

Directorate-General for Internal Market, Industry, Entrepreneurship and SMEs
GROW F.1 – Innovation Policy and Investment for Growth

D06.02 CMISA API Manual

This document provides the required information both for aggregators and 3rd party websites on how to use the CMISA data via an API

Specific Contract n°373 under Framework Contract n° DI/07624:
ABCIV Lot 3: Collaborative mapping of innovation supporting actors

This document has been produced by:



Control Information

Settings	Value
Document Title:	D06.02 CMISA API Manual
Project Title:	Specific Contract n°373 under Framework Contract n° DI/07624 - ABCIV Lot 3: Collaborative Mapping of innovation supporting actors
Document Authors:	Konstantina KYRIAKOPOULOU Kostas ANDROULAKIS Anastasia SOFOU Evangelos NITTIS Anastasios KYREZOPOULOS Stelios KITZIRIS
Project Owner:	Ulla Engelmann
Project Officers (POs):	Szabolcs SZEKACS
Contractor's Project Manager (CPM)	Konstantina KYRIAKOPOULOU
Doc. Version:	5.00
Sensitivity:	Limited
Date:	21/09/2021

Document Approver(s) and Reviewer(s):

Name	Role	Action	Date
SZEKACS Szabolcs	Project Officers	Review / Validation	15/09/21
SZEKACS Szabolcs	Project Officers	Review	15/07/21
SZEKACS Szabolcs	Project Officers	Review	09/06/21

Revision history:

Revision	Date	Created by	Short Description of Changes
5.00	21/09/21	Konstantina KYRIAKOPOULOU	Creation of the final deliverable
4.01	16/09/21	Konstantina KYRIAKOPOULOU	Implementation of comments received from PM
4.00	01/09/21	Konstantina KYRIAKOPOULOU	Submission of the final document to PMs
3.04	31/08/21	Konstantina KYRIAKOPOULOU	Review and preparation of the final deliverable
3.01-3.03	30/07/21	Konstantina KYRIAKOPOULOU Kostas ANDROULAKIS	Implementation of PM's feedback
3.00	28/06/21	Konstantina KYRIAKOPOULOU	Submission of the final document to PMs
20.1-2.03	23/06/21	Konstantina KYRIAKOPOULOU Anastasia SOFOU Evangelos NITTIS Anastasios KYREZOPOULOS	Implementation of comments received from PM

2.00	11/06/21	Konstantina KYRIAKOPOULOU	Version shared with project partners
2.00	10/06/21	Konstantina KYRIAKOPOULOU	Creation of final version
1.01-1.04	10/06/21	Konstantina KYRIAKOPOULOU Anastasia SOFOU Evangelos NITTIS Anastasios KYREZOPOULOS	Implementation of comments received from PM
1.00	08/06/21	Konstantina KYRIAKOPOULOU	Submission of the document to PMs
0.07	08/06/21	Konstantina KYRIAKOPOULOU	Implementation of internal comments and creation of final version before submission
0.06	04/06/21	Ludovic MAYOT	Quality Review
0.03-0.05	04/06/21	Konstantina KYRIAKOPOULOU Anastasia SOFOU Evangelos NITTIS Anastasios KYREZOPOULOS	Review and update of the document
0.01-0.02	26/03/21	Ksenia BOCHAROVA	Initiation of the document

Disclaimer

This deliverable was prepared for the ISA² programme by Trasys International under SC373 and is the European Commission's property. The views expressed in this document are purely those of the authors and may not, in any circumstances, be interpreted as stating an official position of the European Commission. The European Commission does not guarantee the accuracy of the information included in this document, nor does it accept any responsibility for any use thereof.

Reference herein to any specific products, specifications, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favouring by the European Commission.

All care has been taken by the authors to ensure that they have obtained, where necessary, permission to use any parts of manuscripts including illustrations, maps, and graphs, on which intellectual property rights already exist from the titular holder(s) of such rights or from her/his or their legal representative.

© European Union, 2021

TABLE OF CONTENT

INTRODUCTION	6
SCOPE.	6
1. CMISA APIS – HOW TO SEND AND RETRIEVE DATASETS TO/FROM THE CMISA CENTRAL DATABASE	7
1.1. API - Sending aggregator datasets to CMISA central database	7
1.2 API – 3rd Party web sites retrieving datasets from CMISA central database.....	8
2. SENDING DATASETS TO CENTRAL CMISA DATABASE – FOR AGGREGATOR WEBSITES	9
2.1. Preparation of datasets	9
2.2. Format and structure of API JSON	9
2.3. API Specification and Swagger access.....	13
2.4. Sending the data to the Server	14
2.5. Access the API – Identification and Authorisation.....	16
2.6. Upload of the datasets.....	17
3. RETRIEVING DATASETS FROM CENTRAL CMISA DATABASE – FOR ALL 3RD PARTY WEBSITES	20
3.1. Querying the database.....	20
4. ANNEXES	22
4.1. Annex 1 – CMISA JSON Template payload.....	22
4.2. Annex 2 – Information on how to use Controlled Vocabularies in JSON	22

LIST OF FIGURES

<i>Figure 1 – Overview of API receiving datasets from Aggregators.....</i>	<i>7</i>
<i>Figure 2 - Overview of API retrieving datasets from Central Database</i>	<i>8</i>
<i>Figure 3 - Swagger File preview</i>	<i>14</i>
<i>Figure 4 - Swagger – Save Actor area</i>	<i>14</i>
<i>Figure 5 - Swagger – Save Actor parameters</i>	<i>15</i>
<i>Figure 6 - Example of Access Token.....</i>	<i>16</i>
<i>Figure 7 - Example on how to invoke the API.....</i>	<i>17</i>
<i>Figure 8 - Invalid token.....</i>	<i>17</i>
<i>Figure 9 - Swagger – Try it out - Responses</i>	<i>18</i>
<i>Figure 10 – Overview: the process of API sending datasets to demonstrator website</i>	<i>20</i>

LIST OF TABLES

<i>Table 1 – JSON template</i>	<i>10</i>
<i>Table 2 – Required attributes in JSON file</i>	<i>15</i>
<i>Table 3 – Error 404 message</i>	<i>18</i>
<i>Table 4 - Error 401 message/ Unauthorised</i>	<i>18</i>
<i>Table 5 – Error 409 message</i>	<i>19</i>

INTRODUCTION

Echoing the [EU Industrial Policy](#), the European Commission (EC) recognises that innovation plays a key role for the European Union (EU) industry competitiveness. In this line, Member States' (MS) public administrations and businesses already have access to various websites that provide information on advanced technologies, testing facilities, financing, etc. in the area of innovative solutions (e.g. the European Network for Pilot Production Facilities and Innovation Hubs).

However, many of these websites provide only fragmented information rather than a comprehensive overview, forcing the authorities and businesses to visit many websites to obtain all needed data about innovation aspects. This fragmentation and lack of data interoperability, represents a burden and sets additional costs for the different interested stakeholders.

In this context, the EC via the ISA² Programme finances the current assignment 'Collaborative Mapping of innovation supporting actors' (CMISA) with the aim to develop and implement a: 1) central online database to serve as a basis for participating websites to upload their datasets; 2) common semantic data model to be used by participating websites to share their information on innovation supporting actors and instruments; 3) a central demonstrator website allowing the search of collected datasets within a graphical user interface, and; 4) an API to ease exchange of data between various 3rd party European and national websites and allow public administrations and enterprises to receive needed information faster and in a more efficient manner. For the purpose of this document, Project Team refers to all stakeholders sharing their dataset with CMISA as 'aggregators'.

SCOPE

This deliverable provides useful information to aggregators and 3rd party websites on how to share their datasets with CMISA online demonstrator website, retrieve and use these data by accessing the related API. Specifically, it outlines the steps that aggregators should follow in order to import their data to the central database via a CMISA API as well as how to call the API in order to retrieve data from CMISA¹.

The implementation of the CMISA API aims at increasing the data quality by elaborating a flexible common set of rules with the descriptions of the actors and instruments. It also helps reducing effort for EU data providers by enabling a harmonised central dataset, making available information from the participating websites. This endeavour should allow policy makers to analyse more efficiently innovation-related datasets, enabling decision-making on accurate evidence.

It is important to note, that the current manual is a 'live' document and as such it evolves over time with the progress on the project but also via feedback received from aggregators.

¹ Please note, that in addition to an API, CMISA provides additional means, such as an embeddable map, or datasets in downloadable excel format for websites to reuse the data it collects. For more information, please visit <https://joinup.ec.europa.eu/collection/cmisa>

1. CMISA APIS – HOW TO SEND AND RETRIEVE DATASETS TO/FROM THE CMISA CENTRAL DATABASE

The overall process followed in order to send and retrieve datasets to/from the CMISA central database is the following: 1) aggregators collect the data from different websites/actors, 2) they merge and structure the information under a common format (JSON) which are then, 3) sending to the CMISA central database via CMISA API. Any 3rd party websites will be able to access the central database of the system via the CMISA API in order to retrieve the desired datasets and use them for their purpose.

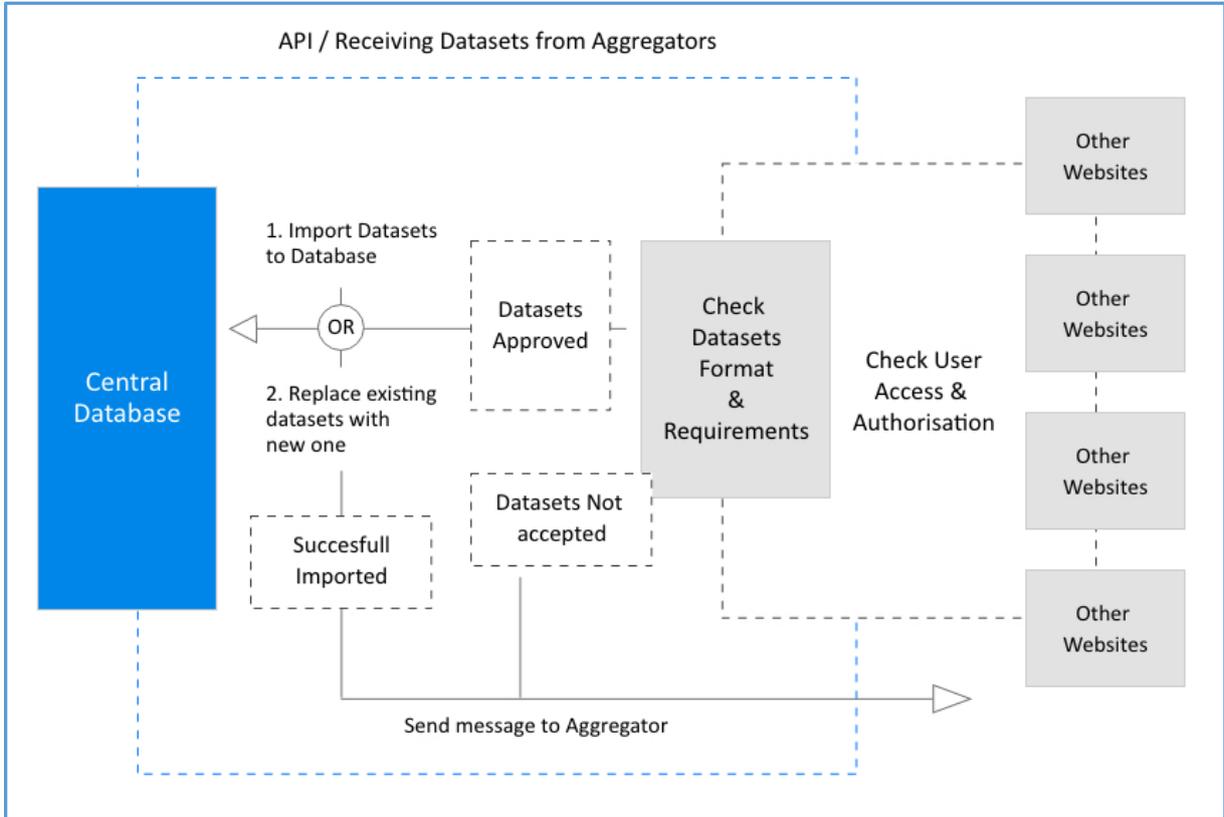
In this context, the Project Team developed two (2) APIs in PHP v7.4 / Symphony v5, as the standard technology suggested from the EC.

- The first API is used to **receive datasets** from aggregators and import them to the main central database (see [1.1 API – Sending aggregator datasets to CMISA central database](#)).
- The second API allows the CMISA demonstrator website, as well as 3rd party websites, to **query** the central database and **retrieve the requested datasets**. Any 3rd party website will be able to use this API by requesting access to the server/API and then CMISA will grant access by adding the related and required domains (i.e. Domain.com) of the 3rd party website in the API allowed origins area (see [3. API – Retrieving datasets from central CMISA database – For all 3rd party websites](#)).

1.1. API - Sending aggregator datasets to CMISA central database

The overview of the process followed by the API when receiving datasets from aggregators is presented in *Figure 1* below.

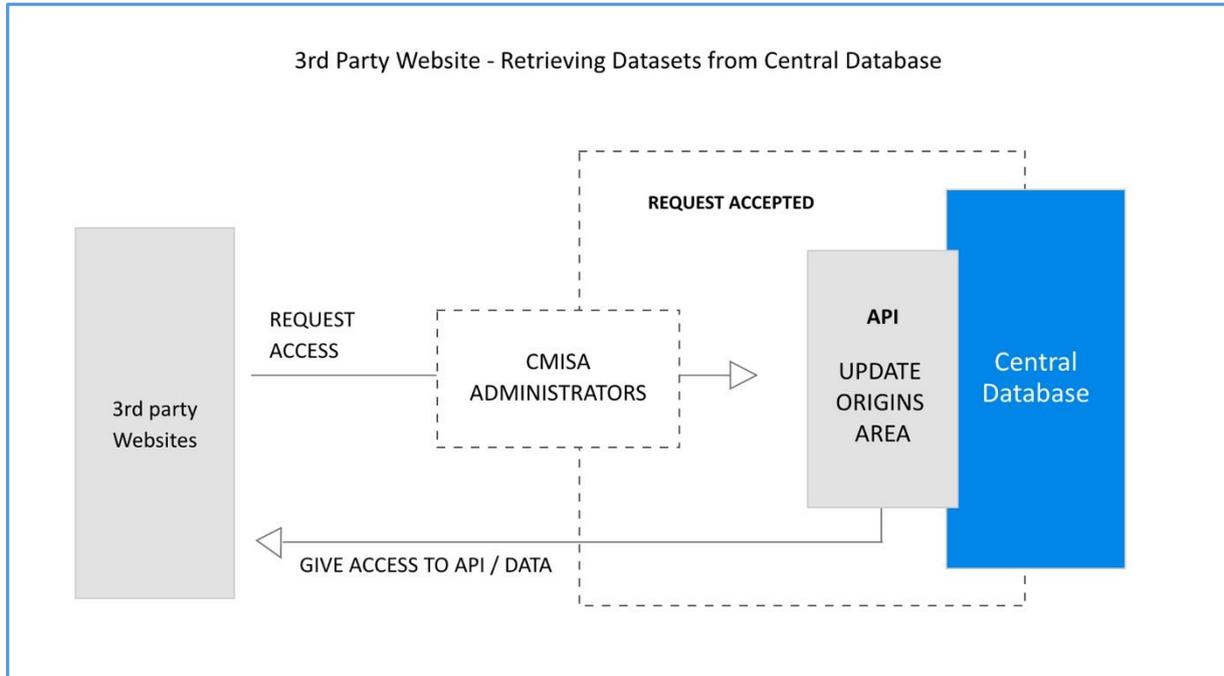
Figure 1 – Overview of API receiving datasets from Aggregators



1.2 API – 3rd Party web sites retrieving datasets from CMISA central database

The overview of the process followed by the API when retrieving datasets from Central Database and sending datasets to 3rd party websites is presented in *Figure 2* below.

Figure 2 - Overview of API retrieving datasets from Central Database



2. SENDING DATASETS TO CENTRAL CMISA DATABASE – FOR AGGREGATOR WEBSITES

2.1. Preparation of datasets

The web API used to upload datasets received from aggregators will merge the data into one dataset in the central databased. These datasets must meet certain format and structure, as described below.

1. The datasets should meet the requirements set by the common data model²:

- All data pushed to the database need to have the same characteristics, and at least the mandatory data must be provided.
- In case the datasets do not match the existing format in the common data model, then the data will not be added to the database. The Aggregators will receive a message in case of success or failure (see 2.6).

2. The aggregators should use JSON format to send CMISA their data:

- The Aggregators will be able to send their datasets to the central database by using a predefine JSON file via the API. The JSON format and values can be found in section '2.2. Format and structure of API JSON' below.

3. The Aggregators are authorised to upload datasets:

- The system will check if the aggregator is authorised to use the API and send data. This authorisation will be completed via the Commission's API Gateway (see section '2.3 Access the API – Identification and Authorisation').

Importantly, the transfer of the data will only be possible if the above-mentioned criteria are followed.

2.2. Format and structure of API JSON

A template of the JSON³ format that should be used is provided in [Annex 1](#) and it is shown in *Table 1 below*. To support the aggregators in the creation of JSON, there is important information below:

- The JSON includes tags that are associated with controlled vocabularies. More information on how to use controlled vocabularies when sharing data via JSON can be found in [Annex 2](#).
- In the JSON in *Table 1*, wherever you see the value '0', it should be replaced by string data of your choice.
- In the JSON in *Table 1*, wherever you see the value 0, it should be replaced by numerical data of your choice.
- Regarding the platform ID, each aggregator is assigned a unique number. Please consult the list below.

² Common data model available in all serialisation here:

<https://joinup.ec.europa.eu/collection/cmisa/solution/cmisa/release/101>

³ More information on JSON: <https://jsonapi.org/format/>

- ATI : **platform ID 1**
 - DIH : **platform ID 2**
 - EEN : **platform ID 3**
 - ECCP : **platform ID 4**
 - KET4CP : **platform ID 5**
- If your aggregator is not listed here, please get in touch with us on cmisa@trasysinternational.com.
 - When an aggregator sends us a new dataset, all previous data from the given aggregator is deleted from the central database and only the new dataset is saved.
 - The maximum JSON file size the system can support is **5MB**⁴.

Table 1 – JSON template⁵

```

{
  "platform_id": 0,
  "actors": [
    {
      "actor_classification__id": 1,
      "actor_type__id": 1,
      "actor_name": "0",
      "alternative_actor_name": "0",
      "actor_description": "0",
      "aggregation": [
        {
          "platform_actor_access__url": "0",
          "aggregation_identifier": "0"
        }
      ],
      "linked_actor": [],
      "contact_details": [
        {
          "email": "0",
          "telephone": "0",
          "name": "0",
          "position": "0"
        },
        {
          "email": "0",
          "telephone": "0",
          "name": "0",
          "position": "0"
        }
      ]
    }
  ]
}

```

⁴ We are currently testing the platform in order to ensure that it can support an even larger file size.

⁵ Note to aggregators: we continuously work on improving the JSON. Hence, certain fields might become obsolete in updated versions.

```

],
"actor_identifiers": [
  {
    "actor_identifier": "0"
  },
  {
    "actor_identifier": "0"
  }
],
"actor_urls": [
  {
    "actor_url": "0"
  },
  {
    "actor_url": "0"
  },
  {
    "actor_url": "0"
  }
],
"actor_collaboration_relationships": [
  {
    "collaboration_id": 1,
    "organizational_collaboration": [
      {
        "collaboration_identifier": [
          {
            "collaboration_identifier": "0"
          }
        ],
        "collaboration_classification": [
          {
            "collaboration_classification": "Project"
          }
        ],
        "collaboration_name": "0"
      }
    ]
  }
],
"actor_classification": [
  {
    "actor_classification_label": "0",
    "actor_classification_notation": "0"
  }
],
"actor_type": {
  "actor_type_id": 1,
  "actor_type_label": "0",
  "actor_type_notation": "0",
  "actor_type_definition": "0"
},

```

```

"suborganization": [],
"location": [
  {
    "geographic_name": "0",
    "geographic_identifier": "0",
    "geometry": "0",
    "address": [
      {
        "full_address": "0",
        "admin_unit_l1": "0",
        "admin_unit_l2": "0",
        "state": "0",
        "postal_code": "0"
      }
    ]
  }
],
"actor_catalog_relationship": [
  {
    "catalog_id": 1,
    "generic_catalog": [
      {
        "catalog_name": "0",
        "catalog_description": "0",
        "catalog_value": "0",
        "catalog_attribute": [
          {
            "attribute_id": 0,
            "catalog_id": 0,
            "attribute_name": "0",
            "attribute_value": "0",
            "attribute_description": "0"
          }
        ]
      }
    ]
  }
],
"actor_sector_relationship": [
  {
    "sector_id": 0,
    "sector": [
      {
        "sector_name": "0",
        "sector_description": "0",
        "sector_type": [
          {
            "sector_type_label": "0",
            "sector_type_notation": "0",
            "sector_type_parent_id": 0
          }
        ]
      }
    ]
  }
]

```

```

    }
  ]
}
],
"actor_technology_relationship": [
  {
    "technology__id":1,
    "technology": [
      {
        "technology__id": 0,
        "technology_name": "0",
        "technology_description": "0",
        "technology_type": [
          {
            "technology_type__id":0,
            "technology_type_label": "0",
            "technology_type_notation": "0"
          }
        ]
      }
    ]
  }
],
"actor_trl_relationship": [
  {
    "technology_readiness_level__id": 0,
    "technology_readiness_level": {
      "technology_readiness_level__id": 0,
      "technology_readiness_level": "0",
      "technology_readiness_level_description": "0"
    }
  }
]
}
]
}

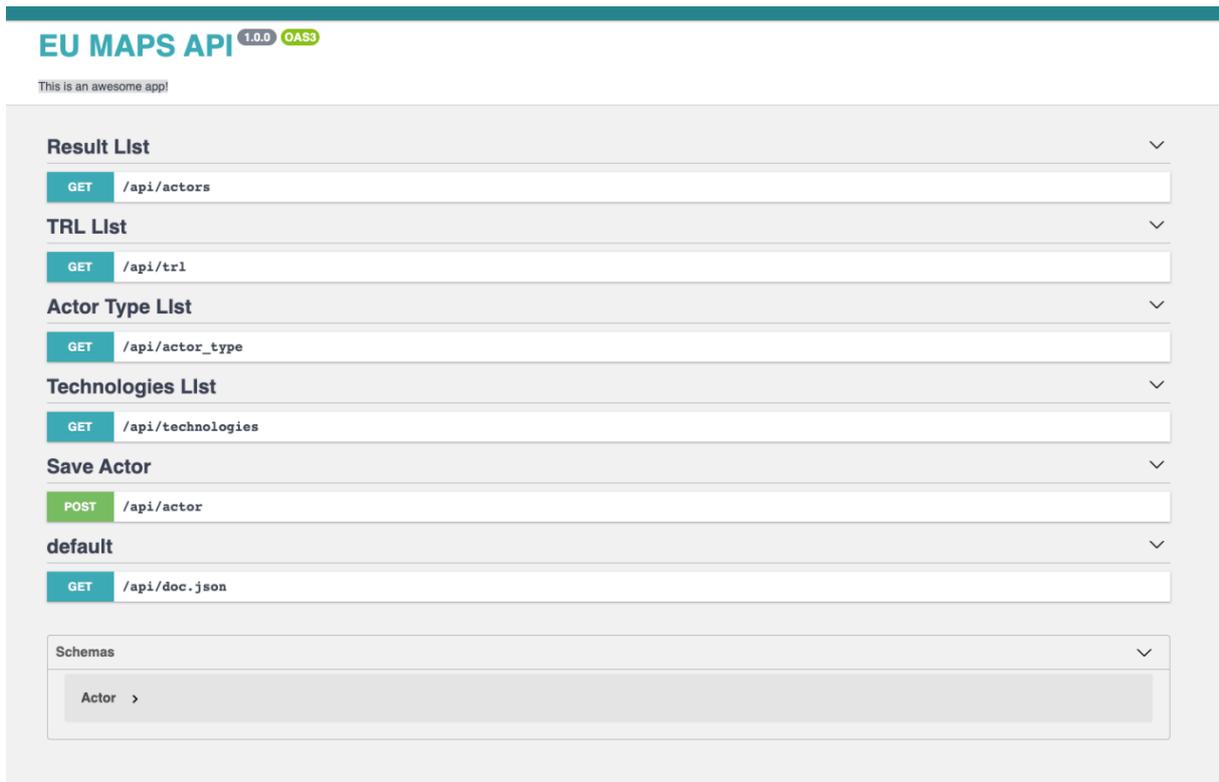
```

2.3. API Specification and Swagger access

The aggregators can access the API JSON files and find the following available end points on a swagger file. This swagger file is shown in *Figure 3*⁶.

⁶ Access to the Swagger file as downloadable document will soon become available.

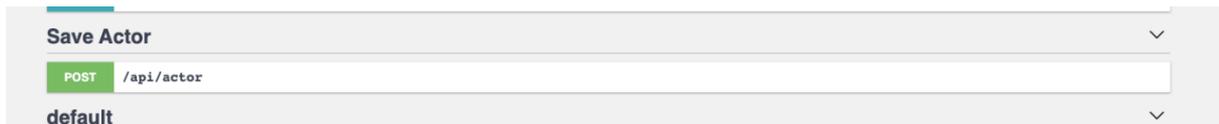
Figure 3 - Swagger File preview



2.4. Sending the data to the Server

The aggregators can send their data to the server using the information that appears under the section 'Save Actor' in the swagger file shown in *Figure 4*.

Figure 4 - Swagger – Save Actor area



The field 'Save actor' area provides the information of all required mandatory fields that the aggregators' JSON must contain and send to the central server. These required fields are shown in *Table 2* and *Figure 5*.

Table 2 – Required attributes in JSON file

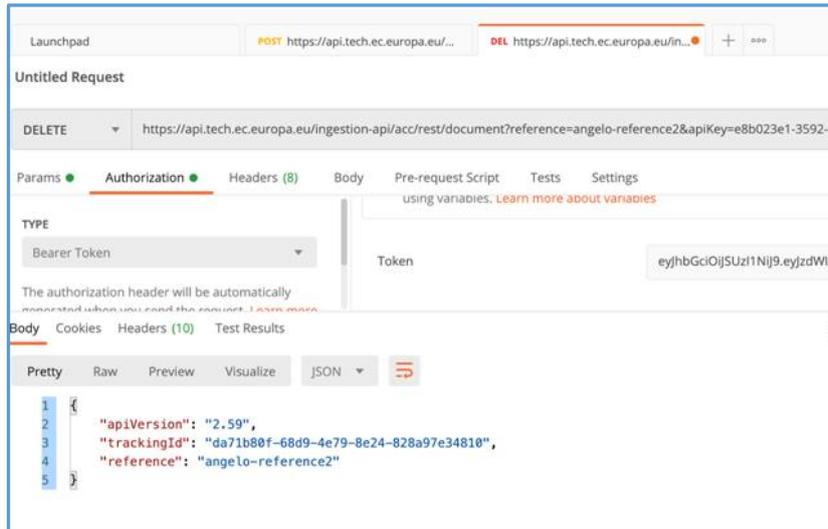
Name	Description
actor_classification__id	The id of the actors classification
actor_description	Actors description field
actor_name	Actors name
actor_type__id	The type of the actor
actor_url	Array Of Actors URL
alternative_actor_name	Short Actor Name
contact_details	The contact details of Actor
linked_actor	An Array of ids of the actors that are linked with actor
location	The location of the Actor
suborganization	Array of ids of the actors that are a sub-organisation of the Actor

Figure 5 - Swagger – Save Actor parameters

The screenshot shows the Swagger UI for the 'Save Actor' endpoint. The parameters section is titled 'Parameters' and includes a 'Try it out' button. The parameters are listed in a table with their names, types, and descriptions, and each has a corresponding input field.

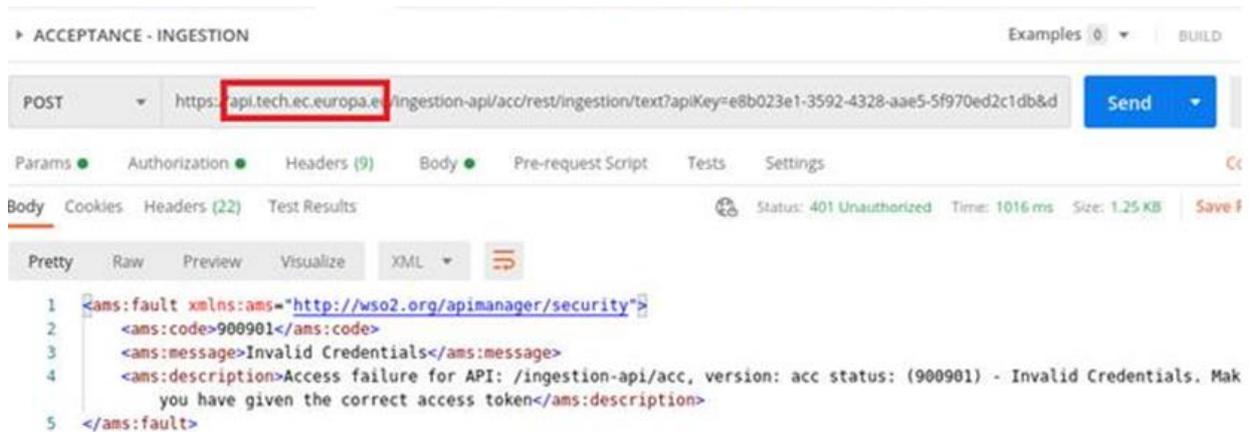
Name	Description
actor_classification__id number (query)	Thr id of the actors classification
actor_description string (query)	Actors description field
actor_name string (query)	Actors name
actor_type__id string (query)	The type of the actor
actor_url string (query)	Array Of Actors URL
alternative_actor_name string (query)	Short Actor Name
contact_details string (query)	The contact details of Actor
linked_actor string (query)	An Array of ids of the actors tha linked with actor
location string (query)	The location of the Actor
suborganization string (query)	Array of ids that connected with the Actor

Figure 7 - Example on how to invoke the API



The Aggregator should note that in case the token is not valid, the gateway will provide a message as shown in Figure 8.

Figure 8 - Invalid token



2.6. Upload of the datasets

Regarding the platform ID, each aggregator is assigned a unique number. All data that will be sent to the server will check the aggregator ID. If the ID already exists in the database, it will automatically overwrite the records. As mentioned in [section 2.2 Format and Structure of API JSON](#), please consult the list below for the known platform IDs. If your aggregator is not listed here, please get in touch with us on cmisa@trasysinternational.com.

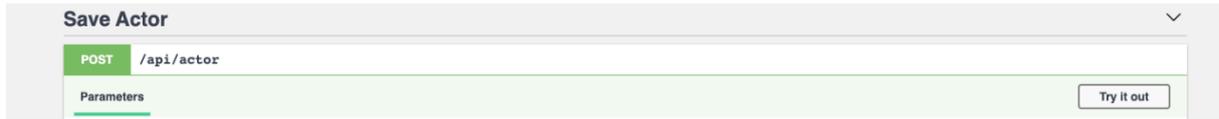
- ATI : **platform ID 1**
- DIH : **platform ID 2**
- EEN : **platform ID 3**
- ECCP : **platform ID 4**

- KET4CP : **platform ID 5**

For each submitted data, the system will respond back with a message type to allow aggregators understand, whether their submission was successfully uploaded, rejected, etc. The API also keeps track of uploaded attempts and results. These results will be kept in the central database and will be available through a direct query to the database by CMISA admins, for trouble shooting purposes.

From the swagger file and for each case, aggregators can generate and see these messages by using the “Try it out” button and then execute option, as shown in *Figure 9*.

Figure 9 - Swagger – Try it out - Responses



The API provides feedback to aggregators in case of successful / unsuccessful operations via different error codes and descriptions. These types of errors are:

- **200 – Success OK:** Indicates that the user request was accepted successfully.
- **404 – Not Found:** Either there is no API method associated with the URL path of the request, or the request refers to one or more resources that were not found, as shown in *Table 3*.

Table 3 – Error 404 message

```
{
  "error": {
    "errors": [
      {
        "domain": "global",
        "reason": "notFound",
        "message": "Not Found"
      }
    ],
    "code": 404,
    "message": "Not Found"
  }
}
```

- **401 – Unauthorised**

Access to a Requester Pays bucket requires authentication shown in *Table 4*.

Table 4 - Error 401 message/ Unauthorised

```
{
  "error": {
    "errors": [
      {
        "domain": "global",
```

```
"reason": "required",
"message": "Login Required",
"locationType": "header",
"location": "Authorization"
}
],
"code": 401,
"message": "Login Required"
}
}
```

- **500 – Internal Server error**

- **409 – Conflict**

A request to change a resource, usually a storage.*.update or storage.*.patch method, failed to commit the change due to a conflicting concurrent change to the same resource as shown in *Table 5*. The request can be retried, though care should be taken to consider the new state of the resource to avoid blind replacement of another agent's changes.

Table 5 – Error 409 message

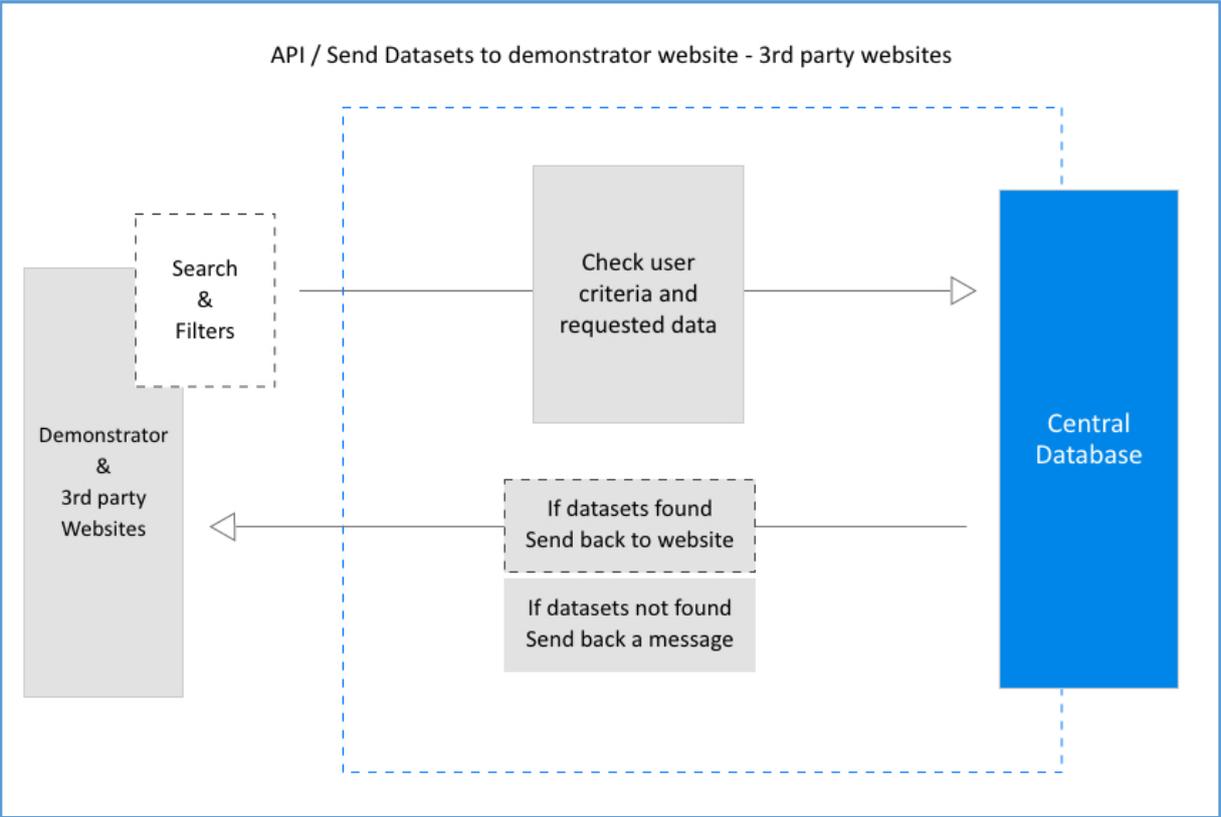
```
{
"error": {
"errors": [
{
"domain": "global",
"reason": "conflict",
"message": "Missing required data"
}
],
"code": 409,
"message": " Missing required data "
}
}
```

For any questions or clarifications, please use the Helpdesk Support via [cmisa\(at\)trasysinternational.com](mailto:cmisa(at)trasysinternational.com).

3. RETRIEVING DATASETS FROM CENTRAL CMISA DATABASE – FOR ALL 3RD PARTY WEBSITES

The datasets of the central database are accessible via an API⁹. This API is the same for both the demonstrator website operated by the Commission and 3rd party websites and can expose the same datasets with common characteristics. The demonstrator website can show the number of datasets available as well as the way of filtering, previewing and exporting the data. It has been developed to demonstrate the value of re-using datasets by 3rd party websites. Any 3rd party website may use the same API, which will give access to CMISA datasets so that they can enrich their websites with CMISA data. The overview of the activities is presented in *Figure 10* below.

Figure 10 – Overview: the process of API sending datasets to demonstrator website



3.1. Querying the database

The web API allows the querying of the central database and the retrieval of structured data. The structure of this JSON file is the same as the JSON file aggregators use (*Table 1*). The **Common Data Model** is available in various formats to be used by the websites to share / publish their datasets on innovation supporting actors in a standard and harmonised way. Users can access it on Joinup by following this link: <https://joinup.ec.europa.eu/collection/cmisa/solution/cmisa/release/101>. The Common Data Model represents a **robust semantic model** that promotes Innovation Supporting Actors by facilitating data exchange, sharing and publishing of data.

⁹ The collected data is also accessible via an embeddable map and downloadable in Excel format.

In case the data is not available or the server has an internal error the following messages will be displayed:

- HTTP Error 404 (Not Found)
This error will be displayed when a resource on the server does not exist or it is a broken link, or mistyped url, etc.
- HTTP Error 500 (Internal Server Error)
This error appears when the server encounters some form of internal error.

The API provides all necessary information to the demonstrator website related to user's actions. In more details, a sample of the query parameters for each option will be the following:

Query parameters Samples

<p><u>All options selected:</u></p> <p>domain.eu/api/results?q=searchterm<String>&countries=[1,2,3]<ID[]>&<String>actortype=[1,2,3]<ID[]>&<String>competences=[1,2,3]<ID[]>&<String>trl=[1,2,3]<ID[]>&<String>marketsectors=[1,2,3]<ID[]></p>
<p><u>Search with term:</u></p> <p>domain.eu/api/results?q=searchterm</p>
<p><u>Countries – Filter:</u></p> <p>domain.eu/api/results?<String>&countries=[1,2,3]<ID[]></p>
<p><u>Actor type – Filter:</u></p> <p>domain.eu/api/results?<String>actortype=[1,2,3]<ID[]></p>
<p><u>Technical Competences – Filter:</u></p> <p>domain.eu/api/results?<String>competences=[1,2,3]<ID[]></p>
<p><u>Focus on TRL – Filter:</u></p> <p>domain.eu/api/results?<String>trl=[1,2,3]<ID[]></p>
<p><u>Market Sectors – Filter:</u></p> <p>domain.eu/api/results?<String>marketsectors=[1,2,3]<ID[]></p>

4. ANNEXES

4.1. Annex 1 – CMISA JSON Template payload



CMISA sample.json

4.2. Annex 2 – Information on how to use Controlled Vocabularies in JSON



SC373_D06.02
Aggregators Manual_