Enabling Data Enrichment Pipelines for AI-driven Business Products and Services

HORIZON-CL4-2021-DATA-01-03

D3.1 enRichMyData integrated toolbox v1

Work Package 3

Type of document:	Report
Dissemination level:	Public
Lead beneficiary:	CS
Authors:	Cosmin Udroiu, Cosmin Cara, Laurentiu Nicola (editors), all other technical partners contributors
Version:	1.0
Due Date of document:	30.09.2023
Delivery Date of document:	29.09.2023

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

Document History

Version	Date	Contributor	Comments
0.1	05.05.2023	Cosmin Udroiu (CS)	Structure and ToC
0.2	10.09.2023	Technical partners	Content to sections
0.3	20.09.2023	Cosmin Udroiu (CS)	Updates after technical review
1.0	29.09.2023	Dumitru Roman	Final QA and final version submitted

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

Executive summary

This document represents the user manual of the enRichMyData Integration Toolbox. The document presents the hardware and software needed to execute the toolbox but also how the toolbox can be installed and configured.

Once the toolbox is installed, the user can find in the document instructions how to login into to the web interface, access the monitoring dashboard, add Docker images and define the processing components, how to perform user administration and how to create and execute workflows.

Besides the access through the web interface, the toolbox can also be accessed programmatically via its REST API interface, allowing integration with third-party systems.

At the end of the document, a concrete example of how a workflow along with all needed components can be developed and executed.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

Contents

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

List of figures

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

List of abbreviations

Abbreviation	Description
EMD	enRichMyData
TAO	Tool Augmentation by User Enhancements and Orchestration

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

1 Deliverable overview

1.1 Deliverable purpose, scope and context

This document represents the Software User Manual of the enRichMyData integration platform. It provides users with guidance on how to install the platform, how to use it from the web interface but also how to use it programmatically, from third party applications, using its REST API. Additionally, it provides an example on how to add all components (docker images, processing components, workflow, etc.) from scratch and executing a concrete pipeline.

1.2 Target audience

This deliverable is specifically designed for the stakeholders of the enRichMyData (EMD in short) platform, encompassing professionals including developers, researchers, data workers, and other technical staff.

1.3 Deliverable structure

This deliverable contains the following sections:

- Section 3 describes the purpose of the integration platform.
- Section 4 presents the hardware and software configuration used for running the platform.
- Section 5 is the operations manual that describes how to install and configure the platform, how to access and use the platform, both via the web interface but also via the REST API that it exposes.
- Section 6 presents an example with a pipeline created from scratch.

1.4 Applicable documents

- D1.2 - enRichMyData architecture
- D2.1 - enRichMyData tools

1.5 Updates of this deliverable

This deliverable corresponds to the current version of the enRichMyData integration toolbox. It will be evolved during the project according to the evolutions of the toolbox.

1.6 Definitions and interpretations

N/A

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

2 Purpose of the software

The EMD integration toolbox is derived from the "Tool Augmentation by User Enhancements and Orchestration (TAO)" toolbox that was initiated by the European Space Agency (ESA) with the aim to provide extension and reusability (i.e., integration) of commonly used toolboxes in the Earth Observation domain.

The TAO toolbox core started to be refactored in the context of enRichMyData project in order to contain a generic core component that is not coupled anymore to Earth Observation data model and that would facilitate the integration of EMD tools and their composition in the desired pipelines.

This framework should allow for processing composition and distribution in such a way that end users could define processing workflows by themselves and easily integrate additional processing components.

EMD Integration platform provides a means for the orchestration of heterogeneous processing components and libraries in order to process EMD specific data. This can be achieved in following steps:

- Preparation of resources like the execution nodes, the processing components, data sources and sinks.
- Definition of a workflow as a processing chain.
- Execution of the workflow.
- Retrieval / visualization of the results.

EMD integration framework will contain, by default, an initial set of tools and applications that can be combined into workflows. This default set of tools and applications will be updated during the project's lifetime with newly developed ones. Additionally, users can add their own developed tools, applications or scripts to the framework.

The execution of the workflows can be performed on a single local machine or distributed on several machines that were defined as execution nodes in the EMD frameworks, using one of the resource managers supported (Torque, Slurm, SSH, Kubernetes).

3 Operations environment

In the next sections the principal configuration of the hardware and software components required for the functioning of the system are described.

3.1 Hardware configuration

The master node (or the workstation – in the case of a local service), where the actual TAO services are executing, shall have the following minimum hardware configuration:

- Processor: x64 architecture, quad core, 2.0 GHz
- Memory: 16GB RAM DDR3
- Storage: 200GB

Each execution node, where the tools instances are executed, shall have at minimum the following hardware configuration:

- Processor: x64 architecture, dual core (with SMT/HT) or quad core, 2.0 GHz
- Memory: 24GB RAM DDR3
- Storage: 200GB for processing nodes

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

A common storage space, accessible from all the nodes (both master and execution nodes) should be provided and its size depends on the input and intermediate data produced during the execution of the workflow steps, typically from several hundreds of GB to several TB size.

NOTE: The above values are the minimum requirements of the systems but they may vary very much, depending on the size of data that will be used (requiring for instance more storage space), on the algorithms involved (an algorithm that creates a huge model of several GB that needs to be loaded in the memory might easily require 128 GB or more of RAM) and on the computations performed (an intensive computing algorithm might require more processors in order to have reasonable computing times).

3.2 Software configuration

The following auxiliary software should be installed on the machine before deploying the EMD integration platform:

- Operating System: Windows 10, Windows 11 or Centos 7 for standalone version and Centos 7 for distributed version
- Java Virtual Machine >= 14.x
- Apache Tomcat servlet container >= 8.x
- PostgreSQL >= 12

Nevertheless, if these software applications are not installed, the EMD toolbox installer will try to install them.

4 Operations manual

4.1 Set-up and initialization

4.1.1 Installation

This paragraph presents the first steps to be performed for installing the TAO platform.

The installation is performed in an automatic manner, using the installation script “install_emd.bat” for Windows and “install_emd.sh” for Linux. During the installation, the interactions with the user are limited as much as possible, the user is prompted only when absolutely needed such as demands for directory locations or passwords.

In order to start the installation, the following command should be executed in the command line:

- For Windows (only the standalone mode of the TAO system):
  ```
  cd <INSTALL_SCRIPTS_DIRECTORY>
  install_emd.bat
  ```
- For Linux:
  ```
  cd <INSTALL_SCRIPTS_DIRECTORY>
  ./install_emd.sh
  ```

The installation scripts handle the following operations:

- Java installation (if missing)
- Postgres installation (if missing)
- Database installation and configuration

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

- Tomcat installation and configuration
- Resource manager installation (Torque or SLURM in the case of platform mode installation)
- EMD Toolbox server and EMD Toolbox Web UI deployment and configuration

A list of available parameters of the installation script is available using the command:

```
./install_emd.sh --help
```

The script can be executed with no parameters and in this case the default values for the configurable options are used.

4.1.2 Configuration

The configuration of the EMD Integration framework is performed via the “tao.properties” file. In this file the following settings can be configured:

- server.port - the server port where EMD services are deployed (ex. 8880).
- proxy.type – if the system is behind a proxy, this key specifies the type of the proxy. The type of the proxy can be one of the following values: direct – no proxy is present, and a direct connection is available, http – a proxy for high level protocols such as HTTP or FTP is used, socks – a SOCKS (V4 or V5) proxy is used.
- proxy.host – the hostname or IP address of the proxy is specified.
- proxy.port – the port for the proxy connections.
- proxy.user – the user for the proxy.
- proxy.password – the password for the proxy.
- spring.datasource.url – the EMD database connection URL in the format jdbc:postgresql://SERVER:PORT/DB_NAME?stringtype=unspecified, where SERVER is the server name or IP (default = localhost), PORT is the port for the postgresql server (default 5432) and DB_NAME is the data base name (default emddata).
- spring.datasource.username – the EMD database connection username (default tao).
- spring.datasource.password – the EMD database connection password (default tao).
- tao.drmaa.sessionfactory – the EMD defined session factory for DRMAA. This is the Java class name that implements the DRMAA interface (default ro.cs.tao.execution.local.DefaultSessionFactory but can be also ro.cs.tao.execution.drmaa.slurm.SlurmSessionFactory or ro.cs.tao.execution.drmaa.torque.TorqueSessionFactory, for example, for resource manager Slurm or Torque respectively).
- master.docker.mappings – the binding between the host and the docker container of the working directory, configuration, input and output directories (the first value is the host directory and the second is the containner directory). This allows mapping various directories into the Docker container when executing “docker run” commands. An example would be a value like { "workspace.mount": "/mnt/tao/working_dir:/mnt/tao", "temp.mount": "/home/tao/workdir:/mnt/workdir", "cfg.mount": "/mnt/tao/cfg:/mnt/cfg" } that defines the mappings for workspace, working or configuration directories.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

4.1.3 First administration steps

In the case of using Kubernetes as a Resource Manager through the DRMAA API, the Kubernetes cluster needs to be created a priori, as this cannot be done automatically by the EMD framework.

4.2 Getting started

The EMD system can be accessed at https://<server_name>:<port>/

using one of the following Web browsers:

- Firefox version 30 or higher
- Chrome version 14 or higher

The first page displayed when accessing the EMD framework (https://<server_name>:<port>/ui/login.html) is the login page as shown below.

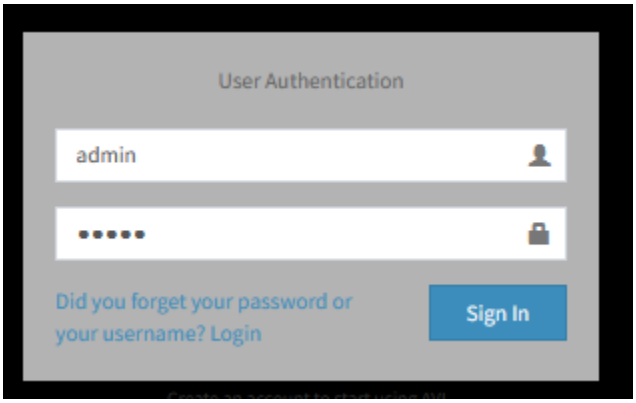


Figure 1: User authentication

In order to access the EMD framework, the user must be registered on the application. The user has to authenticate with username and password on the login page.

Any access of anyURL in the web application when the user is not authenticated already, is followed by a redirection to the login page.

4.3 EMD toolbox interaction via web interface

4.3.1 Monitoring dashboard

4.3.1.1 System monitoring

This interface allows the users to view the current status of a node (if it is available or offline), the execution status of a certain workflow (not running, executing, paused completed or in error), the status of a certain task in the workflow (awaiting to be executed, executing, finished or error) or the quota status for that user (percentage and the value of the disk space used).

It presents also general information about the platform/system status and synthetic overview about resources/load/nodes/activity.

The dashboard facilitates access to different parts/components of the system/platform through menus.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

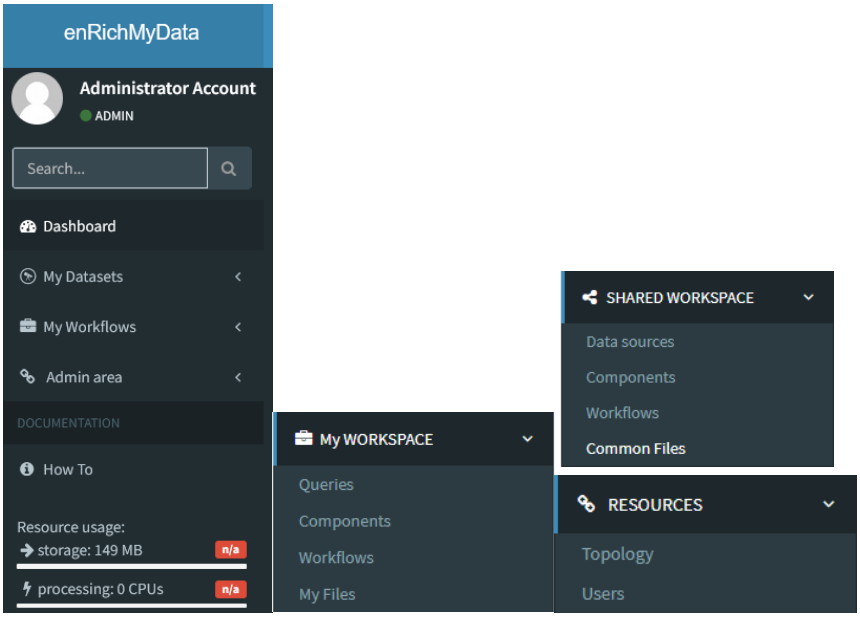


Figure 2: Toolbox menus

The Dashboard is the first page that is displayed to the user after the login. In the dashboard, the following information is available:

- Recent notification messages.
- The status of the EMD master node, such as the last start time, the number of processors of the master node, the memory used and a graph displaying its processor and memory usage.
- A summary information about the load of each processing node defined in the EMD framework.
- A list of present workflows running and the percentage of completion.

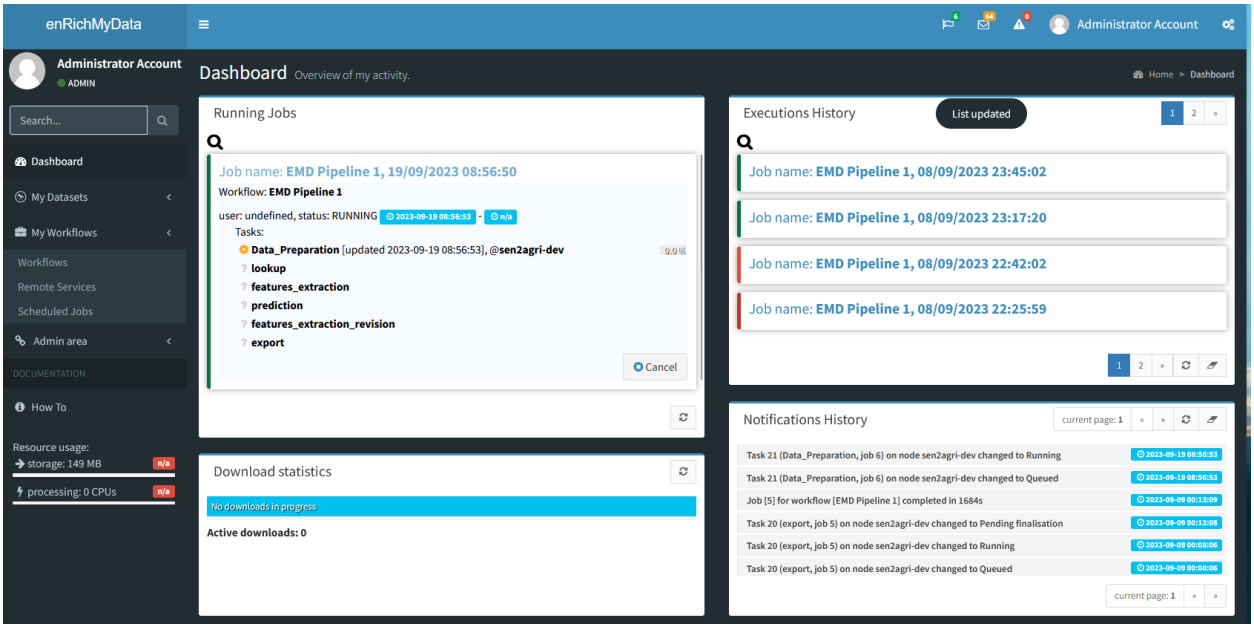


Figure 3: Dashboard view

4.3.1.2 Notifications in the EMD framework

EMD framework offers a notification mechanism that informs the user about the status of the last operations performed. These notifications are displayed as popups in the upper right side of the interface. The notifications can be general information messages (Figure 4) or error messages (Figure 5).

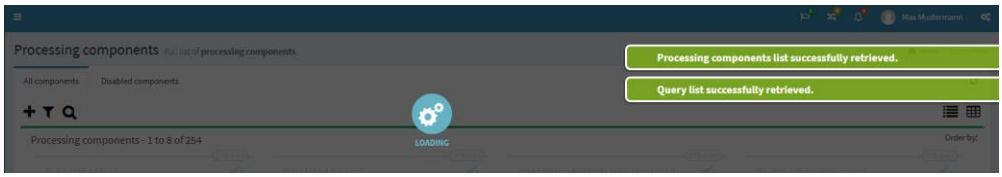


Figure 4: Showing notification messages

The error messages are also presented to the user upon validation of inputs.

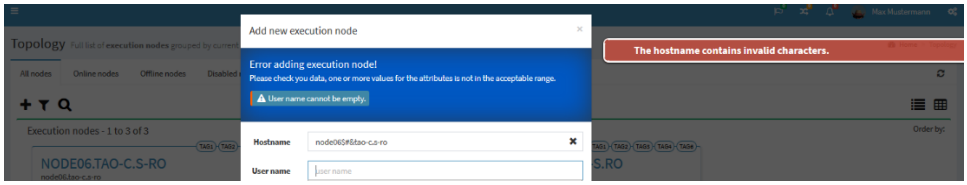


Figure 5: Showing the error messages

Notifications are also displayed during workflow runtime in the Dashboard main page, as presented in Figure 3".

4.3.2 Toolbox Docker images

Normally, the executable tools (applications) are packed into Docker images. A docker image might contain one or several tools or applications. These applications are the ones that are actually executed in the pipelines, from the Docker containers.

The EMD framework allows the user to register Docker images containing the tools that could be used inside pipelines. When the user accesses the Toolbox Image Descriptors page, a list of existing Docker images is displayed.

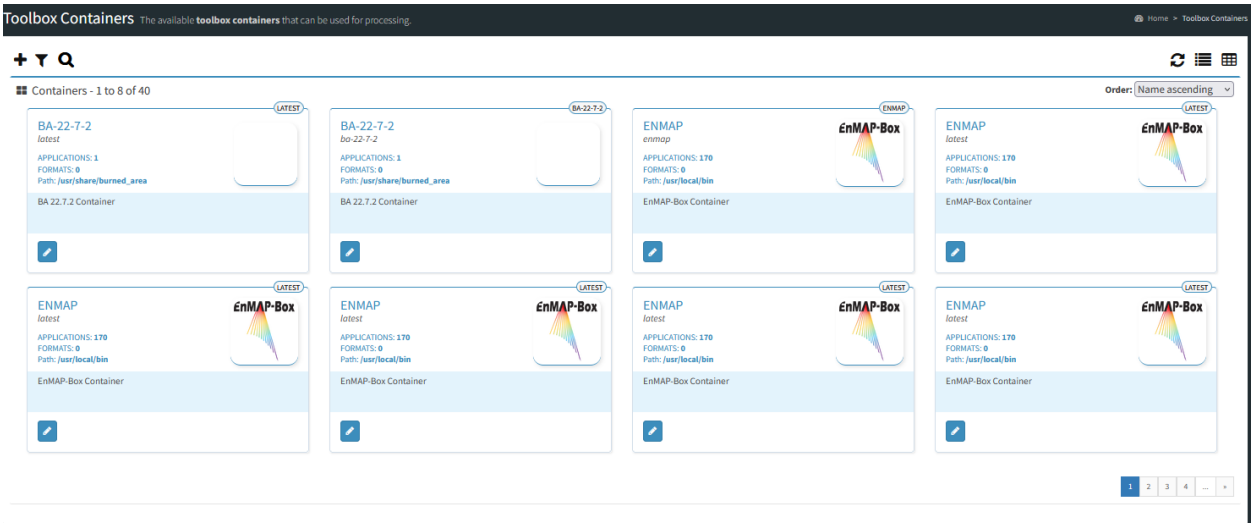


Figure 6: Toolbox Docker image descriptors page

The user has the option to edit existing delete or add new images.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

In order to add a new Docker image, the user must submit necessary information as shown in Figure 7:

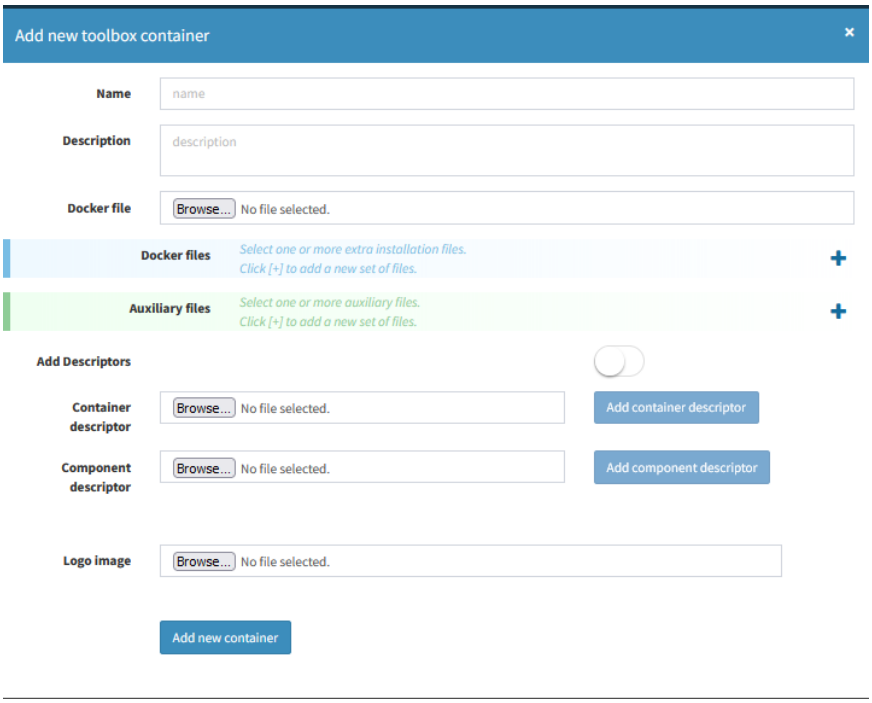


Figure 7: Add a new docker image

In the page shown in Figure 7, the user will need to provide the following information:

- The name of the Docker image. This is the name that is actually used for the name of the image in Docker itself (as seen when running the command “docker images”).
- A short description of the Docker image.
- The Dockerfile file that is used to create the Docker image.
- The docker files that need to be added into the Docker image when it is created using the above-mentioned Docker file (as when the “docker build” command is run).
- Logo image – a png file that will be used in the toolbox to be displayed for this Docker image.

In order to create also a Docker image descriptor, the user will have to press “Add container descriptor” button. The Docker image descriptor is what is registered in the toolbox database and contains information about the Docker image registered in Docker, a name to be displayed to the user in the Web Interface (may or not be the same as the registered Docker image name), the tool executable(s) that are encapsulated in that docker image and the path where they are located inside the docker image.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

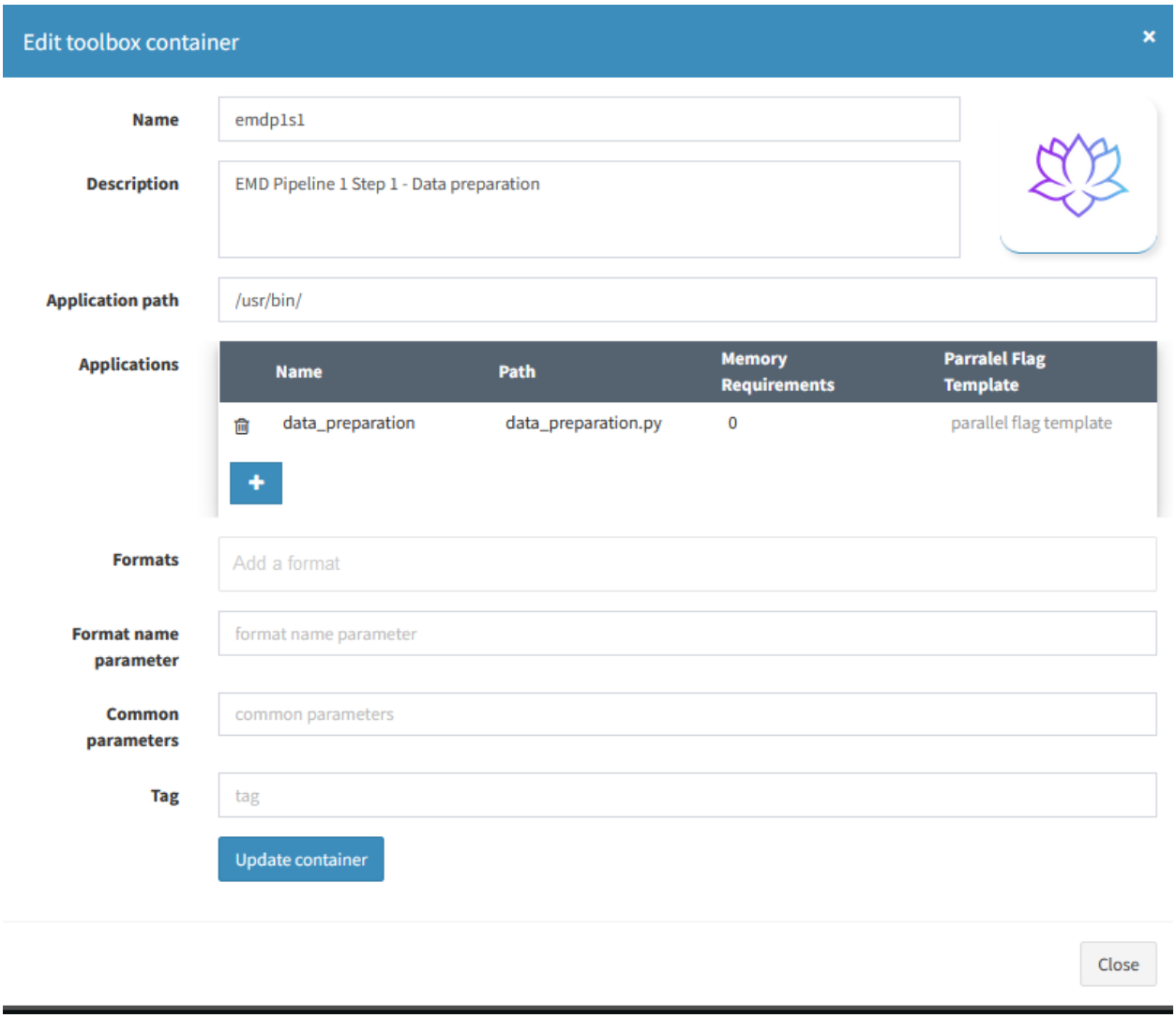


Figure 8: Adding the Docker image descriptor

At this stage, are defined only the tools executables that are contained in the Docker image (name, path in the docker image) but not how these executables behave when added in a workflow or when they are executed (parameters, inputs, outputs etc.). This is performed later, as described in the next section.

4.3.3 Processing components

The processing components are wrappers describing the attributes of the tools that can be used in the construction of workflows. These attributes are the name of the component, the executable to be invoked, input and output parameters, etc.

In the “Components” section, which is located under the “Admin area” of the menu, the user will have access to the page presenting a list of processing components.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

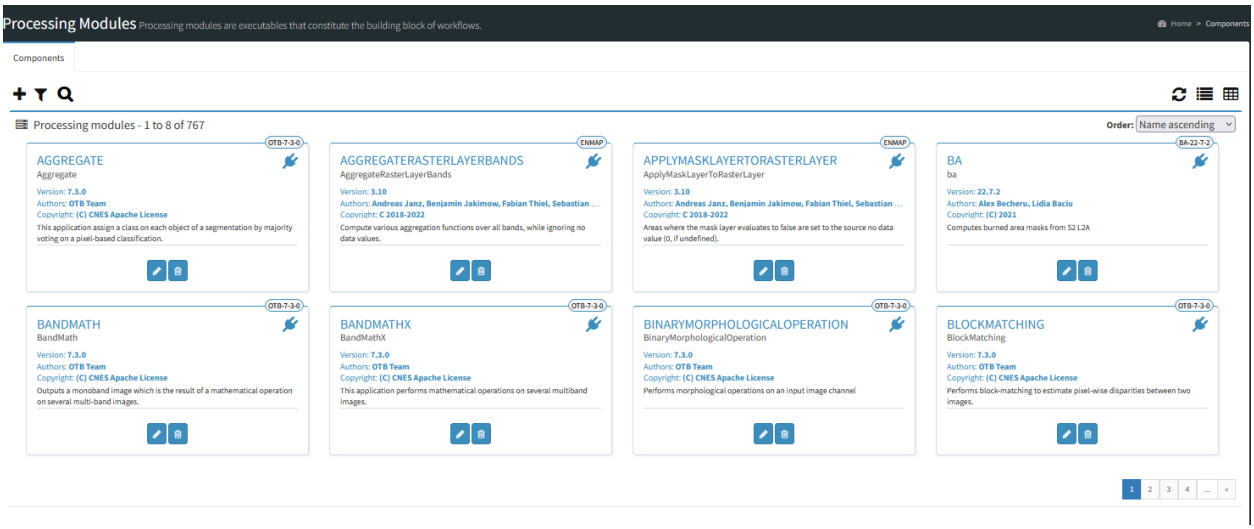


Figure 9: Processing components page

In this page, the user can manage processing components.

To add a new component, select the “+” button. This will allow opening the “Create a processing module” window as shown in Figure 10.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

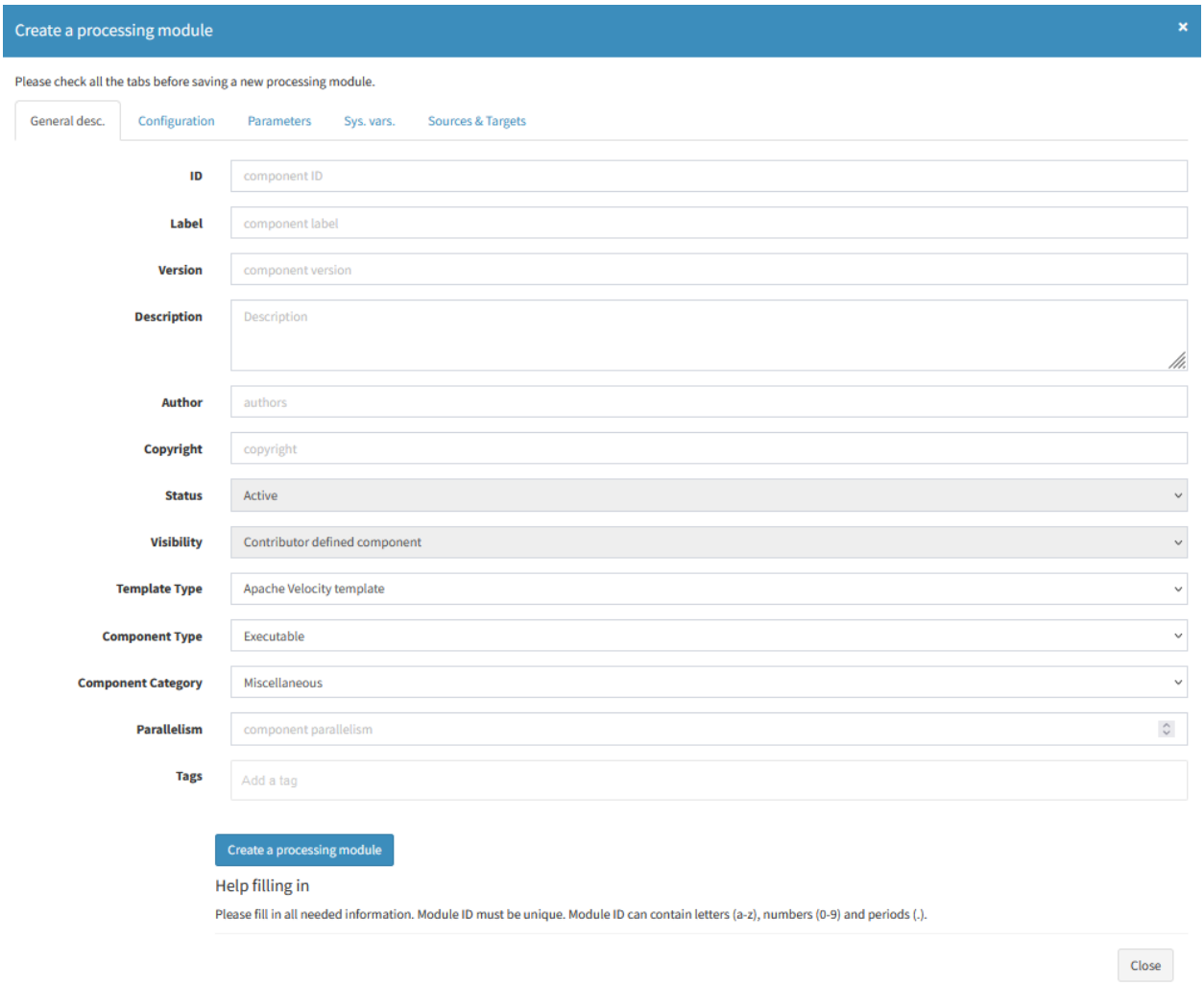


Figure 10: Creating a new processing component

In the processing component window shown in Figure 10, the user needs to fill in various fields about the component. The required fields in the General Description tab are:

- ID – a unique ID in the toolbox of this processing component.
- Label – a short description of the processing component that will be displayed in the web interface.
- Version – the current version of this processing component.
- Description – a detailed description of this processing component.
- Other fields that are optional

The “Configuration” tab describes mainly the template for the command line that is used for invoking the tool described through this processing component descriptor.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

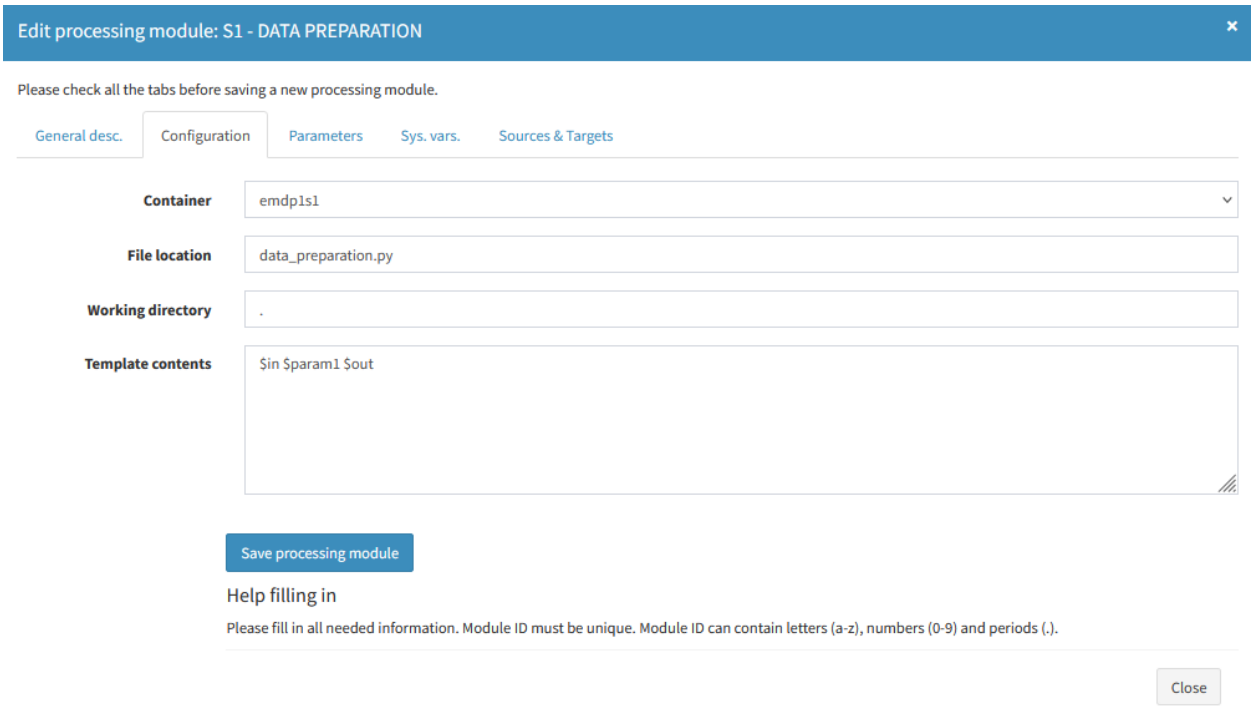


Figure 11: Processing component configuration

In this page, the user will have to fill:

- The Docker image – represents the docker image that encapsulates the executable that is described by this processing component. This can be selected from a dropdown control listing all the Docker images registered in the toolbox.
- The File location – this is the path of the executable in the Docker image as defined when it was added to the Docker image descriptor (see Figure 8: Adding the Docker image descriptor).
- The working path inside the Docker container that is used for saving intermediate files.
- Template contents – describe the command line template for the tool added. This describes actually the invocation arguments of the application that will be invoked. The template should contain the inputs and the outputs but also the parameters from the command line. Whenever a parameter starts with $ this is used as a placeholder for an input or a parameter that will be replaced during execution with concrete values (paths or other values). To be noted that there is made a distinction between inputs, outputs, parameters and fixed values. Thus:
- An input is a file that is defined with a placeholder (ex. $in) and describes what is received by this tool which means that when defining a pipeline, it can accept to connect with outputs from other components. This is also referred to as a "source" of the processing component.
- An output is a file that is defined with a placeholder (ex. $in) and describes what is produced by this tool which means that when defining a pipeline, it can be the input to other components. This is also referred to as a "target" of the processing component.
- Parameters are some arguments of the tool that cannot be used to interconnect the component but that can be still changed by the user before execution. Some examples would be a parameter accepting either value "score" or "rho2".

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

- Fixed parameters are arguments in the command line defined by the “Template contents: that cannot be changed before the execution of the workflow.

The parameters can be defined in the page “Parameters”:

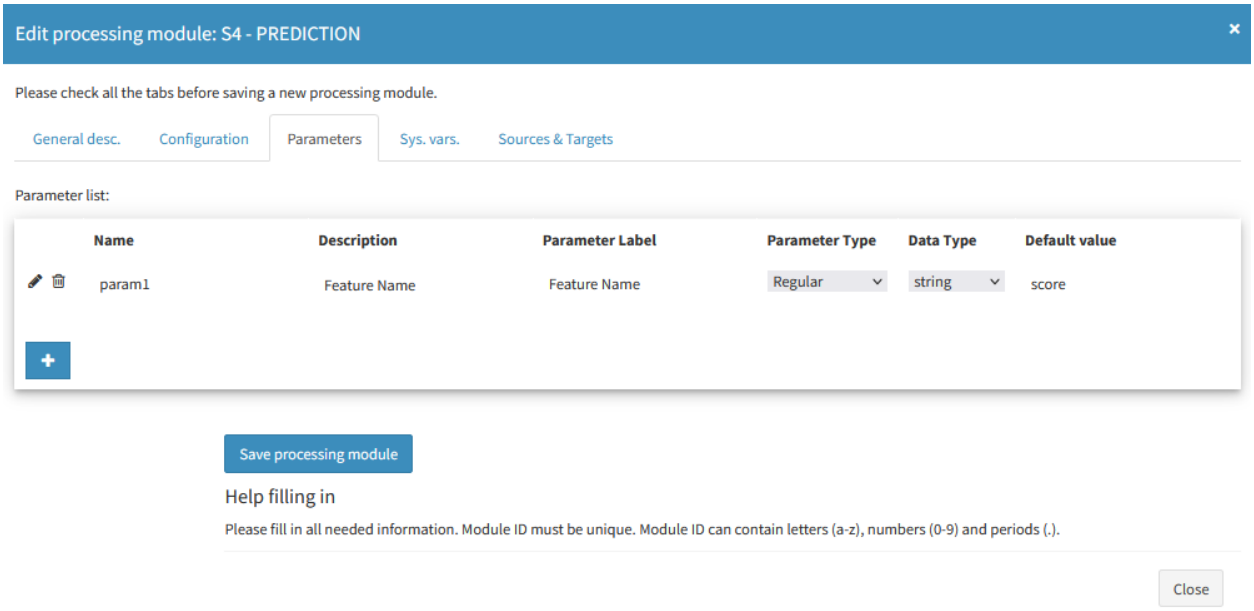


Figure 12: Processing component parameters

For defining the inputs and outputs, the “Source & Targets” tab should be selected:

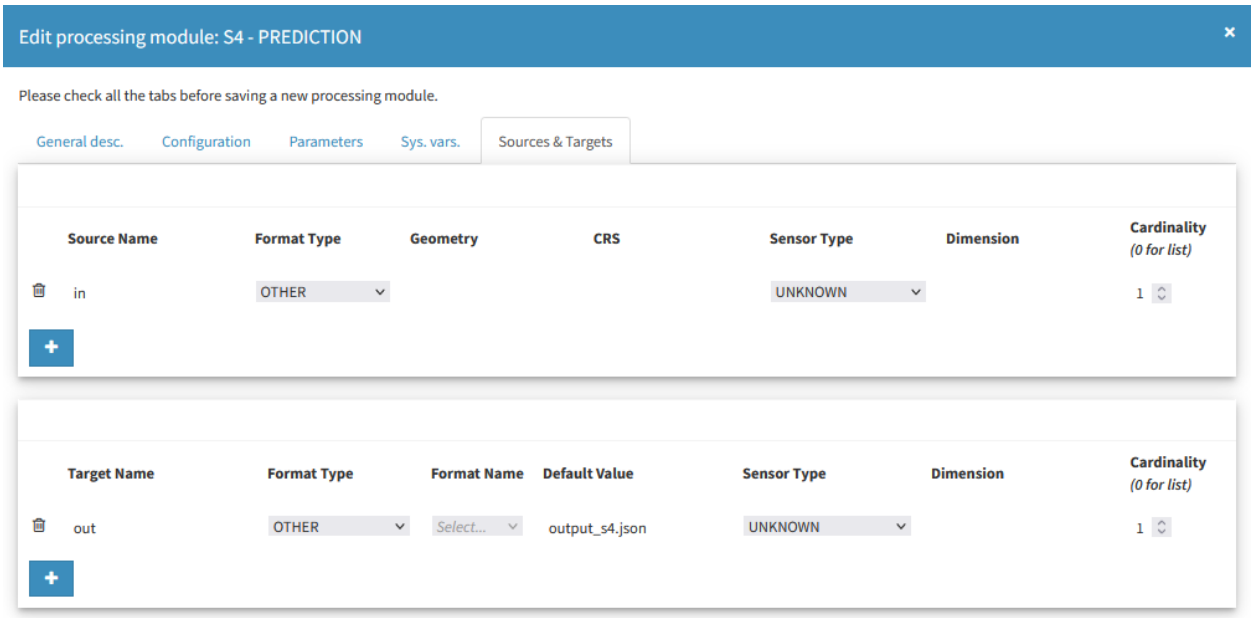


Figure 13: Processing components inputs and outputs

NOTE: The parameters, inputs and outputs are defined in the corresponding fields by removing the placeholder prefix ($).

For the editing, the user will be directed to a similar window as for new components definition.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

4.3.4 User administration

The user administration can be performed from the menu “Resources” -> “Users” option.

This submenu is displayed only for administrators.

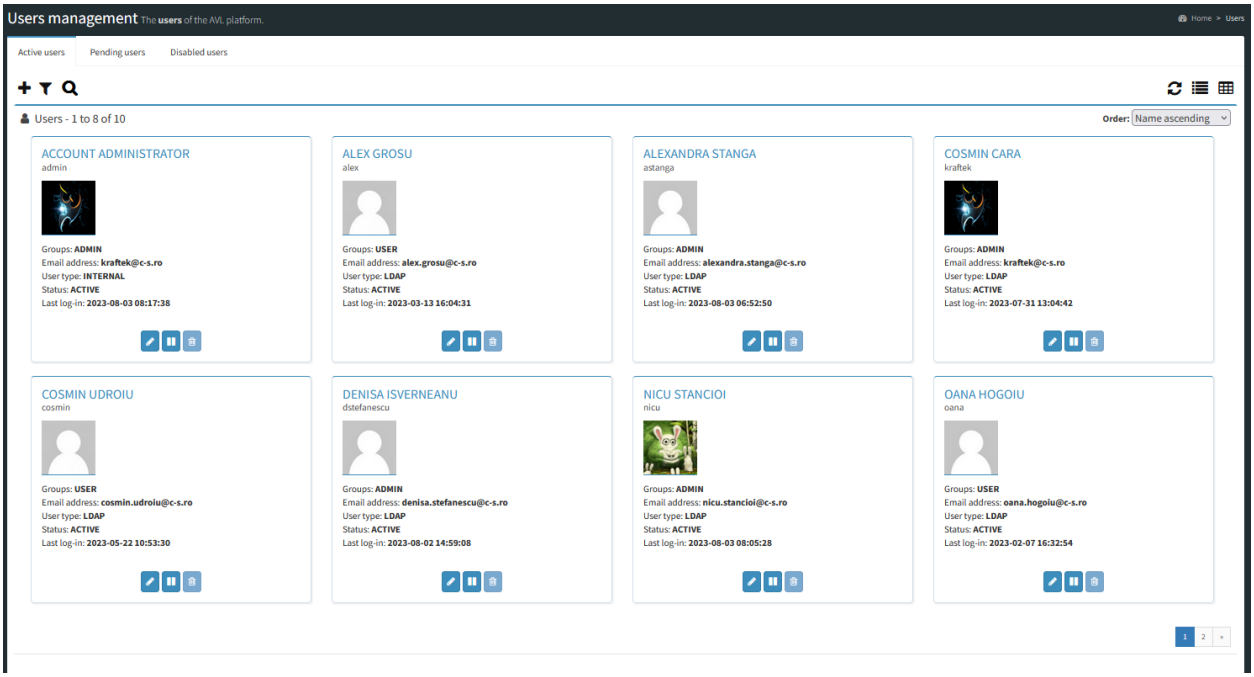


Figure 14: Users visualization

The Users page is illustrated in Figure 14 where available actions are:

- add a new user.
- edit user details.
- disable user.
- delete user.

For adding a new user, the following details should be provided:

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

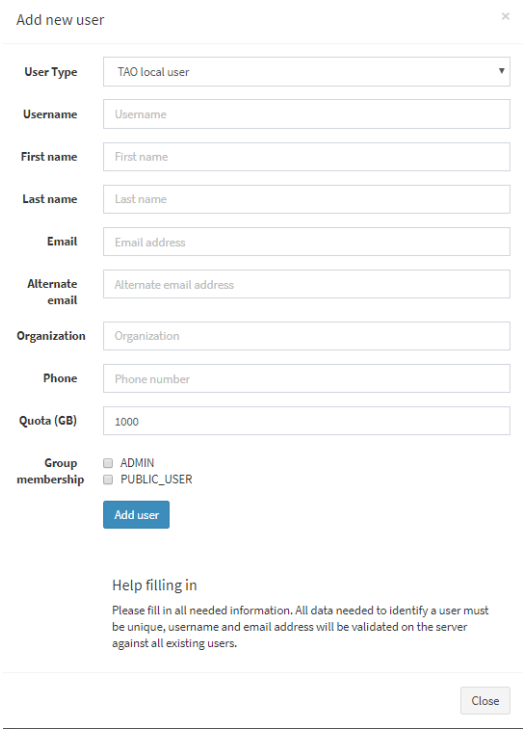


Figure 15: Adding a user

A non admin user can edit his/her details by clicking on the avatar from toolbar and then on the 'Profile' button.

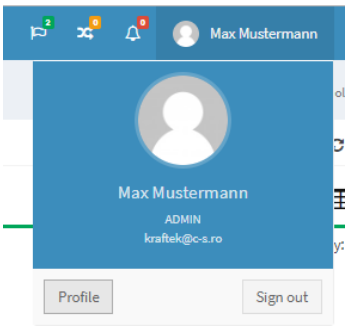


Figure 16: User's avatar options

The following pop-up window will be displayed:

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

Figure 17: User profile visualization

The user can edit only the inputs which become editable after 'Edit' button is pressed.

4.3.5 Workflow management

To start performing workflow operations, the user will have to access the section "Workflows" from "My Workflows" section. Selecting this option, the user will be redirected to the main workflows page where a list of existing (if any) workflows is presented.

On this page, the user has the option to create, delete or edit workflows.

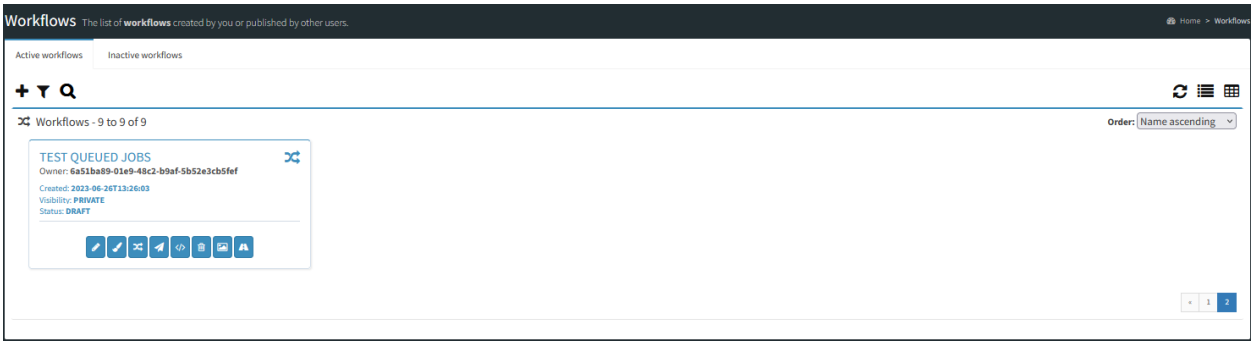


Figure 18: Workflows page

By pressing the "+" button, the editing page of a new workflow is displayed. On this page, the user has the option to drag and drop tools from the "Toolbox" menu available on the left side and then interconnect these components according to the definition of the pipeline.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

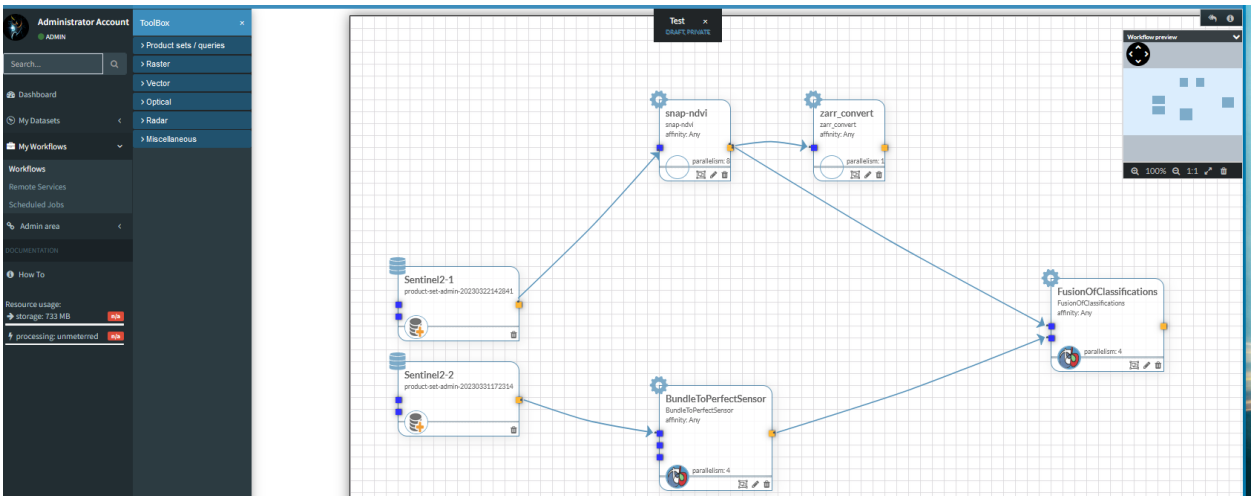


Figure 19: Workflow creation

Once a workflow is created, from the Workflows page the following operations become available:

- Edit description.
- Edit the workflow – in this case the user is redirected to the same page as for workflow creation.
- Clone the workflow. This will create a completely new independent copy of the original workflow. Any change or enhancement of this new copy will not affect in any way the original workflow definition.
- Start the execution, monitor the execution and see the results, when finished.
- Export the workflow as a script. At present it is possible to export into JavaScript, Bash and CWL scripts.

4.3.6 Accessing 'How to' section

The 'How to' describes the steps required to perform an action. For example, if the user has a question about how to add a new executing node, he/she will find the answer selecting the panel 'Workflows'.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

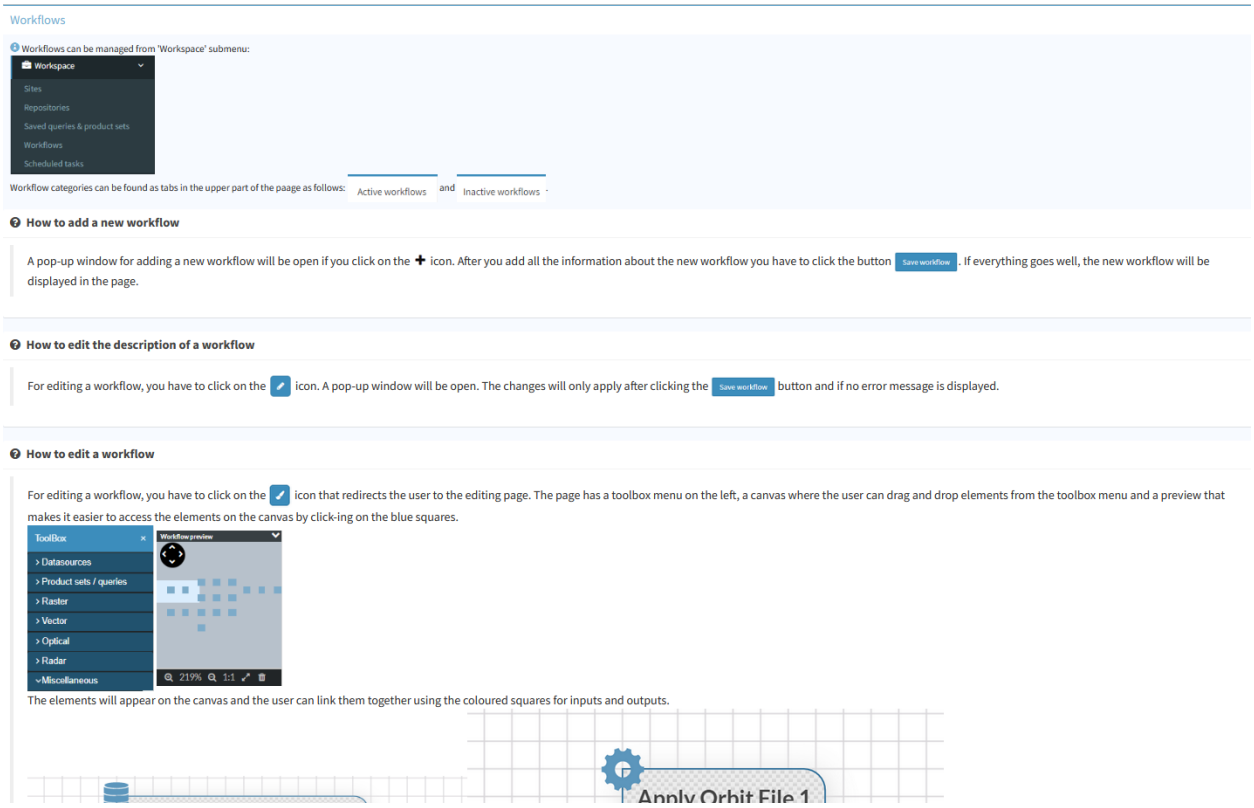


Figure 20: "How To" help page

This page can be seen as a short overview of this document 'EMD toolbox interaction via website' section.

4.4 EMD toolbox interaction via REST API

Besides interactions via the web interface, the toolbox allows accessing almost all the functionalities also via a REST API. The complete list of the operations via the API are provided via a dedicated documentation page in the web interface of the toolbox that is generated in OpenAPI format using Swagger from the REST API endpoints java doc. These documentation pages are accessible at the following address:

<server_name>:<server_port>/docs/swagger-ui/index.html

The API documentation is structured by categories of operation, each category containing the endpoints description. For each endpoint the following information is provided:

- The path and the access type of the endpoint.
- The description of the endpoint.
- The parameters of the endpoint with their name, type, description and if required or not.
- The description of the response generated (along with some examples).

Additionally, Swagger generated documentation allows the user to query endpoints directly from the documentation page by clicking on the "Try it out" button and then execute a request.

In the following sections wepresent the API for the main functionalities of the toolbox.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

4.4.1 Authentication API

4.4.1.1 Login

The following endpoint describes the login request.

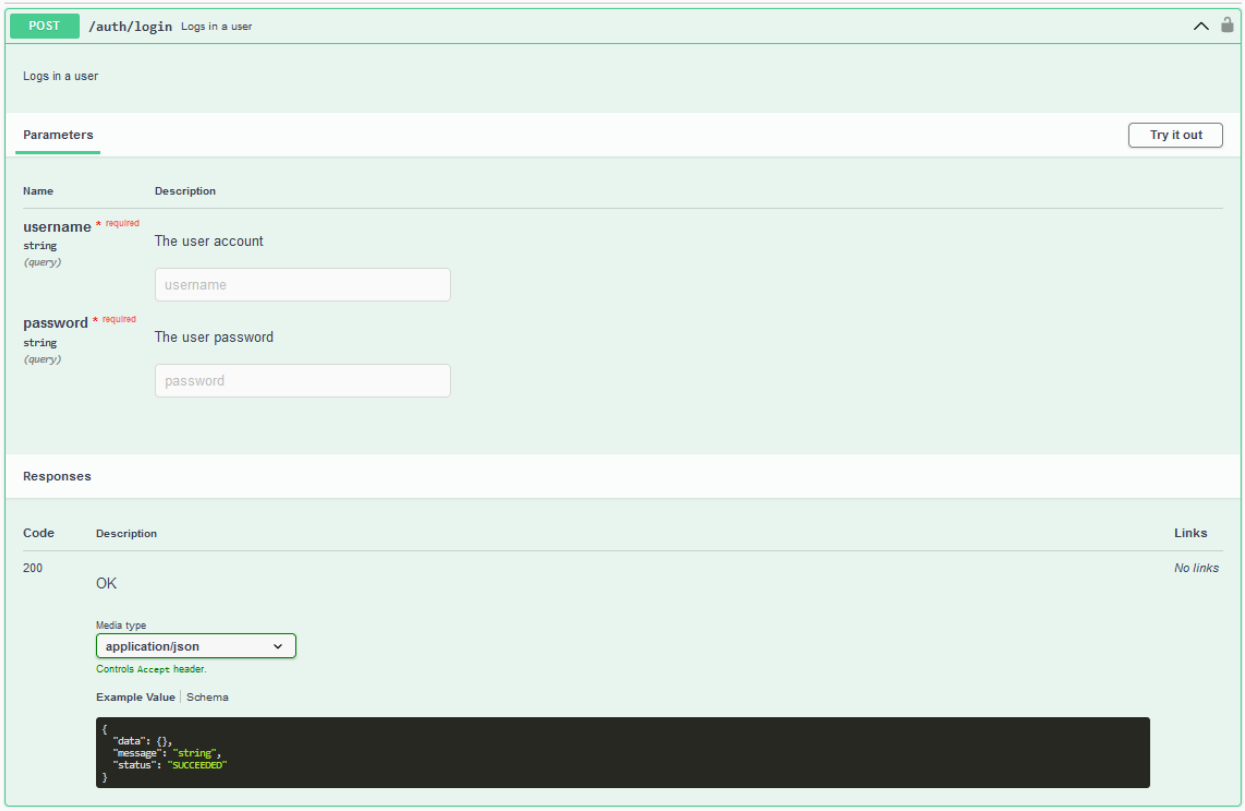


Figure 21 Login REST API endpoint

4.4.1.2 Logout

The following endpoint describes the logout request.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

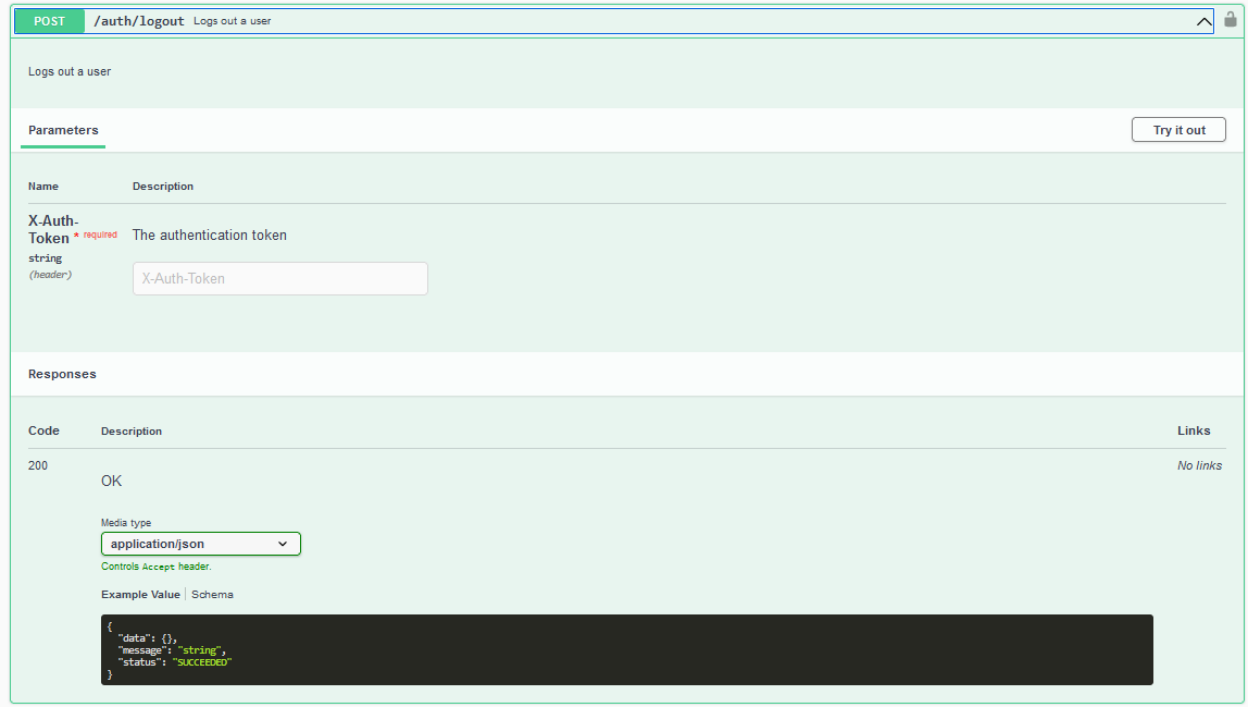


Figure 22: Logout REST API endpoint

4.4.2 Docker images integration API

4.4.2.1 List available images

The following endpoint describes the request for retrieving the docker images available in the system.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

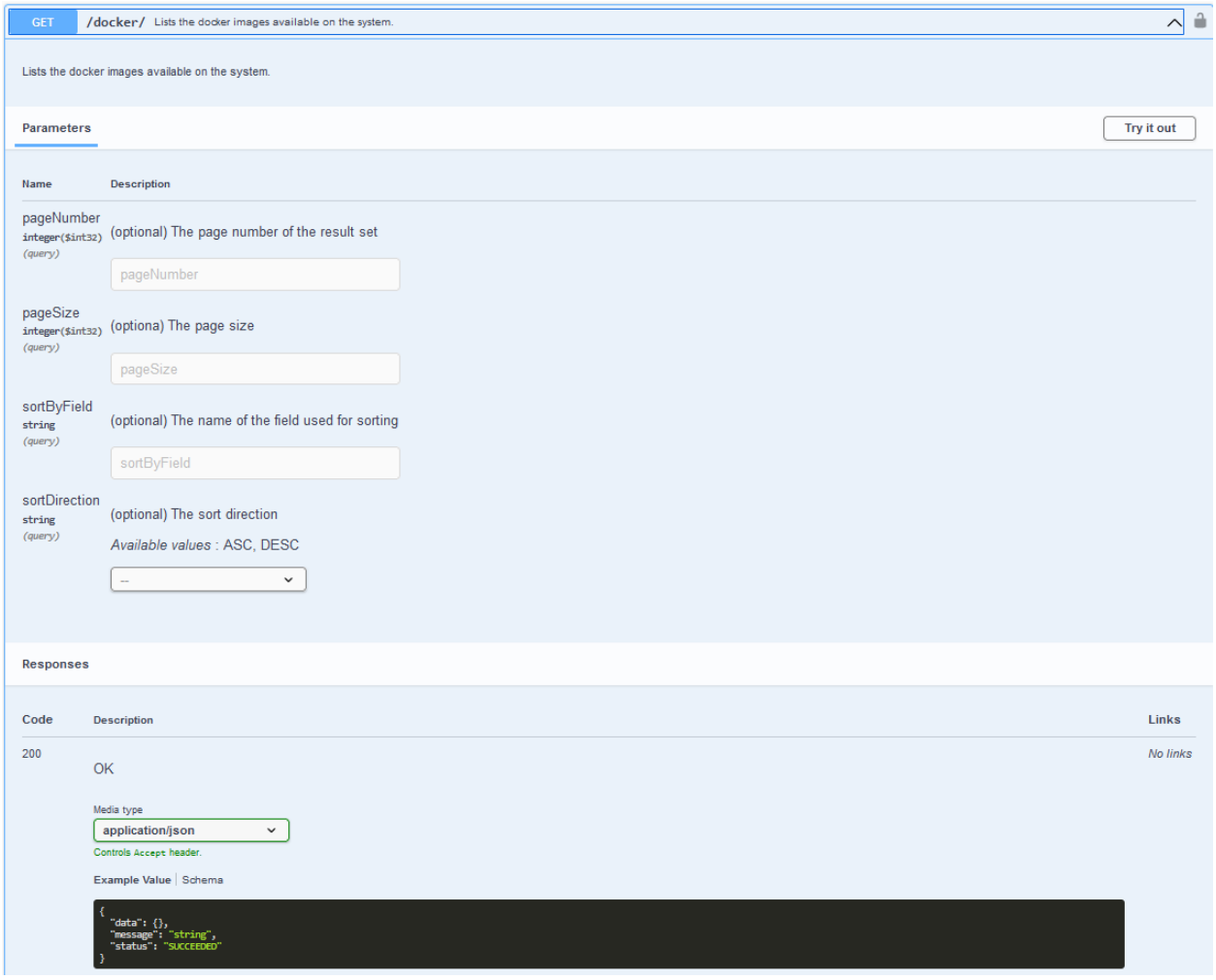


Figure 23: List available Docker images REST API endpoint

4.4.2.2 Add a new docker image

The following endpoint describes the request for adding a new docker image into the system.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

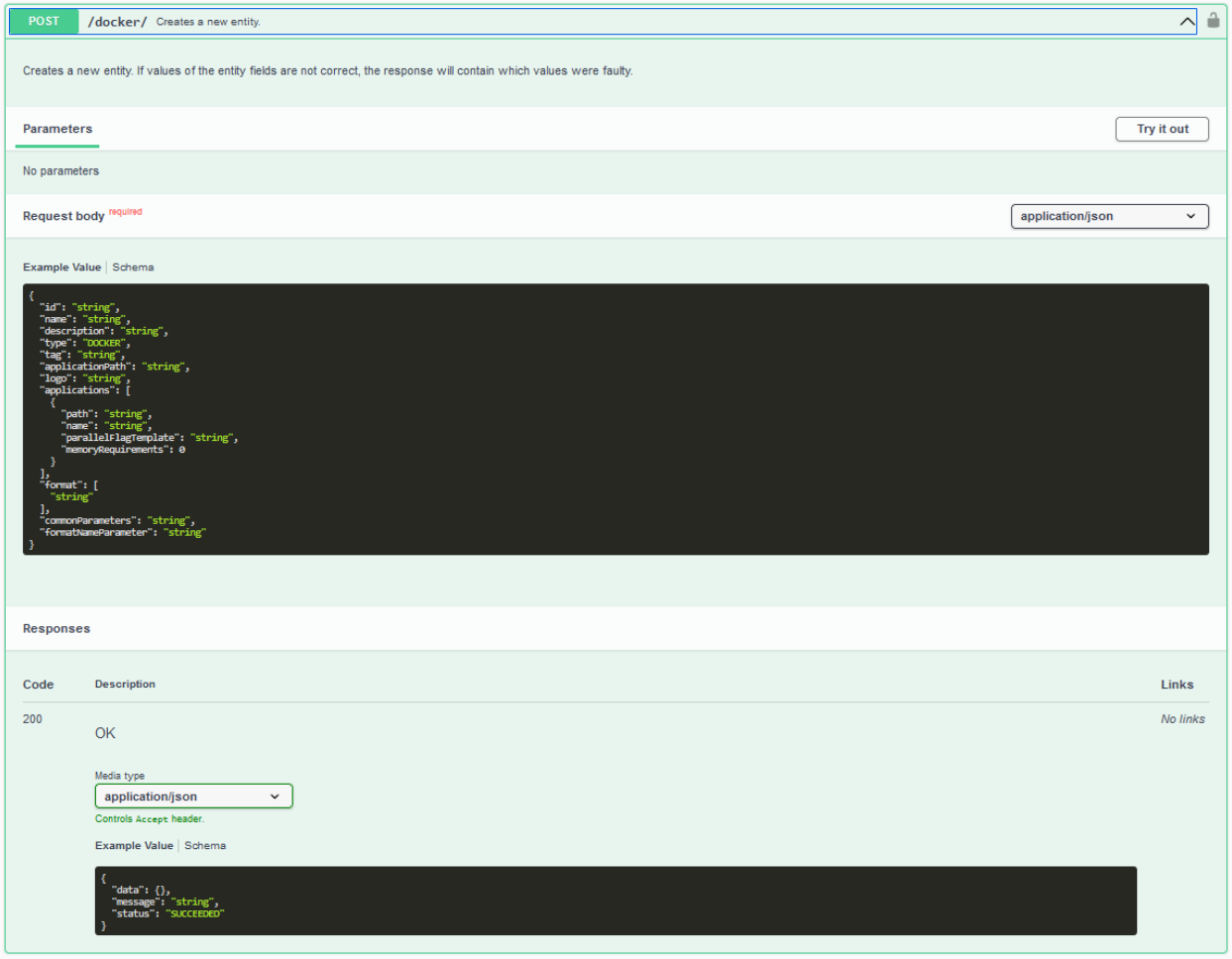


Figure 24: Add new Docker image REST API endpoint

4.4.2.3 Returns the details of a image

The following endpoint describes the request for retrieving the details of a docker image already existing in the system.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

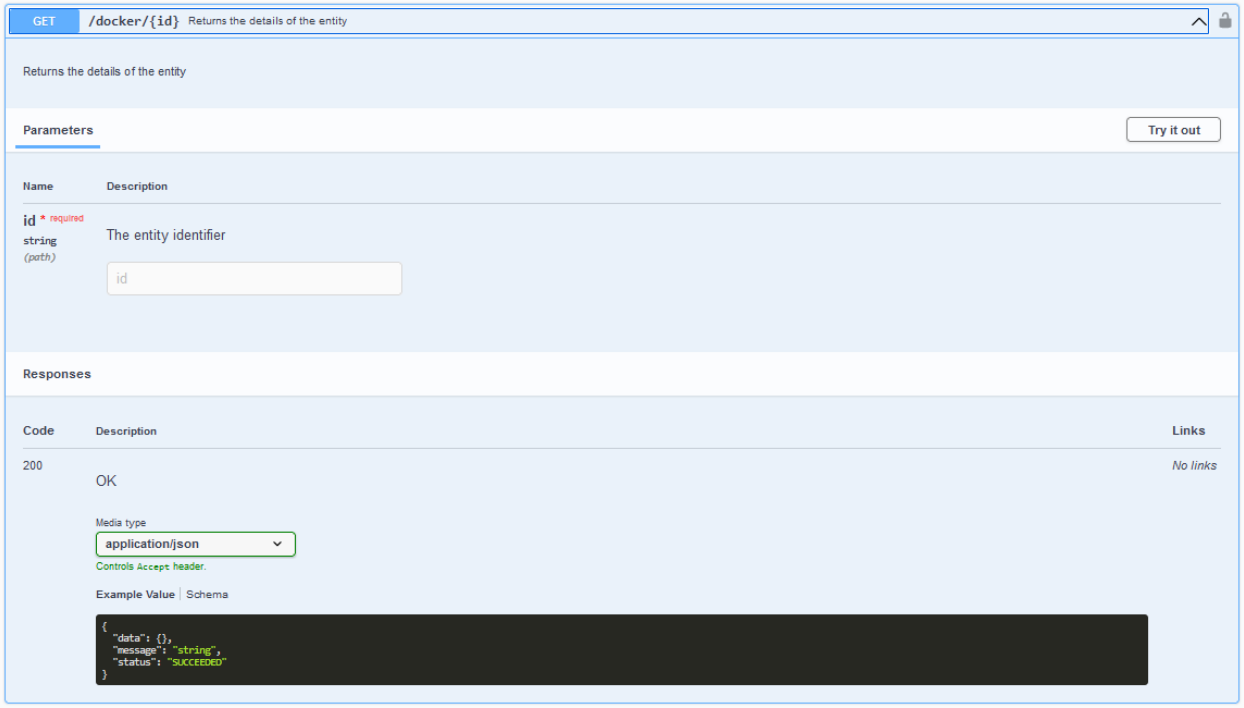


Figure 25: Get Docker image properties REST API endpoint

4.4.2.4 Delete a Docker image entity

The following endpoint describes the request for deleting a Docker image from the system.

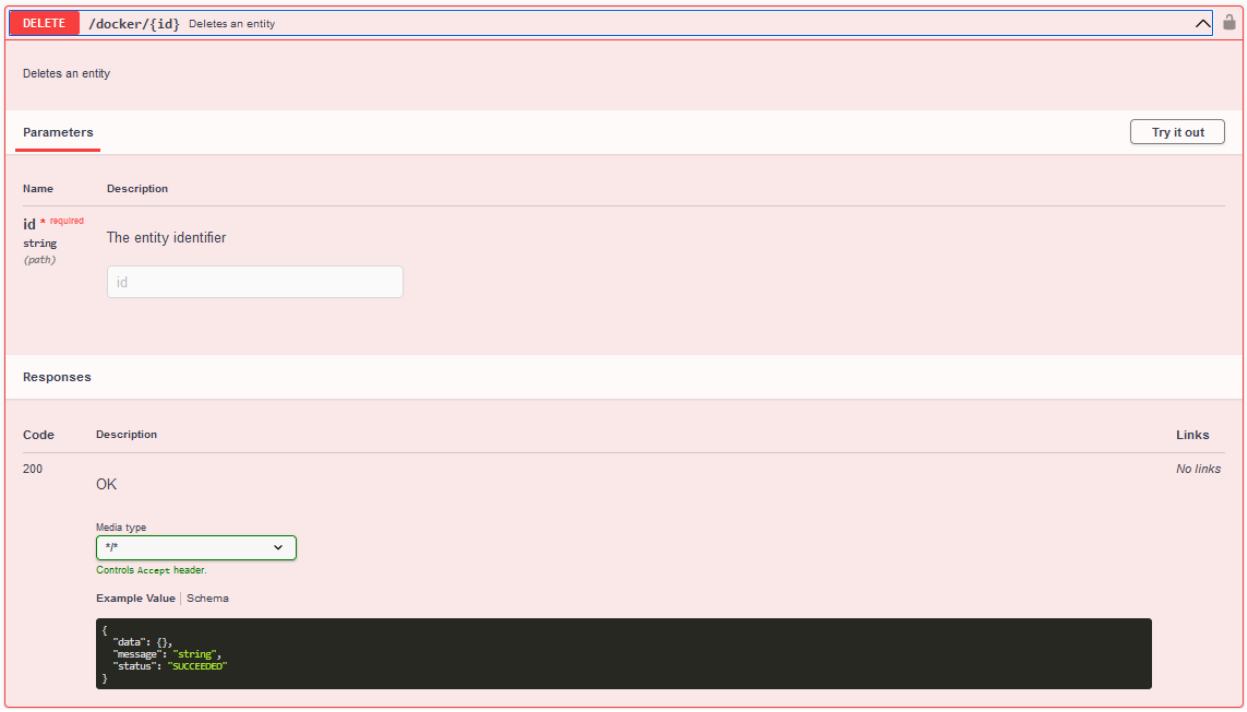


Figure 26: Delete a Docker image REST API endpoint

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

4.4.3 Processing components administration API

4.4.3.1 List available processing components

The following endpoint describes the request for listing the processing components defined in the system.

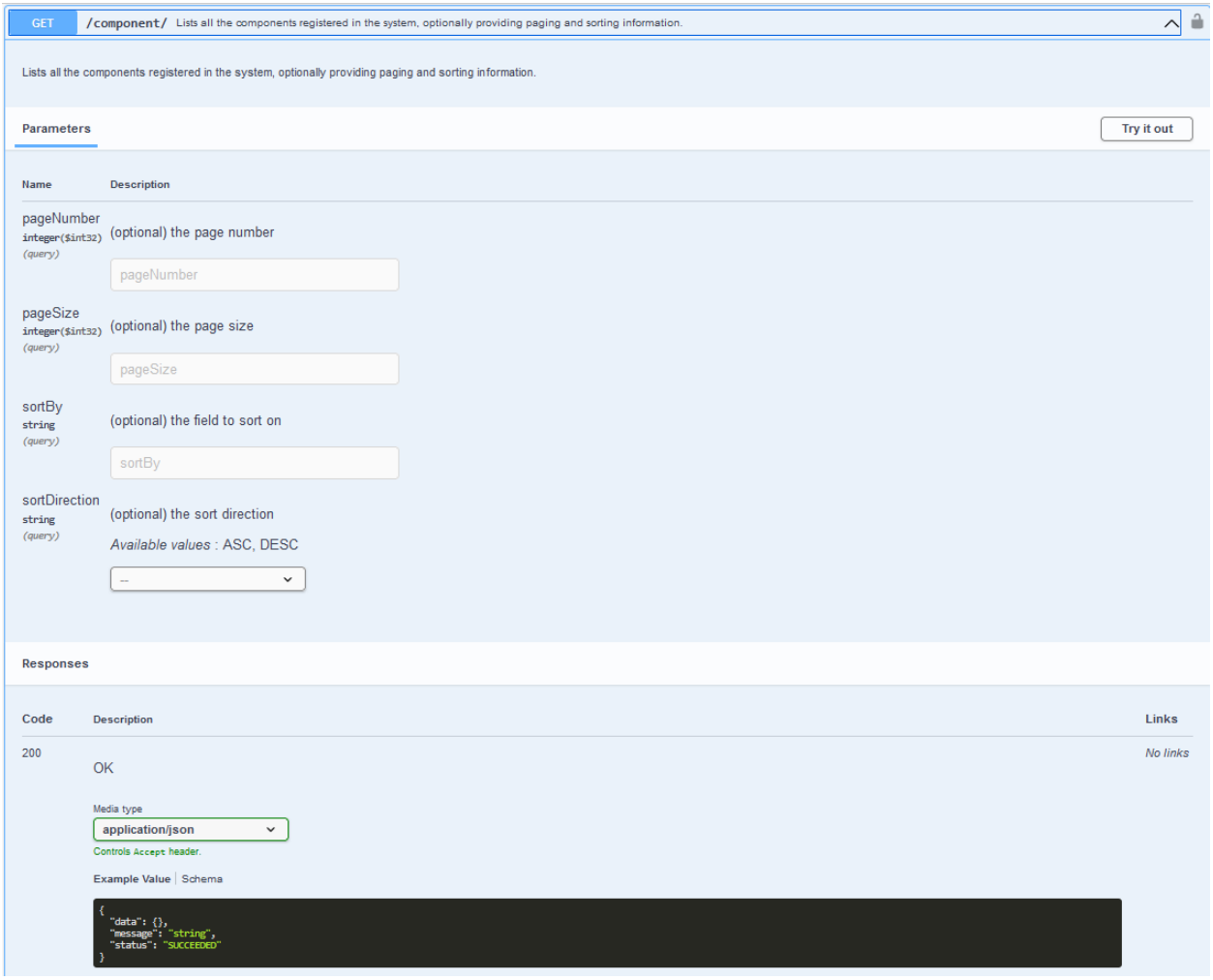


Figure 27: List processing components REST API endpoint

4.4.3.2 Create a new processing component

The following endpoint describes the request for creating a new processing component.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

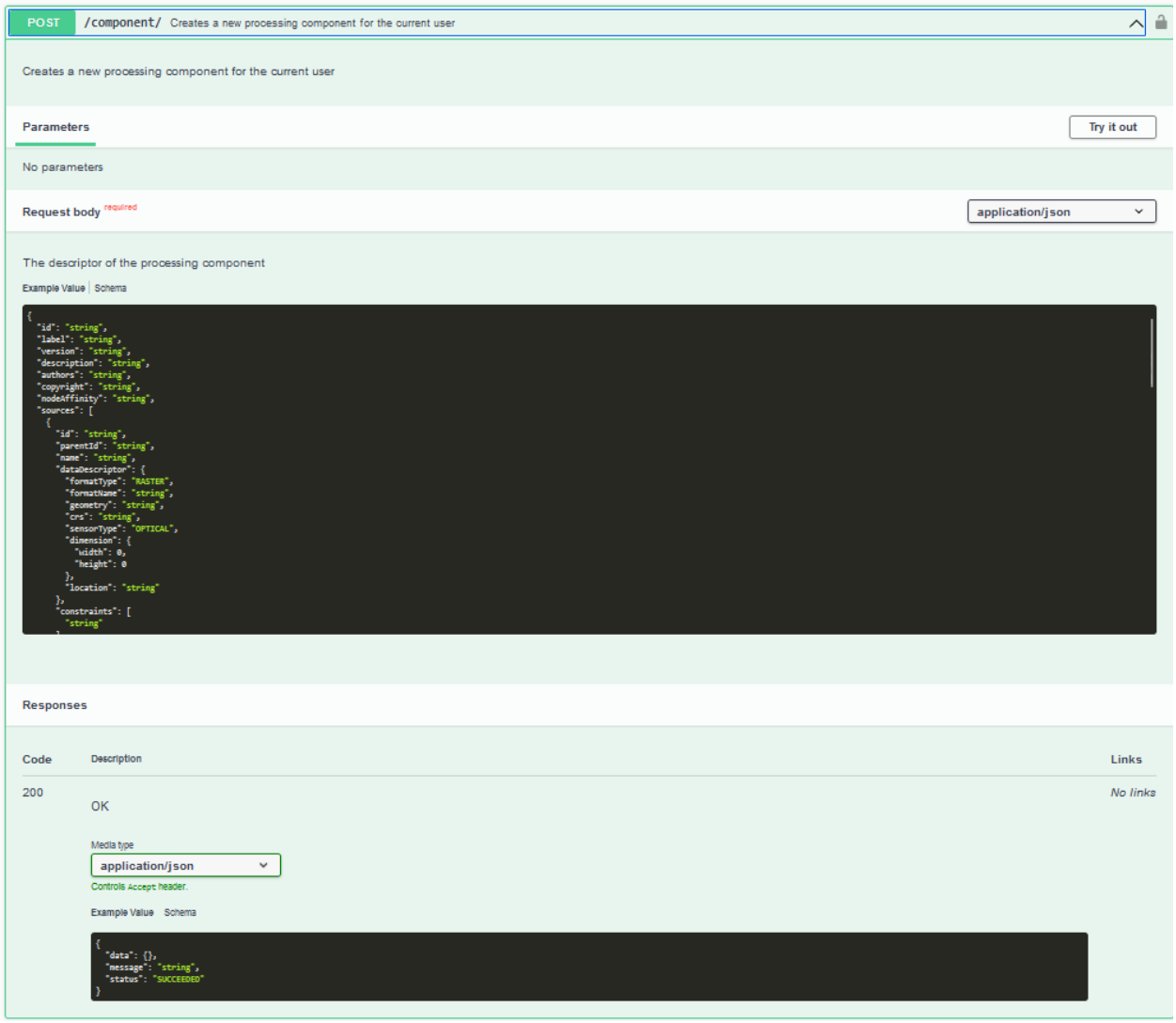


Figure 28: Add new processing component REST API endpoint

4.4.3.3 Get a processing component description

The following endpoint describes the request for retrieving the properties of an existing processing component.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

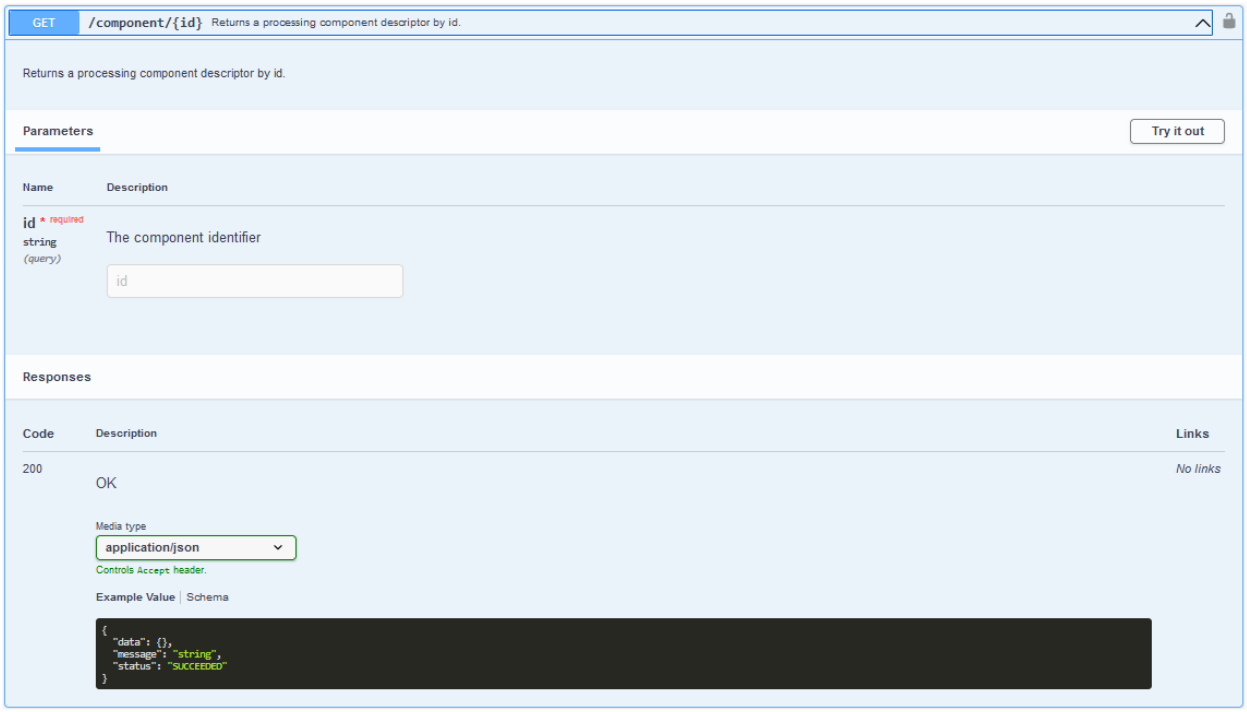


Figure 29: Retrieve a processing component REST API endpoint

4.4.3.4 Remove a processing component

The following endpoint describes the request for removing an existing processing component.

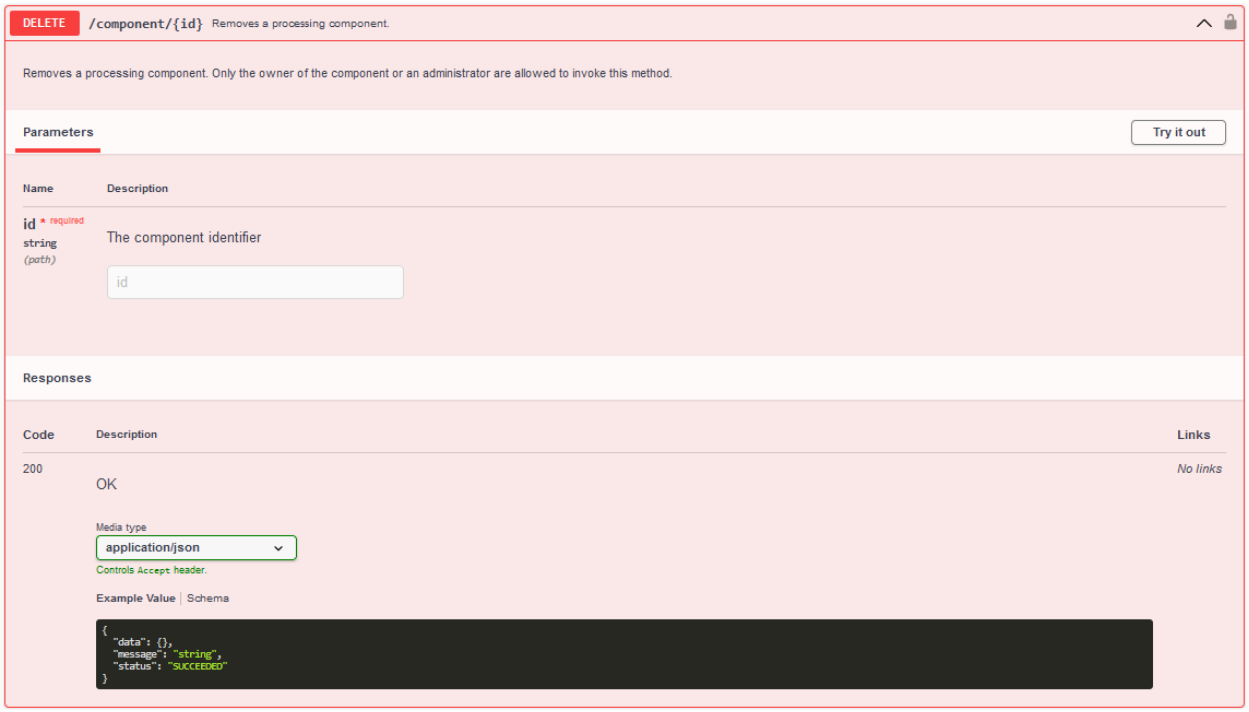


Figure 30: Remove a processing component REST API endpoint

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

4.4.4 Workflows API

4.4.4.1 Lists the workflows that are visible to the current user

The following endpoint describes the request for retrieving the list of workflows that can be executed by the current user.

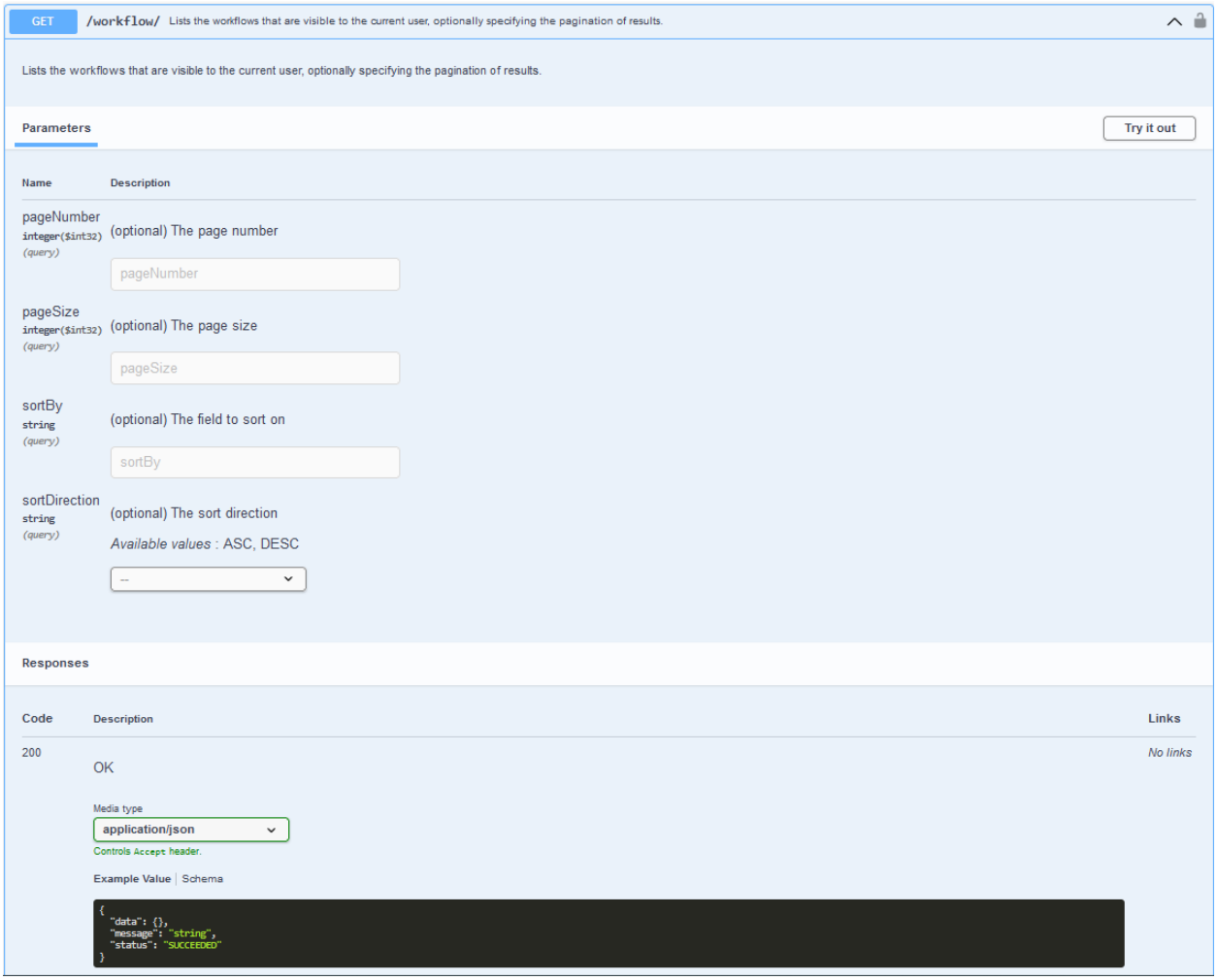


Figure 31: List workflows REST API endpoint

4.4.4.2 Create a new workflow

The following endpoint describes the request for creating a new workflow.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

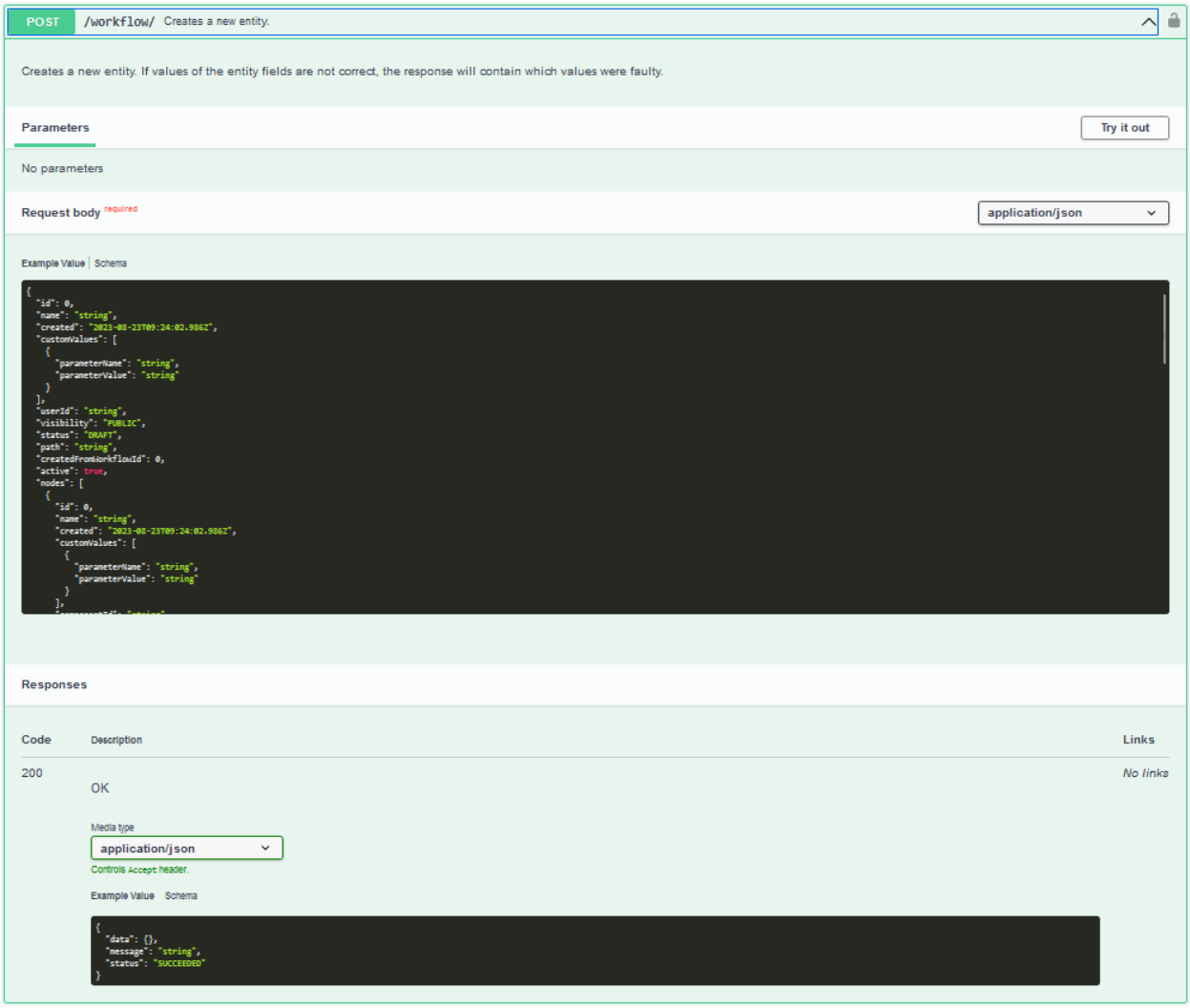


Figure 32: Create a new workflow REST API endpoint

4.4.4.3 Get the description of a workflow

The following endpoint describes the request for retrieving the descriptor of an existing workflow.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

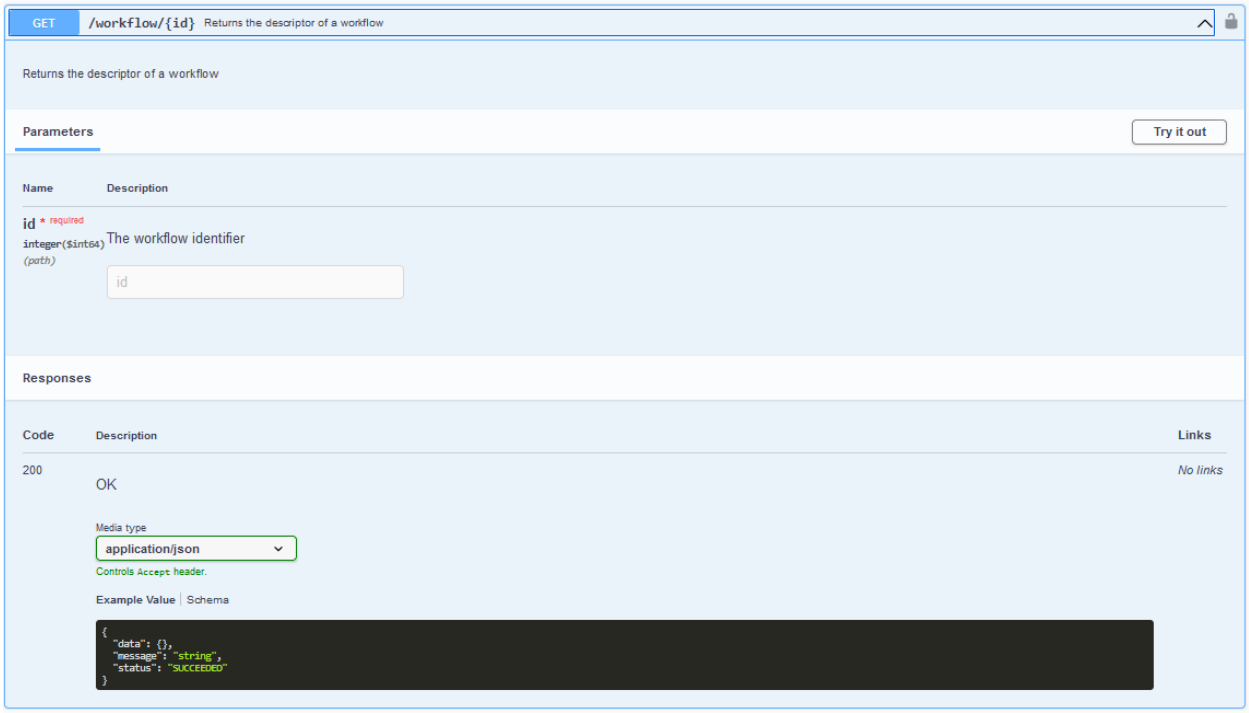


Figure 33: Get description of a workflow REST API endpoint

4.4.4.4 Delete a workflow

The following endpoint describes the request for deleting an existing workflow.

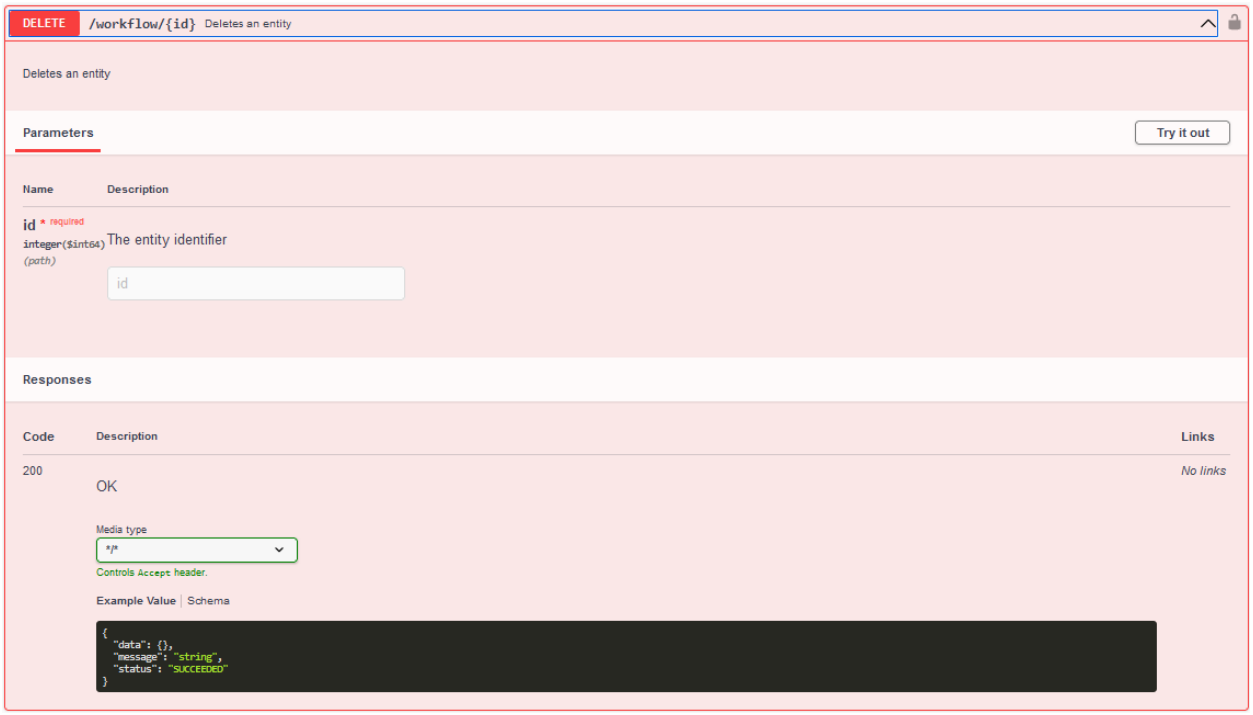


Figure 34: Remove a workflow REST API endpoint

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

5 Deployment infrastructure

The current EMD integration platform represented by TAO is deployed at the following address:

http://enrichmydata.c-s.ro

The platform does not allow in this moment self-registration so any user desiring to access it will have to contact EMD or CS GROUP – ROMANIA team.

The next table presents the list of current deployments for all tools available at this moment in the project and enumerated in D2.1:

Tool	Deployment
ABSTAT	http://abstat.disco.unimib.it/apis
SemTUI	Available as docker image : ghcr.io/i2tunimib/tui:latest Pulled in https://enrichmydata.c-s.ro
Ontotext GraphDB	https://graphdb.innograph.enrichmydata.eu
Ontotext Semantic Objects	https://graphql.innograph.enrichmydata.eu
Ontotext Semantic Search	https://graphql.innograph.enrichmydata.eu
Ontotext Refine	https://refine.innograph.enrichmydata.eu
RMLMapper	Available as docker image : rmlio/rmlmapper-java Latest docker image also pulled in https://enrichmydata.c-s.ro
selBat	https://alligator.hel.sintef.cloud/Alligator
Ontotext Reconciliation	https://reconciliation.innograph.enrichmydata.eu
Wikifier	https://wikifier.org
Expert AI Platform Document Analyser	https://deployenv6.expertcustomers.ai:8084/services/emd/analyze
Event Registry Relation Classifier	http://eventregistry.org/api/v1/
InfoMiner	https://infominer.ijs.si/datasets Image can be also built using: https://github.com/Infominer-JSI/infominer-js/blob/main/docker-compose.yml
Expert AI Platform Document Classification	https://deployenv6.expertcustomers.ai:8084/services/emd/classify
LamAPI	https://lamapi.hel.sintef.cloud/

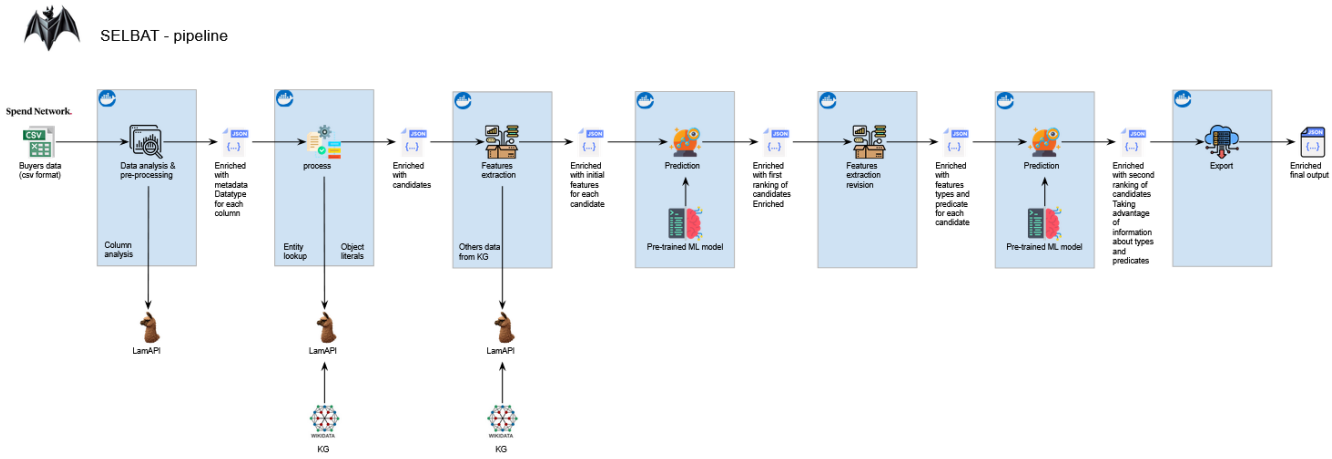
THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

TAO	https://enrichmydata.c-s.ro
StreamStory	http://atena.ijs.si:8080/login
Carbontracker	Available as python plugin : pip install carbontracker Installed in https://enrichmydata.c-s.ro

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

6 Pipeline creation example

This section presents step by step the implementation of a pipeline using the EMD Toolbox. The flow of the pipeline exemplified is presented in the following diagram:



Usually, when starting from scratch, after a fresh installation of the EMD Toolbox, the following preparatory steps need to be done:

- Add in the toolbox any new Docker image that contains tools executables (applications) that can be used in pipelines (see 4.3.2 Toolbox Docker images).
- Describe in the toolbox the tools executables by creating the corresponding processing components (see 4.3.3 Processing components).
- Create the workflow.
- Execute the workflow.

To start creating the Docker images, the following structure of directories and files was created for each step in the above diagram:

- A directory was created for each step.
- In each directory, the following files were created and added:
- the main script implementing the step.
- some additional helper scripts that are used within the main script.
- a text file containing the Python dependencies for all the scripts needed in PIP. This text file is used when creating the docker image such that all the necessary dependencies are added.
- The Dockerfile that will be used for the process of image creation. This docker file contains the instructions for copying into the image all the above scripts and installing all the necessary Python dependencies.

Nevertheless, creating individual Docker images for each step is a matter of user choice. Another solution is also to group all the applications (or subset of applications) in a single docker image, if the dependencies of the applications allow this configuration.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

After a successful login with an administrator account, the user needs to first access the menu “Toolbox Docker Image Descriptors”, from the “Admin Area” section. The existing Docker image descriptors will be displayed.

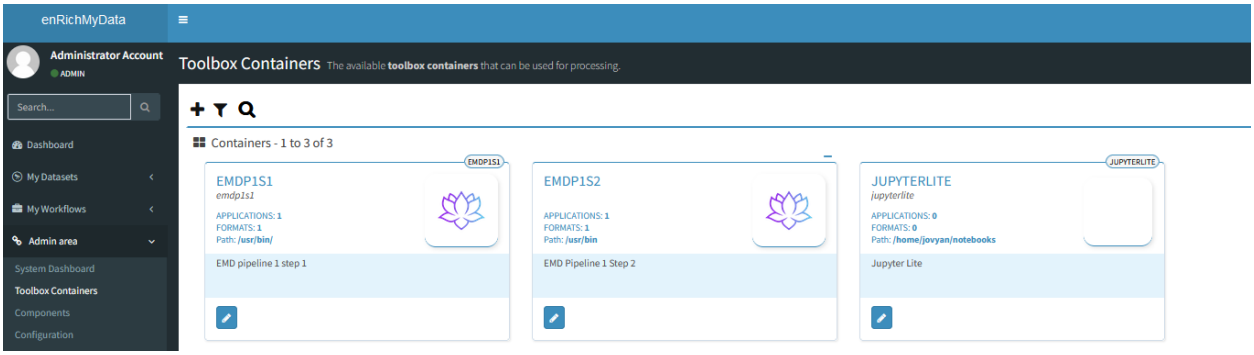


Figure 35: Accessing Toolbox Docker Image Descriptors

On this page, using the ✚ button the user can create one image containing all scripts or several Docker images corresponding to each step. The choice of using one image or several depends on how specific the scripts or applications covering a certain step are.

For this example, we assume that all scripts have specific needs, so we create Docker images for each step of the pipeline.

Once the ✚ is pressed, the following dialog opens allowing the user to create the Docker image and the TAO descriptor associated with it.

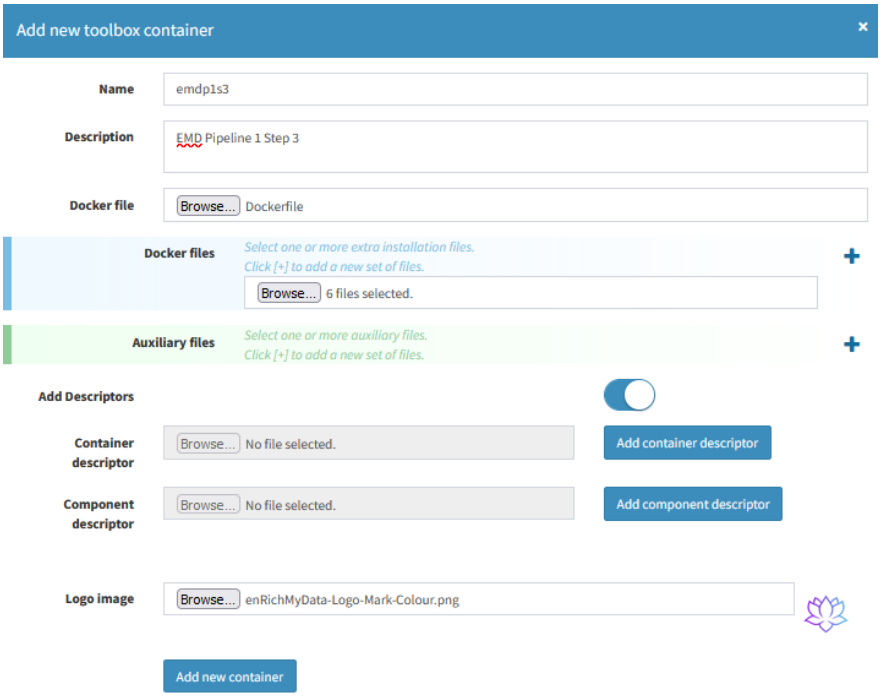


Figure 36: Add new Docker Image

The following fields need to be filled:

- Name – the name of the image as it will be registered in Docker.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

- Description – a brief description of the docker image.
- Docker file – a Dockerfile that will be used to create the image.
- Docker files – the files that are needed by the selected Docker file to be added into the image (the Python scripts, file containing Python pip requirements, etc).
- Logo image – a png file that will be used for the docker image.

Further, the user will have to add a Docker Image descriptor by pressing the "Add image descriptor" button that will open the following dialog box:

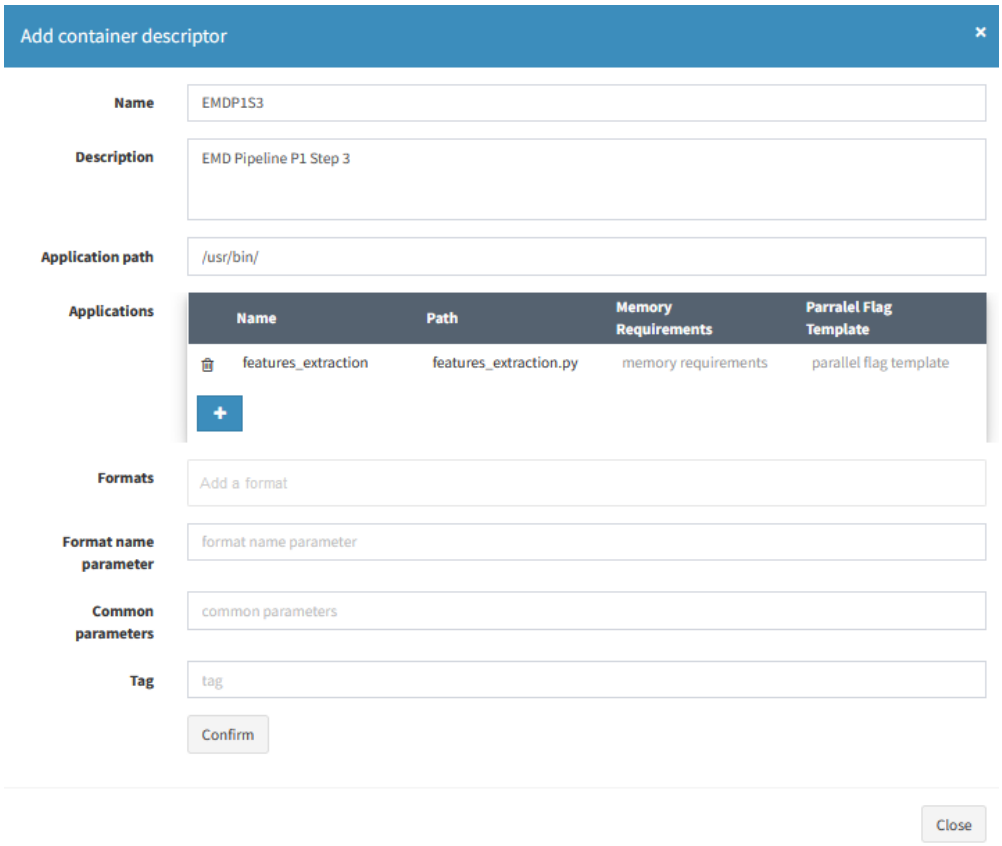


Figure 37: Add Docker image descriptor dialog

In this dialog box, the user will have to fill the following mandatory information (other fields are optional):

- Name - the name of the toolbox Docker image descriptor.
- Description – a brief description of the Docker image.
- Application path – the default location where the scripts are in the docker image.
- Applications - the applications exposed by this docker image.

After the confirmation that the descriptor was completed, the user is redirected to the "Add new toolbox Docker image" dialog. Once the user presses the "Add new image" button, the creation of the container will be launched in the background. Once the creation is finished and a refresh of the "Toolbox Docker Image Descriptors" page is performed, the new Docker image descriptor will be displayed and can be used for creating new processing components.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

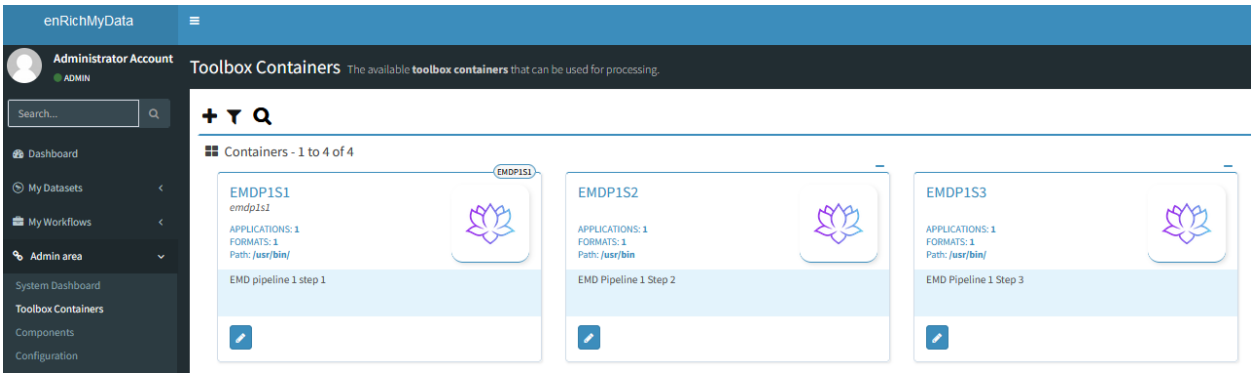


Figure 38: New toolbox docker image descriptor added

In order to add new processing components from the added Docker image, the user will need to access the "Components" section located into the same "Admin area" menu. This page will display the existing processing components:

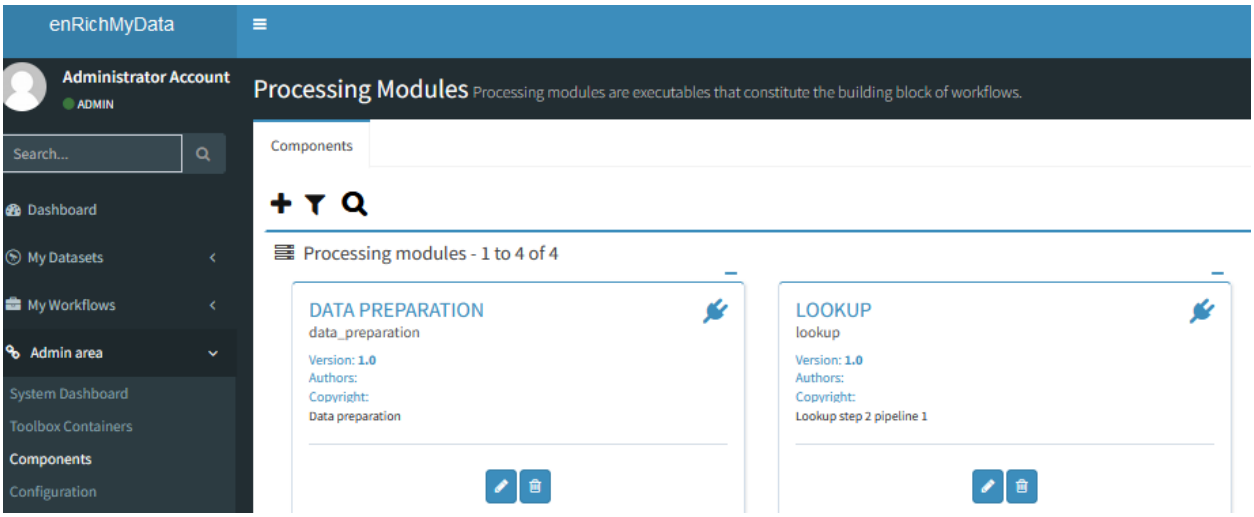


Figure 39: Processing modules page

Once the ✚ is pressed, the user will have the option to create a new processing module:

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

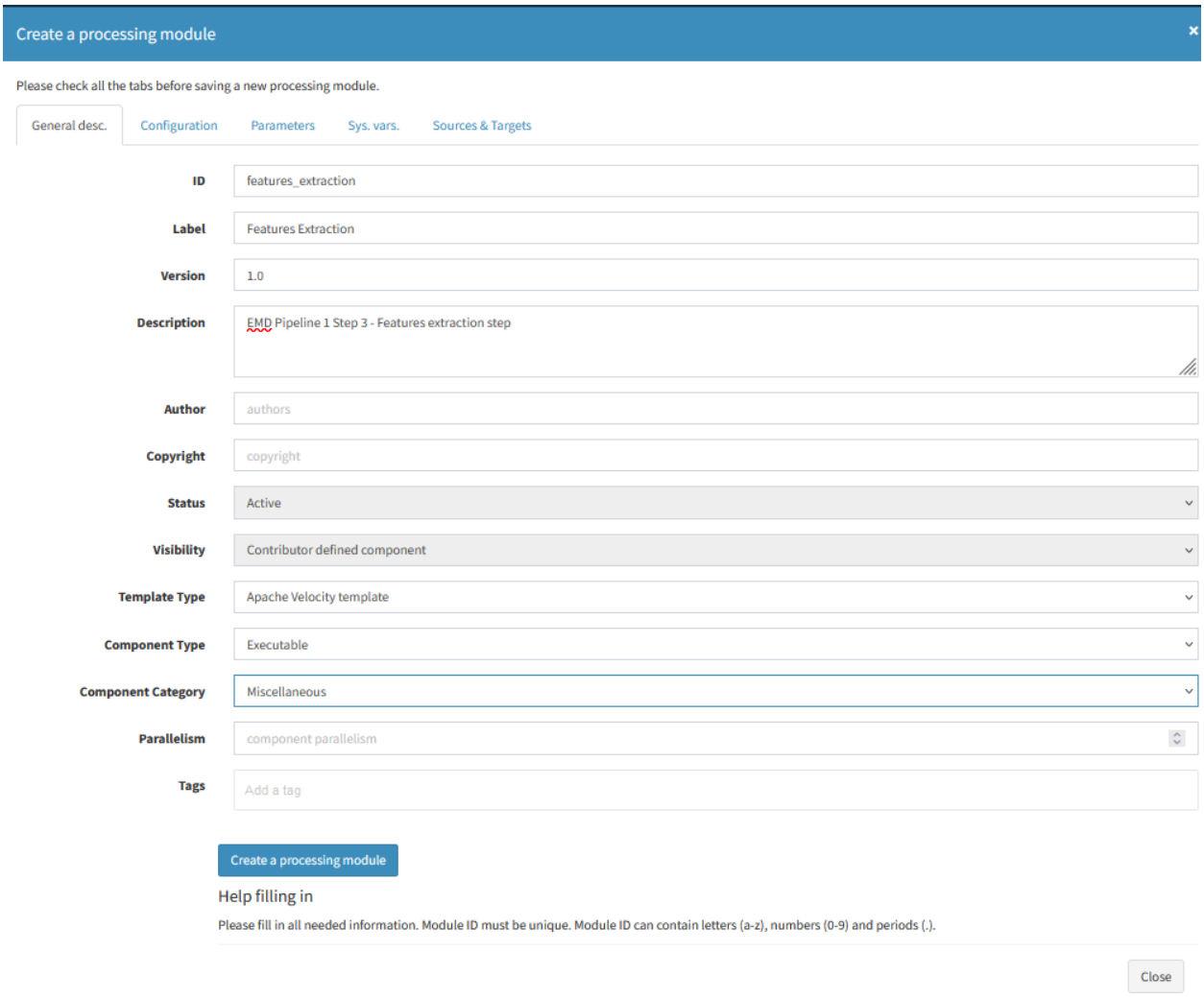


Figure 40: Creating a new processing module page

On this page, on the "General desc." Tab, the following fields is mandatory to be filled out:

- ID – the ID of the processing module that needs to be unique.
- Label – the short name of the component.
- Version – the current version of the component.
- Description – the description of the component.
- Component Category = Miscellaneous – the category of the component.

On the "Configuration" tab of the same page, the following fields need to be completed:

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

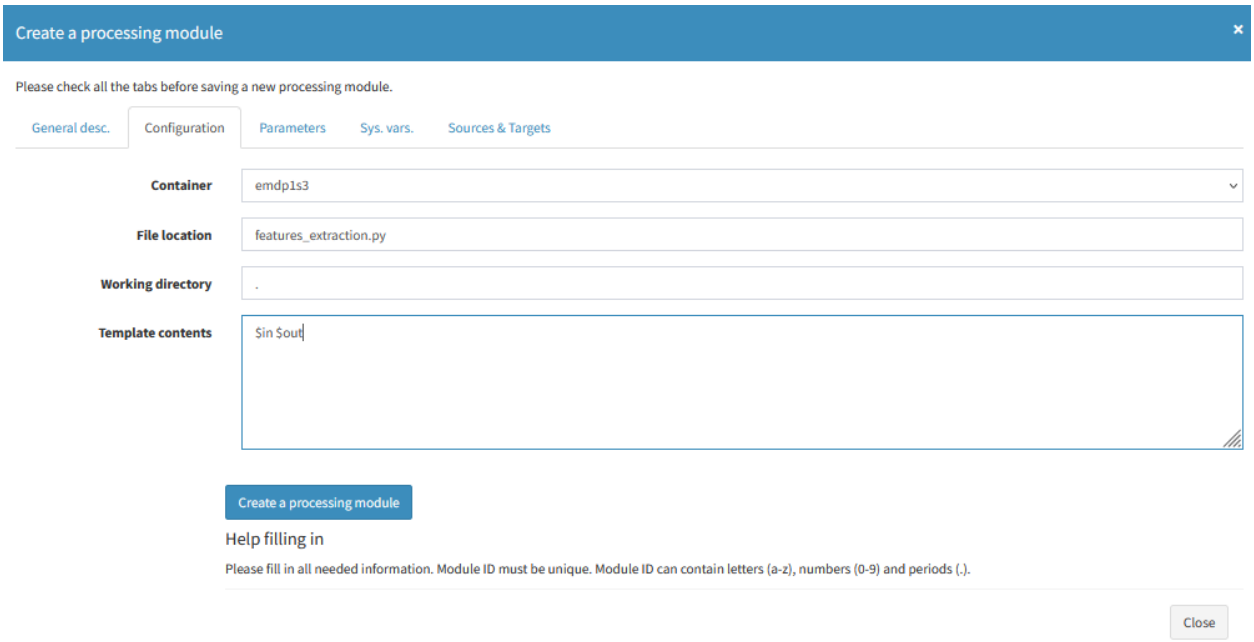


Figure 41: Processing module configuration page

The fields are:

- Docker image – the user will have to select one of the available images defined in the toolbox (in our case, the newly created Docker image "emdp1s3").
- File location: this is the actual path of the executable in the selected Docker image as defined when it was added to the Docker image descriptor (see Figure 8: Adding the Docker image descriptor) (in our case, "features_extraction.py"). If an invalid value is provided (ex. a path that was not defined in the Docker image description, the user will be prompted).
- Working directory: the working directory from the docker image (if needed).
- Template contents: the invocation arguments of the application that will be invoked. As our application is invoking as "features_extraction.py input.json output.json" then it has only one input file and one output file, the "$in" and "$out" being placeholders for these parameters. Nevertheless, other examples might be:
- features_extraction.py -i input.json -o output.json. In this case, the "Template contents" should be filled as "-i $in -o $out".
- features_extraction.py -i input.json -o output.json -param1 some_value. In this case, the template will be, for example "-i $in -o $out -param1 $param1". But for this particular case, the param1 should be added also in the "Parameters" tab.

The next section to be filled is the "Parameters" tab from the same dialog, where need to be defined parameters in the command line, other than the ones defined as "Sources & Targets", that usually contain fixed or specific values (as in the second example of the template content).

The last section to be filled is the "Sources & targets" section where are defined the parameters from the command line (or template contents) represent input and the output files in the command. In our case, $in is a source file and $out is a target file:

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

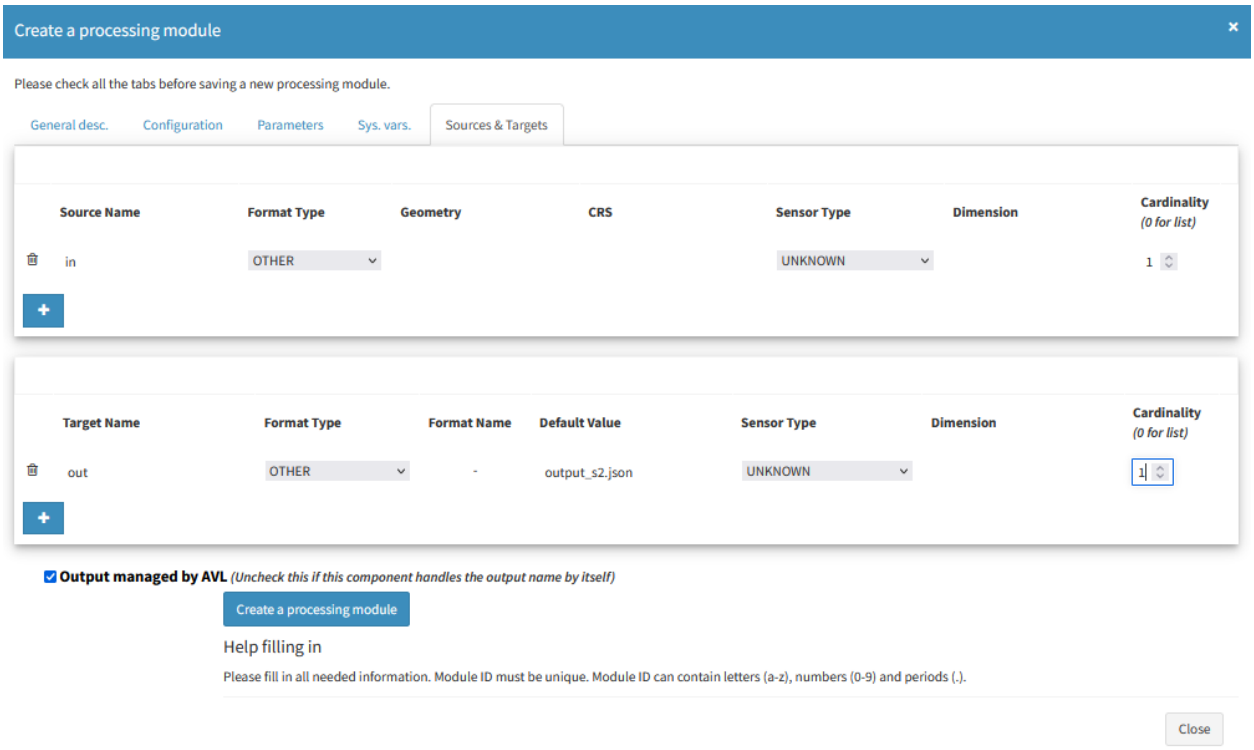


Figure 42: Define sources and targets of a new processing module

In order to add a source or target, the "+" button should be used. This will add new entries in the corresponding table. For each entry, the following fields should be completed:

- Source name – the name of the parameters from the template (but without the $ prefix)
- Format Type – select OTHER.
- Cardinality – in our case 1 as our python script accepts only one file for the input and one file for the output. Nevertheless, there are some cases when an input parameter accepts, for example, a list of files in which case the cardinality should be set to 0. Another particular case is for the applications that are intended to be used as the first node in a workflow and we want to customize its input (during the launch of the workflow in execution) and in this case we should set the cardinality to -1.

Once all the fields are set, the user can press "Create a processing module" button that will create and display the new processing component in the "Processing modules" page:

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

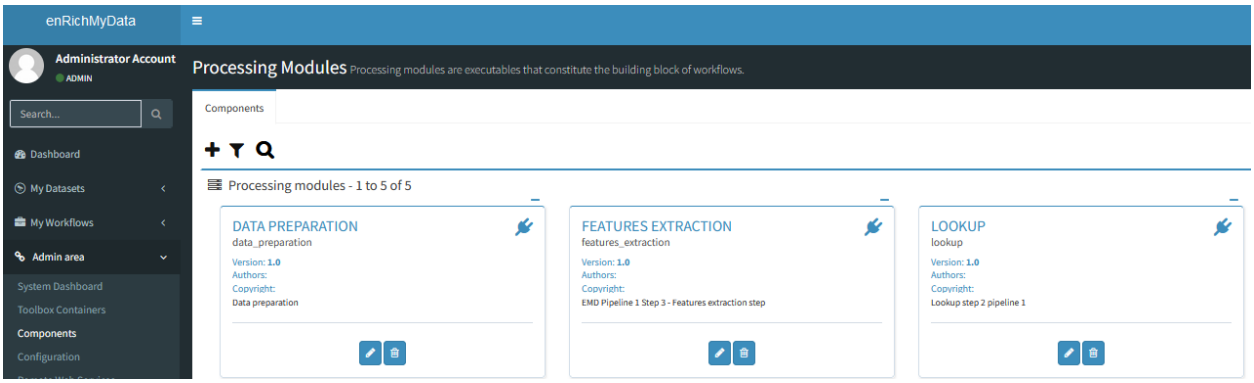


Figure 43: New Processing module added

If any changes are needed to the processing component, it can be edited by pressing the  button that will open a dialog similar to the one used for the creation.

Once a processing component has been created, it can be further used in the creation of a new or existing workflow.

To access workflows section, the “My Workflows” => “Workflows” options should be selected that will display the existing created workflows:

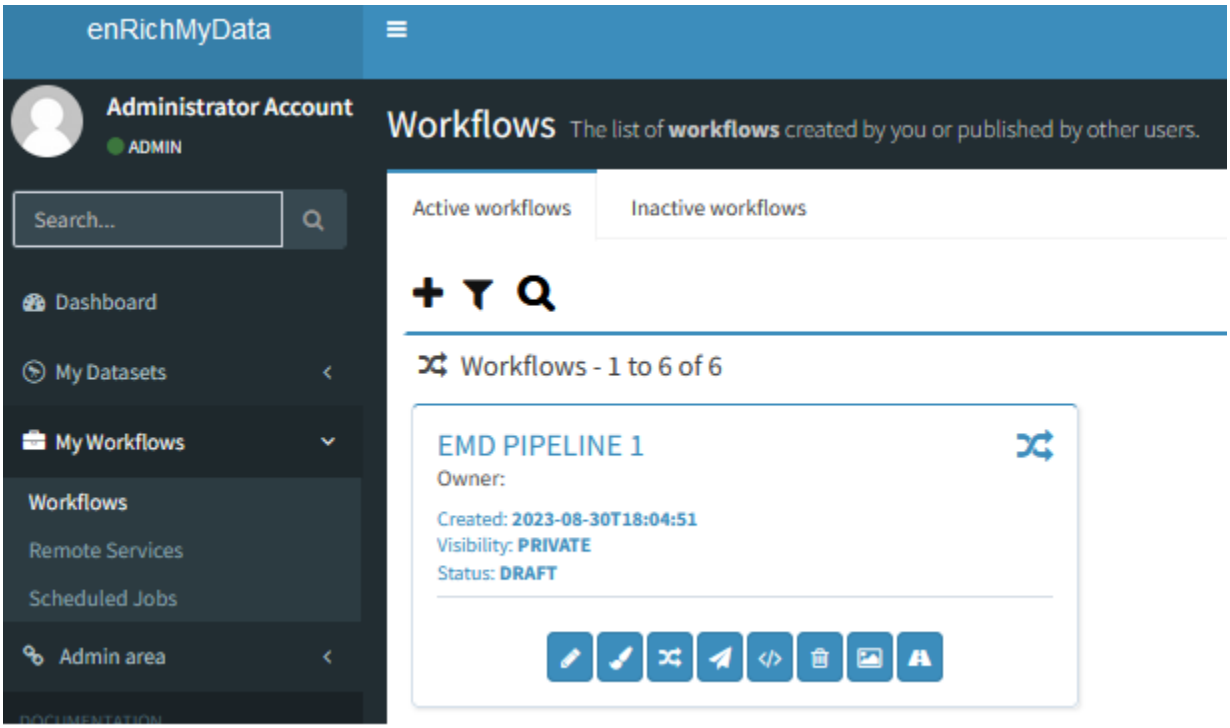


Figure 44: Workflows page

On this page, to add a new workflow, the “+” button can be pressed:

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

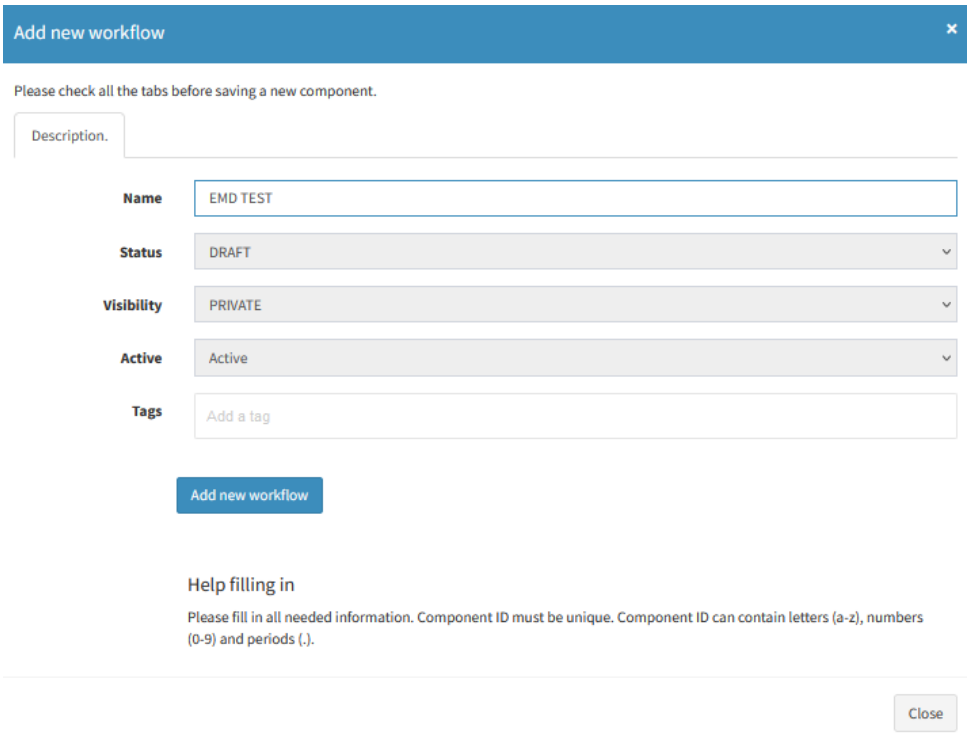


Figure 45: Create new workflow dialog

Once the "Add new workflow" button is selected, the user is redirected to the Workflow design page where he can drag and drop processing components and make connections between them. Same happens if the user is selecting the "Edit workflow" from a specific workflow:

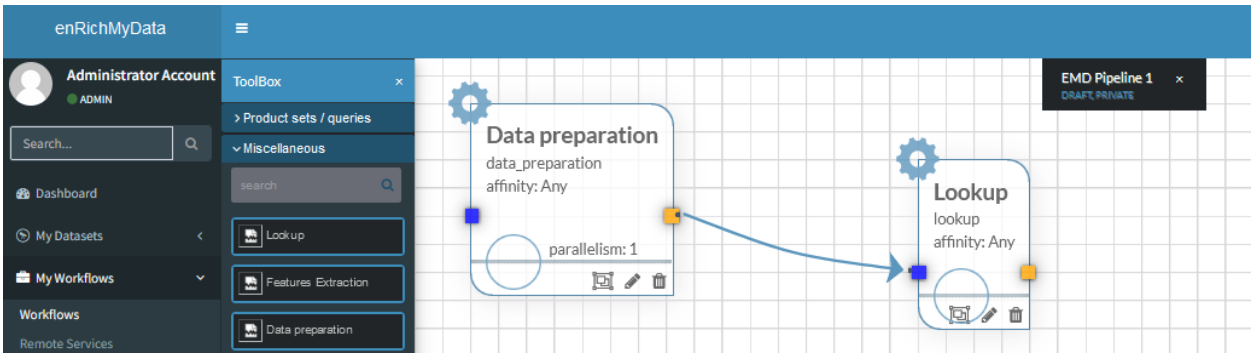


Figure 46: Workflow editor

In order to add new components on the dashboard, the user can drag and drop components from the left side of the panel:

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

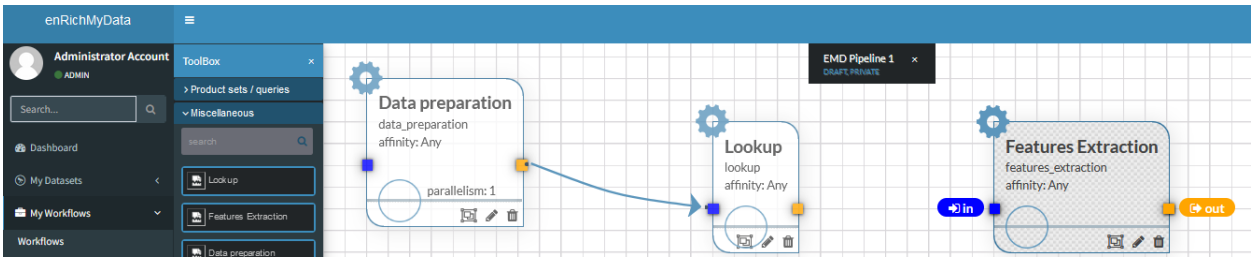


Figure 47: New workflow component added

Once a new component is added, its input can be connected to the output of another component, according to the definition of the pipeline:

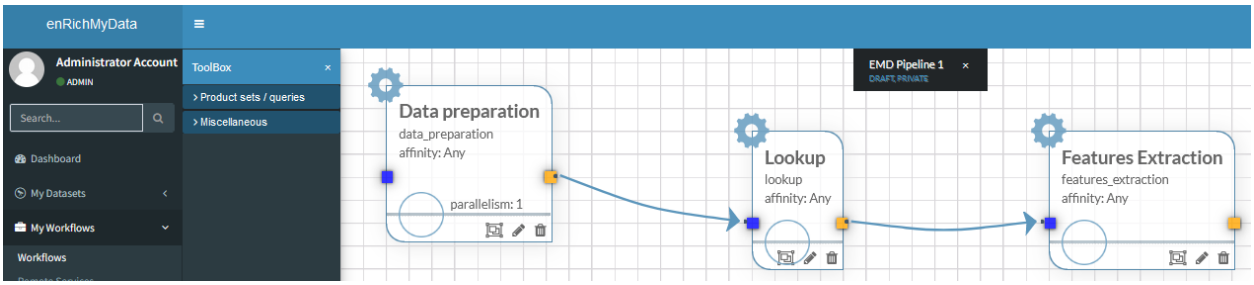


Figure 48: Interconnecting components

Additionally, the component can be edited by using the  button in which case, a dialog allowing to the user to provide custom values for certain parameters of the component:

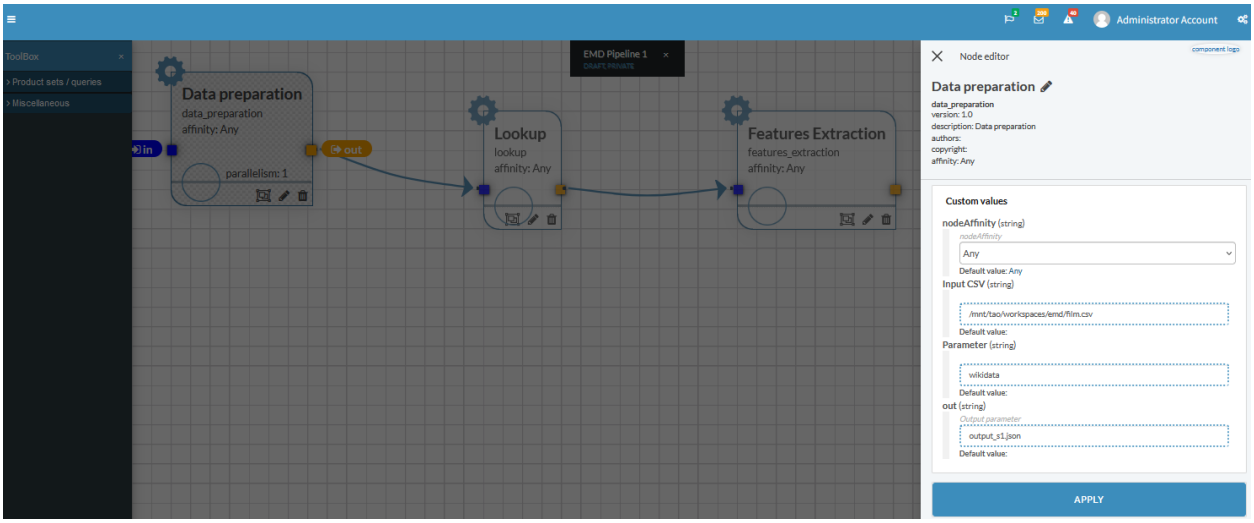


Figure 49: Customizing workflow component parameters

Once the editing of the workflow is done, the user can return to the main workflows page and launch the workflow in the execution by pressing the “Start execution”  button. Following this action, the following dialog is displayed:

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

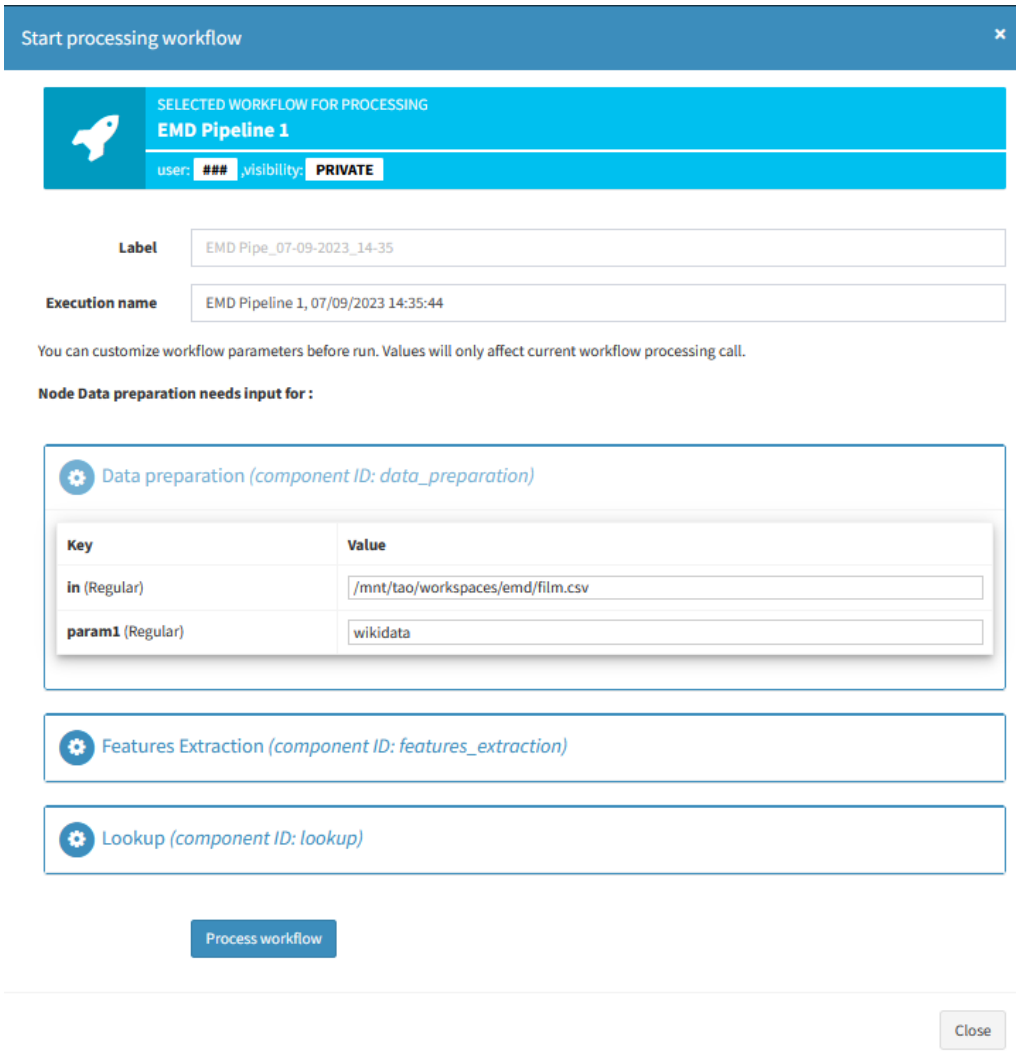


Figure 50: Executing a workflow

On this page, the user can review and change some of the parameters before execution. Additionally, the user needs to provide a label and an execution name (they are automatically filled but the user can change them). If everything is OK, the user can launch the execution by pressing “Process workflow”.

Once the execution is started, the user can go to the “Dashboard” page and see the status of the running job:

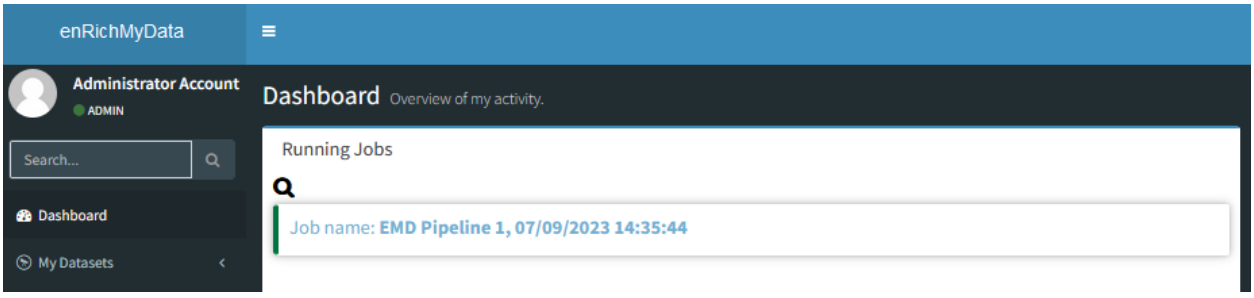


Figure 51: Workflow execution status

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

Selecting a running job, the user can see more details about the execution (current status, steps statuses etc.):

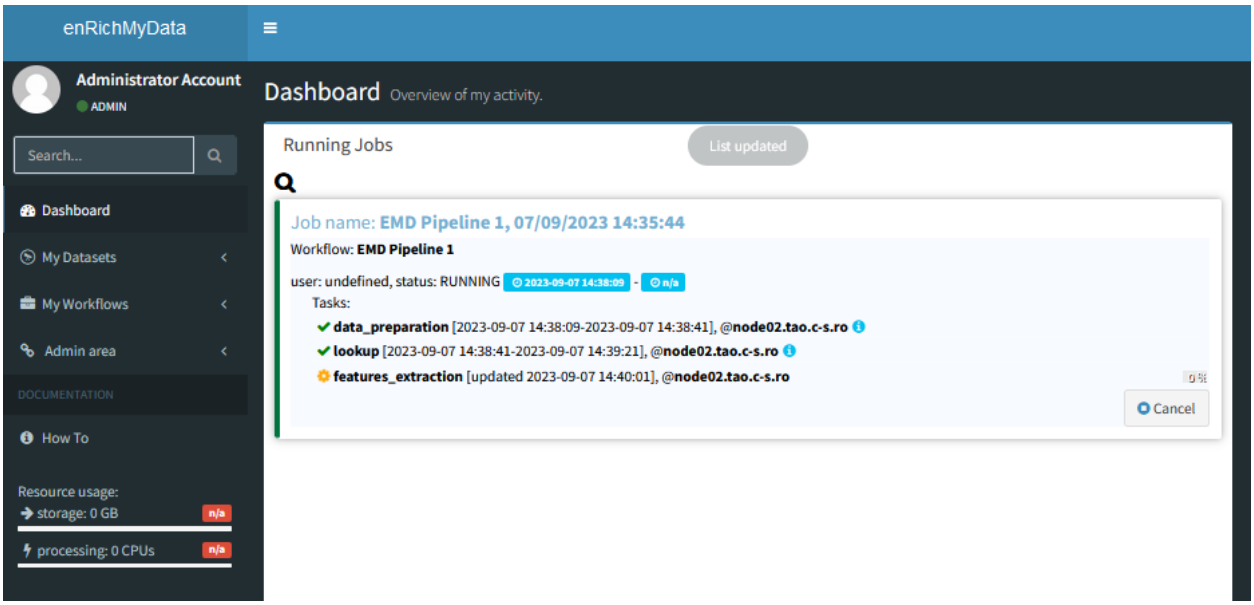


Figure 52: Workflow execution details

If a job is finished, the user can see it in the “Execution history" and also see notifications in the “Notifications History” and the online notifications popup:

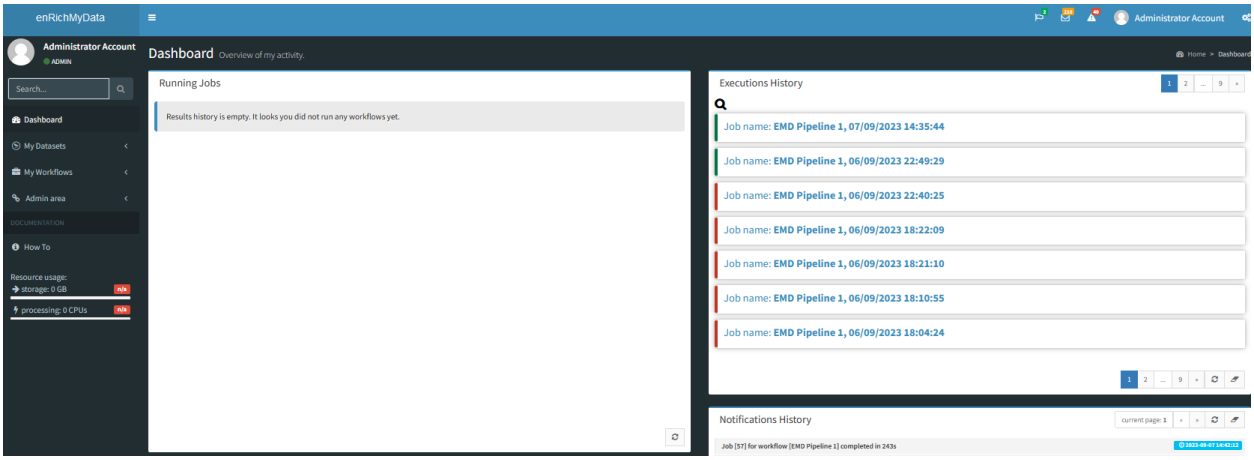


Figure 53: Workflows execution history

After the execution of a workflow, based on the job execution label, the user can see the files that were created by each step accessing the LOCAL repository:

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.

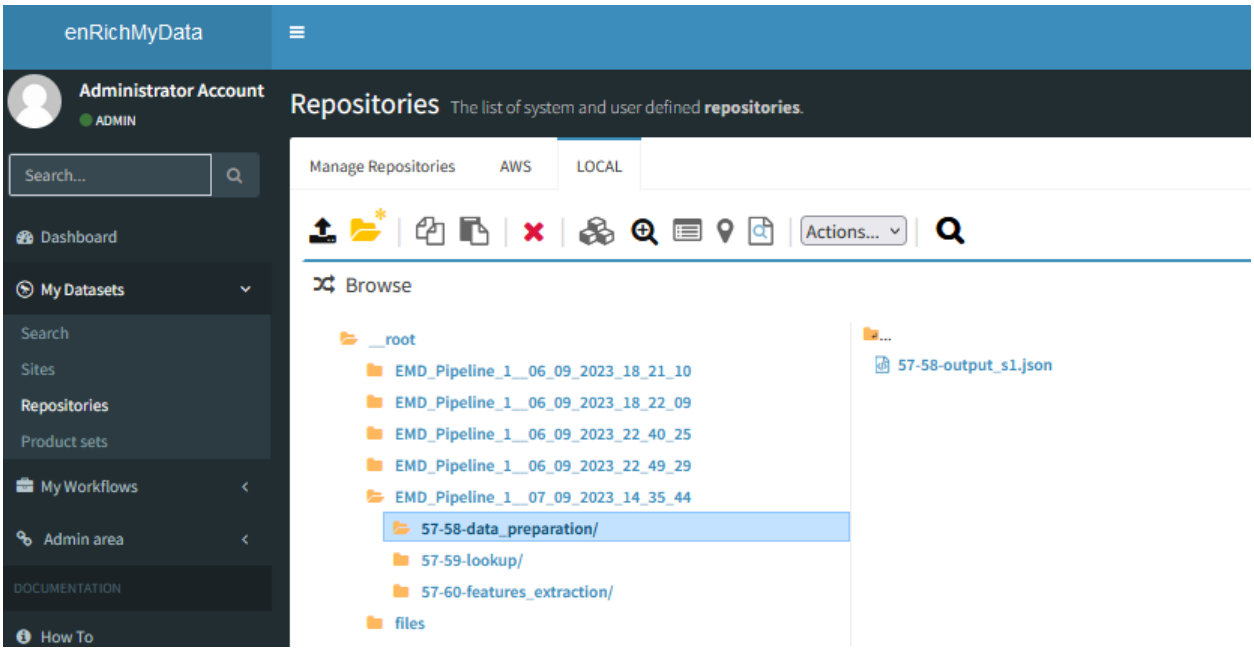


Figure 54: Accessing workflow execution result files

In this page, it has the option to upload new files, to create folders, copy and paste files from a directory to another, to delete, to view the file content, to move to another repository etc.

THIS PROJECT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON EUROPE RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO 101070284.