ReDREAM
change your energy

D2.6: Co-creative connection with third parties and devices

June 2022

Main Authors (in alphabetic order):

Carlos Álvarez Vereterra.

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

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, Stephane Galland, Andres Nikoglou	
Other authors/contributors	☑	*STEMY: Carlos Rodriguez, Antonio Hernandez, Jaime Boal, Antonio Vázquez, Sergio Diaz, Alberto Castillo, Ignacio Basallote, Ada Pouseu, Valeria Pascual.*
	☑	*RIMOND : Hamed Abbasi, Andreea Miroslav*
	☑	*Soulsight: Juan Martino*
	☑	*Comillas: Francisco Martín Martinez, Alessandra Porfido, Teresa Freire, Alvaro Sanchez Miralles, Carmen Valor*
	☑	*ENER: Maria Regidor*
	☑	*ZeZ : Lucija Nad, Mislav Kirac*
	☑	*BWCE: Alison Turnbull*
	☑	*BIO : Andrea Ferrante, Giacomo Nardoni*

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.

	☑	*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	*M15 – 20-Dec-21*	
Actual submission date	*M15 (First), M21(Second))*	

[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	Hamed Abbasi (RIMOND), Andreas Nikoglou (NTUA)
Reviewing period	
Approved by reviewers	

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
V1.6	16/06/22	Carlos Álvarez	Added Mobility service API documentation from Deliverable D3.5. Include how credentials can be achieved. Add explanation for third party developers.
V1.7	27/06/33	Badr Ghorbal Ruben Rodríguez Olga Rico.	Add examples of Mobility previously located in D3.5, check Grammar and format.

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).

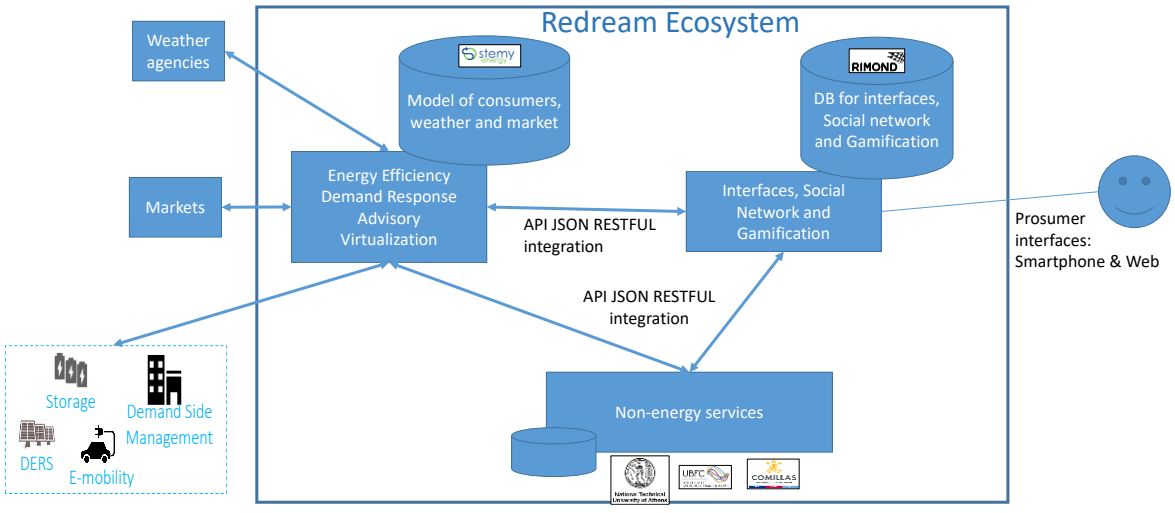


Figure 1: Interoperability of different systems in REDREAM

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.

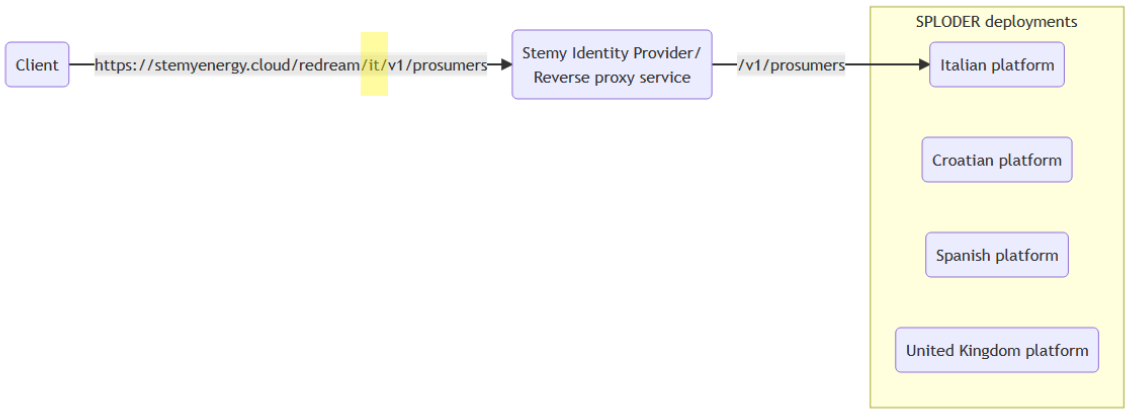


Figure 2: ReDREAM Energy API deployments

- Communications between services will be done using REST APIs.
- There will only be one Identity Provider Service for REDREAM, the one managed by STEMY. This Identity Provider Service will serve all users interacting with User Interfaces, such as web or mobile front-ends.
- 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 to obtain the unique id of the prosumer.
- 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 to identify the prosumer Id. 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 REDREAM API is open to any user, not only ReDREAM partners but credentials must be requested to access it. To request user credentials, please refer to Annex 2.

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)

Disclaimer

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.

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.

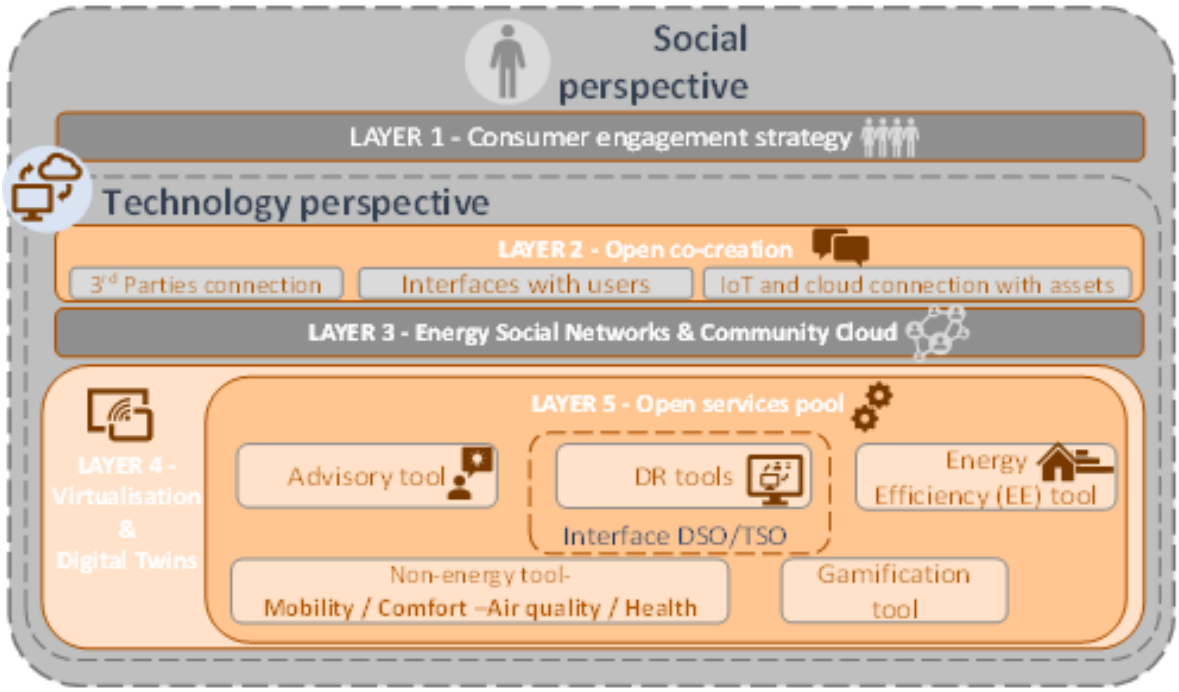


Figure 3: ReDREAM Ecosystem Layers

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 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 documentation of the Energy API will be shared using the OpenAPI 3.0 format and will be accessible to 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.

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 mobility service will be documented thoroughly in Deliverable D3.5.

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 STEMY, 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. This will be used as the REDREAM Proxy and the Energy API will be the one that stores the prosumerID linked with the user account. 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.

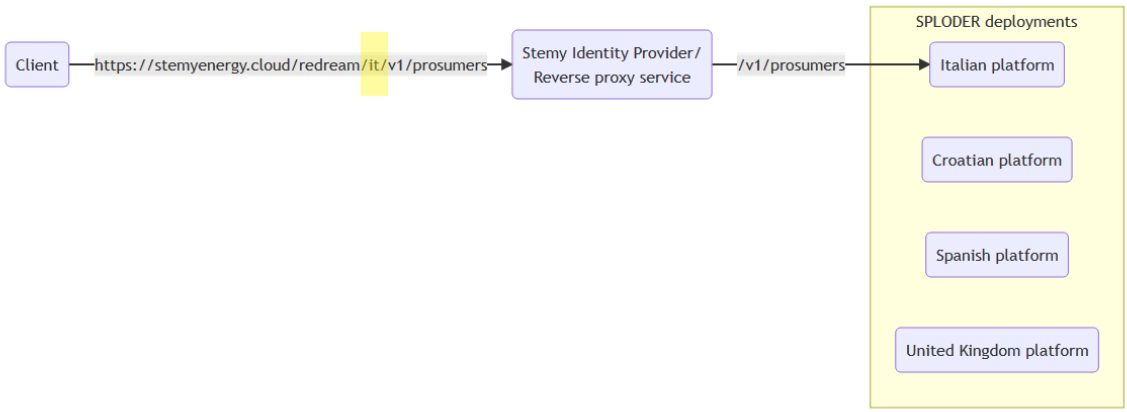


Figure 4: Request to the Italian Energy API deployment

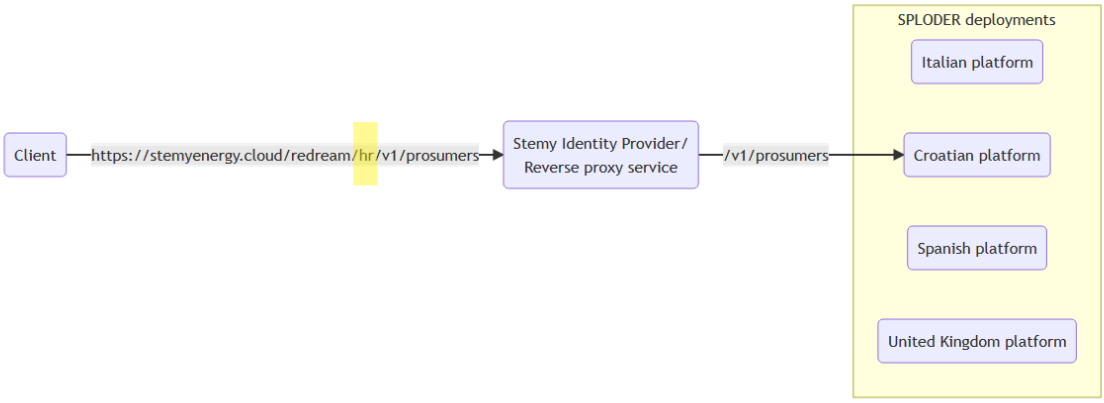


Figure 5: Request to the Croatian Energy API deployment

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 partner 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, 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 D1.6 (Not the final one to avoid disadvantages)

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

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.

- STEMY has the master table of IDs since the users make the initial registration on their proxy that has been established for REDREAM purposes.
- 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 the 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.

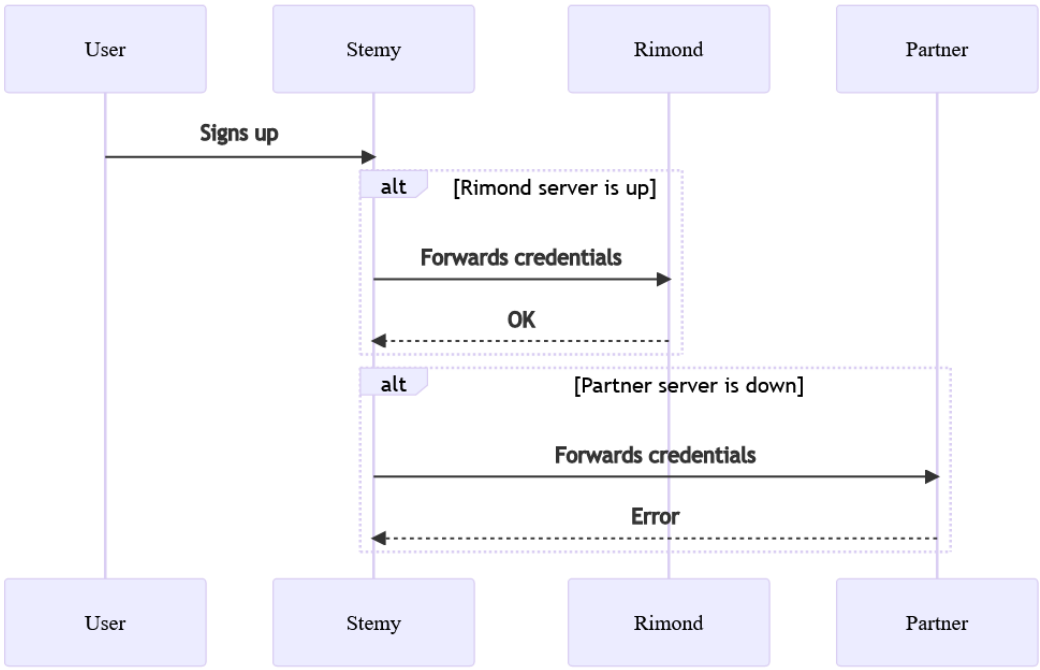


Figure 6: Disadvantages of the proposed approach in Deliverable D1.6

2.2.2. New agreed approach (Final one)

The newly agreed approach greatly simplifies the previous one. Instead of replicating the user credentials multiple times, STEMY will set up a unique identity provider in the project, but ReDREAM partners' services will not be behind STEMY's reverse proxy. This means partners' services will have their own URL.

- New users will sign up using the invitations received by the Demo Leaders.
- Existing users will be able to sign in using the REDREAM Ecosystem.
- Existing users will be able to restore their passwords using the REDREAM 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.

- 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 REDREAM Ecosystem.
    - RIMOND will request an access token with the user credentials to REDREAM's Identity Provider Service managed by STEMY. This eliminates the need for RIMOND to store the user's credentials.
    - RIMOND's interface will use the provided access token to authenticate and obtain the prosumerId and start using the Energy 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 Energy API too, obtain the prosumer Id and continue with their service.

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 REDREAM'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 REDREAM's authentication server. The first step 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 REDREAM 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 REDREAM's Identity Provider server responds with a valid token, that would mean that the credentials entered by the user are correct. If REDREAM''s Identity Provider server responds with an Unauthorized, the credentials are invalid. If this happens, users could be redirected to the REDREAM 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.

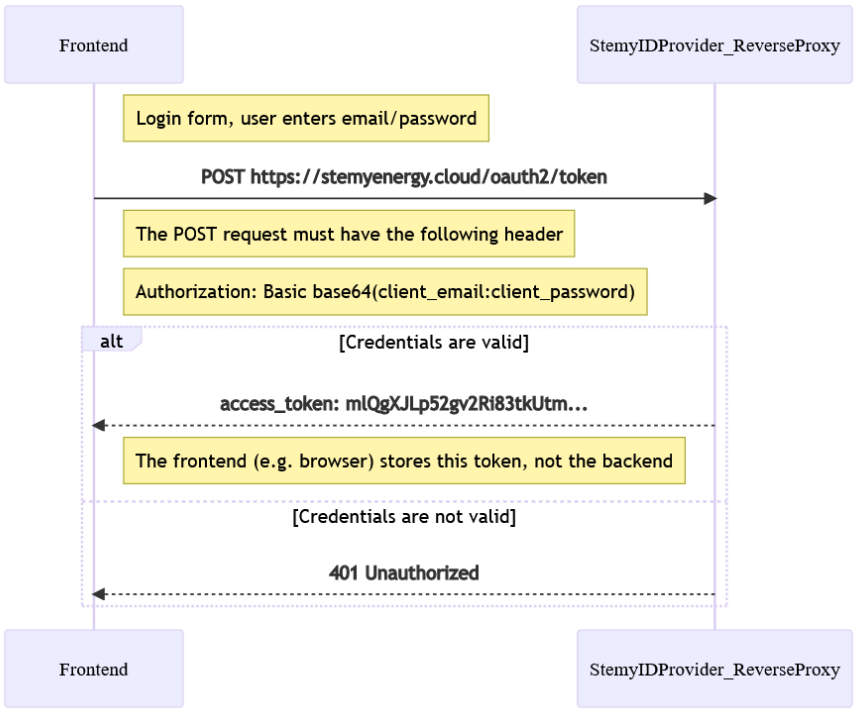


Figure 7: Requesting a valid token to REDREAM's ID Provider

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.BniLlK9oOaWptBUr_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* corresponds 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.

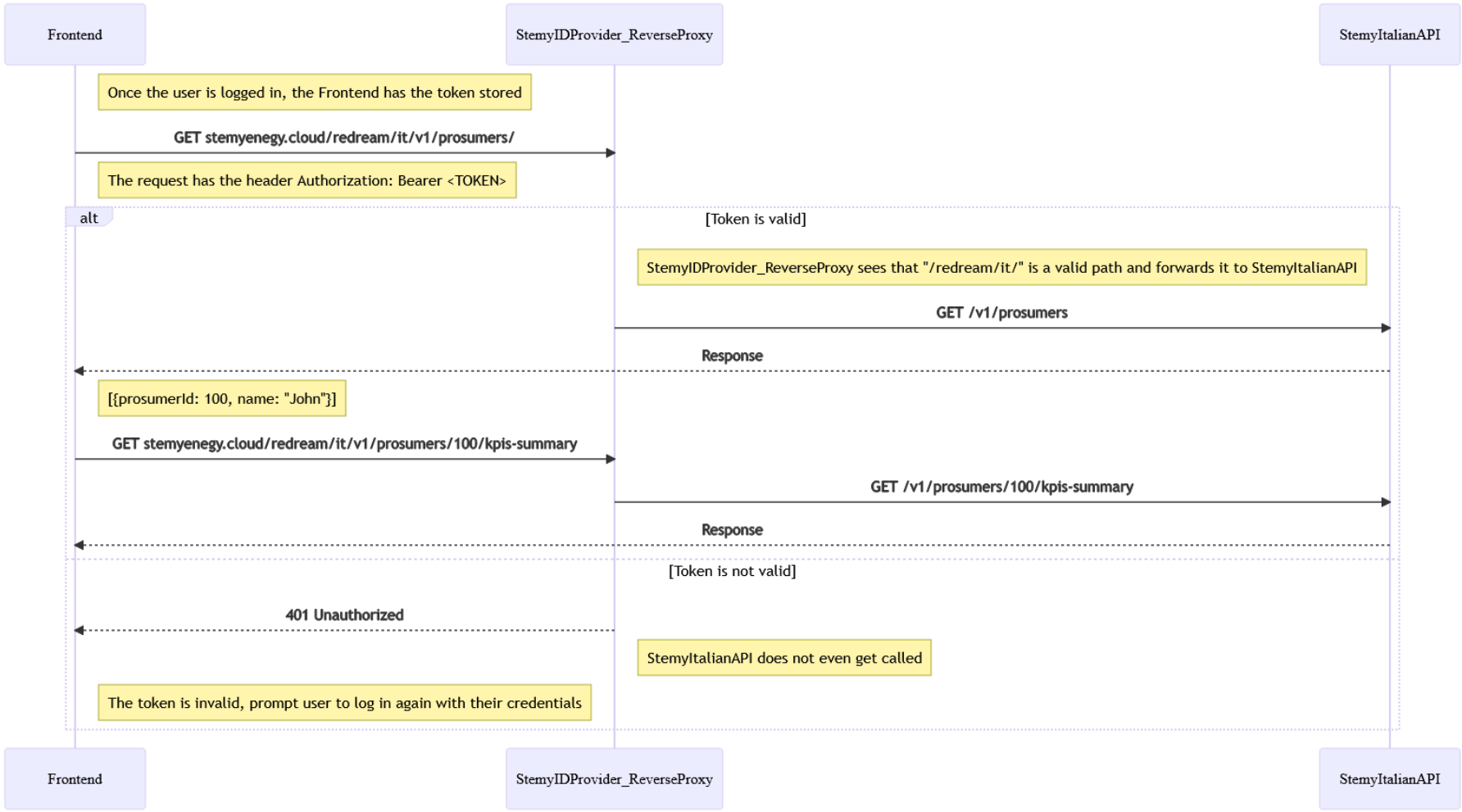


Figure 8: Front-end interacting with the Energy API

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 REDREAM authentication server .
- 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/ or just continue with its service.*
- 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.

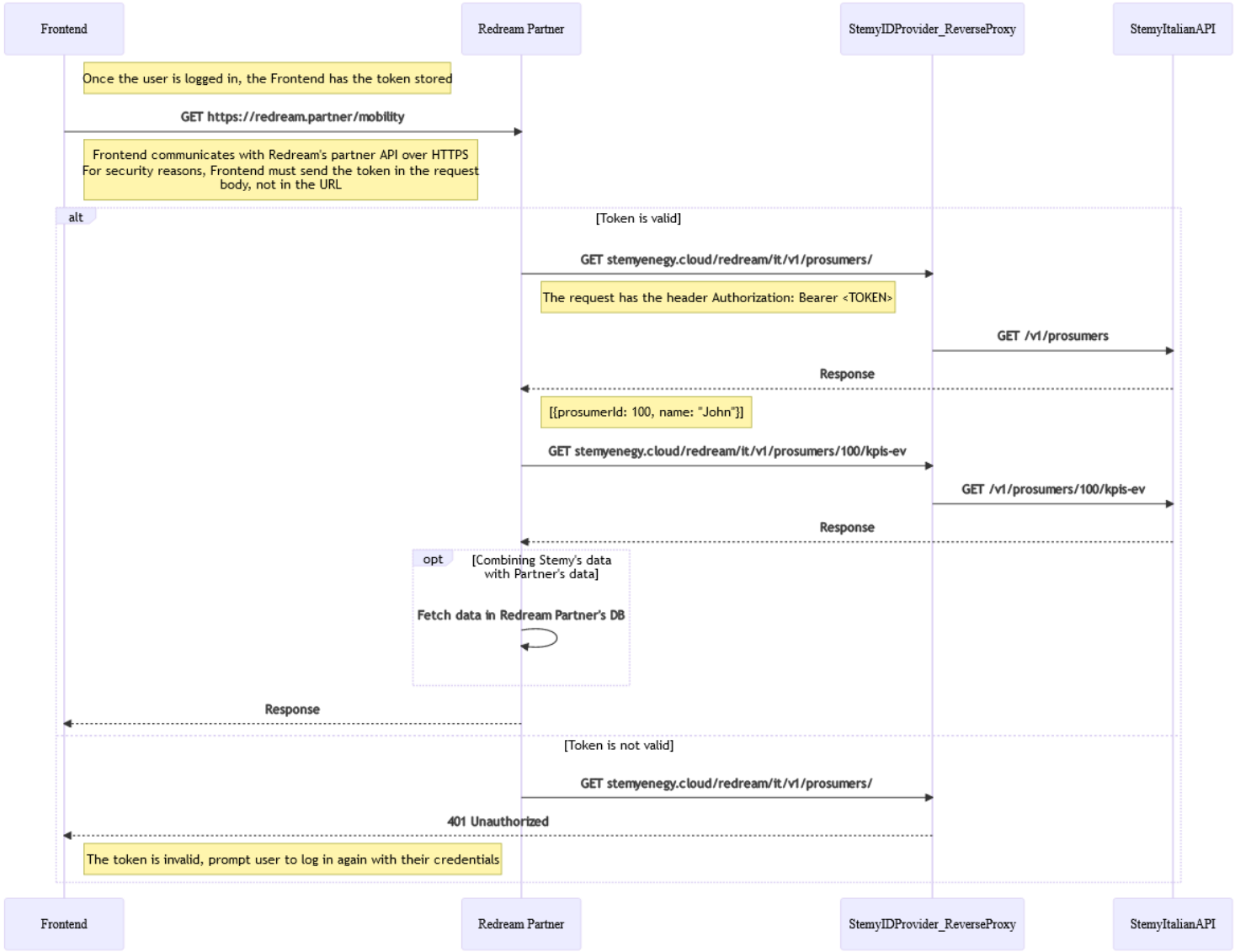


Figure 9: Front-end and one ReDREAM partner interacting with the Energy API

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

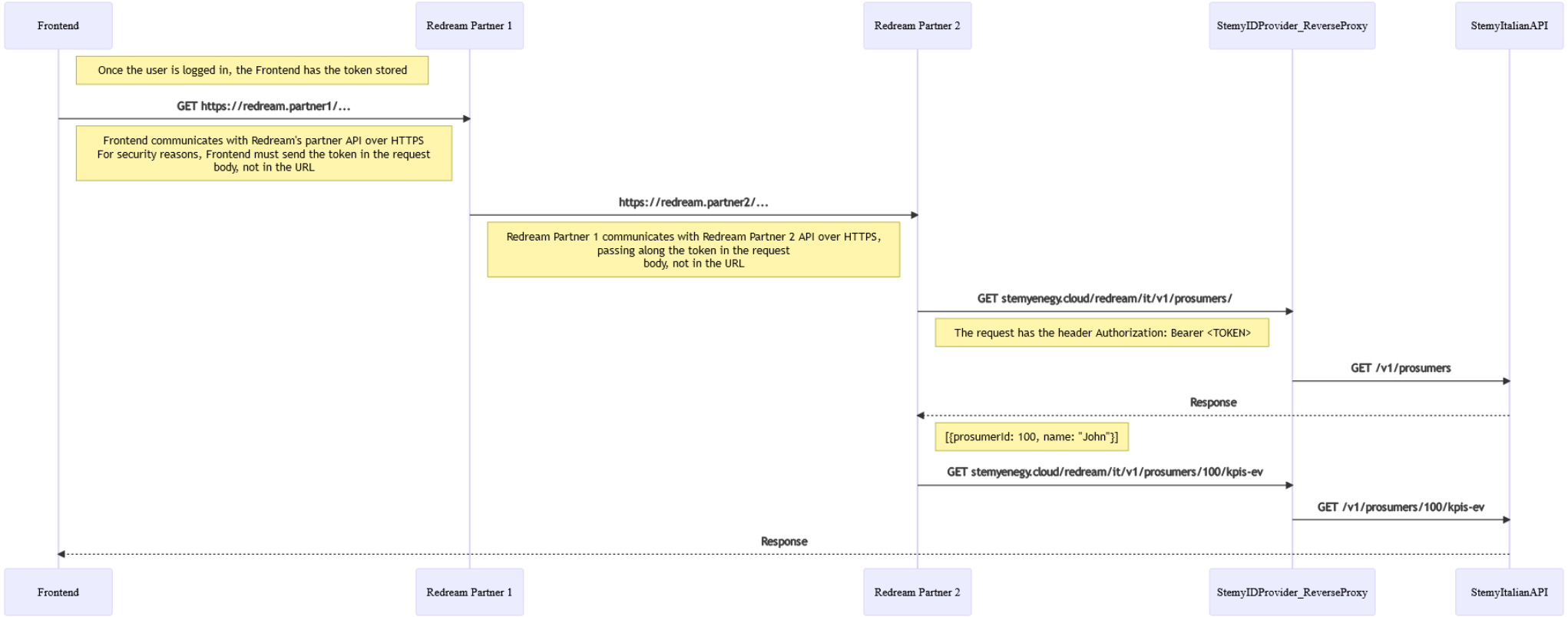


Figure 10: Front-end and multiple partners interacting with the Energy API

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 ReDream 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

You may want to use some of the demo account to test the API: refer to https://redream.energy.

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.

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

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-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
DELETE /locations/{location_id}	Delete 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***
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

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. Consumption warning	Average prices of the day
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

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 /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

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

This section provides a summary of the technical documentation of the mobility service. To enable the test of the mobility API by any demo user, a specific username and its password have been defined. This specific user account enables anyone to test the mobility API without the requirement to be registered in the REDREAM ecosystem. However, this demo user account has limitations regarding the number of queries he could do per day. This limitation will be displayed on the mobility service webpage (http://h2020-redream.utbm.fr). An example of how to Access the mobility service is given in Annex 2.

5.1. Auth-service

5.1.1. Get token

Requests a token to authenticate requests in other services. Returns a JSON containing the auth token to add as Bearer Authentication header. Also returns token information such as delay before token expiration. You may want to use some of the demo account to test the API: refer to https://redream.energy. The request has the format:

POST /api/v2/auth/getToken

Table 18 Parameters of the Get-token request

Parameter	Type	Description	Example
username	String	The username to authenticate the user with	xyz
password	String	The password to authenticate the user with	xyz

Table 18 lists the parameters of the request. Figure 11 provides an example of a response. Table 19 provides the possible errors that could be provided by the microservice to the client.

For any user of the mobility service, it is important to note that:

ALL FOLLOWING ENDPOINTS MUST BE CALLED USING BEARER AUTHENTICATION HEADER WITH THE ACCESS_TOKEN FROM THE PREVIOUS REQUEST AS VALUE.

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.

```
{
   "access_token":
"eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJvMHpRTVBMQU40U0p6YVRtUFJ4bUx0T1VKTjBKWjlYVXl4VjVWS3
h3cXZZIn0.eyJleHAiOjE2MzkzOTMyOTgsImlhdCI6MTYzOTM4OTY5OCwianRpIjoiZjViZWI2MGEtNzhjOC00YzVhLWJkZjktM
zg1NDQ4ZjcyMGE5IiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdDo4ODg5L2F1dGgvcmVhbG1zL3JlZHJlYW0iLCJhdWQiOiJhY2Nv
dW50Iiwic3ViIjoiZDVjNWNhZjAtZjIwOC00MDI4LWIxZDktMDU5NWFhMzg0N2U2IiwidHlwIjoiQmVhcmVyIiwiYXpwIjoicm
VkcmVhbS1hcGkiLCJzZXNzaW9uX3N0YXRlIjoiOTExZGZjZGEtZTRhMC00YmJjLTk4OTEtYjg1Y2I5Njk3MmRkIiwiYWNyIjoiMS
IsInJlYWxtX2FjY2VzcyI6eyJyb2xlcyI6WyJkZWZhdWx0LXJvbGVzLXJlZHJlYW0iLCJFTkVSR1kiLCJWRUhJQ0xFIiwiUk9VVElORyI
sIm9mZmxpbmVfYWNjZXNzIiwidW1hX2F1dGhvcml6YXRpb24iXX0sInJlc291cmNlX2FjY2VzcyI6eyJyb2xlcyI6WyJFTkVSR1kiLCJWRUhJQ0xFIiwiUk9VVElORyJdfSwiYWNjb3VudCI6eyJyb2xlcyI6WyJtYW5hZ2UtYWNjb3VudCI
sIm1hbmFnZS1hY2NvdW50LWxpbmtzIiwidmlldy1wcm9maWxlIl19fSwic2NvcGUiOiJlbWFpbCBwcm9maWxlIiwic2lkIjoiOT
ExZGZjZGEtZTRhMC00YmJjLTk4OTEtYjg1Y2I5Njk3MmRkIiwiZW1haWxfdmVyaWZpZWQiOmZhbHNlLCJwcmVmZXJyZWRf
dXNlcm5hbWUiOiJ0b3RvIn0.jUYiYQU3MNE8QcxuLfAe9xdRQ1U5Sf_GvtYc_5GtfQnpOZ5jGMTTem7lX0bNoJvKhwErH99F8
QPWW2kwiUnFPAP0Pbnanlw8a80tMwQOjdkaAGcGrCKhhVneWeDHlw7pims8ZAHYSfdEwqF-yzrMuJeY3ZAKqIB1n-
EXV4RMWDfKjLZJEeRwE3D75forsWeTRSPME4h7cDbriLTnERMjX30yZcdpUANhp2kDSEgaUma_cjV-
vCafu_gCayaIWzkvND-v0tTvSAhEHa8V-
J8BphECRKUXBICKKqT5qNkjPoyqvJPFVsh_B6HKyNEu6HJpgrJ9VrmM86cLIwYVJYKZlA",
   "expires_in": 3599,
   "refresh_expires_in": 0,
   "token_type": "Bearer",
   "not-before-policy": 1635865960,
   "session_state": "911dfcda-e4a0-4bbc-9891-b85cb96972dd",
   "scope": "email profile"
}
```


Figure 11 Example of response to the Get-token request

Error code	Meaning
400	Bad request: a compulsory parameter is missing or has a wrong type.
401	Unauthorized: authorization header is missing or invalid or expired
500	Error: internal error. A subsequent API is offline, or an internal exception occurs. Retry and check logs if the error is still there.

Table 19 List of errors for the Get-token request

5.2. Energy service

5.2.1. Electricity price

Retrieve the price of the kWh of electricity in Euro at a given location. Returns a double-precision floating-point number. The format of the request is:

GET /api/v2/energy/getKwhUsdCost

Table 20 provides the accepted parameters for the *getKwhUsdCost* request. Table 21 provides the list of errors that could be generated for a *getKwhUsdCost* 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.

Table 20 Parameters of the getKwhUsdCost request

Parameter	Type	Description	Example
lat	Double	The latitude of the location where to get the electricity price from	4.19323
lon	Double	The longitude of the location where to get the electricity price from	34.32425

Table 21 List of errors for getKwhUsdCost request

Error code	Meaning
400	Bad request: a compulsory parameter is missing or has a wrong type.
401	Unauthorized: authorization header is missing or invalid or expired
500	Error: internal error. A subsequent API is offline, or an internal exception occurs. Retry and check logs if the error is still there.

5.2.2. Petrol price

Retrieve the price of a litre of petrol in Euro at a given location. Returns a double-precision floating-point number. The format of the request is:

GET /api/v2/energy/getPetrolLiterEurPrice

Table 22 provides the accepted parameters for the *getPetrolLiterEurPrice* request. Table 23 provides the list of errors that could be generated for a *getPetrolLiterEurPrice* request.

Table 22 Parameters of the getPetrolLiterEurPrice request

Parameter	Type	Description	Example
lat	Double	The latitude of the location where to get the petrol price from	4.19323
lon	Double	The longitude of the location where to get the petrol price from	34.32425

Table 23 List of errors for the getPetrolLiterEurPrice request

Error code	Meaning
400	Bad request: a compulsory parameter is missing or has a wrong type.
401	Unauthorized: authorization header is missing or invalid or expired
500	Error: internal error. A subsequent API is offline, or an internal exception occurs. Retry and check logs if the error is still there.

5.2.3. Diesel price

Retrieve the price of a litre of diesel in Euro at a given location. Returns a double-precision floating-point value. The format of the request is:

GET /api/v2/energy/getDieselLiterEurPrice

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 24 provides the accepted parameters for the *getDieselLiterEurPrice* request. Table 25 provides the list of errors that could be generated for a *getDieselLiterEurPrice* request.

Table 24 Parameters of the getDieselLiterEurPrice request

Parameter	Type	Description	Example
lat	Double	The latitude of the location where to get the diesel price from	4.19323
lon	Double	The longitude of the location where to get the diesel price from	34.32425

Table 25 List of errors for the getDieselLiterEurPrice request

Error code	Meaning
400	Bad request: a compulsory parameter is missing or has a wrong type.
401	Unauthorized: authorization header is missing or invalid or expired
500	Error: internal error. A subsequent API is offline, or an internal exception occurs. Retry and check logs if the error is still there.

5.3. Vehicle service

5.3.1. Get all vehicles

Retrieve all vehicle types in the vehicle database. Returns a list of *Vehicles*. The format of the request is:

GET /api/v2/vehicle/

The request has no accepted parameter.

Figure 12 provides an example of a response to this request. Table 26 provides the list of errors that could be generated for the request.

Table 26 List of errors for the getAllVehicles request

Error code	Meaning
400	Bad request: a compulsory parameter is missing or has a wrong type.
401	Unauthorized: authorization header is missing or invalid or expired
500	Error: internal error. A subsequent API is offline, or an internal exception occurs. Retry and check logs if the error is still there.

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.

```
[
  {
    "id": 1,
    "name": "Tesla Model 3",
    "description": "Tesla Model 3 2021",
    "type": "CAR",
    "energy": "ELECTRICITY",
    "maxCharge": 54.0,
    "freeFlowSpeedTable":
"0,0.239,27,0.239,45,0.259,60,0.196,75,0.207,90,0.238,100,0.26,110,0.296,120,0.337,130,0.351,250,0.351",
    "traficSpeedTable":
"0,0.349,27,0.319,45,0.329,60,0.266,75,0.287,90,0.318,100,0.33,110,0.335,120,0.35,130,0.36,250,0.36",
    "auxiliaryConsumption": 1.8,
    "ascent": 9.0,
    "descent": 4.3,
    "chargingCurve": "0,239,32,199,56,167,60,130,64,111,68,83,72,55,76,33,78,17,80,1",
    "maxChargingVoltage": 400.0,
    "maxChargingCurrent": 600.0,
    "chargingSetupDuration": 300,
    "connectorType": "iec62196Type1Combo,iec62196Type2Combo,Chademo,Tesla",
    "whToLiterFactor": 0.0,
    "literToCO2Factor": 0.0,
    "elsewhereCO2Factor": 0.0
  },
...
  {
    "id": 7,
    "name": "Peugeot 2008",
    "description": "Peugeot 2008",
    "type": "CAR",
    "energy": "DIESEL",
    "maxCharge": 1000000.0,
    "freeFlowSpeedTable":
"0,0.29401,10,0.254172,20,0.221614,30,0.196336,40,0.178338,50,0.16762,60,0.164182,70,0.168024,80,0.179146,90,0.
197548,100,0.22323,110,0.256192,120,0.296434,130,0.343956",
    "traficSpeedTable":
"0,0.29401,10,0.254172,20,0.221614,30,0.196336,40,0.178338,50,0.16762,60,0.164182,70,0.168024,80,0.179146,90,0.
197548,100,0.22323,110,0.256192,120,0.296434,130,0.343956",
    "auxiliaryConsumption": 1.8,
    "ascent": 4.592435873,
    "descent": 0.0,
    "chargingCurve": "0,1,1000000,1",
    "maxChargingVoltage": 400.0,
    "maxChargingCurrent": 600.0,
    "chargingSetupDuration": 300,
    "connectorType": "iec62196Type1Combo,iec62196Type2Combo,Chademo,Tesla",
    "whToLiterFactor": 2.84972114285714E-4,
    "literToCO2Factor": 2.64,
    "elsewhereCO2Factor": 0.66
  }
]
```


Figure 12 Example of response for the getAllVehicles request

5.3.2. Get a vehicle by id

Retrieve a vehicle in the vehicle database by its id. Returns a *Vehicle*. The format of the request is:

GET /api/v2/vehicle/{id}

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 27 provides the accepted parameters for the *getVehicleById* request. Table 28 provides the list of errors that could be generated for a *getVehicleById* request.

Figure 13 provides an example of a response to the request.

Parameter	Type	Description	Example
In path: id	Int	The id of the vehicle	1

Table 27 Parameters of the getVehicleById request

Error code	Meaning
400	Bad request: a compulsory parameter is missing or has a wrong type.
401	Unauthorized: authorization header is missing or invalid or expired
404	NotFound: no vehicle with this id was found
500	Error: internal error. A subsequent API is offline, or an internal exception occurs. Retry and check logs if the error is still there.

Table 28 List of errors for the getVehicleById request


```
{
   "id": 1,
   "name": "Tesla Model 3",
   "description": "Tesla Model 3 2021",
   "type": "CAR",
   "energy": "ELECTRICITY",
   "maxCharge": 54.0,
   "freeFlowSpeedTable":
"0,0.239,27,0.239,45,0.259,60,0.196,75,0.207,90,0.238,100,0.26,110,0.296,120,0.337,130,0.351,250,0.351",
   "traficSpeedTable":
"0,0.349,27,0.319,45,0.329,60,0.266,75,0.287,90,0.318,100,0.33,110,0.335,120,0.35,130,0.36,250,0.36",
   "auxiliaryConsumption": 1.8,
   "ascent": 9.0,
   "descent": 4.3,
   "chargingCurve": "0,239,32,199,56,167,60,130,64,111,68,83,72,55,76,33,78,17,80,1",
   "maxChargingVoltage": 400.0,
   "maxChargingCurrent": 600.0,
   "chargingSetupDuration": 300,
   "connectorType": "iec62196Type1Combo,iec62196Type2Combo,Chademo,Tesla",
   "whToLiterFactor": 0.0,
   "literToCO2Factor": 0.0,
   "elsewhereCO2Factor": 0.0
}
```


Figure 13 Example of response for the *getVehicleById* 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.

5.4. Routing service

5.4.1. Get electric vehicle route

Computes a route between two points using the HERE Routing API, as described in Deliverable D3.4. This endpoint must be used for **electric vehicles routing only**. Returns a list of possible routes ordered by best choice. The format of the request is:

POST /api/v2/routing/getEvRoute

Table 29 provides the accepted parameters for the *getEvRoute* request. *Table 30* provides the list of errors that could be generated for a *getEvRoute* request.

Figure 14 provides an example of a response to the request.

Table 29 Parameters of the getEvRoute request

Parameter	Type	Description	Example
vehicleId	Integer	The vehicle id to use for this request	4
originLat	Double	The latitude of the location to start from	34.32425
originLon	Double	The longitude of the location to start from	6.4434
destinationLat	Double	The latitude of the location to end to	35.32425
destinationLon	Double	The longitude of the location to end to	7.4434
departureTime	Iso datetime (optional)	Specifies the time of departure as defined by either date-time or full-date T partial-time in RFC 3339 (for example, 2019-06-24T01:23:45). The requested time is converted to local time at origin. When the optional time zone offset is not specified, time is assumed to be local. The special value any can be used to indicate that time should not be considered during routing. If neither departureTime nor arrivalTime is specified, the current time at departure place will be used. All time values in the response are returned in the timezone of each location.	2021-01-13T06:45:01
initialCharge	Double (optional)	Charge level of the vehicle's battery at the start of the route (in kWh). Value must be less than or equal to the value of maxCharge.	30.1
minChargeStation	Double (optional)	Minimum charge when arriving at a charging station (in kWh). The value must be less than the value of maxChargeAfterChargingStation. The algorithm calculates a route as the best possible combination of driving and charging parts so visiting a charging station is planned not when the remaining charge is close to the value of this parameter but	5.5

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.

		when it is part of the best possible charging plan for the given route. For example, it might prefer charging a still half-full battery at the fast-charging station because there are only slower stations after on the route and the remaining charge is not enough to reach the destination without charging at all.	
maxChargeStation	Double (optional)	Maximum charge to which the battery should be charged at a charging station (in kWh). The value must be less than or equal to the value of maxCharge. The algorithm calculates a route as the best possible combination of driving and charging parts so charging at a charging station does not happen strictly to the value of this parameter. Instead, it attempts to leave every station with different charge levels, and only the best possible combination of charging stations and target charge is going to form the final route. For example, if there is a fast but not reachable charging station on the route, the algorithm prefers first to charge at a slower station, but only to a level that enables it to reach the fast station. This way it calculates the best possible combination of driving and charging parts.	25.5
minChargeDestination	Double (optional)	Minimum charge at the final route destination (in kWh). The value must be less than the value of maxChargeAfterChargingStation. The algorithm calculates a route as the best possible combination of driving and charging parts while making sure that the actual value of the charge at the destination would be close to the value of this parameter. I.e., the resulting value is expected to be bigger than this parameter's value by no more than 10% of the battery capacity.	10.1

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.

```
[
  {
    "id": "caf52289-cac6-44ac-a259-2f2a55c477f4",
    "vehicleId": 2,
    "vehicleName": "Tesla Model Y",
    "vehicleDescription": "Tesla Model Y 2021",
    "consumptionInWattsHours": 96242.8,
    "lengthInMeters": 243197.0,
    "durationInSeconds": 11859.0,
    "co2inKg": 0.0,
    "priceInEuros": 16.361276,
    "sections": [
      {
        "id": "45c65e99-93e8-4aaa-9f96-6a202b8836dd",
...
```


Figure 14 Example of response for the *getEvRoute* request

Table 30 List of errors for the getEvRoute request

Error code	Meaning
400	Bad request: a compulsory parameter is missing or has a wrong type. The given vehicle isn't an electric vehicle.
401	Unauthorized: authorization header is missing or invalid or expired
500	Error: internal error. A subsequent API is offline, or an internal exception occurs. Retry and check logs if the error is still there.

5.4.2. Get fuel vehicle route

Computes a route between two points using the HERE Routing API. This endpoint must be used for **fuel vehicles routing only**. Returns a list of possible routes ordered by best choice. The format of the request is:

POST /api/v2/routing/getRoute

Table 31 provides the accepted parameters for the *getRoute* request. Table 32 provides the list of errors that could be generated for a *getRoute* request.

Figure 15 provides an example of a response to the request.

Table 31 Parameters of the getRoute request

Parameter	Type	Description	Example
vehicleId	Integer	The vehicle id to use for this request	4
originLat	Double	The latitude of the location to start from	34.32425
originLon	Double	The longitude of the location to start from	6.4434
destinationLat	Double	The latitude of the location to end to	35.32425
destinationLon	Double	The longitude of the location to end to	7.4434
departureTime	Iso datetime (optional)	Specifies the time of departure as defined by either date-time or full-date T partial-time in RFC 3339 (for example, 2019-06-24T01:23:45).	2021-01-13T06:45:01

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 requested time is converted to local time at origin. When the optional timezone offset is not specified, time is assumed to be local. The special value any can be used to indicate that time should not be considered during routing. If neither departureTime nor arrivalTime is specified, the current time at departure place will be used. All time values in the response are returned in the timezone of each location.	


```
[
  {
    "id": "caf52289-cac6-44ac-a259-2f2a55c477f4",
    "vehicleId": 2,
    "vehicleName": "306",
    "vehicleDescription": "--",
    "consumptionInWattsHours": 96242.8,
    "lengthInMeters": 243197.0,
    "durationInSeconds": 11859.0,
    "co2inKg": 1564.0,
    "priceInEuros": 16.361276,
    "sections": [
      {
        "id": "45c65e99-93e8-4aaa-9f96-6a202b8836dd",
...
```


Figure 15 Example of response for the getRoute request

Table 32 List of errors for the getRoute request

Error code	Meaning
400	Bad request: a compulsory parameter is missing or has a wrong type. The given vehicle is an electric vehicle.
401	Unauthorized: authorization header is missing or invalid or expired
500	Error: internal error. A subsequent API is offline, or an internal exception occurs. Retry and check logs if the error is still there.

5.4.3. Get transit route

Computes a route between two points using the HERE Routing API. This endpoint must be used for **transit routing only**. Returns a list of possible routes ordered by best choice. The format of the request is:

POST /api/v2/routing/getTransitRoute

Table 33 provides the accepted parameters for the *getTransitRoute* request. Table 34 provides the list of errors that could be generated for a *getTransitRoute* request.

Figure 16 provides an example of a response to the 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.

Parameter	Type	Description	Example
originLat	Double	The latitude of the location to start from	34.32425
originLon	Double	The longitude of the location to start from	6.4434
destinationLat	Double	The latitude of the location to end to	35.32425
destinationLon	Double	The longitude of the location to end to	7.4434
departureTime	Iso datetime (optional)	Specifies the time of departure as defined by either date-time or full-date T partial-time in RFC 3339 (for example, 2019-06-24T01:23:45). The requested time is converted to the local time at origin. When the optional timezone offset is not specified, time is assumed to be local. If neither departureTime nor arrivalTime is specified, the current time at departure location will be used. All Time values in the response are returned in the time zone of each location.	2021-01-13T06:45:01
arrivalTime	Iso datetime (optional)	Specifies the time of arrival as defined by either date-time or full-date T partial-time in RFC 3339 (for example, 2019-06-24T01:23:45). The requested time is converted to the local time at destination. When the optional time zone offset is not specified, time is assumed to be local. All Time values in the response are returned in the time zone of each location.	2021-01-13T06:45:01

Table 33 Parameters of the getTransitRoute request


```
[
  {
    "id": "R00502f-C0",
    "vehicleId": null,
    "vehicleName": "Public transports",
    "vehicleDescription": "Public transports",
    "consumptionInWattsHours": 16851.25,
    "lengthInMeters": 61782.0,
    "durationInSeconds": 4080.0,
    "co2inKg": 12.677634855599997,
    "priceInEuros": 30.891,
    "sections": [
      {
        "id": "R00502f-C0-S0",
        "type": "pedestrian",
...
```


Figure 16 Example of response to a getTransitRoute request

Table 34 List of errors for the getTransitRoute request

Error code	Meaning

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.

400	Bad request: a compulsory parameter is missing or has a wrong type. The given vehicle isn't an electric vehicle.
401	Unauthorized: authorization header is missing or invalid or expired
500	Error: internal error. A subsequent API is offline, or an internal exception occurs. Retry and check logs if the error is still there.

5.4.4. Get route with all trip solutions

Computes a route between two points using the HERE Routing API. The computing is done with 4 different types of vehicles: an electric vehicle, a petrol vehicle, a diesel vehicle, and public transport. Returns a list of routes. The format of the request is:

POST /api/v2/routing/getRouteWithAllTravelSolutions

Table 35 provides the accepted parameters for the *getRouteWithAllTravelSolutions* request. Table 36 provides the list of errors that could be generated for a *getRouteWithAllTravelSolutions* request.

Figure 16 provides an example of a response to the 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.

Table 35 Parameters of the getRouteWithAllTravelSolutions request

Parameter	Type	Description	Example
originLat	Double	The latitude of the location to start from	34.32425
originLon	Double	The longitude of the location to start from	6.4434
destinationLat	Double	The latitude of the location to end to	35.32425
destinationLon	Double	The longitude of the location to end to	7.4434


```
[
  {
    "id": "R00502f-C0",
    "vehicleId": null,
    "vehicleName": "Public transports",
    "vehicleDescription": "Public transports",
    "consumptionInWattsHours": 16851.25,
    "lengthInMeters": 61782.0,
    "durationInSeconds": 4080.0,
    "co2inKg": 12.677634855599997,
    "priceInEuros": 30.891,
    "sections": [
      {
        "id": "R00502f-C0-S0",
        "type": "pedestrian",
...
```


Figure 17 Example of response to a getRouteWithAllTravelSolutions request

Table 36 List of errors for a getRouteWithAllTravelSolutions request

Error code	Meaning
400	Bad request: a compulsory parameter is missing or has a wrong type.
401	Unauthorized: authorization header is missing or invalid or expired
500	Error: internal error. A subsequent API is offline, or an internal exception occurs. Retry and check logs if the error is still there.

5.4.5. Get the best vehicle for route

Computes a route between two points using the HERE Routing API. Then, use the criteria to find the "best" vehicle type for this route. Returns a route. The format of the request is:

POST /api/v2/routing/getBestVehicleForRoute

Table 37 provides the accepted parameters for the *getBestVehicleForRoute* request. Table 38 provides the list of errors that could be generated for a *getBestVehicleForRoute* request. Figure 18 provides an example of a response to the request.

Table 37 Parameters of the getBestVehicleForRoute request

Parameter	Type	Description	Example

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.

originLat	Double	The latitude of the location to start from	34.32425
originLon	Double	The longitude of the location to start from	6.4434
destinationLat	Double	The latitude of the location to end to	35.32425
destinationLon	Double	The longitude of the location to end to	7.4434
orderCriteria	OrderCriteria	The order criteria among this list: CO2, TIME, DISTANCE, PRICE, CONSUMPTION	TIME


```
[
  {
    "id": "R00502f-C0",
    "vehicleId": null,
    "vehicleName": "Public transports",
    "vehicleDescription": "Public transports",
    "consumptionInWattsHours": 16851.25,
    "lengthInMeters": 61782.0,
    "durationInSeconds": 4080.0,
    "co2inKg": 12.677634855599997,
    "priceInEuros": 30.891,
    "sections": [
      {
        "id": "R00502f-C0-S0",
        "type": "pedestrian",
...
```


Figure 18 Example of response to a getBestVehicleForRoute request

Table 38 List of errors for the getBestVehicleForRoute request

Error code	Meaning
400	Bad request: a compulsory parameter is missing or has a wrong type.
401	Unauthorized: authorization header is missing or invalid or expired
500	Error: internal error. A subsequent API is offline, or an internal exception occurs. Retry and check logs if the error is still there.

5.5. Json Descriptions

This section contains a brief description of the Json descriptions that are part of the responses provided by the different microservices.

5.5.1. Vehicle

Table 39 defines the Json entry for a vehicle. A vehicle entry describes a category of vehicle, also known as type of vehicle, and all the key attributes for the vehicles in the category.

Table 39 Content of a Json entry describing a vehicle

Field	Type	Description
id	Integer	The vehicle ID. Needed for routing requests.
name	String	The vehicle name.

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.

description	String	The vehicle description
type	String	The vehicle type (CAR, TRUCK, BUS, …)
energy	String	The vehicle energy (HUMAN_POWERED, ELECTRICITY, HYBRID, HYDROGEN, DIESEL, PETROL)
maxCharge	Double	Total capacity of the vehicle's battery (in kWh).
freeFlowSpeedTable	String	Function curve specifying consumption rate at a given speed. The format of the string is a comma-separated list of numbers, as follows: <SPEED_0>,<CONSUMPTION_0>,<SPEED_1>,<CONSUMPTION_1>,...,<SPEED_N>,<CONSUMPTION_N> where speed values are strictly increasing, non-negative integers in units of (km/h), and consumption values are non-negative floating-point values in units of (Wh/m). The function is linearly interpolated between data points. For speeds less than SPEED_0 the value of the function is CONSUMPTION_0, and for speeds greater than SPEED_N the value of the function is CONSUMPTION_N.
traficSpeedTable	String	Function curve specifying consumption rate at a given traffic-reduced speed on a flat stretch of road. See freeFlowSpeedTable for a description of the string format.
auxiliaryConsumption	Double	Rate of energy (in Wh/s) consumed by the vehicle's auxiliary systems (e.g., air conditioning, lights). The value represents the number of Watt-hours consumed per second of trip.
ascent	Double	Rate of energy consumed per meter rise in elevation (in Wh/m, i.e., Watt-hours per meter).
descent	Double	Rate of energy recovered per meter fall in elevation (in Wh/m, i.e., Watt-hours per meter).
chargingCurve	String	Function curve describing the maximum battery charging rate (in kW) at a given charge level (in kWh). The format of the string is a comma-separated list of numbers, as follows: <CHARGE_0>,<RATE_0>,<CHARGE_1>,<RATE_1>,...,<RATE_N>,<CHARGE_N> where charge values are strictly increasing, non-negative floating-point values in units of (kWh), and rate values are positive floating-point values in units of (kW). Charge values must cover the entire range of [0, maxChargeAfterChargingStation]. The charging curve is piecewise constant, e.g., for any charge in the range [CHARGE_0, CHARGE_1), the value of the function is RATE_0. The algorithm calculates a route as the best possible combination of driving and charging parts and uses the charging curve to evaluate the most efficient range of charging. For example, if the rate of charging is high at lower levels of battery, but slows down significantly after charging a little, stopping often and charging less, but quicker, at each station might be better for the overall route. Because batteries lose charging speed with use, providing a charging curve for the exact battery would give a more accurate charging time estimate than providing a generic curve for all batteries of one type.
maxChargingVoltage	Double	Maximum charging voltage supported by the vehicle's battery (in Volt).

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.

chargingSetupDuration	Integer	Time spent after arriving at a charging station, but before actually charging, e.g., time spent for payment processing (in seconds).
connectorType	String	Comma-separated list of connector types that are compatible with the vehicle. If makeReachable is set to true, then only stations with any of these connector types will be evaluated as a potential charging stop. For stations with multiple compatible connectors, the charging time is based on the connector type with the highest power rating among them. Currently supported connector types are: iec62196Type1Combo: Type 1 Combo connector, commonly called "SAE J1772" iec62196Type2Combo: Type 2 Combo connector, commonly called "Mennekes" chademo: CHAdeMO connector tesla: Tesla connector
whToLiterFactor	Double	Factor that is used for converting values expressed in watts/h to an equivalent value expressed in litres
literToCO2Factor	Double	Factor that is used for converting values expressed in litres to an equivalent value expressed in kg of emitted CO2 gas
elsewhereCO2Factor	Double	Default factor that represents the default CO2 gas emission for the vehicle

5.5.2. Route

Table 40 provides the attributes of a route. A route contains all the attributes that are the result of a trip for a specific vehicle.

Table 40 Content of a Json entry describing a route

Field	Type	Description
id	String	Unique identifier of the route.
vehicleId	Integer	The vehicle Id used for this route computation. May be null if public transport.
vehicleName	String	The vehicle name used for this route computation. May be null if public transport.
vehicleDescription	String	The vehicle description used for this route computation. May be null if public transport.
consumptionInWattsHours	Double	The consumption for this whole route in Wh
lengthInMeters	Double	The length of this whole route in meters
durationInSeconds	Double	The duration of this whole route in seconds
co2inKg	Double	The CO2 released for this whole route in Kg
priceInEuros	Double	The price for this whole route in euros
sections	List of sections	An ordered list of vehicle, transit, and pedestrian sections making up the route.

5.5.3. Section

Table 41 provides the attributes of a section. A section describes a sequence of road segments that are traversed by a vehicle.

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 41 Content of a Json entry describing a section

Field	Type	Description
id	String	Unique identifier of the section.
type	String	Section type used by the client to identify what extensions to the BaseSection are available. pedestrian, vehicle or transit.
actions	List of Action	Currently **unused**
postActions	List of post-actions	Actions that must be done prior to departure.
departure	Departure	The section departure
arrival	Arrival	The section arrival.
summary	Summary	Total value of key attributes, e.g., duration or length, summed up for the entire section, including preActions, postActions, and the trip portion of the section.
travelSummary	Summary	Total value of key attributes, e.g., duration or length, summed up for just the trip portion of the section, between departure and arrival. preActions and postActions are excluded.
polyline	String	The polyline of this section
spans	List of Span	Spans attached to a Section describing content.
language	String	Language of the localized strings in the section, if any, in BCP47 format.
transport	Transport	Information about the transport

5.5.4. PostAction

Table 42 provides the attributes of a post-action. A post-action describes a possible action on an ECP that is done on the vehicle before starting a trip, or after ending a trip.

Table 42 Content of a Json entry describing a post-action

Field	Type	Description
action	String	The type of action. Can be charging, chargingSetup, …
duration	Integer	Estimated duration of this action (in seconds). Actions last until the next action, or the end of the route in case of the last one.
consumablePower	Double	Quantity of available power.
arrivalCharge	Double	The arrival charge in kWh (Only specified if the action is charging)
targetCharge	Double	The target charge in kWh (only specified if the action is charging)

5.5.5. Departure

Table 43 provides the attributes of a departure. A departure describes the time and location of a trip's departure.

Table 43 Content of a Json entry describing a departure

Field	Type	Description

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.

Time	String	Expected time of departure of the event. The format is RFC 3339
Place	Place	The place
Charge	Double	The charge in kWh at this place (for EV routing only)

5.5.6. Arrival

Table 44 provides the attributes of an arrival. An arrival describes the time and location of a trip's arrival.

Table 44 Content of a Json entry describing an arrival

Field	Type	Description
time	String	Expected time of arrival of the event. The format is RFC 3339
place	Place	The place
charge	Double	The charge in kWh at this place (for EV routing only)

5.5.7. Place

Table 45 provides the attributes of a place. A place describes a location on the map that has specific features for vehicles, e.g., ECP.

Table 45 Content of a Json entry describing a place

Field	Type	Description
id	String	The place unique identifier
type	String	Place type. Each place type can have extra attributes. Can be chargingStation, place, …
location	Location	The position of this location. This position was used in route calculation. It may be different to the original position provided in the request.
originalLocation	OriginalLocation	If present, the original position of this place provided in the request.
name	String	Place name
attributes	Attributes	Extra attributes of this place
brand	Brand	The brand of this place, if any

5.5.8. Location

Table 46 provides the attributes of a location. A location describes a point on a map, expressed with GPS coordinates.

Table 46 Content of a Json entry describing a location

Field	Type	Description
lat	Double	Location of a point on the Earth north or south of the equator in decimal degrees. WGS84 format.
lon	Double	Location of a place on the Earth east or west of the prime meridian in decimal degrees. WGS84 format.

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.

elv	Double	Ellipsoid (geodetic) height in meters. Difference between the WGS84 ellipsoid and a point on the Earth's surface. Note: Similar elevation can be obtained from a GPS receiver.

5.5.9. Original Location

Table 47 provides the attributes of an original location. An original location describes a point on a map that is the beginning of a trip.

Table 47 Content of a Json entry describing an original location

Field	Type	Description
lat	Double	Location of a point on the Earth north or south of the equator in decimal degrees. WGS84 format.
lon	Double	Location of a place on the Earth east or west of the prime meridian in decimal degrees. WGS84 format.

5.5.10. ECP

Table 48 provides the attributes for an ECP. The ECP attributes include the voltage, the max available power, etc.

Table 48 Content of a Json entry describing ECP

Field	Type	Description
power	Double	The max power of this charging station in Kw
current	Double	The max current of this charging station in Amperes
voltage	Double	The nominal voltage of this charging station in Volts
supplyType	String	The type of energy supply. Can be dc for continuous current of ac for alternating current.
connectorType	String	The type of connector at this charging station. Ex: tesla, iec62196Type2Combo, …

5.5.11. Brand

Table 49 provides the attributes of a brand. A brand is defined in a general way in the mobility service. Multiple types of brands could be found in the different responses, such as the vehicle brand, the place brand, etc.

Table 49 Content of a Json entry describing a general brand

Field	Type	Description
hrn	String	Unique identifier
name	String	The brand name. Can be Lidl, Tesla, Aldi, …

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.

5.5.12. Summary

Table 50 provides the attributes for a trip summary. A summary lists the key attributes of a trip.

Table 50 Content of a Json entry describing a general summary

Field	Type	Description
duration	Integer	Duration in seconds.
length	Integer	Distance in meters.
consumption	Double	The consumption in kWh
baseDuration	Integer	Duration (in seconds) ignoring time-aware information. For pedestrian mode, the reported time is currently equal to that in duration.

5.5.13. Span

Table 51 provides the attributes for a span. A span represents a portion of the road that serves as a building block for describing a route.

Table 51 Content of a Json entry describing a span

Field	Type	Description
offset	Integer	Offset of a coordinate in the section's polyline.
length	Integer	Distance in meters.
functionalClass	Integer	Functional class is used to classify roads depending on the speed, importance, and connectivity of the road. 1: Roads allow for high volume, maximum speed traffic movement between and through major metropolitan areas. 2: Roads are used to channel traffic to functional class 1 roads for trip between and through cities in the shortest amount of time. 3: Roads that intersect functional class 2 roads and provide a high volume of traffic movement at a lower level of mobility than functional class 2 roads. 4: Roads that provide for a high volume of traffic movement at moderate speeds between neighbourhoods. 5: Roads with volume and traffic movement below the level of any other functional class.
speedLimit	Double	Speed in meters per second
consumption	Double	The consumption in kWh

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.

5.5.14. Transport

Table 52 provides the list of fields for a transport mean. A transport mean describes, with a string of characters, a possible mode of transport.

Table 52 Content of a Json entry describing a transport mean

Field	Type	Description
mode	String	The mode of transport. It can be a car, pedestrian, bus, regionalTrain, lightRail, …

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 53.

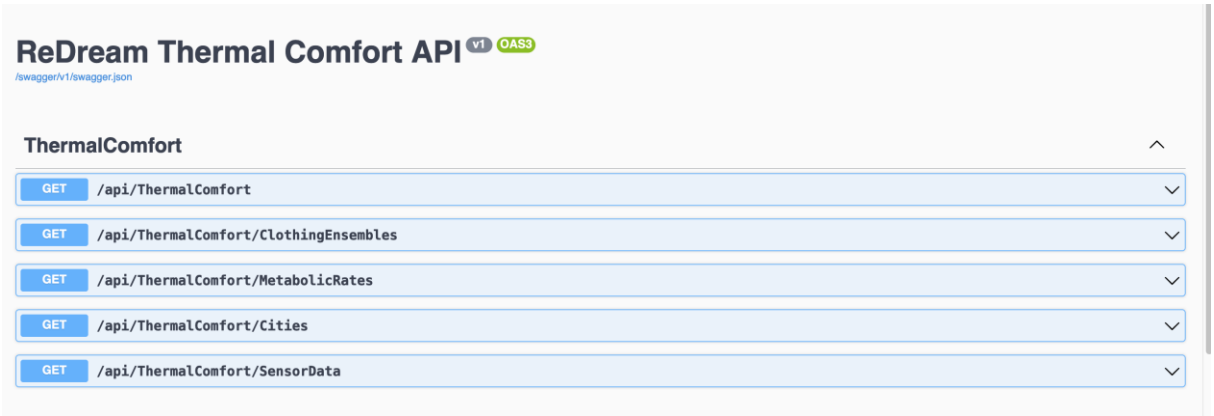
Thermal Comfort	
Request	**Description**
GET /	List thermal comfort data
GET /ClothingEnsembles	List all clothing ensembles by code and description
GET /MetabolicRates	List all possible metabolic rates such as “sleep”, “relax”, “dance”, etc.
GET /Cities	List all registered cities in the service with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

Table 53: 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

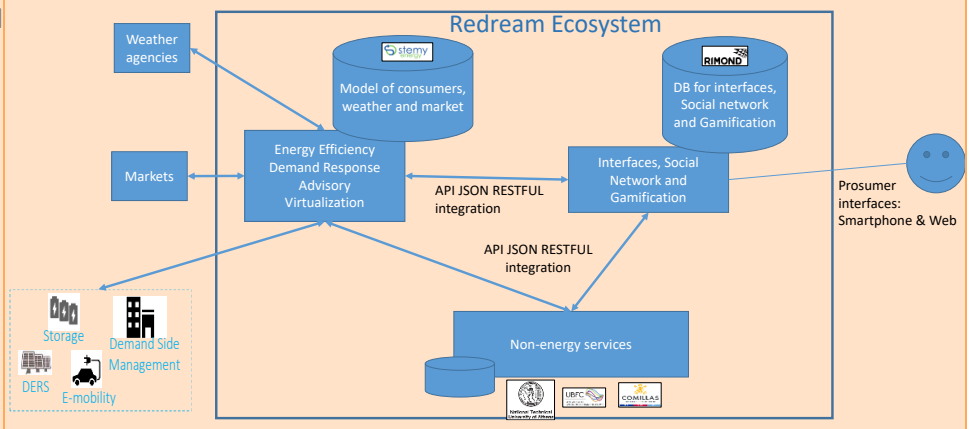


The project leading to this application has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 957837.

7. Conclusion

The contributions of this deliverable (Table 54) are the following:

1. Define the implementation details of the Cloud Architecture for the project. There will only be one REDREAM Identity Provider Service in the project, the one managed 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 REDREAM'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	There will be only Identity Provider in the project (REDREAM Identity Provider) but ReDREAM partners' services will have their own API url. Partners will request a valid token to REDREAM'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 obtained or use demo account for testing. 2. Mobility API. The Mobility API is open to any user, not only ReDREAM partners but credentials must be obtained or use demo account for testing. 3. Comfort API

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.

Energy API and Mobility usage examples	Exhaustive documentation of common use cases for the Energy API that may prove useful for ReDREAM service developers.

Obtain credentials	• Remember you have demo accounts: refer to https://redream.energy
	• If you have already an account because you are already part of ReDREAM. Use your login credentials.
	• To request new user credentials in any of the ReDREAM demos, please write an email to demo leaders or check their webpage to send you an invitation as user.
	  ○ ZEZ: flex@zez.coop
	  ○ BWCE: flexcommunity@bwce.coop or https://www.bwce.coop/flex-community/
	  ○ BIO: https://biodistrettoamerina.com/redream-gallese/ or flex@biodistrettoamerina.com
	  ○ ENER: flex@energetica.coop
	• To request new manager credentials, please write to info@stemyenergy.com.
	• Ask permissions to exiting managers/users to give you the neccesary permissions to access their data and comply to GDPR policies.

Table 54: 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 19.

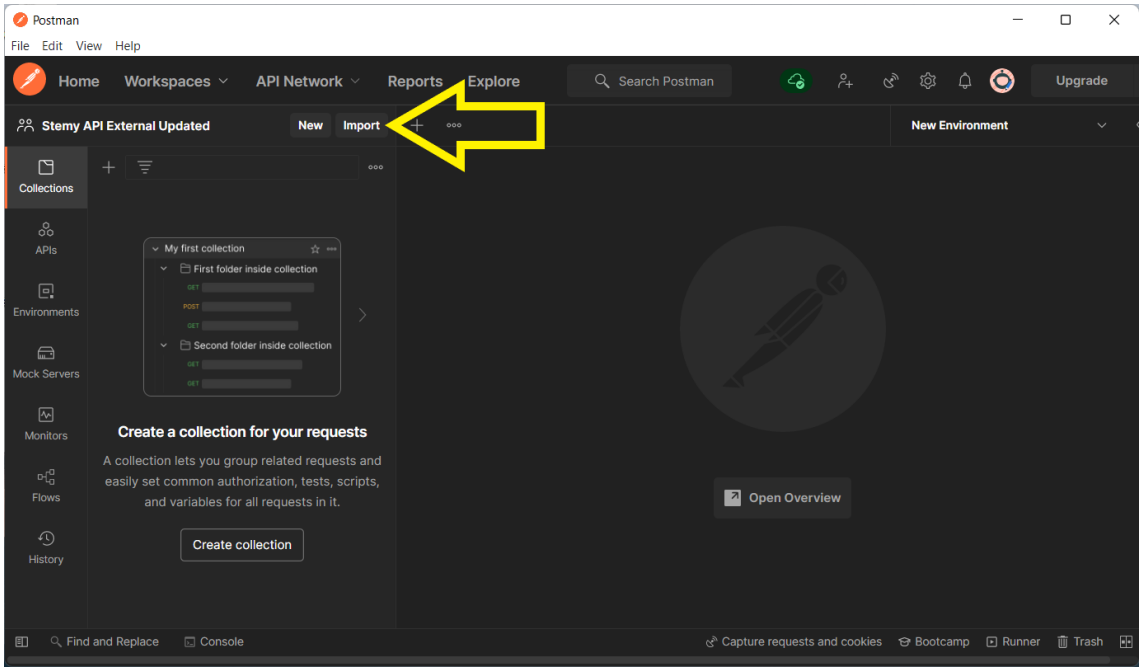


Figure 19: Import files in Postman

Next, click on the *Upload Files* button (Figure 20). 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.

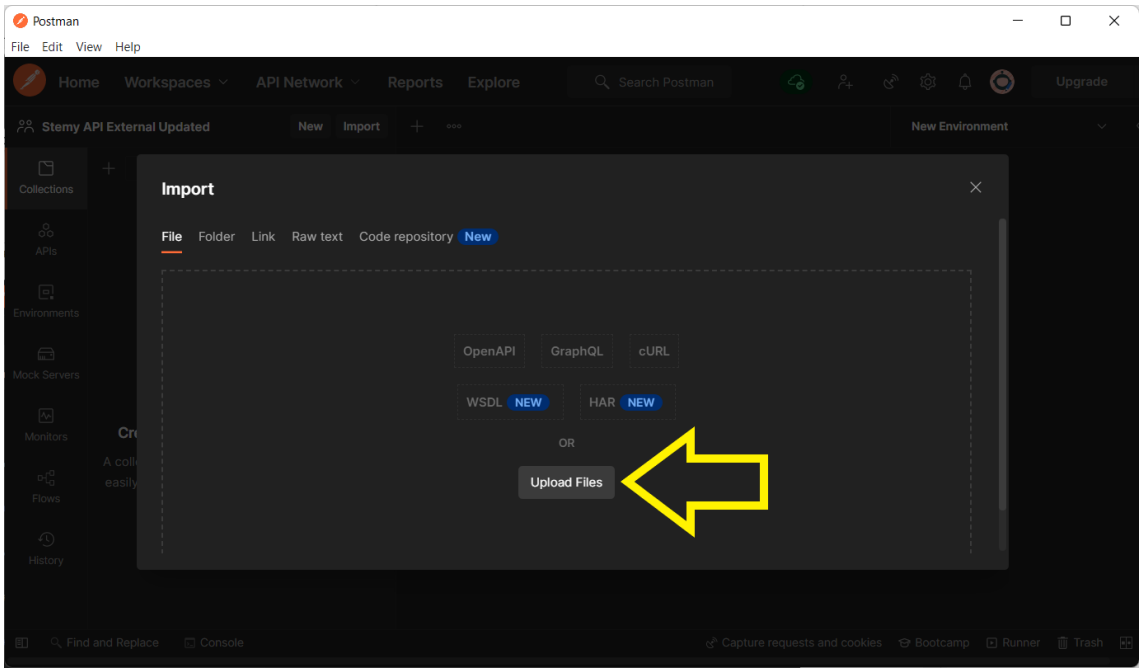
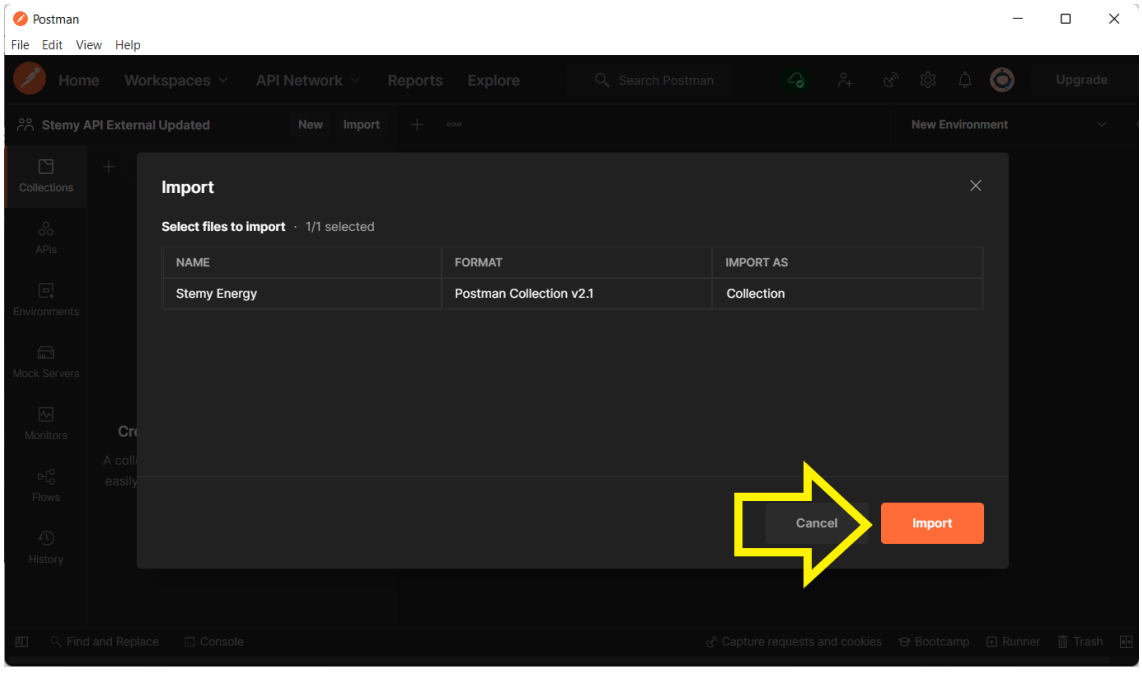


Figure 20: Upload files in Postman

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 21).

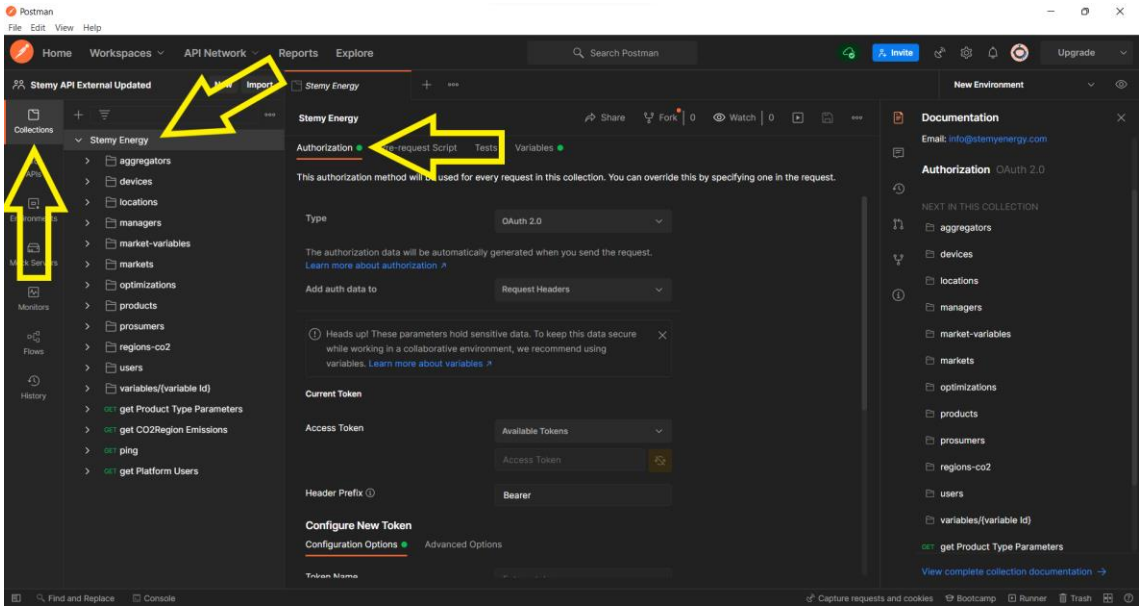


Figure 21: Access Authorization menu in Postman

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 22).

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.

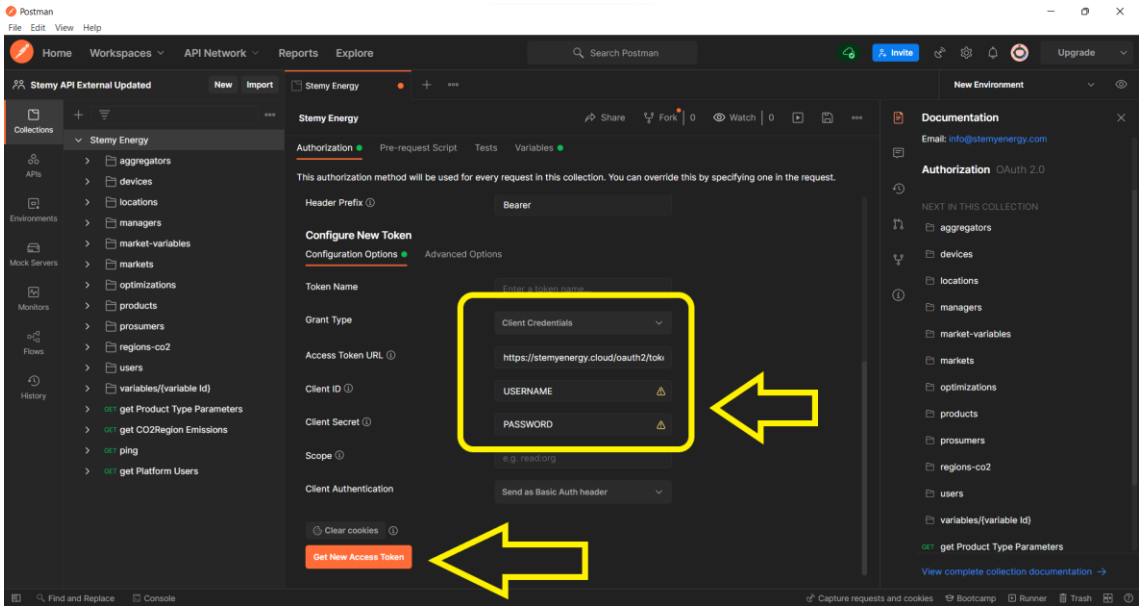


Figure 22: Getting a new Access Token

If the credentials are correct, you will see the authentication complete dialogue (see Figure 23). 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.

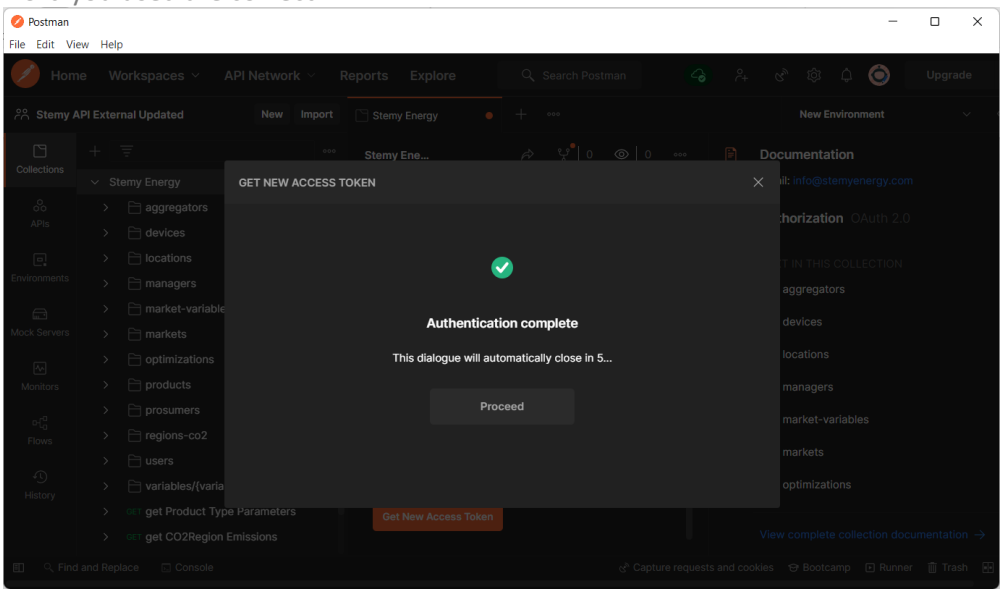


Figure 23: Success authenticating

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 24).

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.

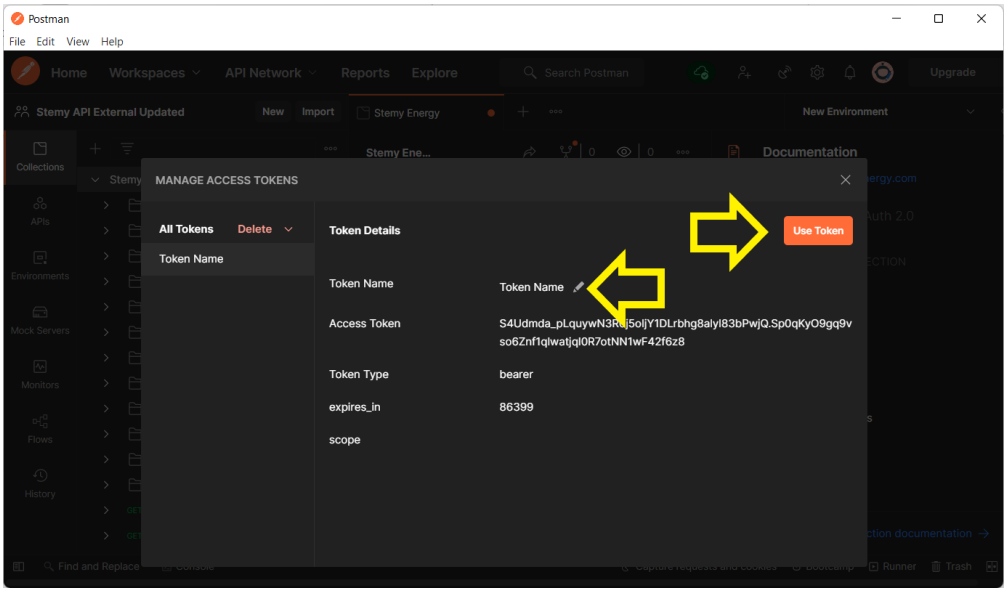


Figure 24: Renaming and saving the token

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 25).

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.

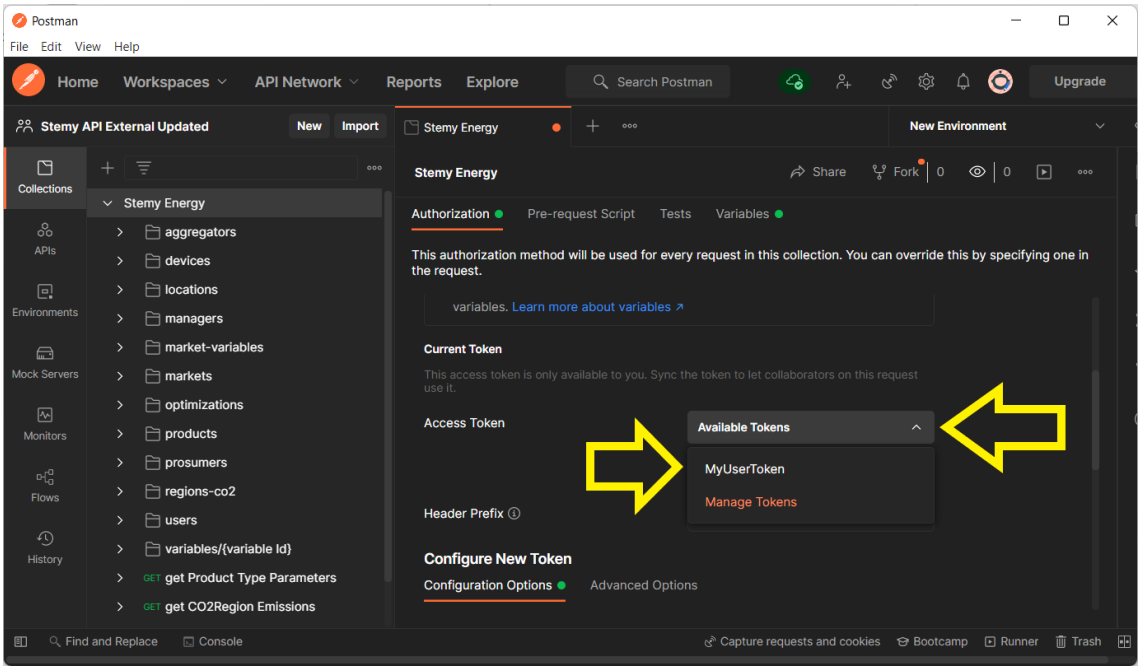


Figure 25: Managing multiple tokens

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 26).

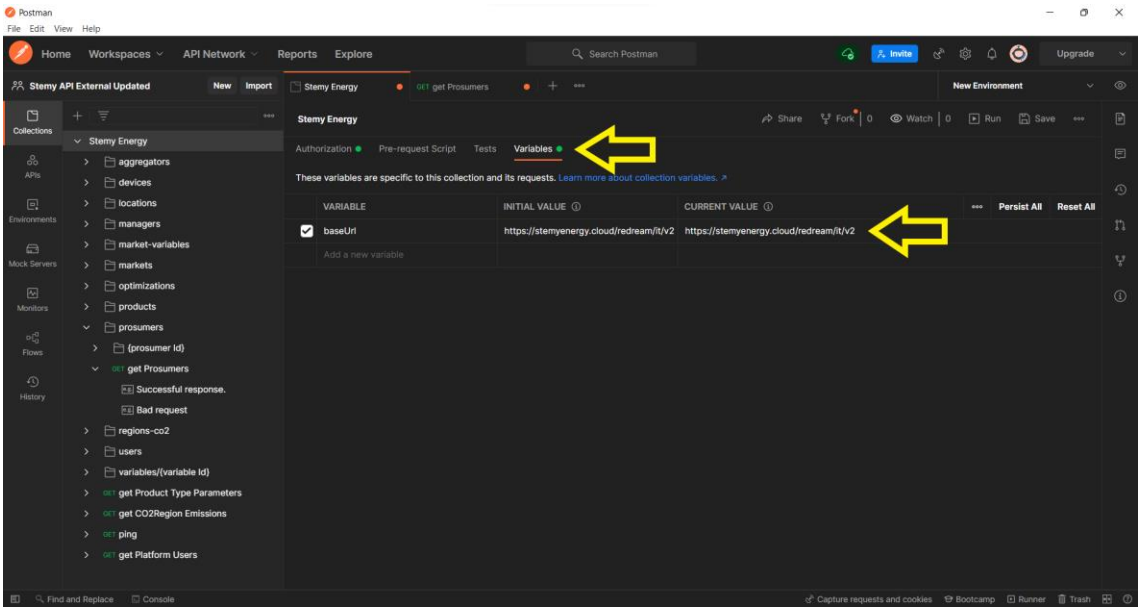


Figure 26: Changing target API deployment

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

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.

to the */prosumers* endpoint, they will get the *prosumerId* in the response body (Figure 27). 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.

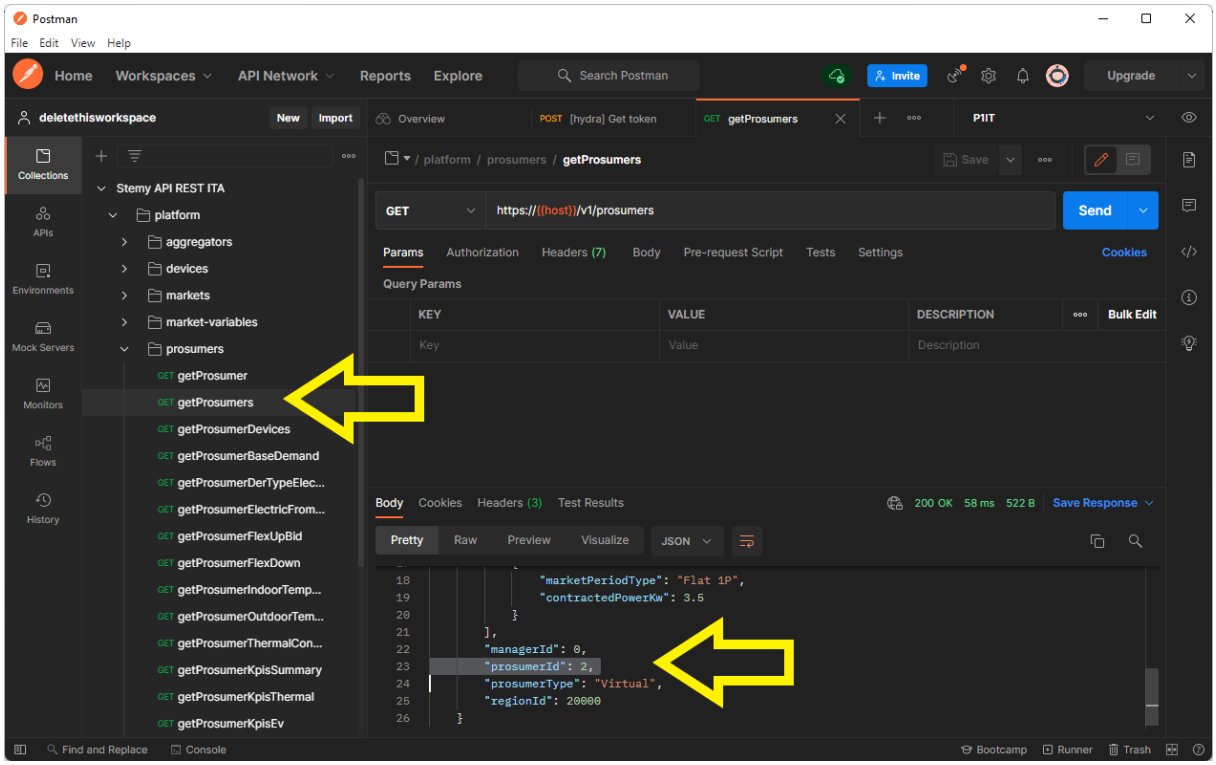


Figure 27: How to find out the prosumerId associated with a valid token

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 28.

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.

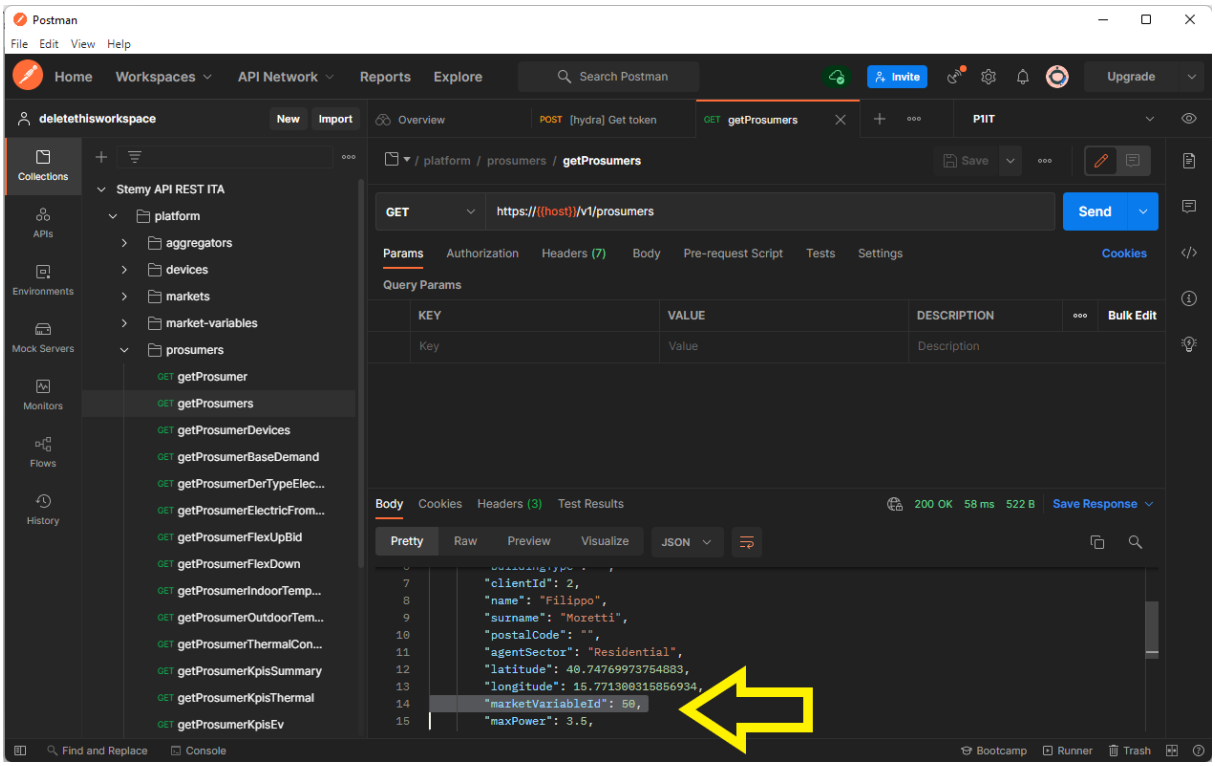


Figure 28: Getting the marketVariableId from the getProsumers request

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 29).

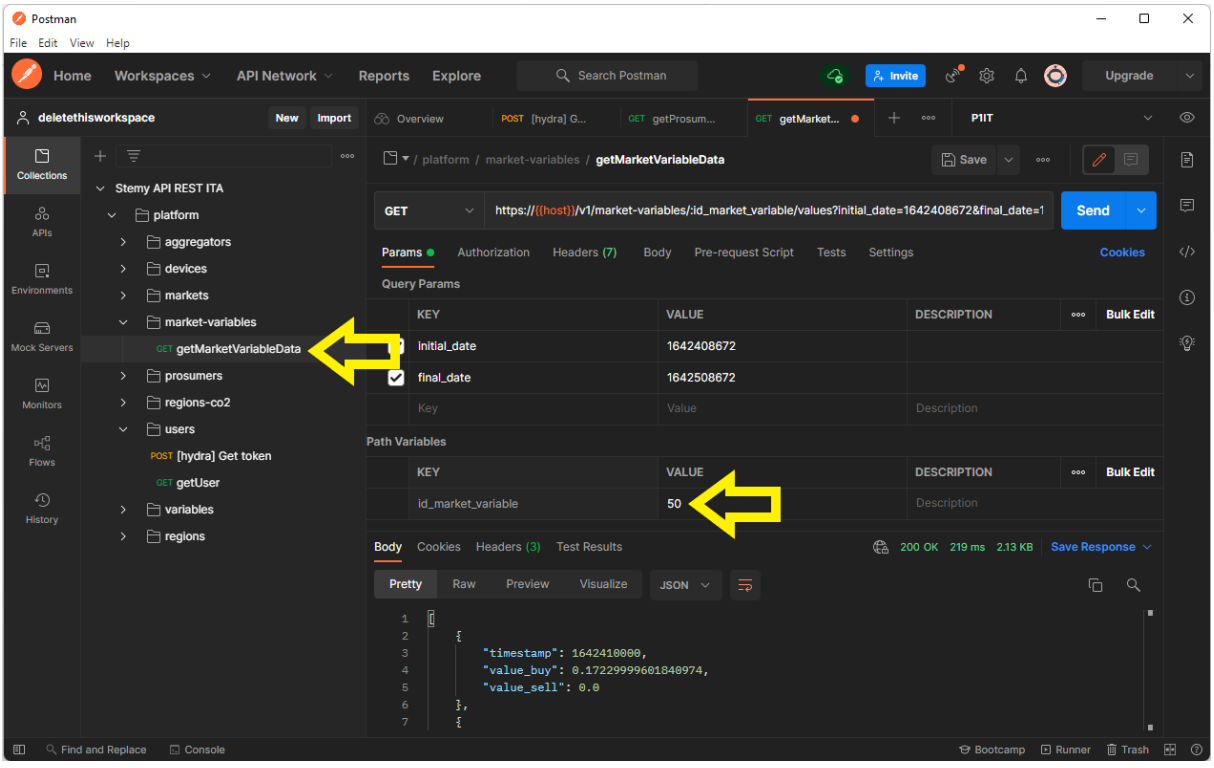


Figure 29: Getting market prices for a prosumer's energy tariff

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.*

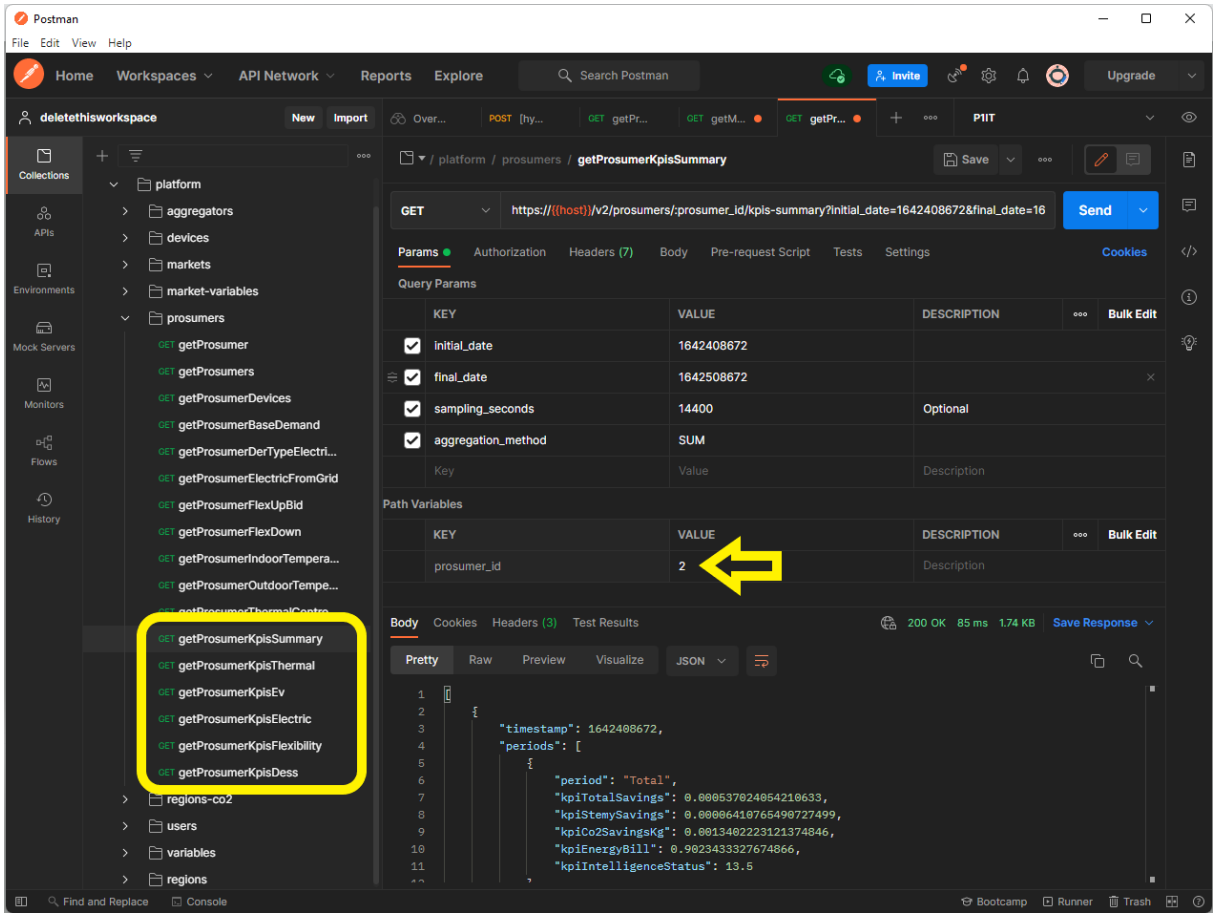


Figure 30: Getting prosumer KPIs

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

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.

- Visibility: Type of event visibility. Must be ‘Advise’ or ‘Advise aggregated’

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

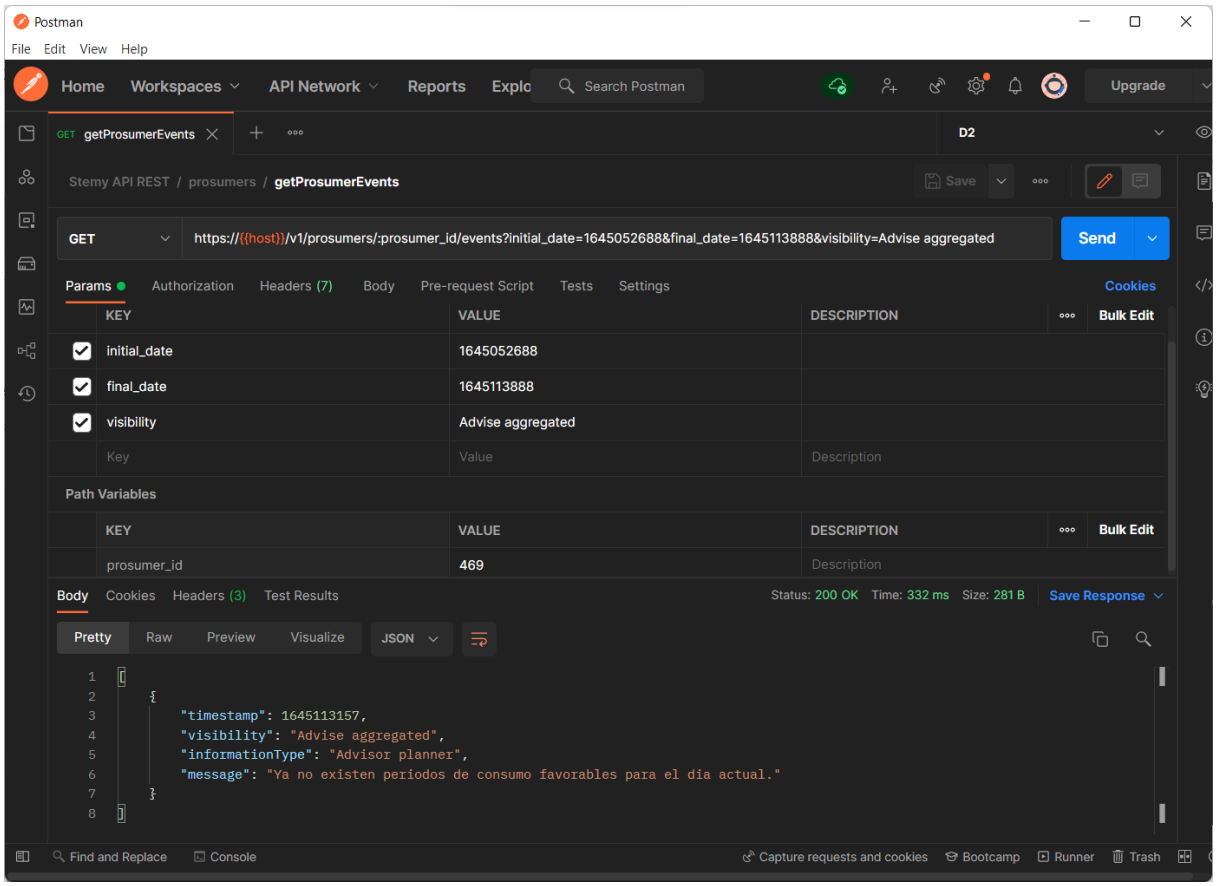


Figure 31: Getting events from the Prosumer Advisor service

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.

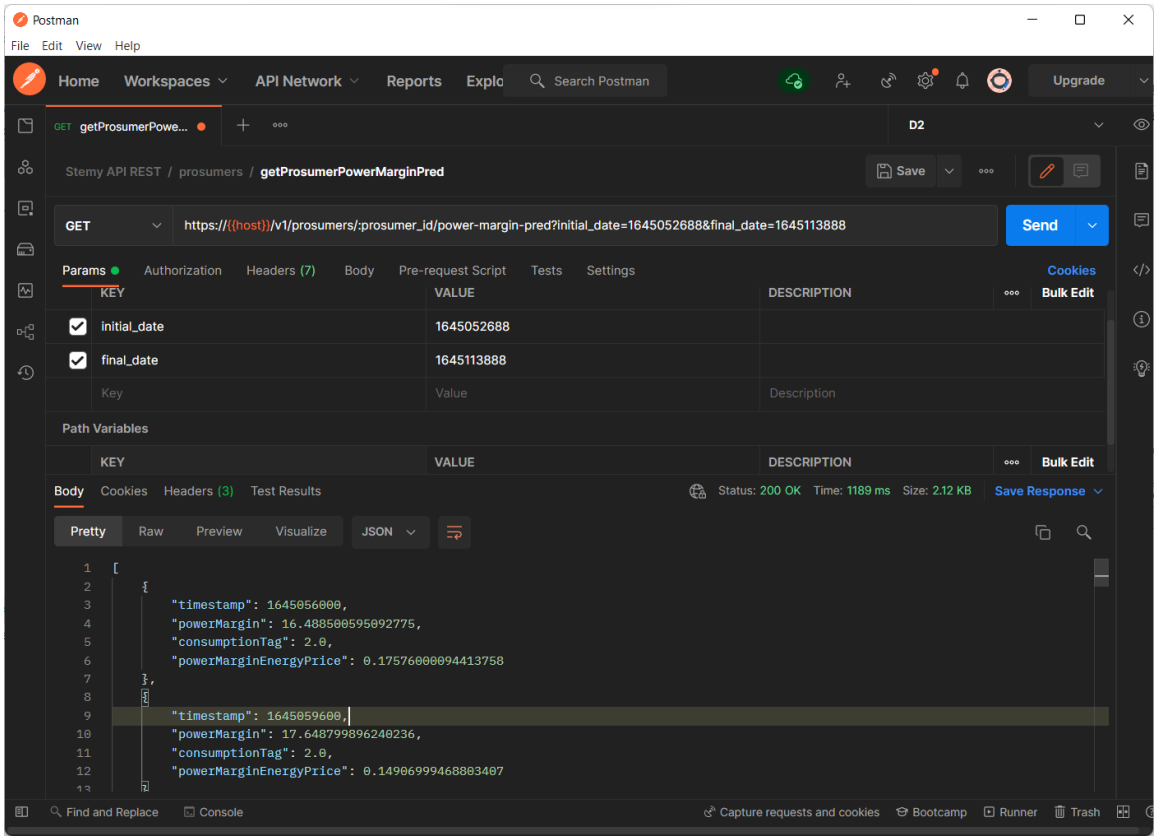


Figure 32: Get prosumer power margin predictions from the Prosumer Advisor service

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.

Annex 2: Example of Access to the Mobility Service and Running of Mobility Service Simulator

The mobility service includes a simulator (as briefly described at the beginning of Section 3 in D3.5) for reproducing the behaviour of an individual. The mobility service simulator is embedded into the microservices, and especially the routing microservice. **To run or launch the simulator, there is no specific action to be done because it is internally managed by the routing microservice connected to the HERE routing API. Consequently, the user of the mobility service needs only to know how to use the endpoints of the mobility service.** This design approach is based on the principle of the Separation of Concern that is well known in software engineering and applied in most of the service-oriented software architectures. **The mobility service simulator is run by the mobility service itself when a service's request needs it.**

This section provides an example process that you need to follow to access to the mobility service through its endpoints. This example is based on the goal to retrieve the energy price for a specific routing.

1. Request a token using the *AuthService:GetToken* endpoint (Section 5.1) with your credentials.
2. Use the auth token as Bearer Authentication token in the header of all the following requests.
3. Find the vehicle ID that you want to use for the routing using the *VehicleService:GetAllVehicles* endpoint (Section 5.3).
4. Use the *RoutingService:GetXXRoute* endpoint (section 5.4) depending on your vehicle type (*XX* should be replaced by the type of vehicle).
    a. If the vehicle you choose is electricity-powered, use the *RoutingService:GetEvRoute* endpoint (Section 5.4.1).
    b. If the vehicle you choose is fuel-powered, use the *RoutingService:GetFuelRoute* endpoint (Section 5.4.2).
5. Use the Route JSON object (see Section 5.5) that you just receive and parse the data in the **priceInEuros** field. The name of the Json field depends on the initial request. It is described in Section 5.5.

You can follow the same process to deal with the other Routing-service endpoints (Section 5.4).

To help a third-party developer to use the mobility service API, the documentation of the API is provided on http://h2020-redream.utbm.fr website. We recommend using the software Postman for exploring the API and obtaining the documentation easily. Postman is a REST API client application that can be downloaded: https://www.postman.com/downloads/. To download the Postman API request collection, please refer to the content of http://h2020-redream.utbm.fr which is provided with the open API full documentation in the form of downloadable JSON files. Before using Postman, download the documentation files to your development machine and import them into the Postman environment, according to the documentation of this software.

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 3: Third-party access and development of external services.

In this section, the methodology of how a third party can use the REDREAM to develop their service is explained. Actually is quite similar to the interactions explained in section 3.

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

1. Get a valid token. Options:
    - Remember you have demo accounts for REDREAM: refer to https://redream.energy.
    - If you have already an account because you are already part of ReDREAM. Use your login credentials.
    - To request new user credentials in any of the ReDREAM demos, please write an email to demo leaders or check their webpage to send you an invitation as user.
        - ZEZ: flex@zez.coop
        - BWCE: flexcommunity@bwce.coop or https://www.bwce.coop/flex-community/
        - BIO:flex@biodistrettoamerina.com or https://biodistrettoamerina.com/redream-gallese/
        - ENER: flex@energetica.coop
    - To request new manager credentials, please write to info@stemyenergy.com.
    - Ask permission to exit managers/users to give you the necessary permissions to access their data and comply with GDPR policies.
2. After getting the credential, you can follow the full API documentation presented in links located in sections 4-6 to request the existing information and manage devices. In those sections, examples of requests are explained. This allows third parties to develop their own interfaces or services to monitor and control the devices and use the ecosystem portals, apps and mobility and comfort services.
3. Once the user has a token and knows the documentation:
    - The Third-party service developer 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 Third-party service developer may do any (*GET, POST…*) request using this prosumerId.
    - It might be the case that the token is expired, so the Third-party will receive an *HTTP* code *of 401 Unauthorized*. In that case, the Third-party developer will have 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.