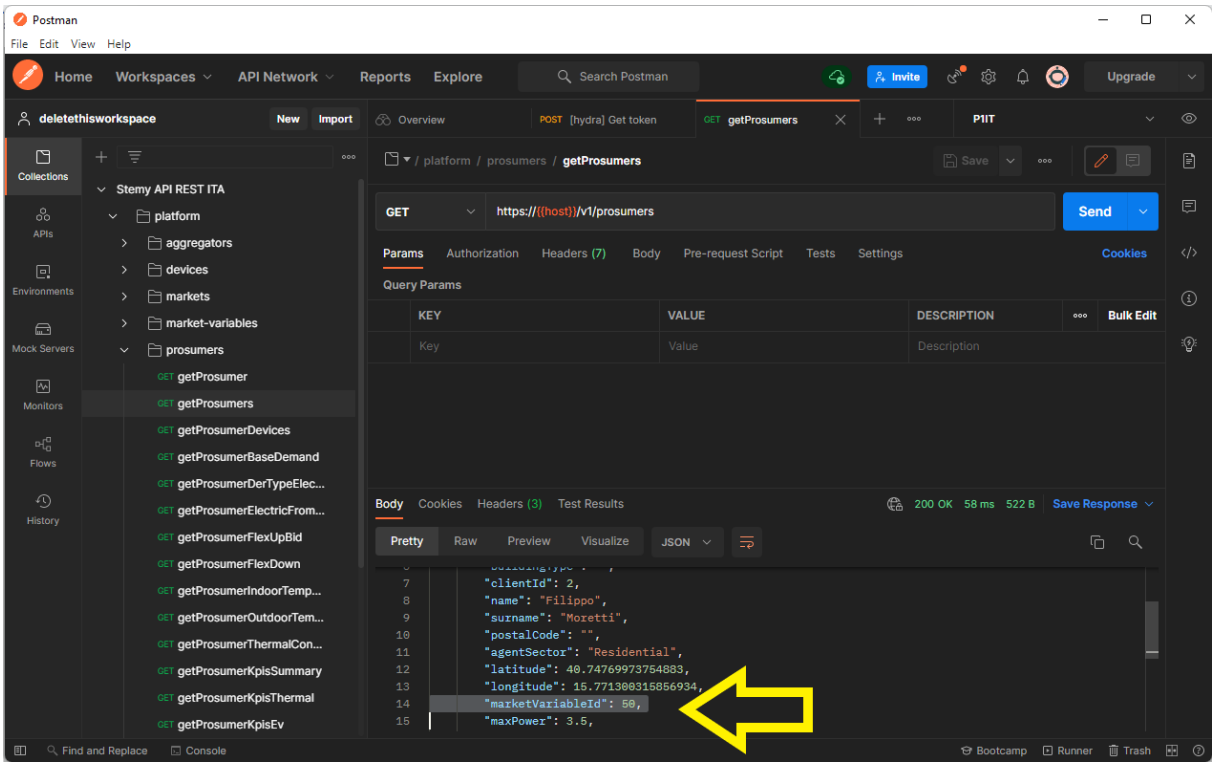


Figure 20: Getting the marketVariableId from the getProsumers request

Once you have the *marketVariableId*, you may navigate to the *market-variables/* folder in the left navigation menu and open the *getMarketVariableData* request. In this request, you will have to replace the value of the path variable *id_market_variable* with the *marketVariableId* (Figure 21).

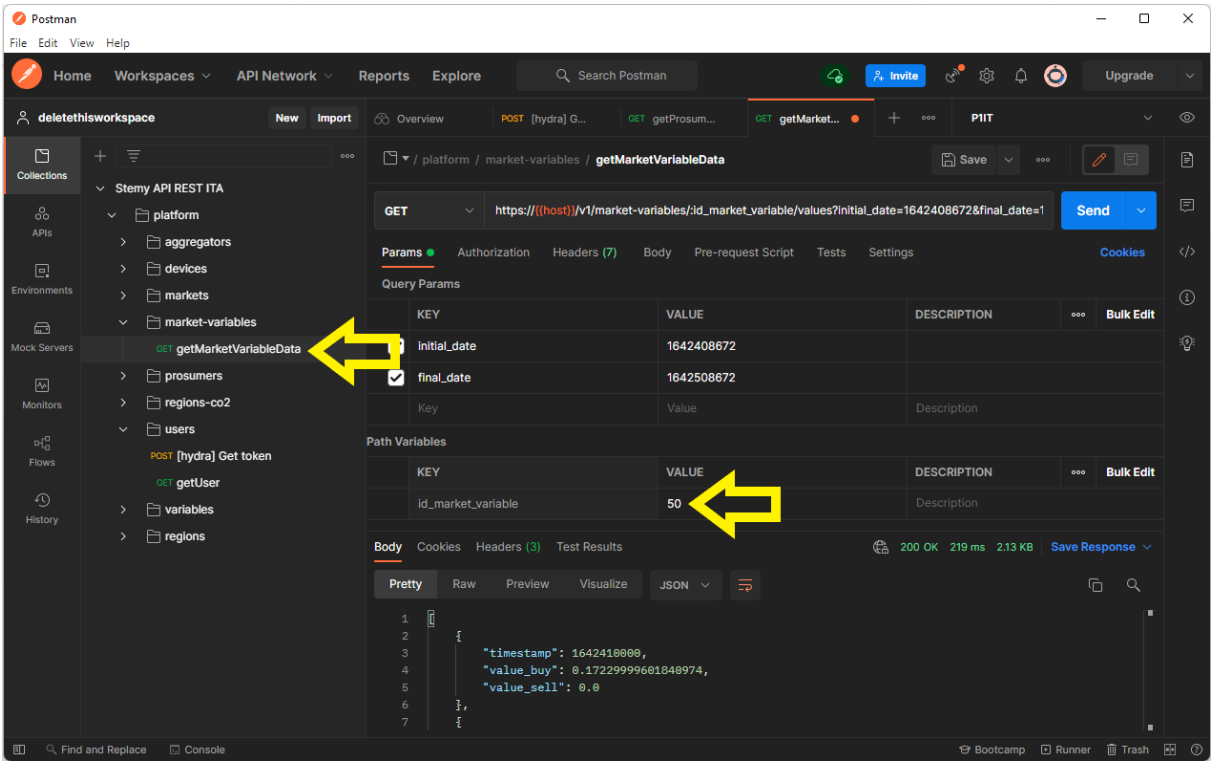


Figure 21: Getting market prices for a prosumer's energy tariff

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

h) Getting prosumer KPIs

Once we know the *prosumerId*, we may navigate to the prosumers/ folder in the left navigation menu. In that folder, you will find several requests related to prosumer KPIs, such as:

- *getProsumerKpisSummary*: general KPIs.
- *getProsumerKpisThermal*: KPIs related to the thermal devices, such as heat pumps, electric radiators, etc.
- *getProsumerKpisEv*: KPIs related to the usage of Electric Vehicles.
- *getProsumersKpisElectric*: KPIs related to electric energy usage.
- *getProsumersKpisFlexibility*: KPIs related to flexibility of said prosumer.
- *getProsumerKpisDess*: KPIs related to distributed energy storage systems.

In all these requests you will have to modify the path variable *prosumer_id.*

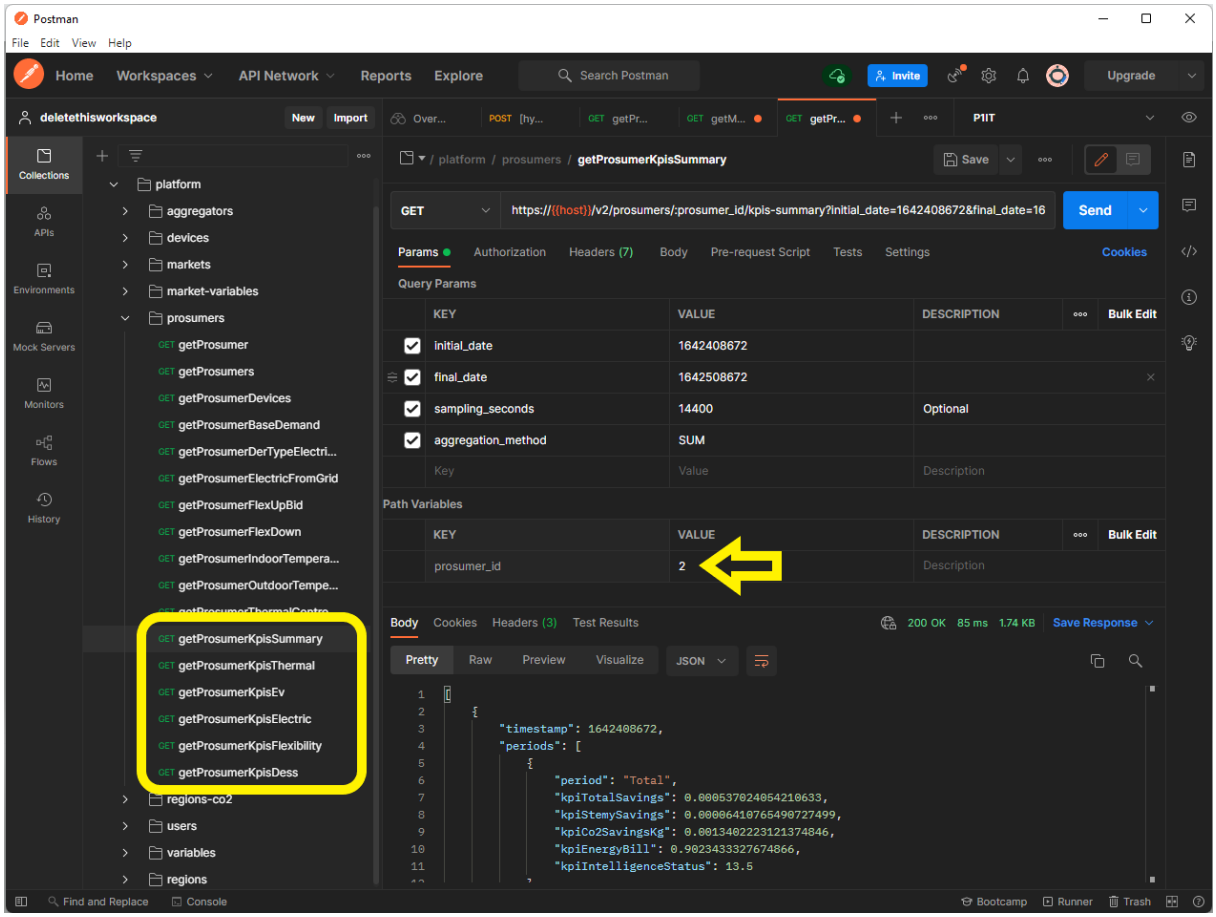


Figure 22: Getting prosumer KPIs

i) Getting events from the Prosumer Advisor service

Once we know the *prosumerId*, we may request data from the prosumer advisor service using the getProsumerEvents request. We will have to specify the following query parameters:

- Initial_date: Initial DateTime in Unix timestamp format
- Final_date: Final DateTime in Unix timestamp format
- Visibility: Type of event visibility. Must be 'Advise' or 'Advise aggregated'

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

The response will be an array of events, each with the message generated by the prosumer advisor service.

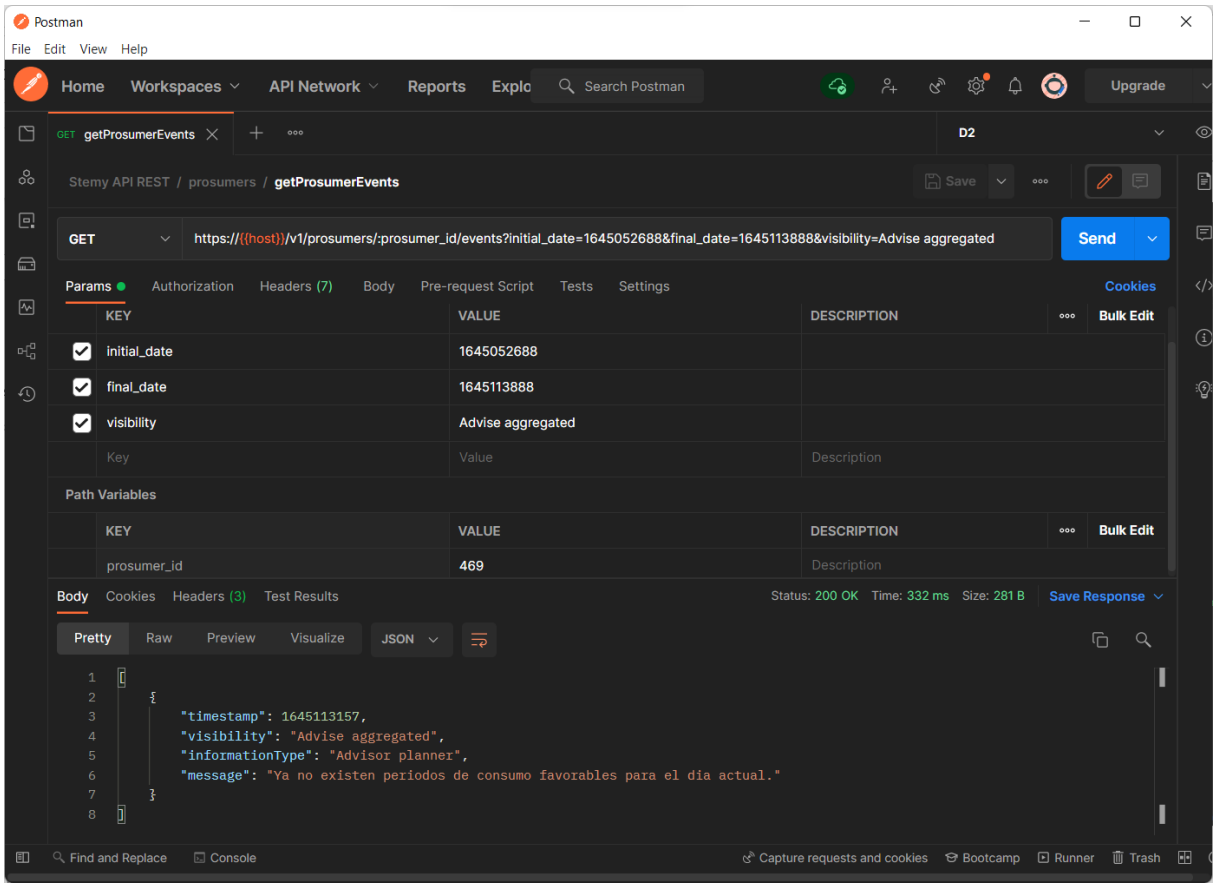


Figure 23: Getting events from the Prosumer Advisor service

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.

Developers can also use the getProsumerPowerMarginPred request, with the following query parameters:

- Initial_date: Initial DateTime in Unix timestamp format
- Final_date: Final DateTime in Unix timestamp format

The response will contain the consumption tags in numeric format, which are documented in the section API Requests: Prosumer Advisor service.

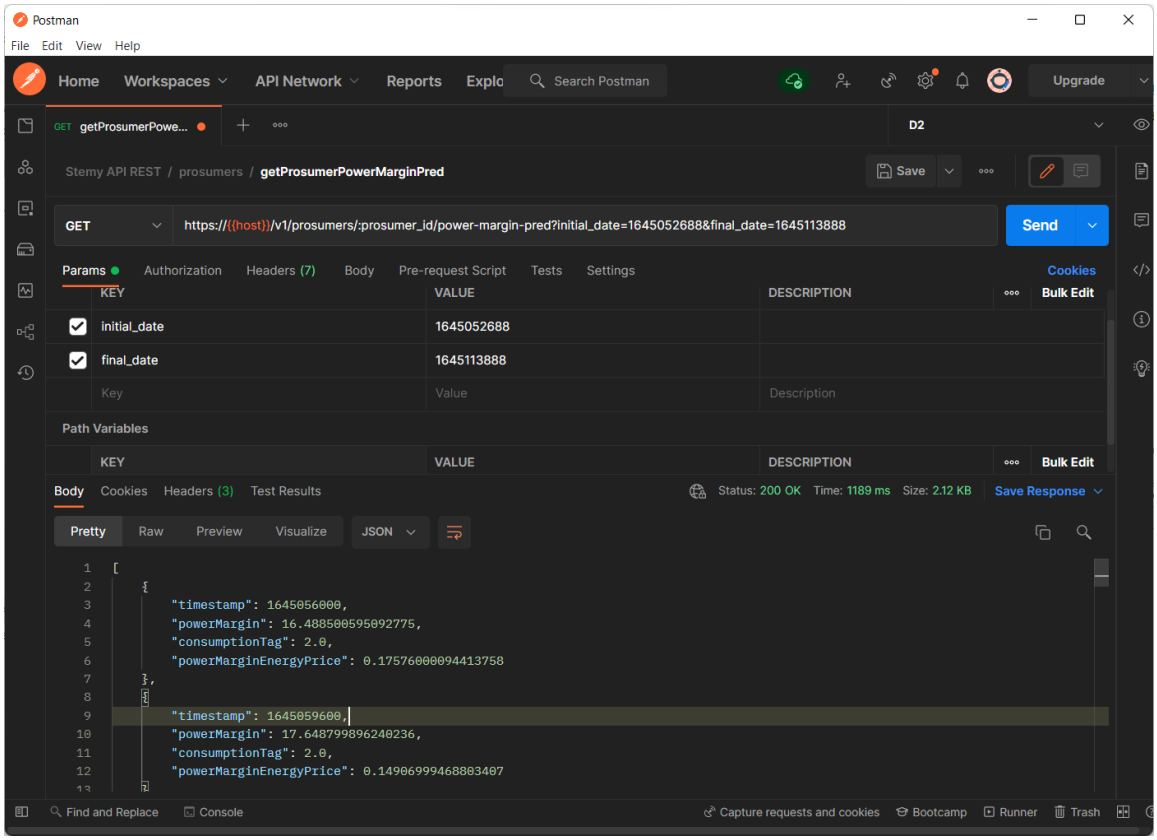


Figure 24: Get prosumer power margin predictions from the Prosumer Advisor service

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957837.