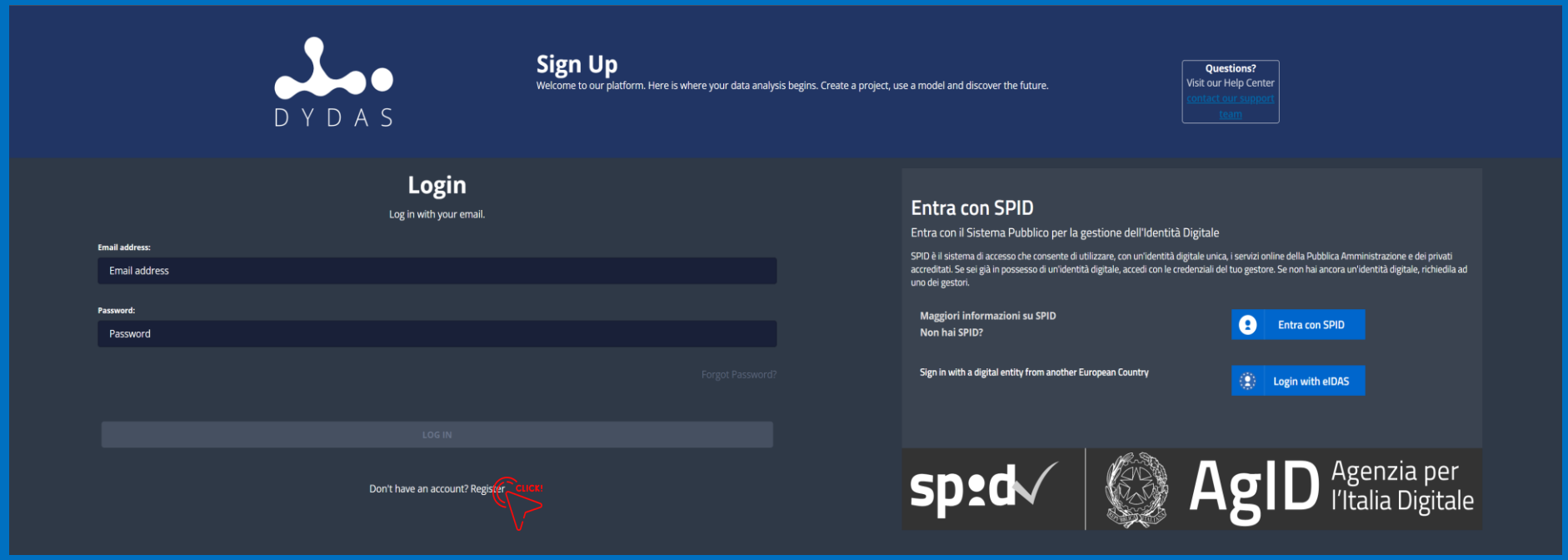
USER MANUAL

DYDAS PLATFORM

OPEN AND PRIVATE DATA FOR THE PUBLIC SECTOR AND INDUSTRIES

Register

Click on the register button as shown in the screenshot.



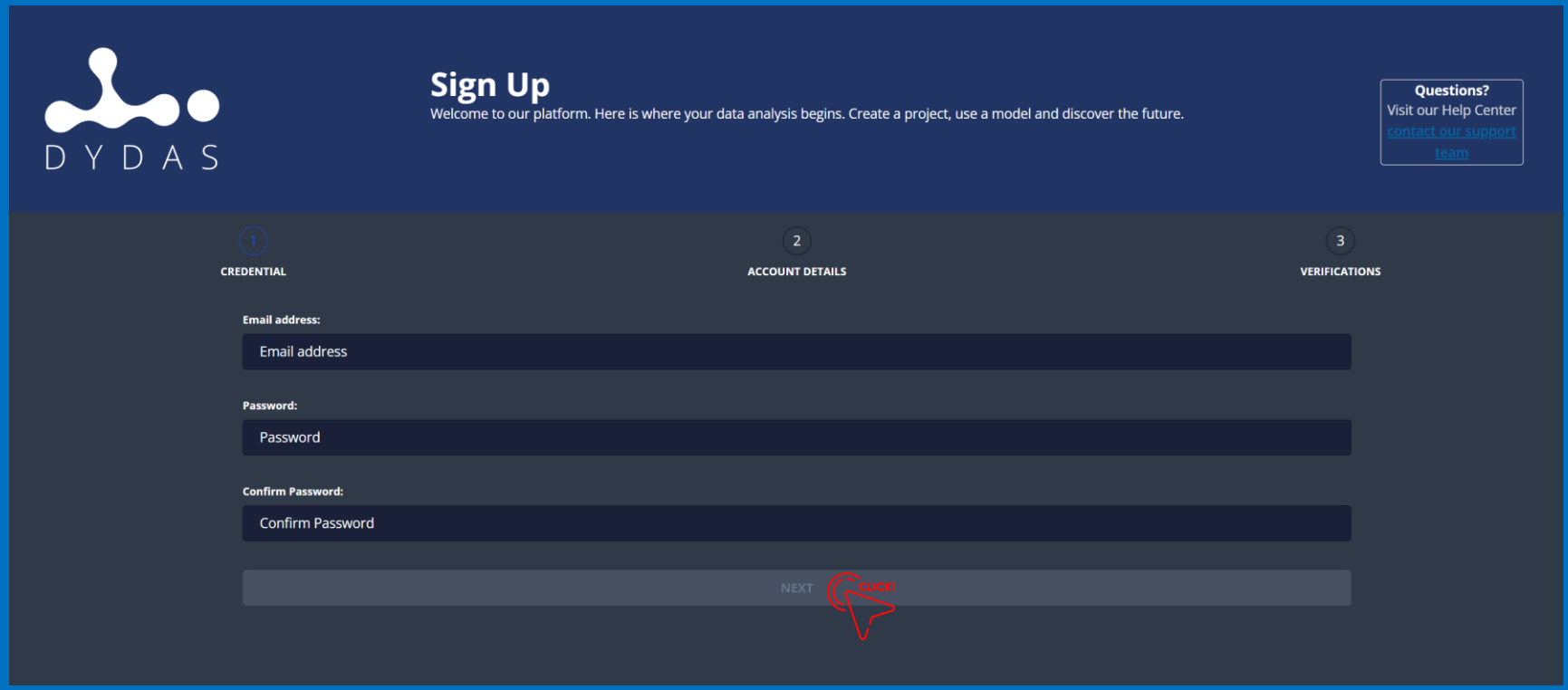
Register

Insert a valid email address and a password and retype it in the "confirm password" section.

The password should be at least 8 character long and it must contain:

- A lower case letter
- An upper case letter
- A number

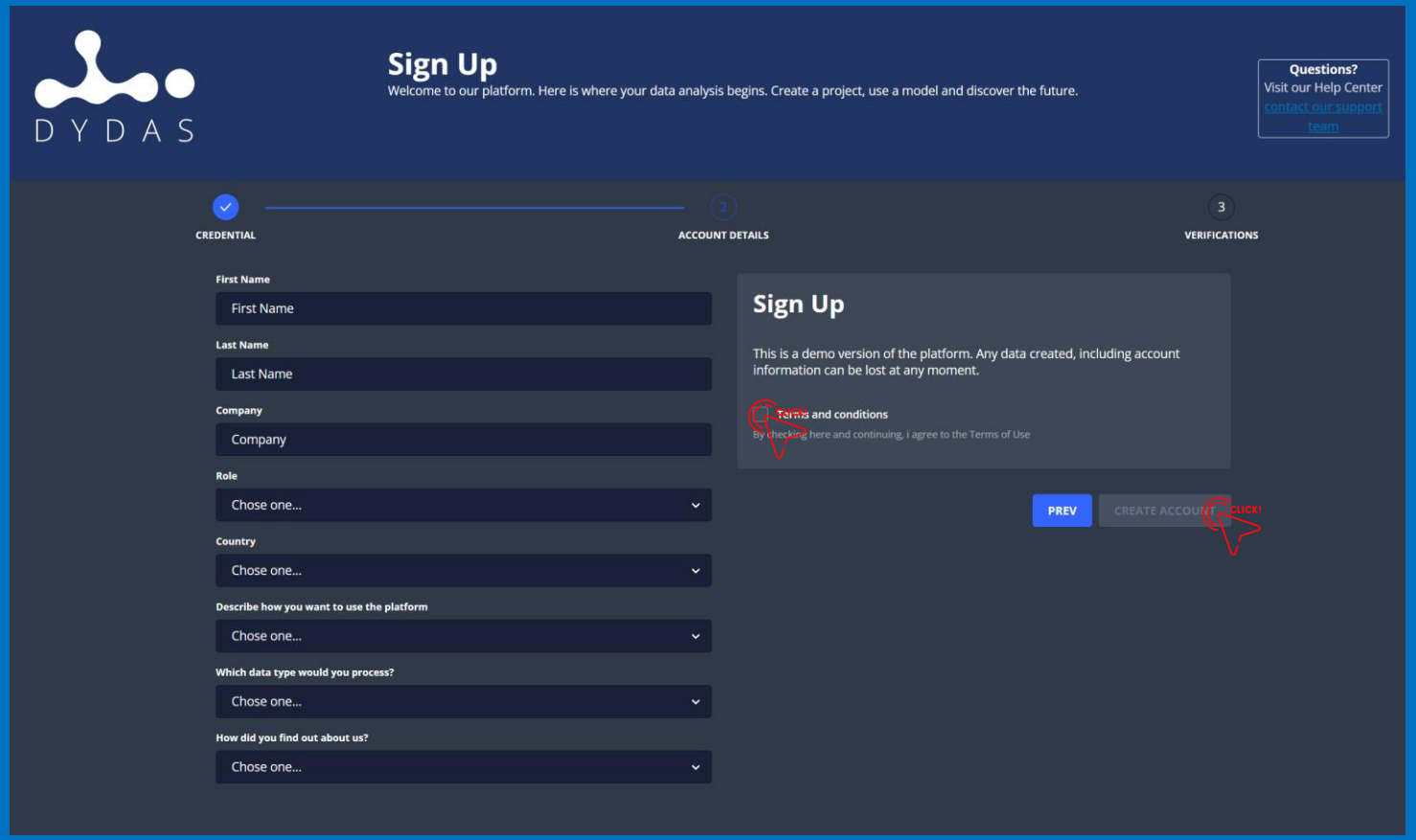
Click on "next".



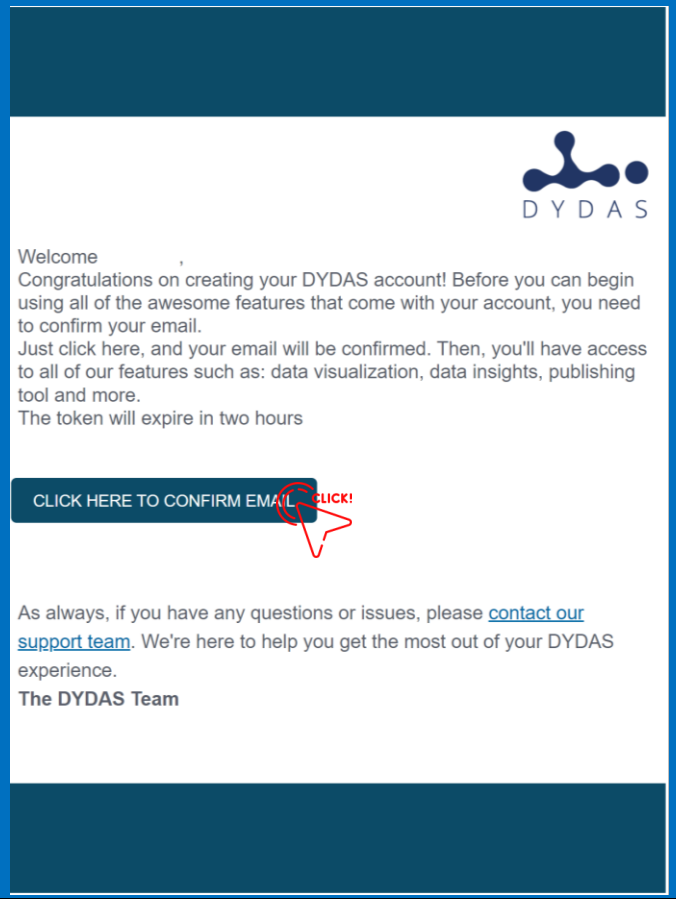
Co-financed by the Connecting Europe Facility of the European Union

Register

Insert account details, agree to terms and conditions and click on “create account”.

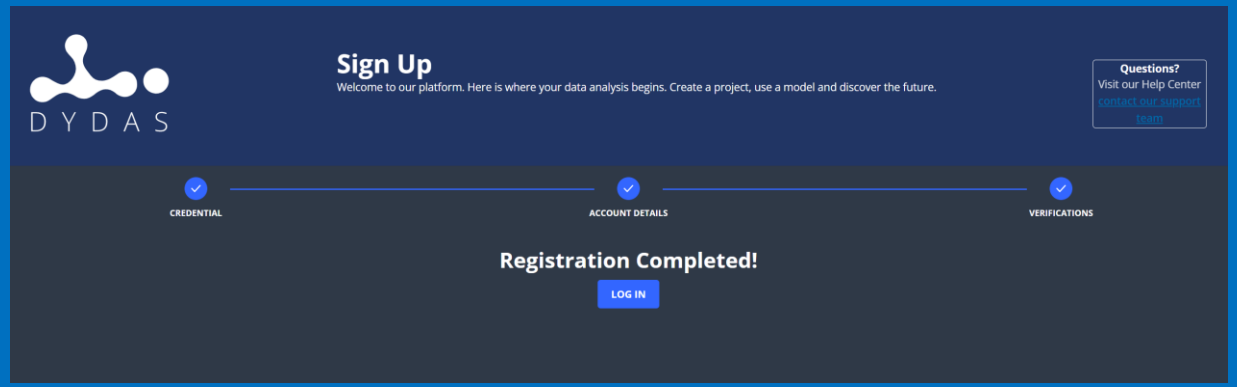


Register



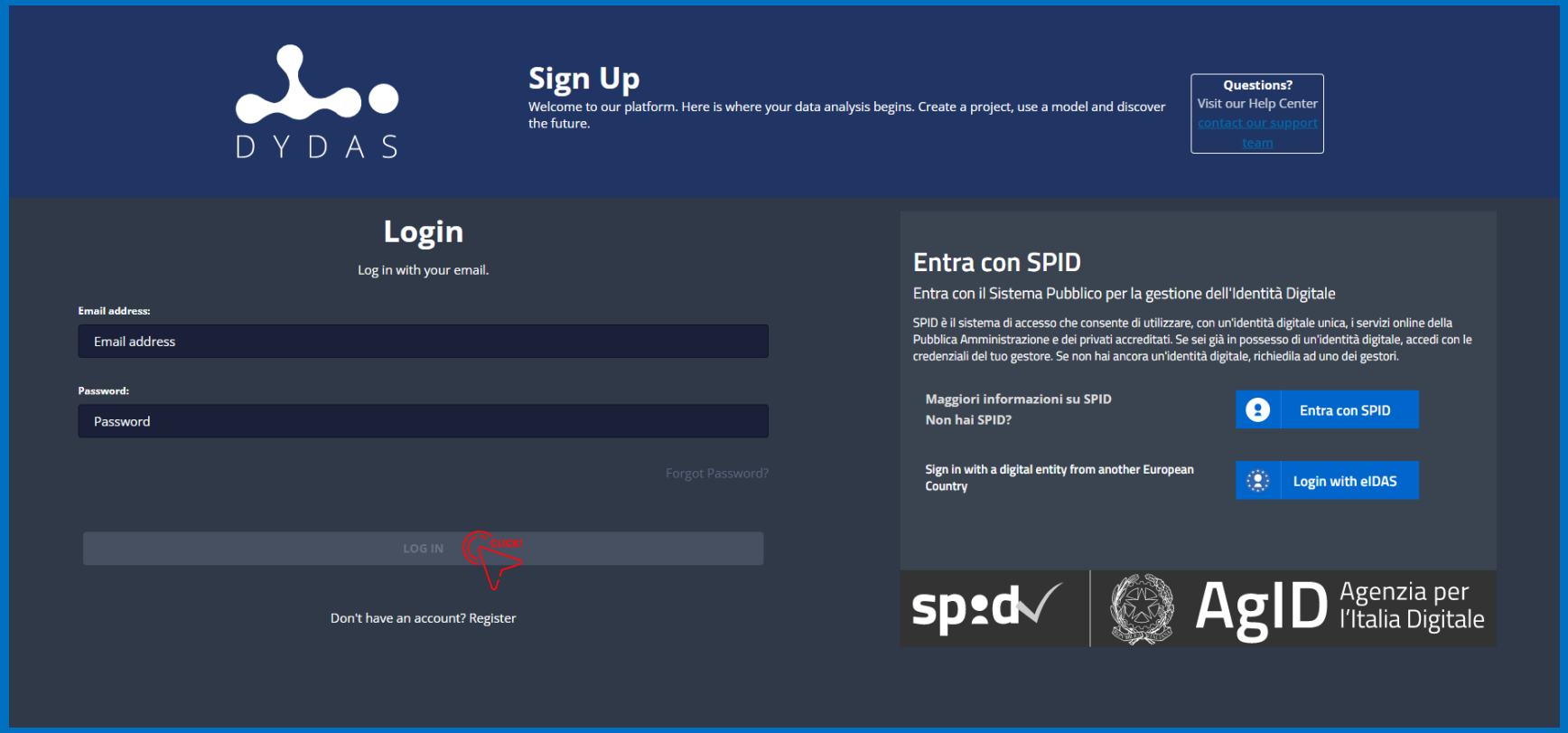
You will receive a email from registration@dydas.eu
Open it and click on the confirm email button.

Once this screen appears you have successfully completed the registration.



Login

Insert your email address and password used in the registration and click on the log in button or log in with SPID credentials by clicking on the SPID button.



Project

The baseline of DYDAS organization are the projects. A project is a container of all features and content that a user can produce in DYDAS: this includes data, algorithms, and views. A project can be considered as a tenant, it can be shared with other users for collaboration and can be published on the marketplace. All data, models and dashboard created are contained into a project: the project content cannot be shared separately from a project.

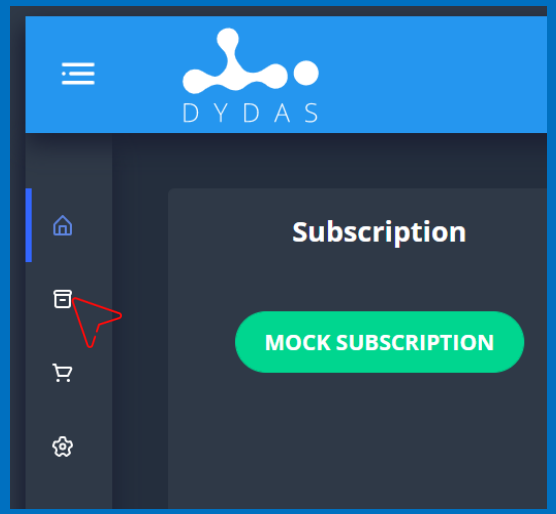
The project contains 3 sub entities:

- **Datasets** (data containers)
- **Model** (scripts containers)
- **Dashboards** (view containers)

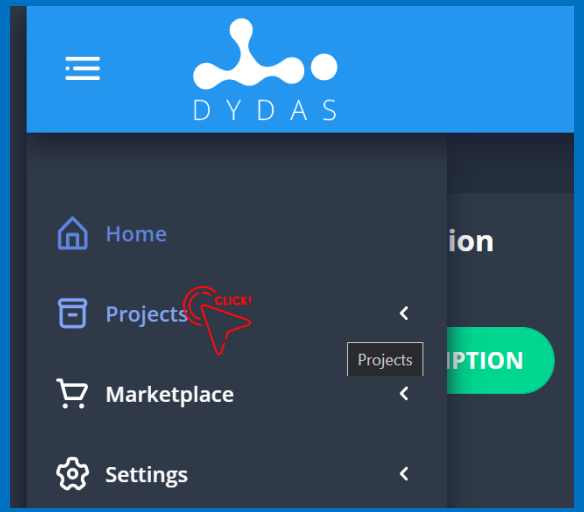
Based on the project type, it can contain one or more type of entities. A project can contain multiple instances of each entity type.
A project can contain private datasets (owned by the project only), open data (referenced only) or datasets contained in acquired projects (referenced only).

Project creation

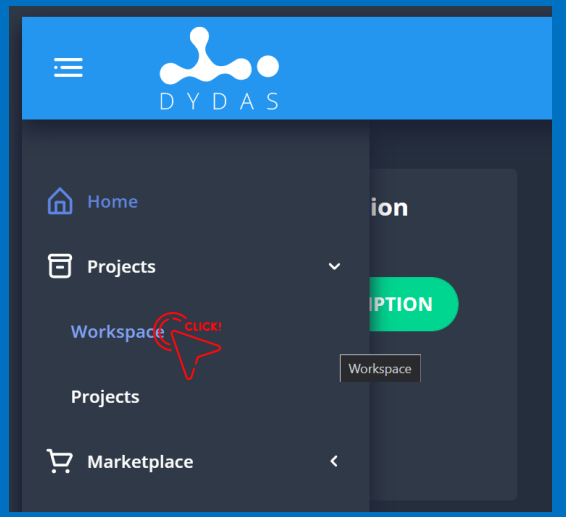
1. Place the mouse in the project icon

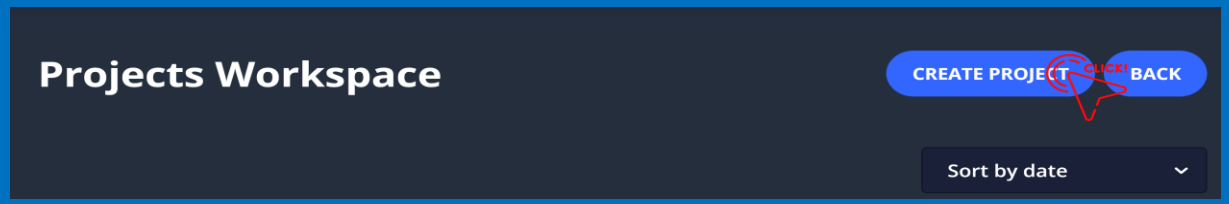


2. Click on the projects section



3. Click on the workspace section



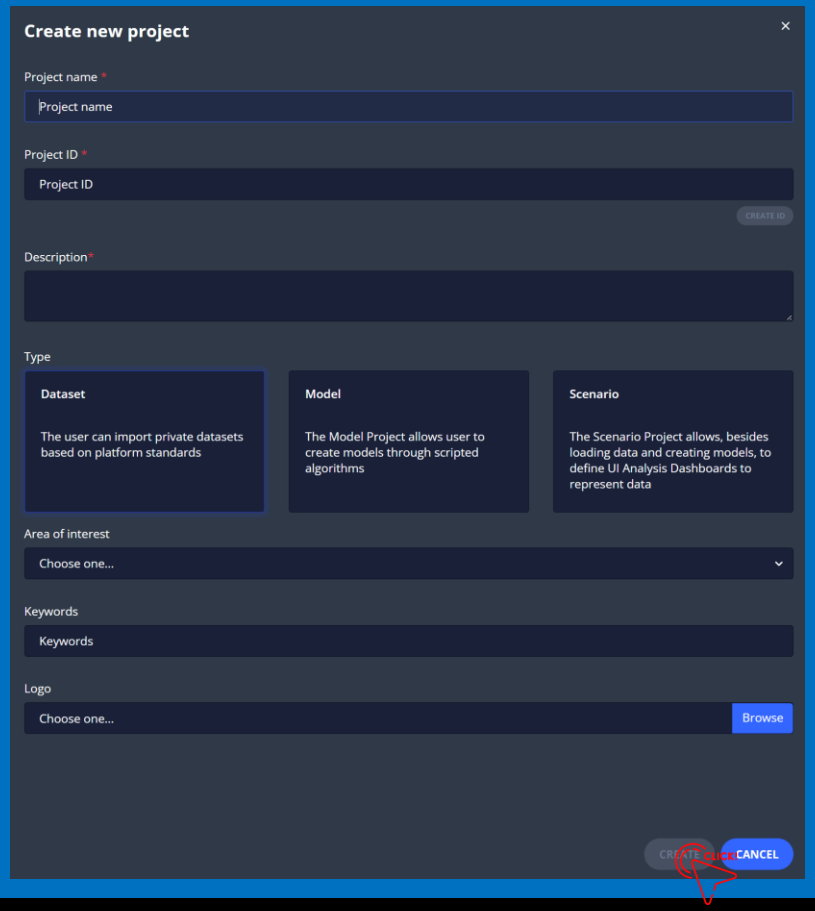


4. Click on the create project button

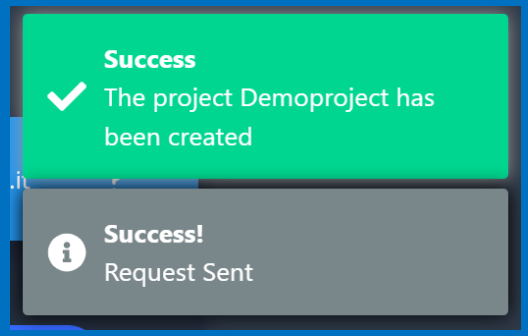
Project creation

5. Fill the form

6. Click on the create button



A request is then sent to create the project.
You will receive a notification that the project has been created like the one shown



Dataset

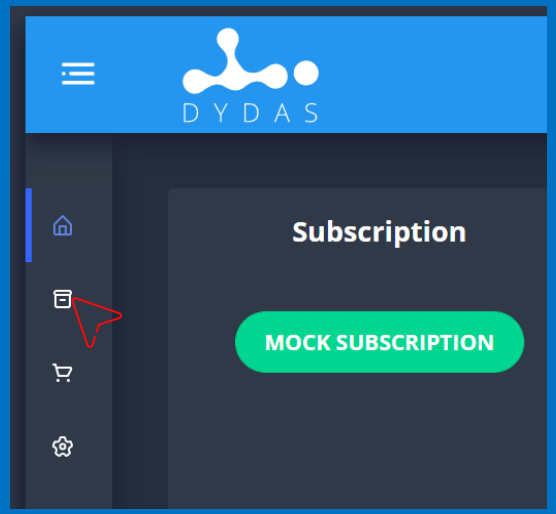
The dataset is the container of data, and it behaves as a database table.
Datasets define the data schema, metadata, and other information regarding the data.

Datasets have a “type” property: this allows to identify the tools that can be used to analyse and view data, and the formats of data. Geographical data have in addition other features that allow to correctly evaluate data quality and optimization.

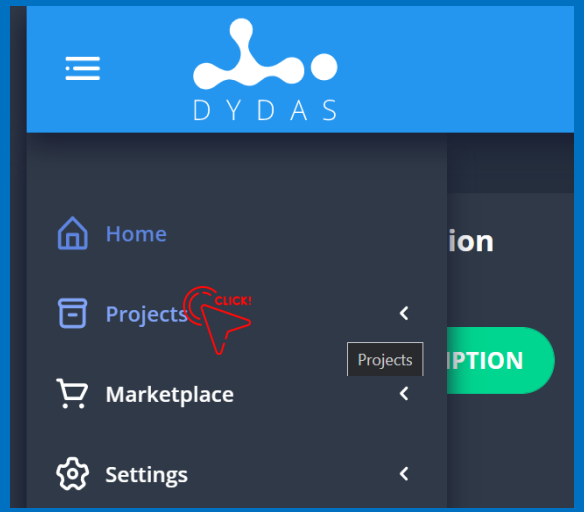
Datasets are the only way to store data into the platform: this includes model elaboration output, proprietary of open data, or any information the user needs to use for their analysis.

Dataset creation

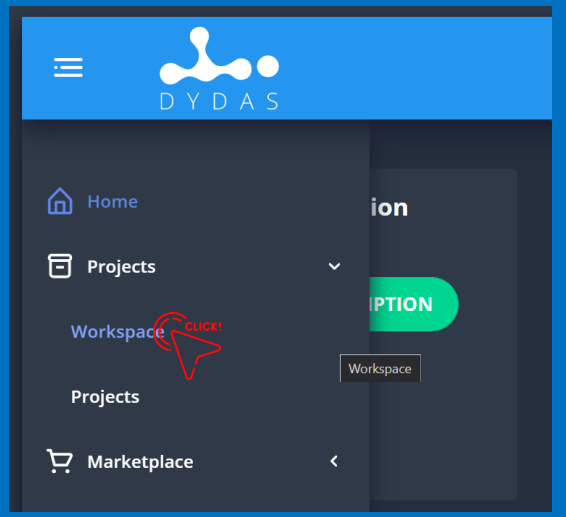
1. Place the mouse in the project icon



2. Click on the projects section



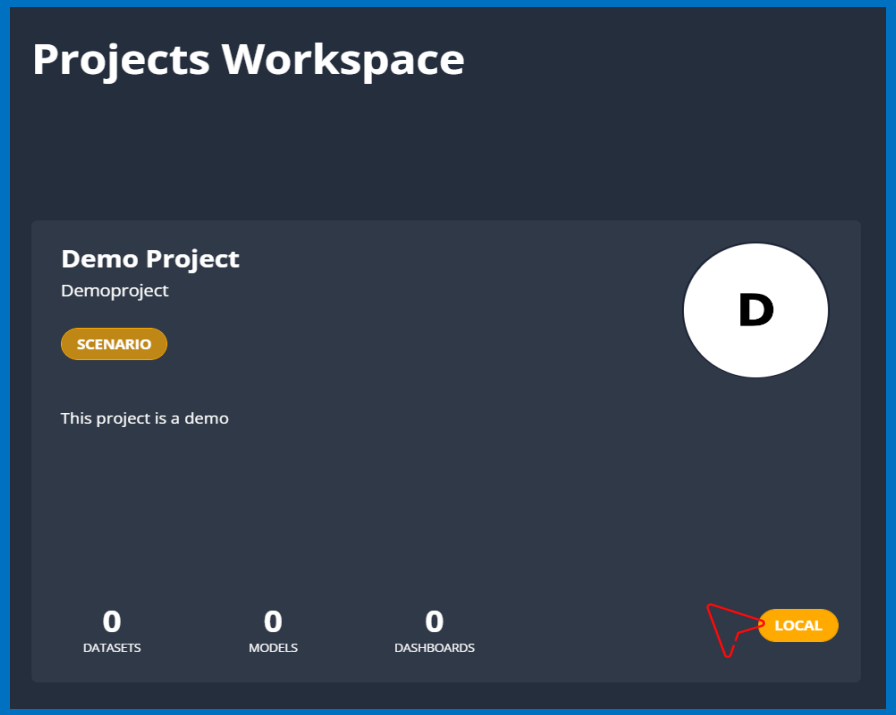
3. Click on the workspace section



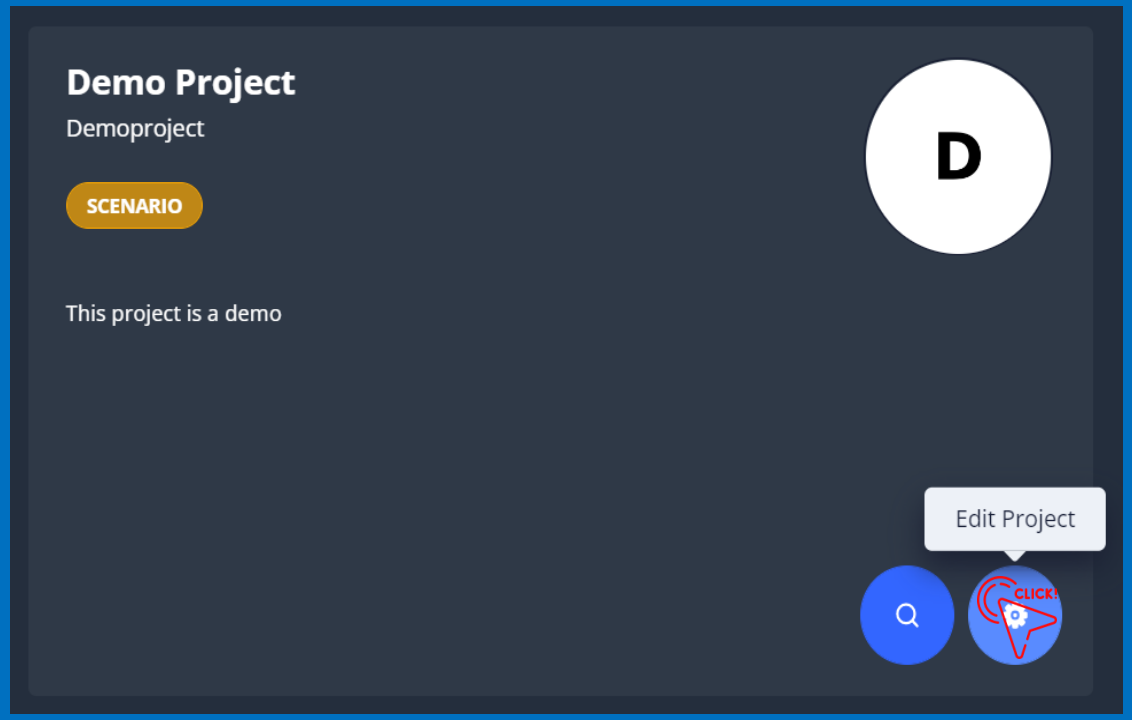
Co-financed by the Connecting Europe Facility of the European Union

Dataset creation

4. Place the mouse in the project card

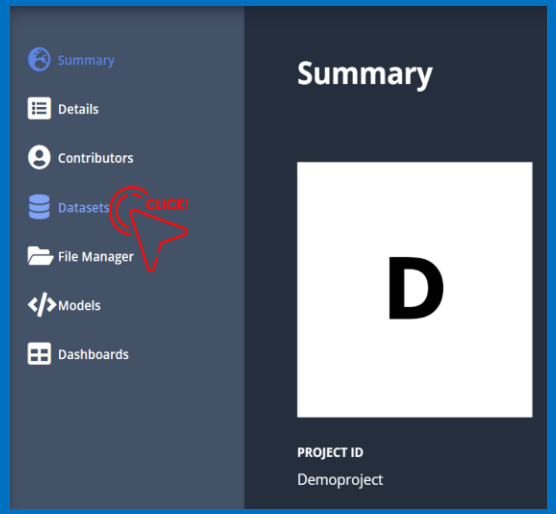


5. Click on the edit project button

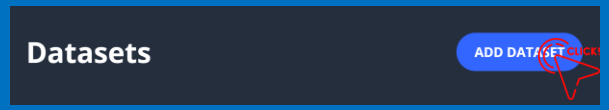


Dataset creation

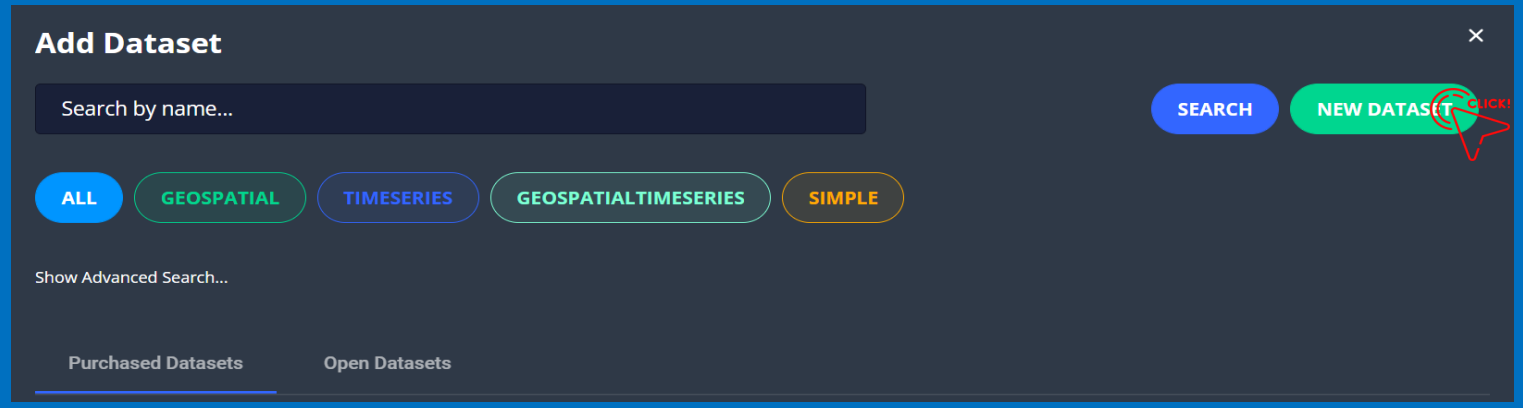
6. Click on the datasets section



7. Click on the add dataset button



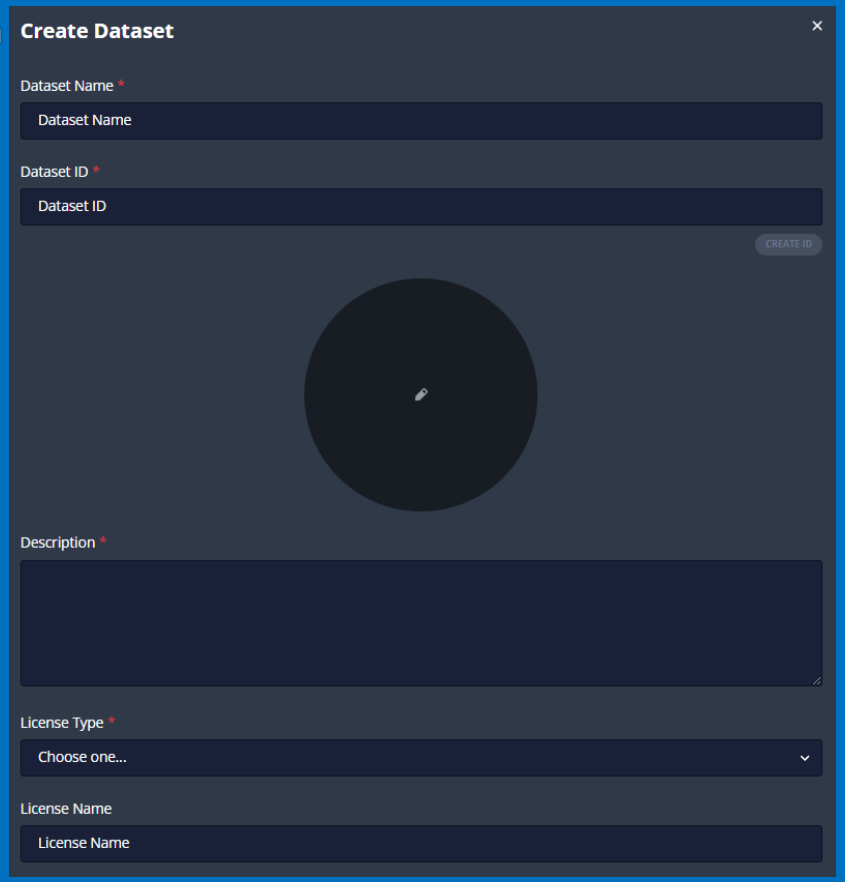
8. Click on the new dataset button



Co-financed by the Connecting Europe Facility of the European Union

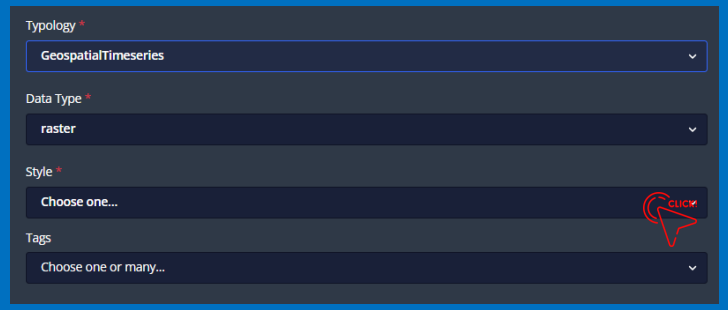
Dataset creation

9. Fill the form

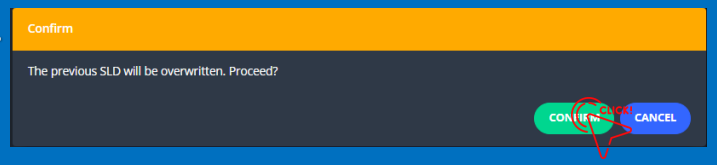


Dataset creation – Style SLD

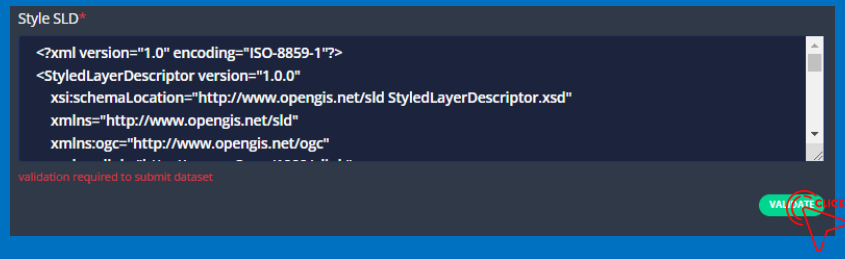
If you choose a **Geospatial** or a **GeospatialTimeseries** as typology you have to select a style by clicking on the style attribute.



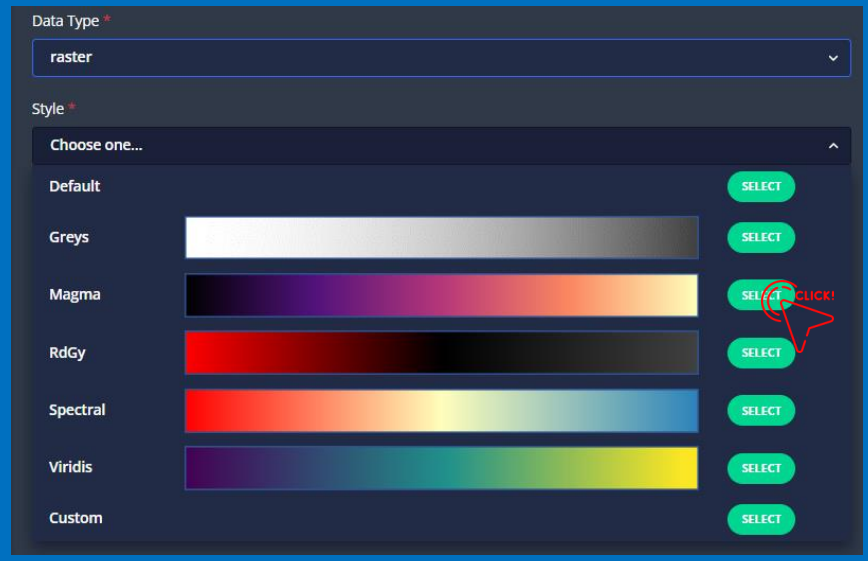
Click on the “confirm” button.



Click on the validate button.



To choose the preferred style click on the select button.



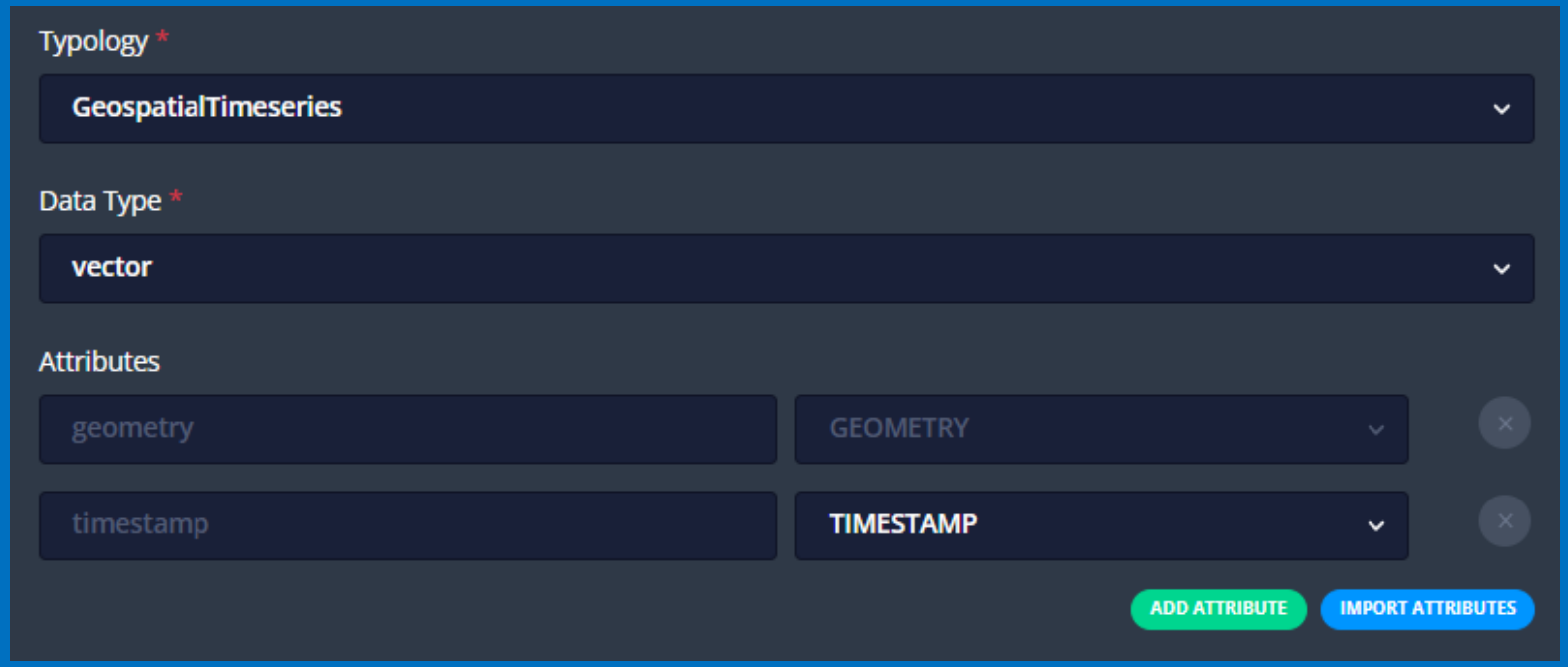
Co-financed by the Connecting Europe Facility of the European Union

Dataset creation – Attributes

If you choose a **Geospatial** or a **GeospatialTimeseries** as typology and selected vector as data type or you choose **Timeseries** or **Simple** as typology you have to declare the attributes in your dataset.

Some of the attributes are mandatory so cannot be changed.

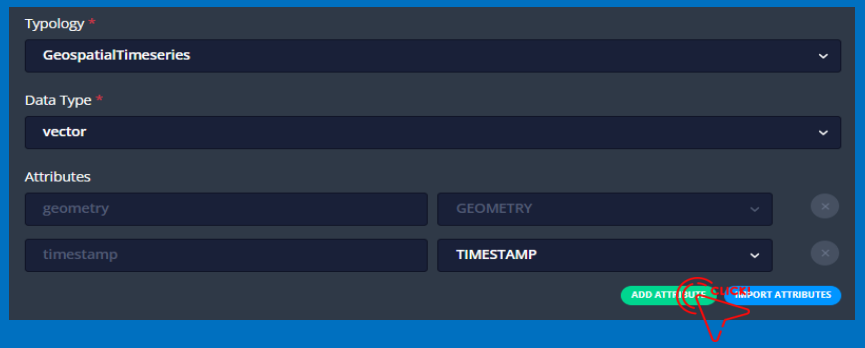
You can add an attribute by clicking on the add attribute button or import the attributes by clicking on the import attributes button.



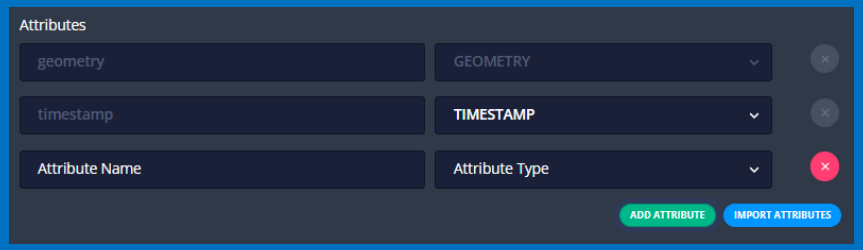
IMPORTANT!!! – These attributes cannot be changed after you created the dataset.

Dataset creation – Attributes

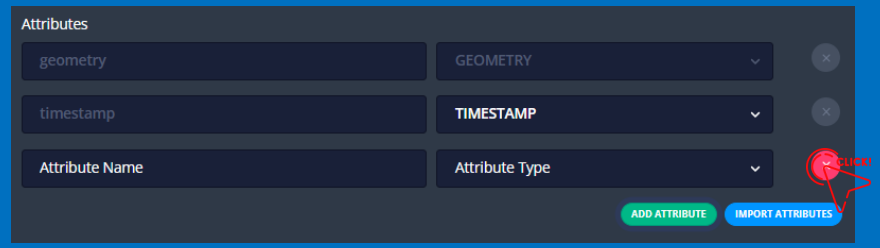
To add an attribute click on the add attribute button.



Fill attribute name and select the attribute type.

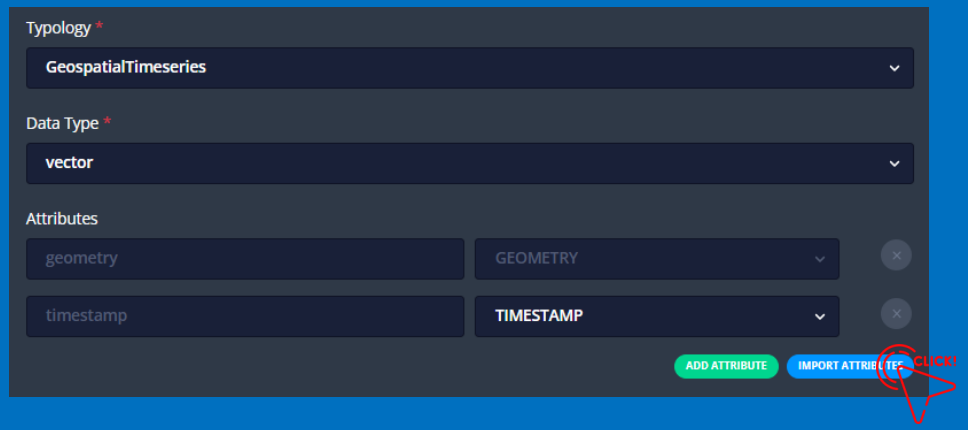


To remove an attribute click on the remove attribute button.

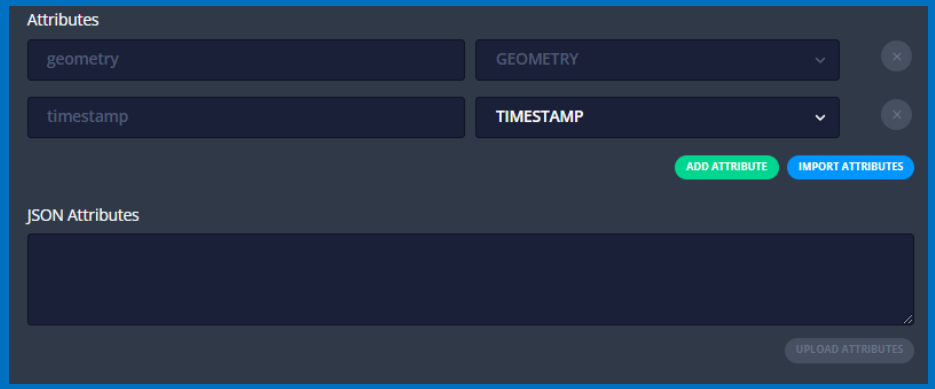


Dataset creation – Attributes

To import attributes click on the import attributes button.



A text input will appear where you have to put the attributes in JSON format.



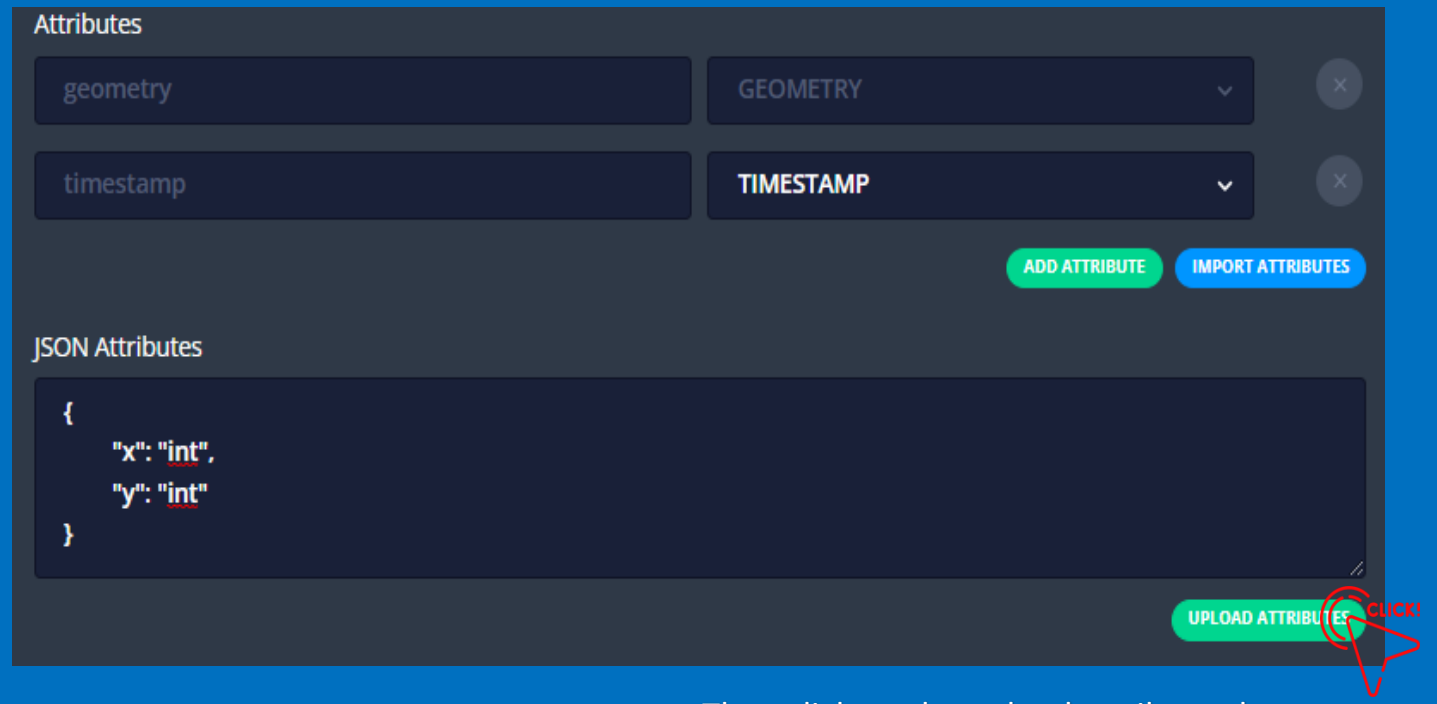
Dataset creation – Attributes

The JSON should be formatted using the following criteria:

```
{
  "Attribute name 1": "Attribute type 1",
  "Attribute name 2": "Attribute type 2"
}
```

The attribute type should be one of the following:

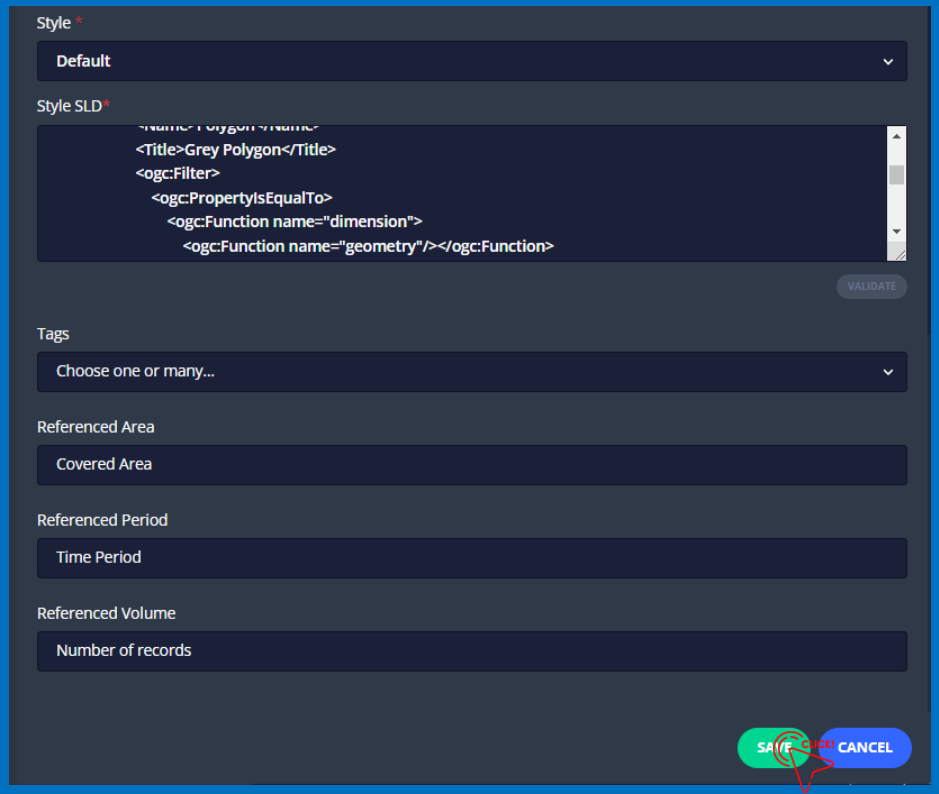
- STRING
- INT
- LONG
- DOUBLE
- DATE
- TIMESTAMP
- BOOLEAN
- GEOMETRY



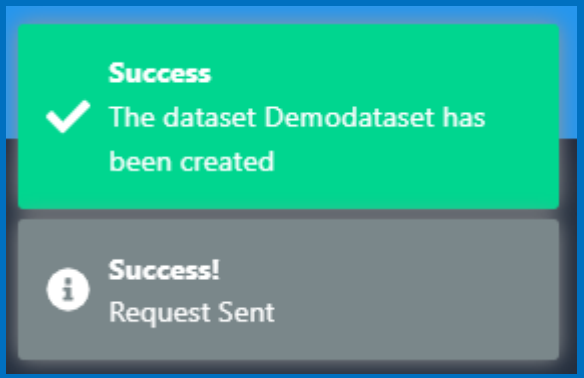
Then click on the upload attributes button.

Dataset creation

10. Click on the save button



A request is then sent to create the dataset.
You will receive a notification that the dataset has been created like the one shown.



Dataset creation – Typologies

Geospatial – This typology is used to store a geospatial referenced dataset.

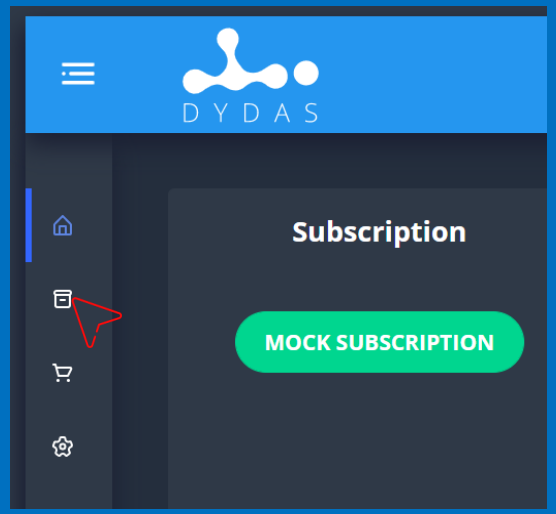
Timeseries – This typology is used to store a timeseries referenced dataset.

GeospatialTimeseries – This typology is used to store a geospatial and timeseries referenced dataset.

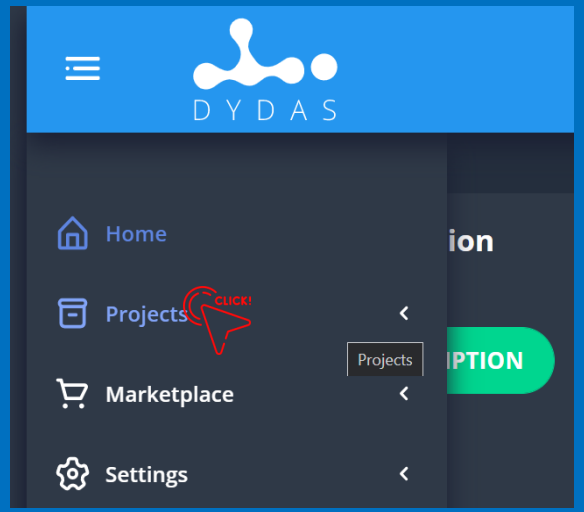
Simple – This typology is used to store a dataset that is not geospatial or timeseries referenced.

Dataset import

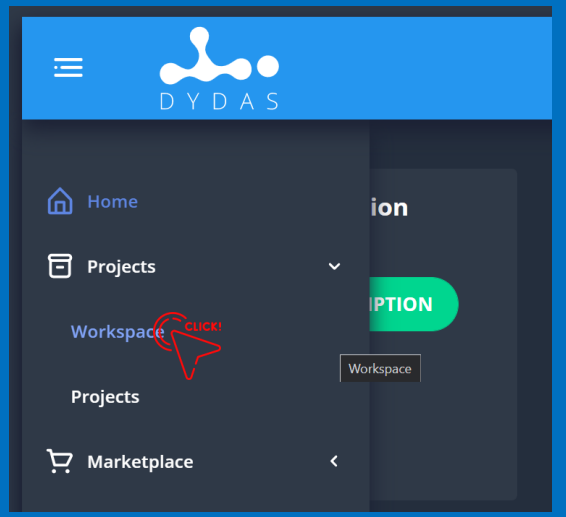
1. Place the mouse in the project icon



2. Click on the projects section



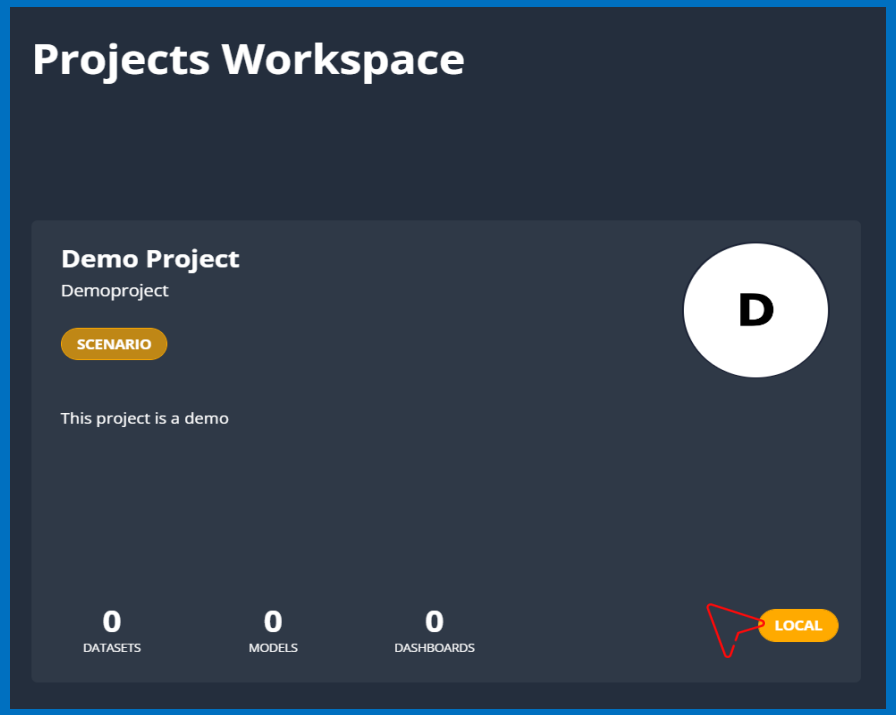
3. Click on the workspace section



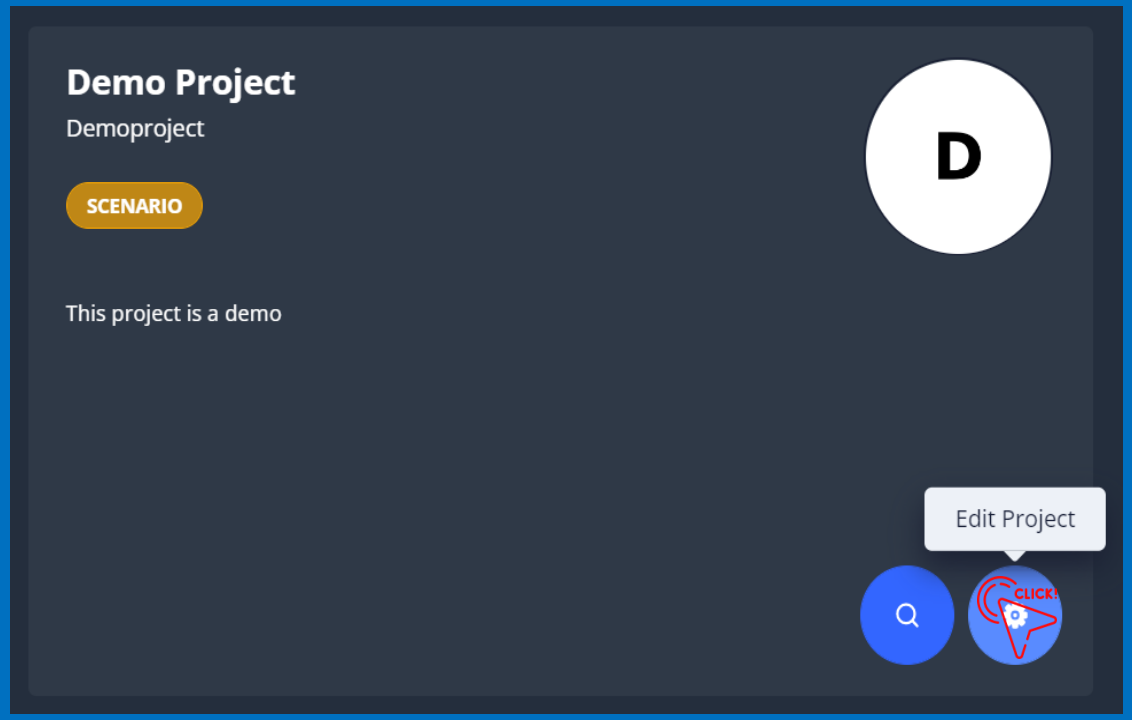
Co-financed by the Connecting Europe Facility of the European Union

Dataset import

4. Place the mouse in the project card

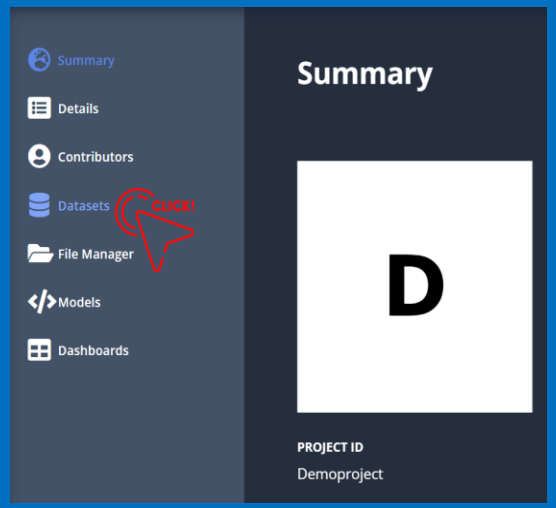


5. Click on the edit project button

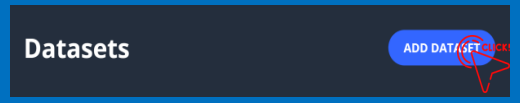


Dataset import

6. Click on the datasets section



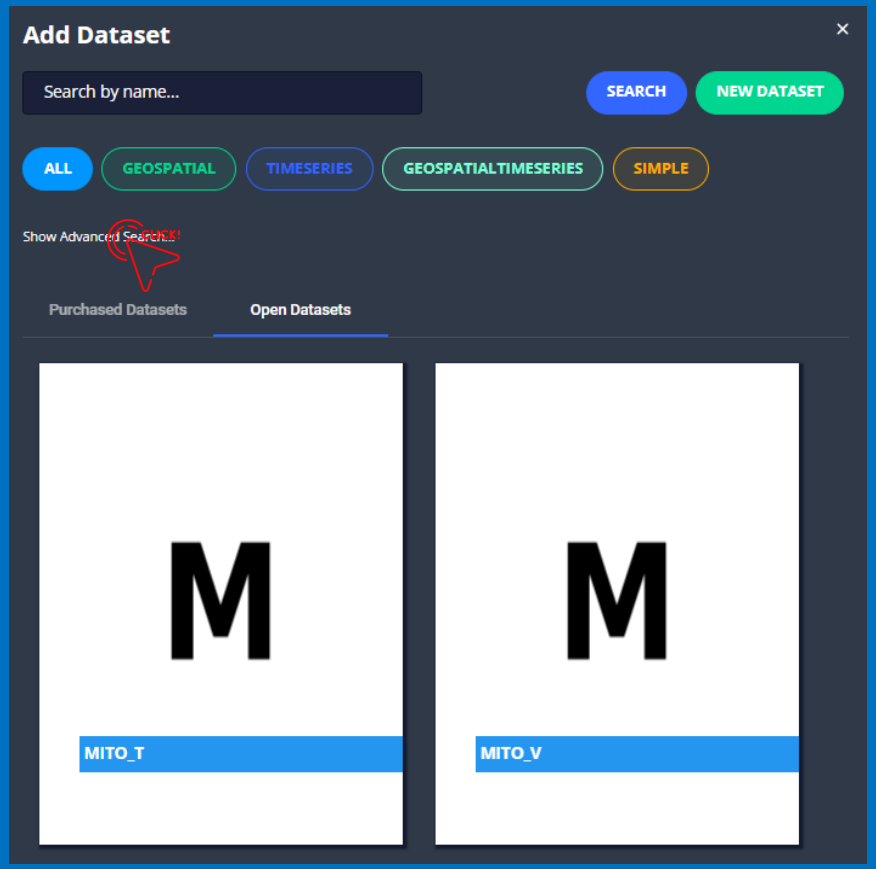
7. Click on the add dataset button



8. In this screen you can search open and purchased datasets to import.

A dataset can be searched by name and by typology.

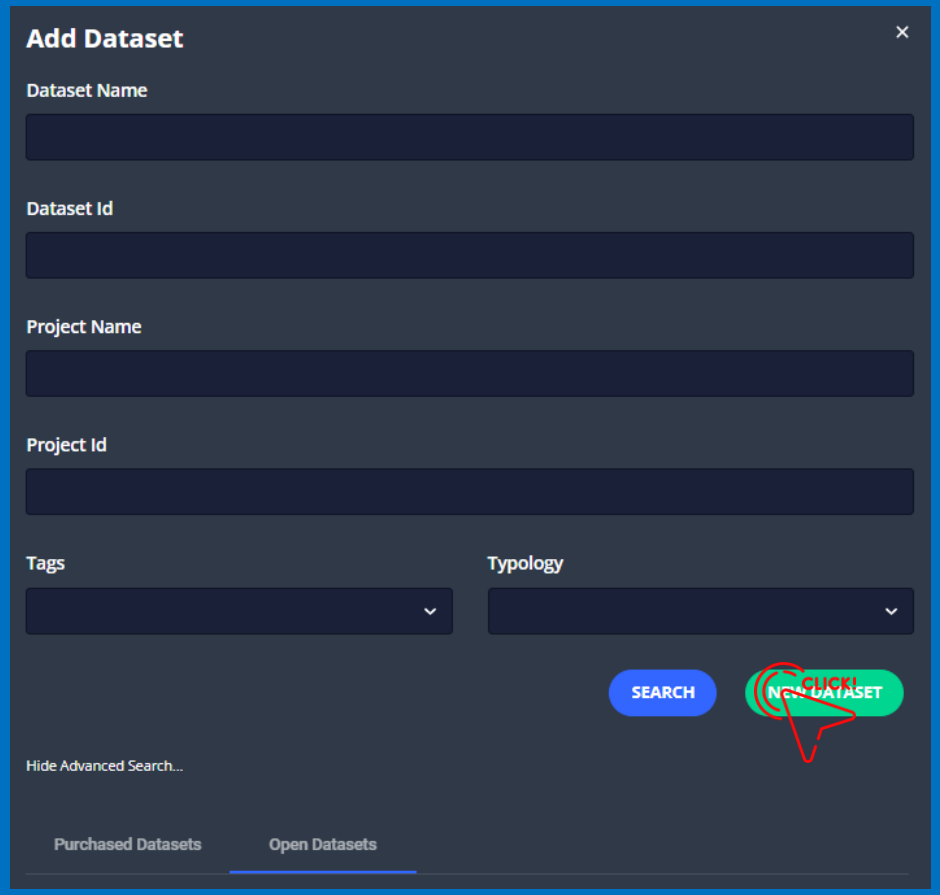
To show more filters click on the show advanced search button.



Co-financed by the Connecting Europe Facility of the European Union

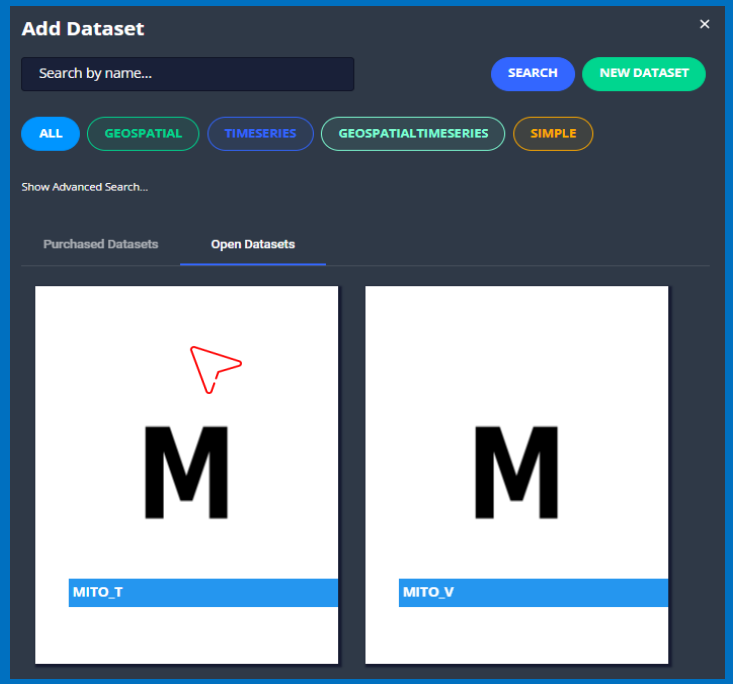
Dataset import

9. Fill the fields you want to filter and click on the search button.



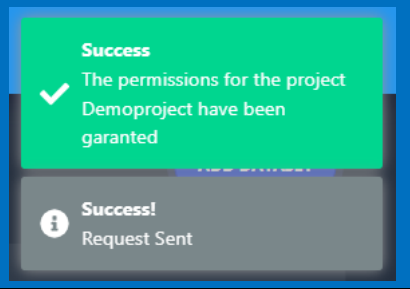
Dataset import

10. To import a dataset hover the mouse on the preferred dataset



11. Click on the add button

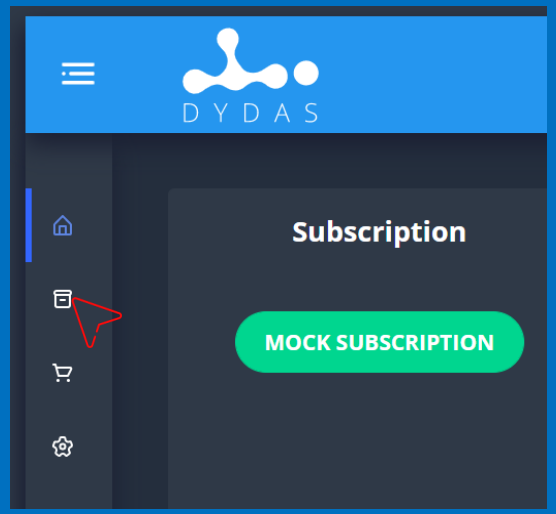
A request is then sent to grant you the permissions to use that dataset.
You will receive a notification that the permissions are granted like the one shown on the right.



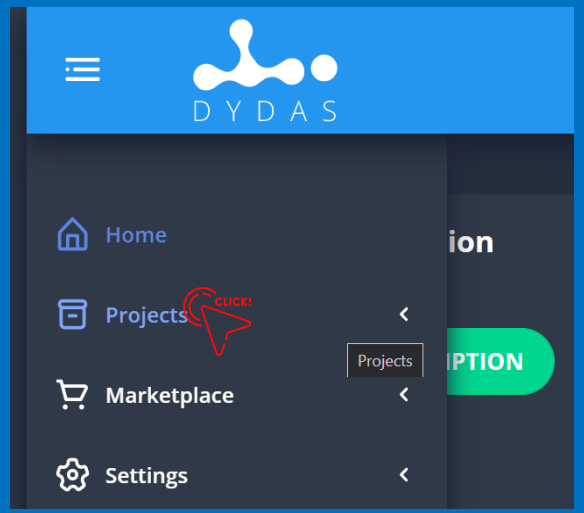
Co-financed by the Connecting Europe Facility of the European Union

Dataset upload

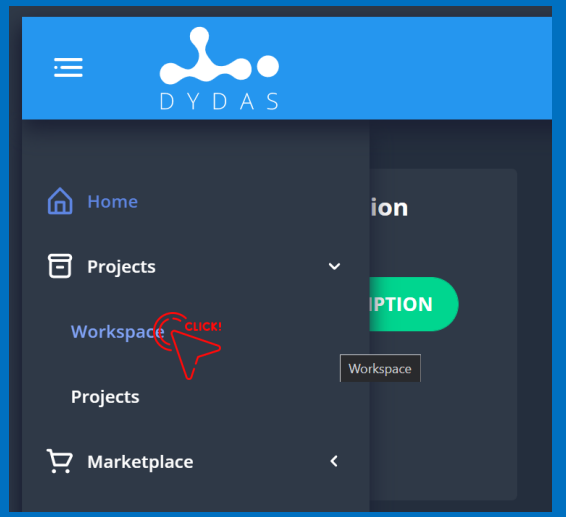
1. Place the mouse in the project icon



2. Click on the projects section



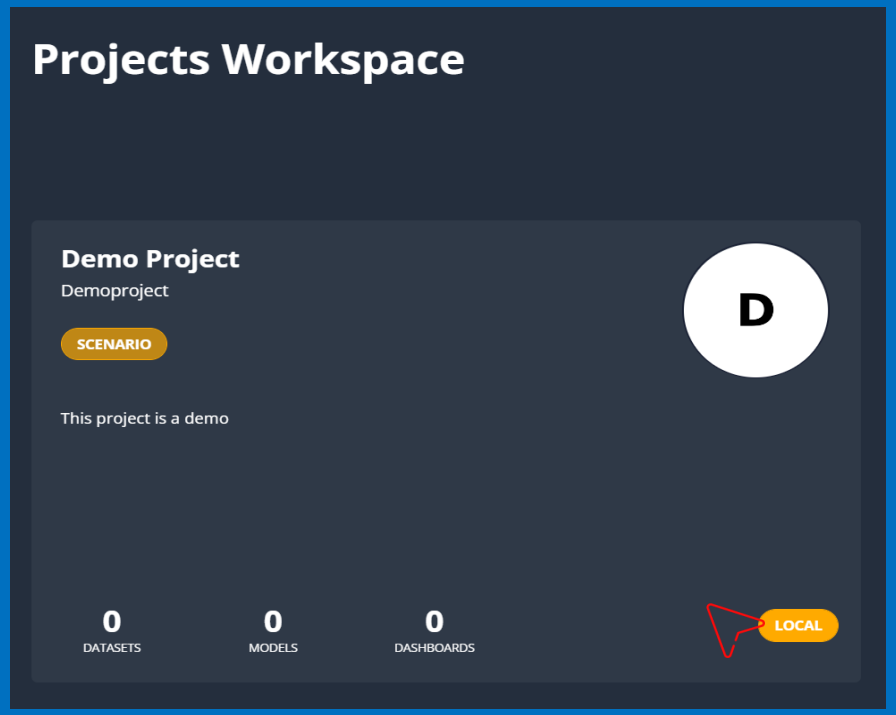
3. Click on the workspace section



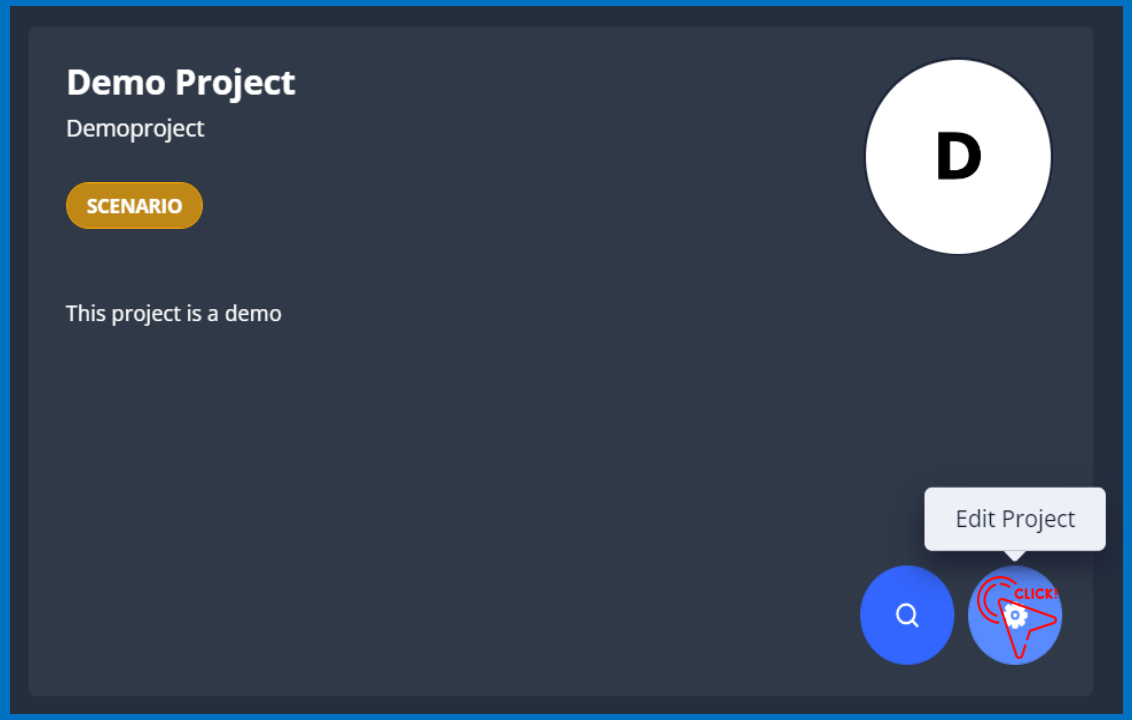
Co-financed by the Connecting Europe Facility of the European Union

Dataset upload

4. Place the mouse in the project card

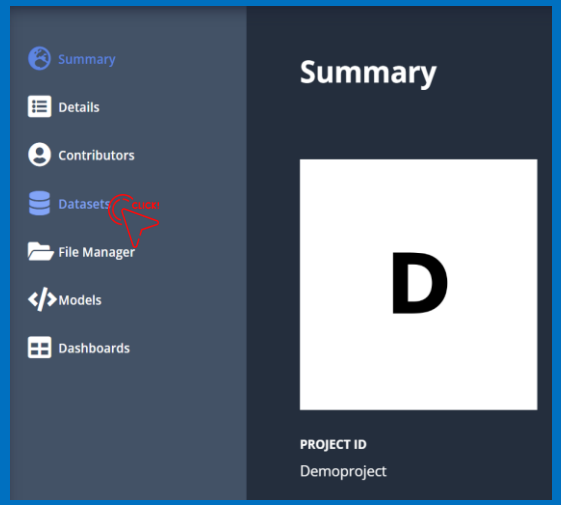


5. Click on the edit project button

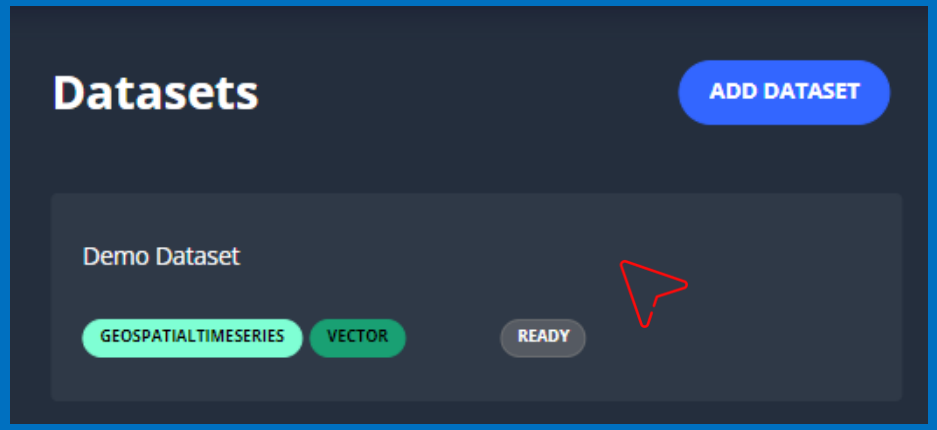


Dataset upload

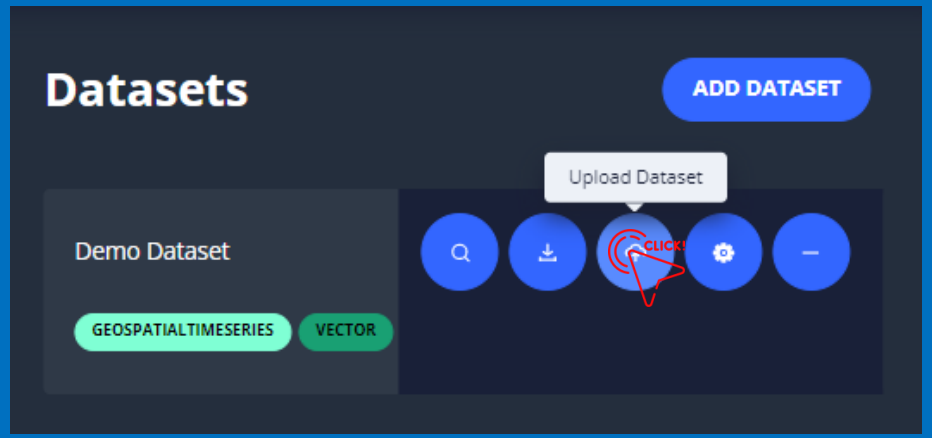
6. Click on the datasets section



7. Hover the mouse over the dataset

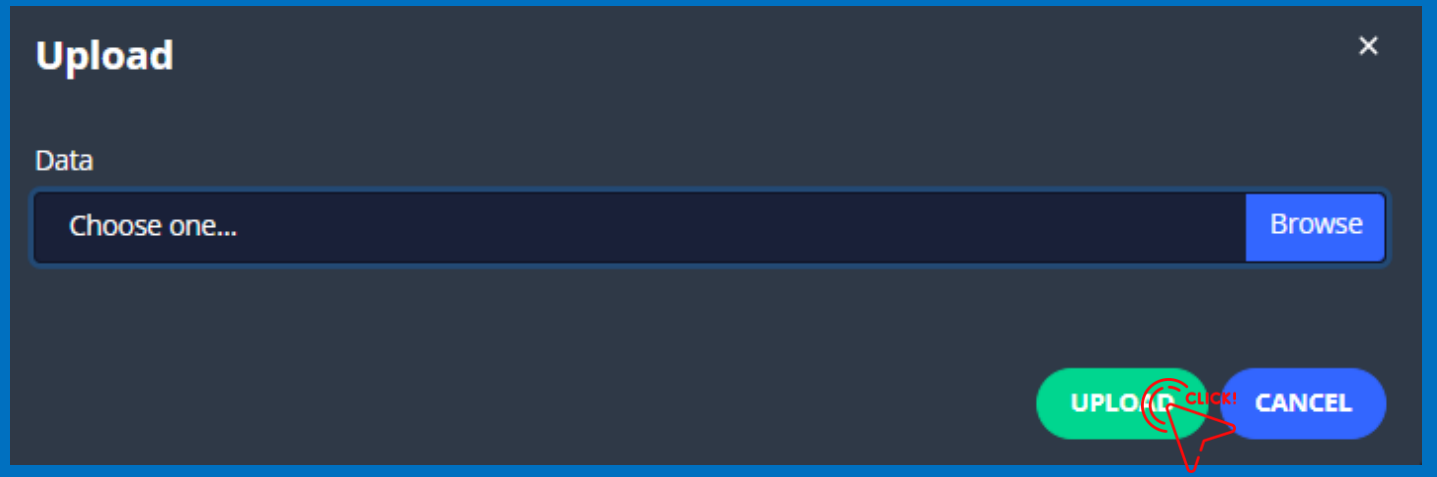


8. Click on the upload dataset button



Dataset upload

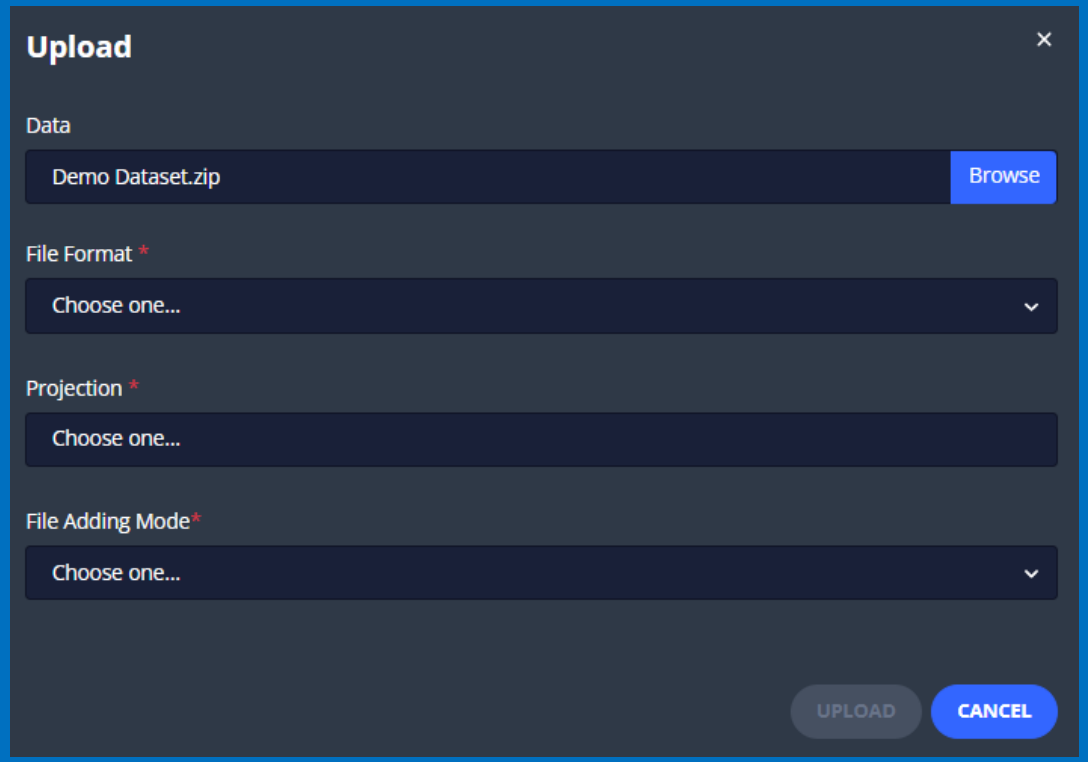
9. The upload dialogue will appear. Click on the data field and choose the file of the dataset you want to upload.



IMPORTANT!!! The file should be inside a zip file.
If you choose a **GeospatialTimeseries** typology with raster as datatype the name of the file inside the zip should be **timestamp.fileformat** (ex. 1614962038000.tif)

Dataset upload

10. Fill the other fields



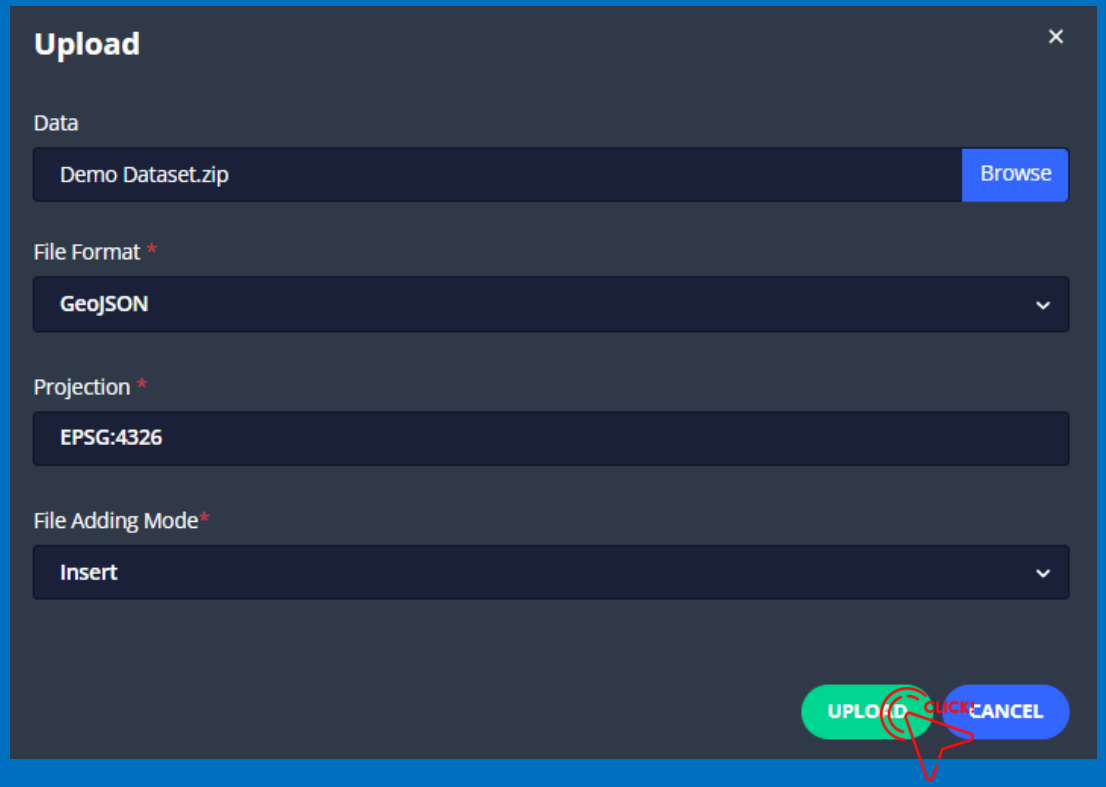
For projection you can digit the numbers and click on the projection (don't leave only numbers, add "EPSG:" before the numbers or it will throw error).

For file adding mode there are 3 choice:

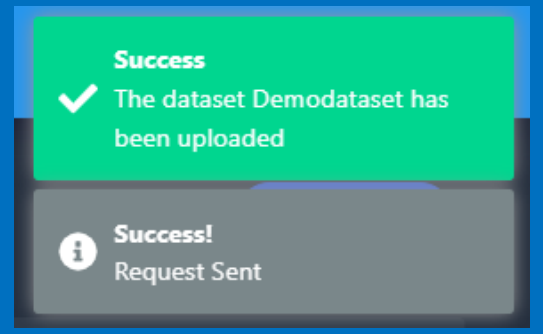
- *Append* – to append the dataset over the existing one;
- *Insert* – the dataset will be overwritten if not present, otherwise an error is thrown;
- *Overwrite* – to overwrite the existing dataset if present.

Dataset upload

11. Click on upload

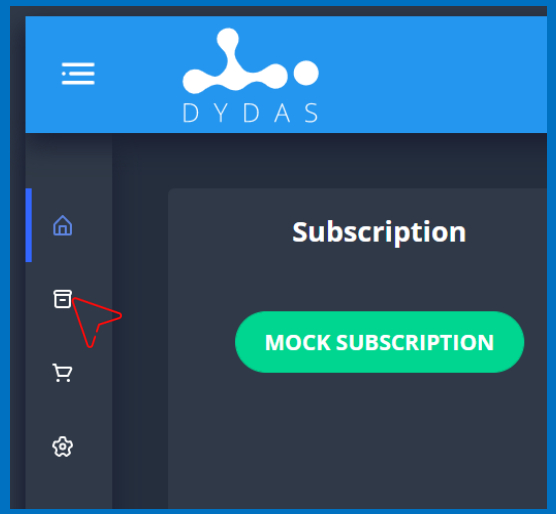


A request is then sent to upload the dataset.
You will receive a notification that the dataset has been uploaded like the one shown.

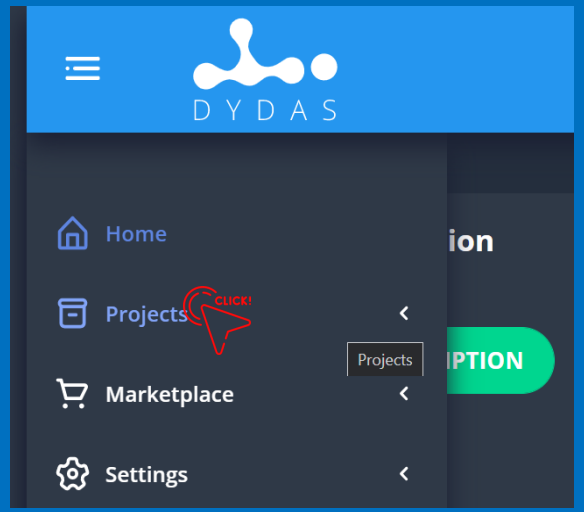


Dataset download

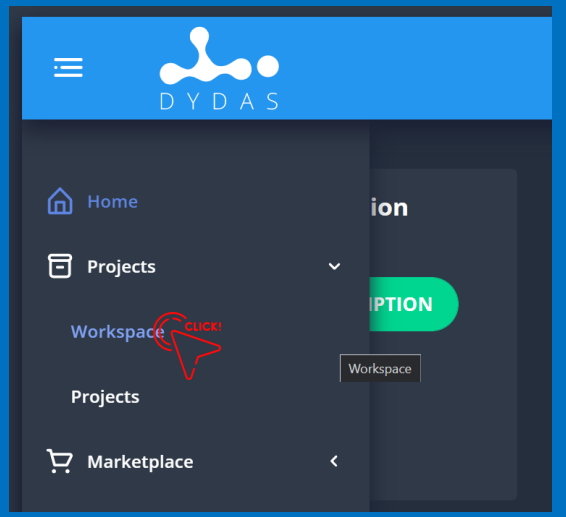
1. Place the mouse in the project icon



2. Click on the projects section



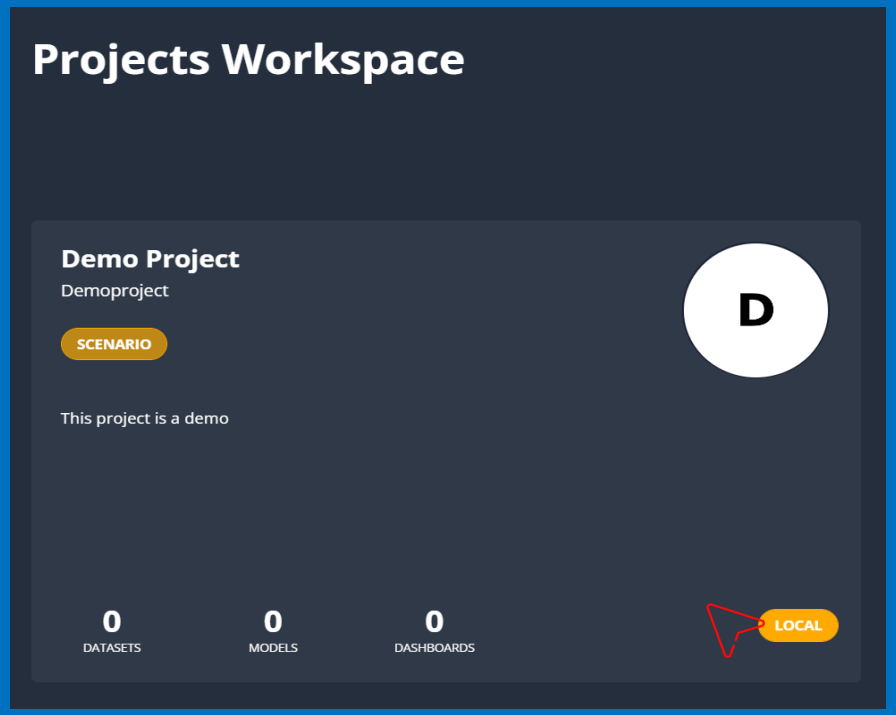
3. Click on the workspace section



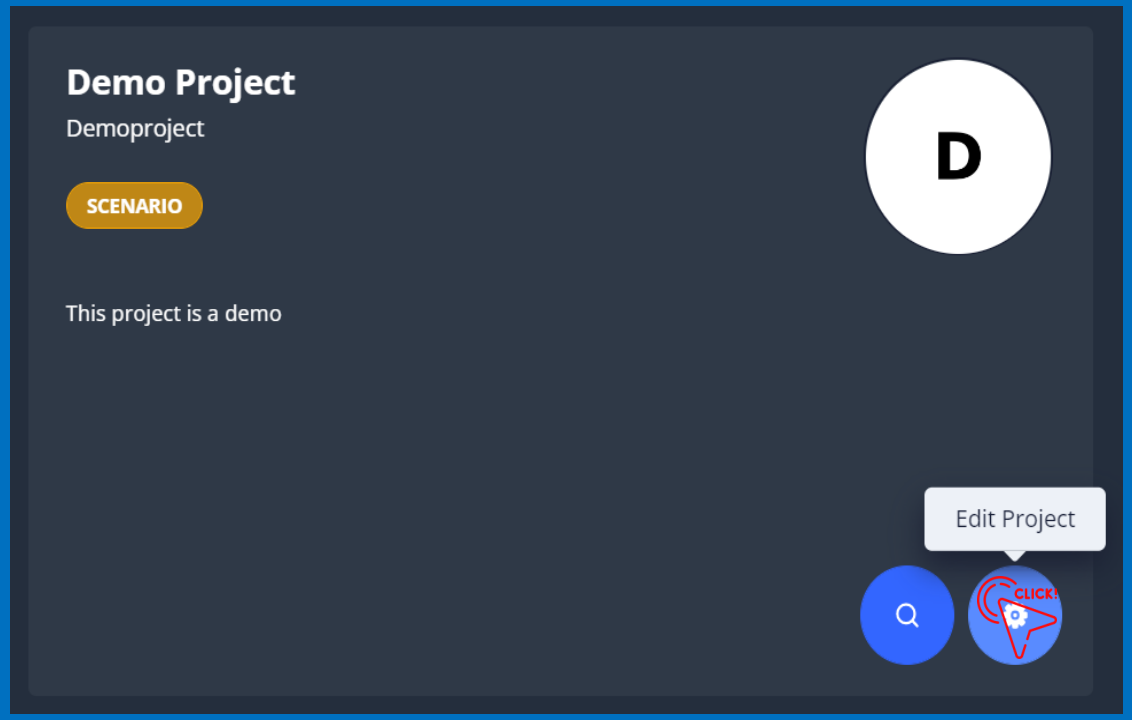
Co-financed by the Connecting Europe Facility of the European Union

Dataset download

4. Place the mouse in the project card

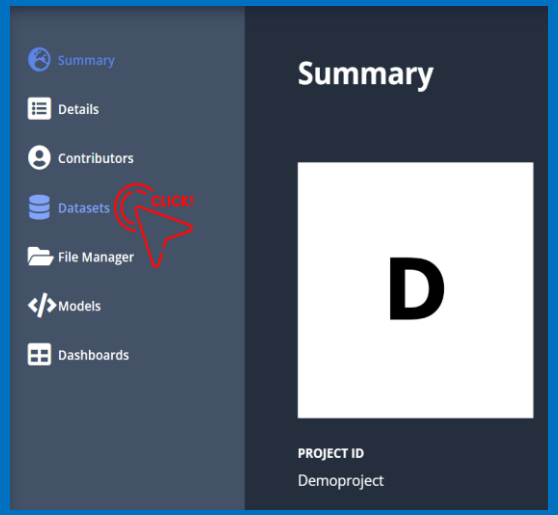


5. Click on the edit project button

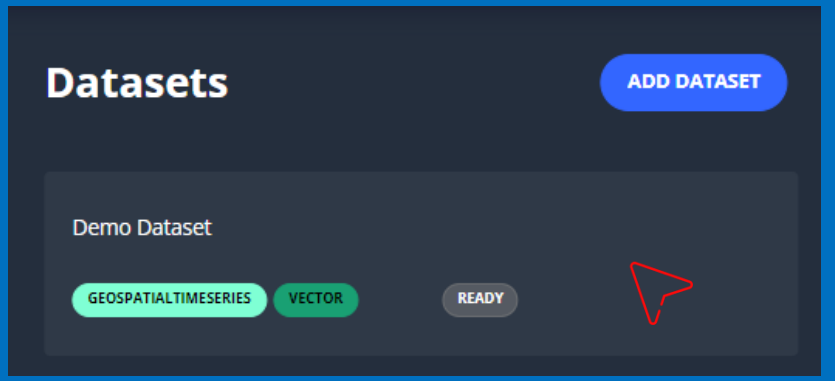


Dataset download

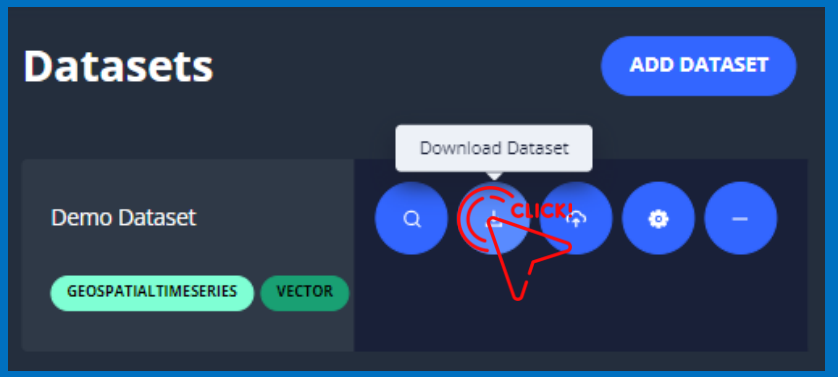
6. Click on the datasets section



7. Hover the mouse over the dataset

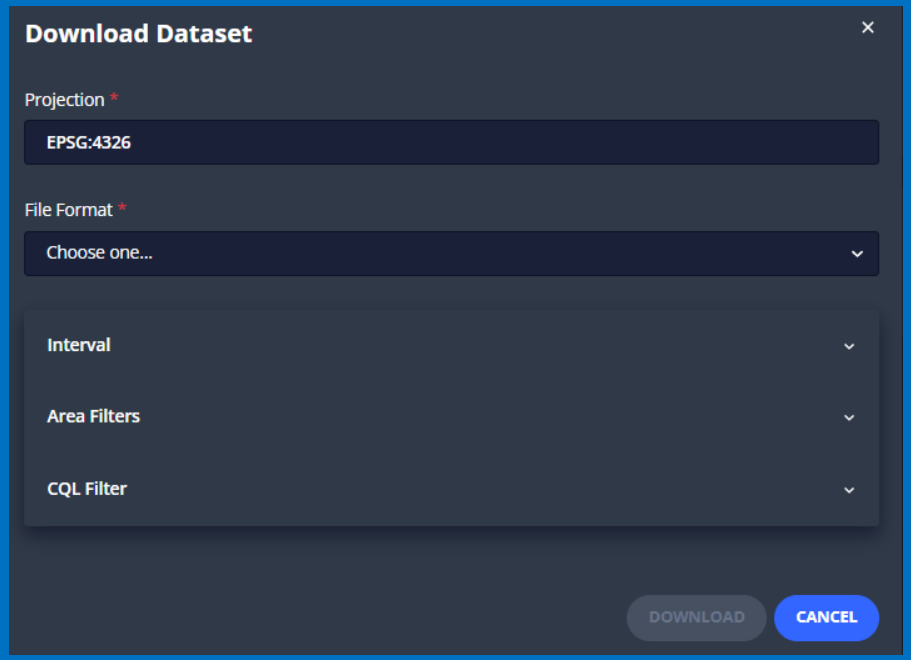


8. Click on the download dataset button



Dataset download

9. The download dialogue will appear. Fill the form.



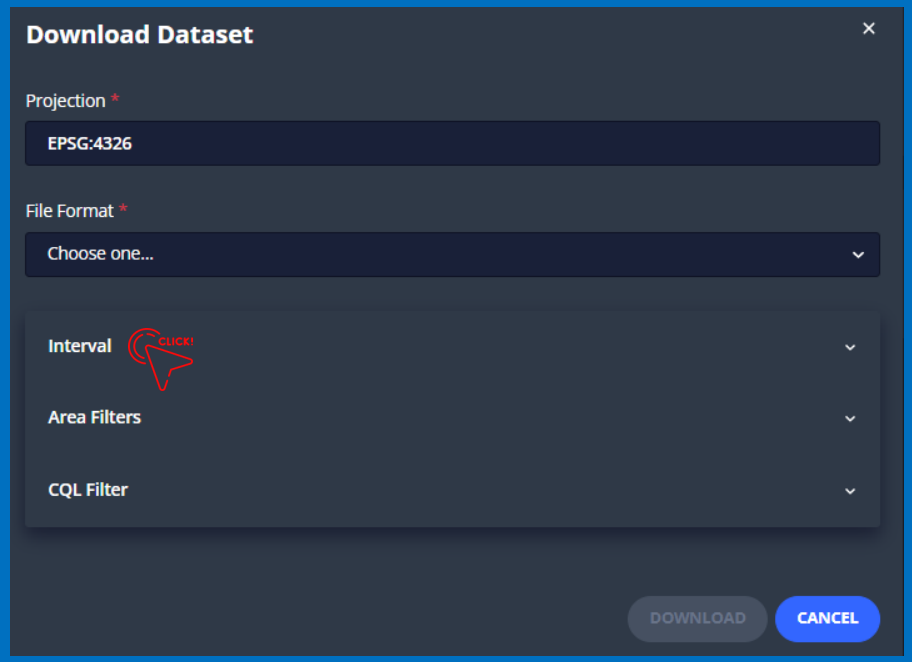
The dataset can filtered by interval if the dataset has **Timeseries** or **GeospatialTimeseries** as typology.

The dataset can filtered with area filters if the dataset has **Geospatial** or **GeospatialTimeseries** as typology.

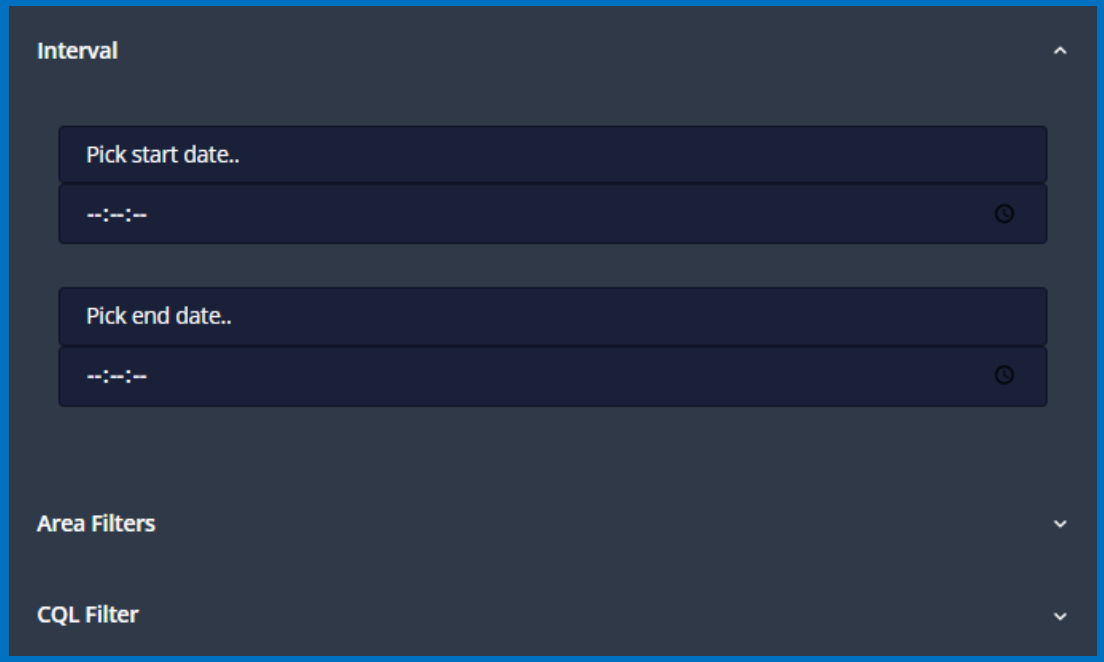
The dataset can also be filtered with a CQL Filter.

Dataset download

To filter by interval, click on the interval section

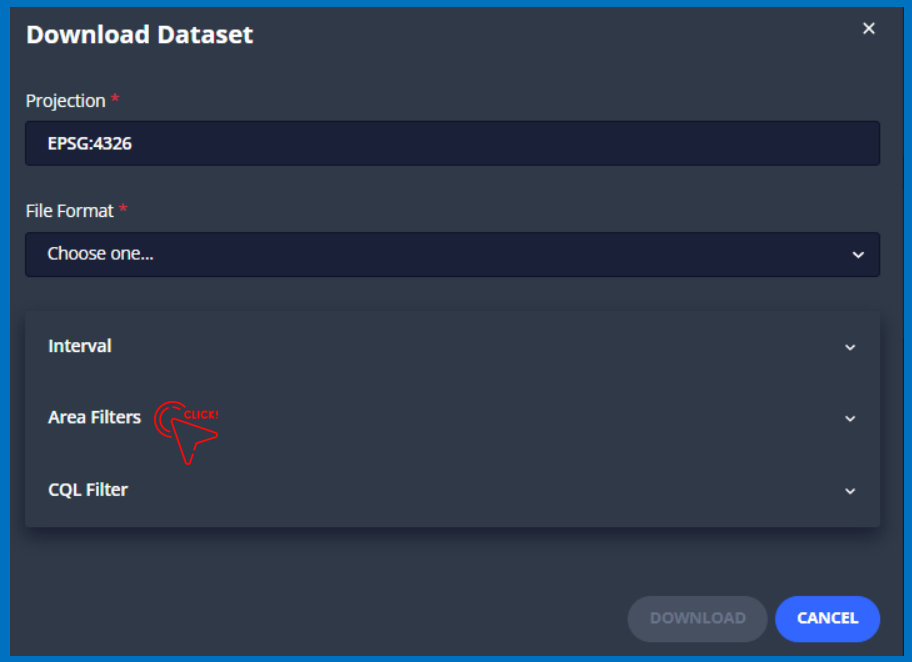


And fill a start date or an end date or both.

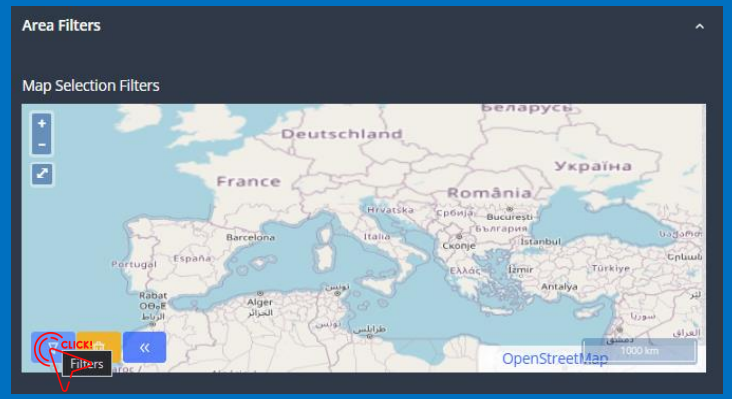


Dataset download

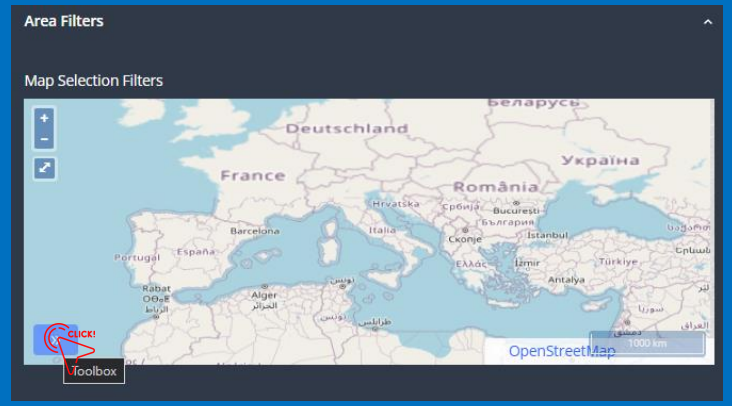
To filter by area click on the area filters section



Click on the filter buttons to draw filters.

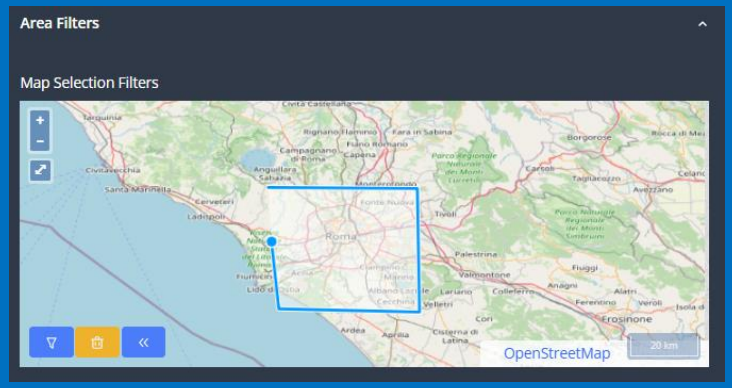


Click on the toolbox button to show the toolbox.

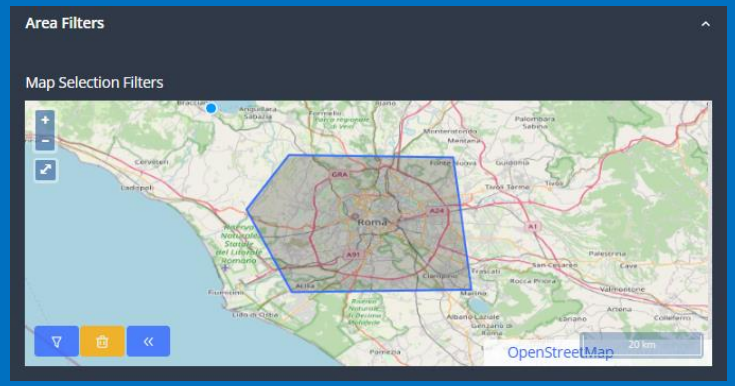


Dataset download

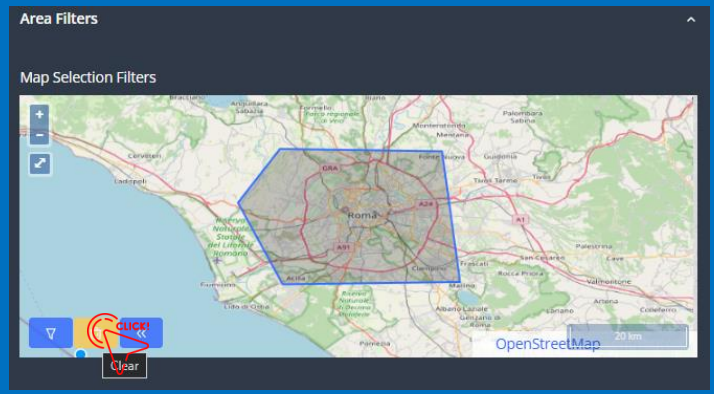
Click on the map to start drawing the filter.



Close the path to create an area to filter.



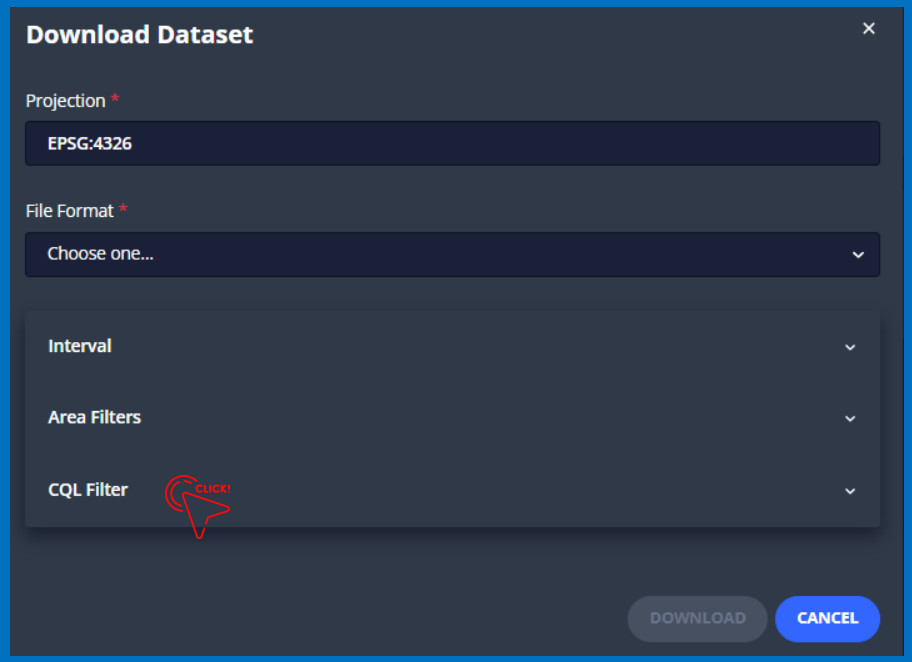
To clear all the filters click on the clear button



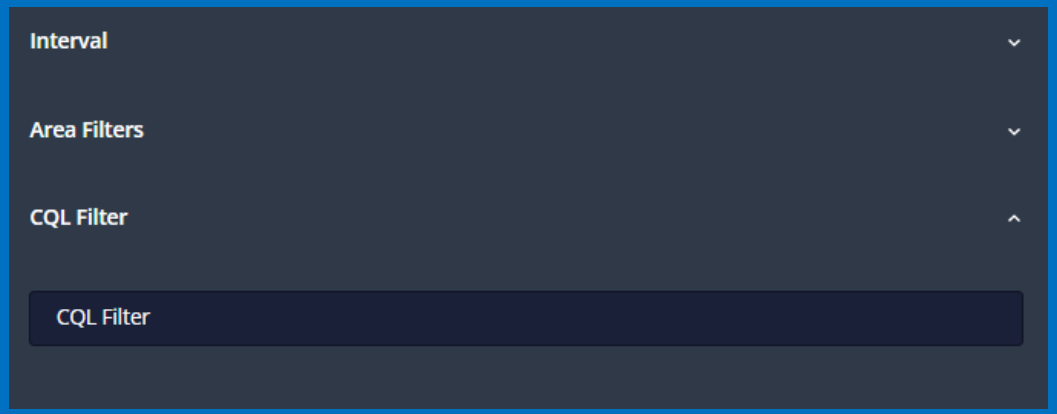
Co-financed by the Connecting Europe Facility of the European Union

Dataset download

To insert a CQL filter click on the CQL filter section

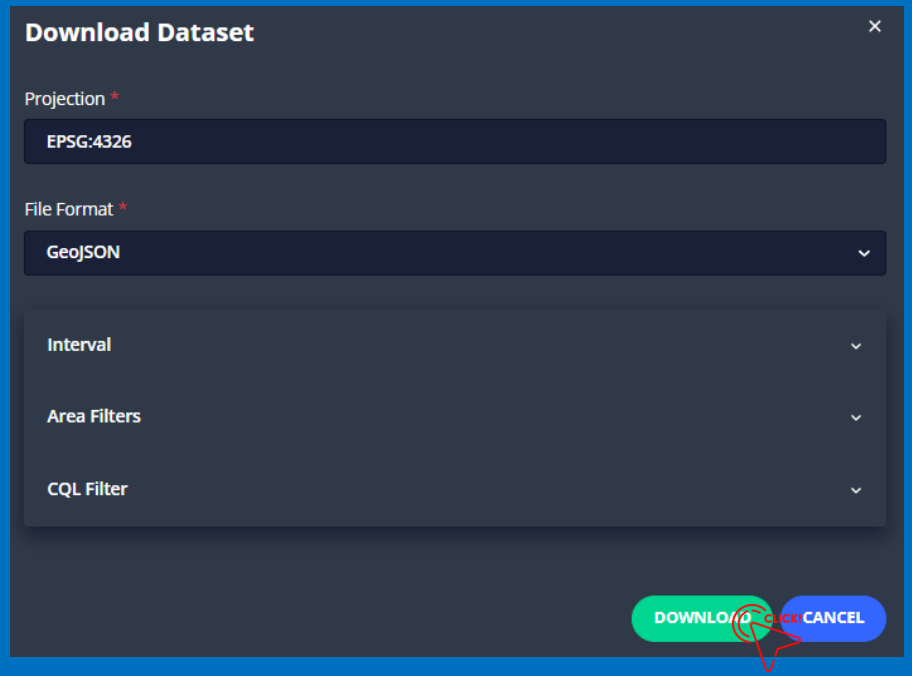


And fill the CQL filter field.



Dataset download

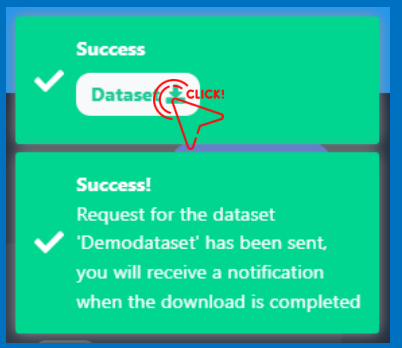
10. Click on the download button to download the dataset.



11. A request is then sent to prepare the download of the dataset.

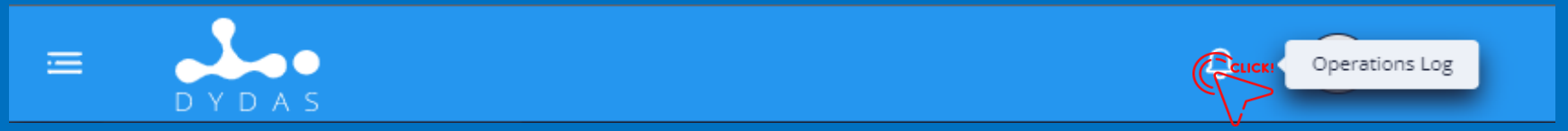
You will receive a notification that the dataset is ready to download like the one shown.

Click on the download dataset button.



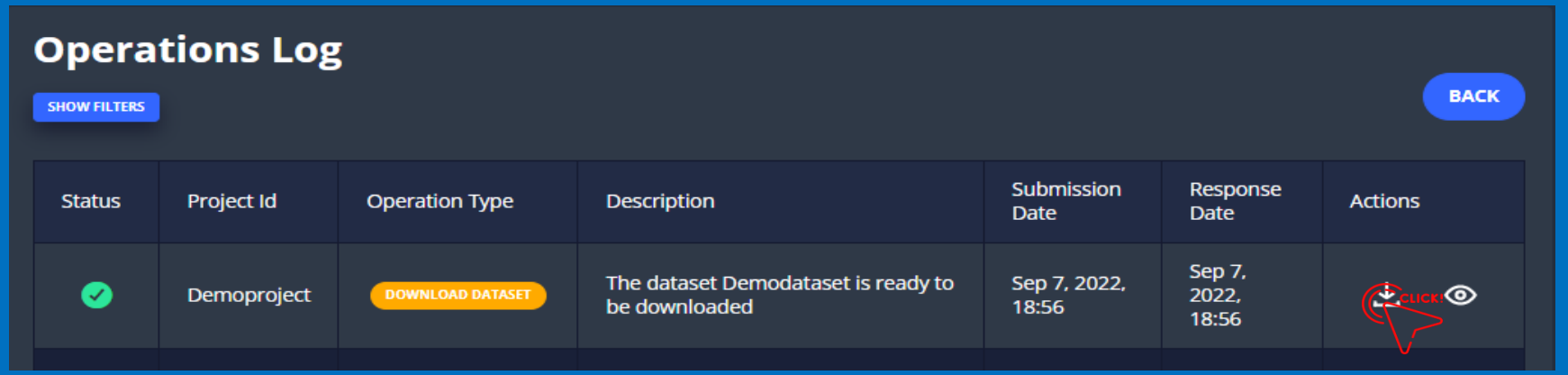
Dataset download

If you miss the notification click on the operation log button in the navbar



Check the operations log and click on the download button.

The download is available only for 24h



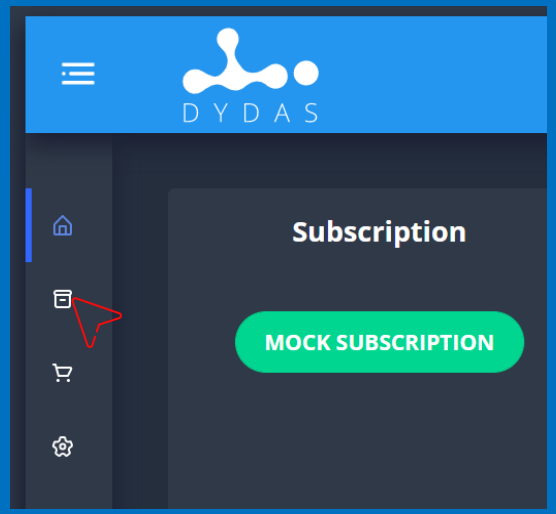
File Manager

The File Manager is a cloud repository that allows users to store folders and files in a project.

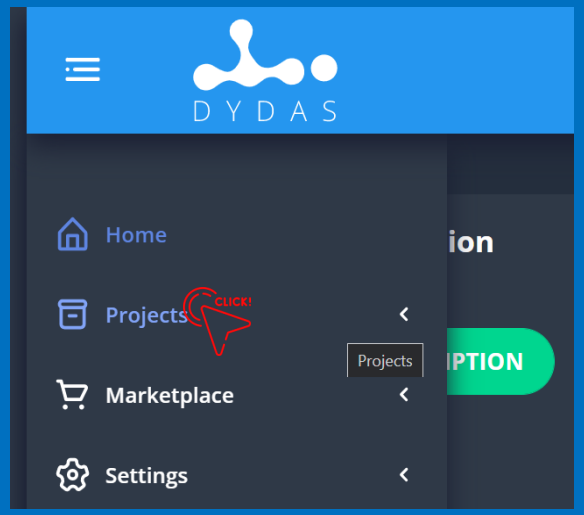
All the files can be retrieved by a python script in a model using the DYDAS package.

File Manager

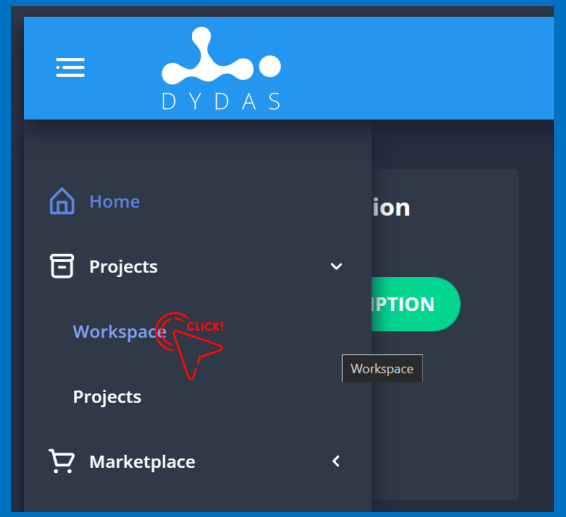
1. Place the mouse in the project icon



2. Click on the projects section



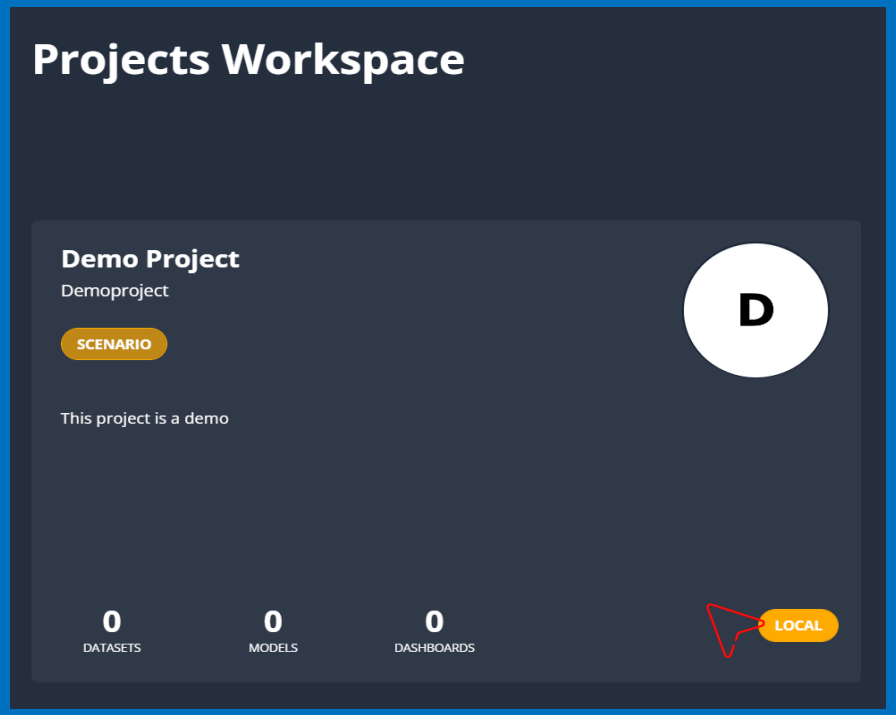
3. Click on the workspace section



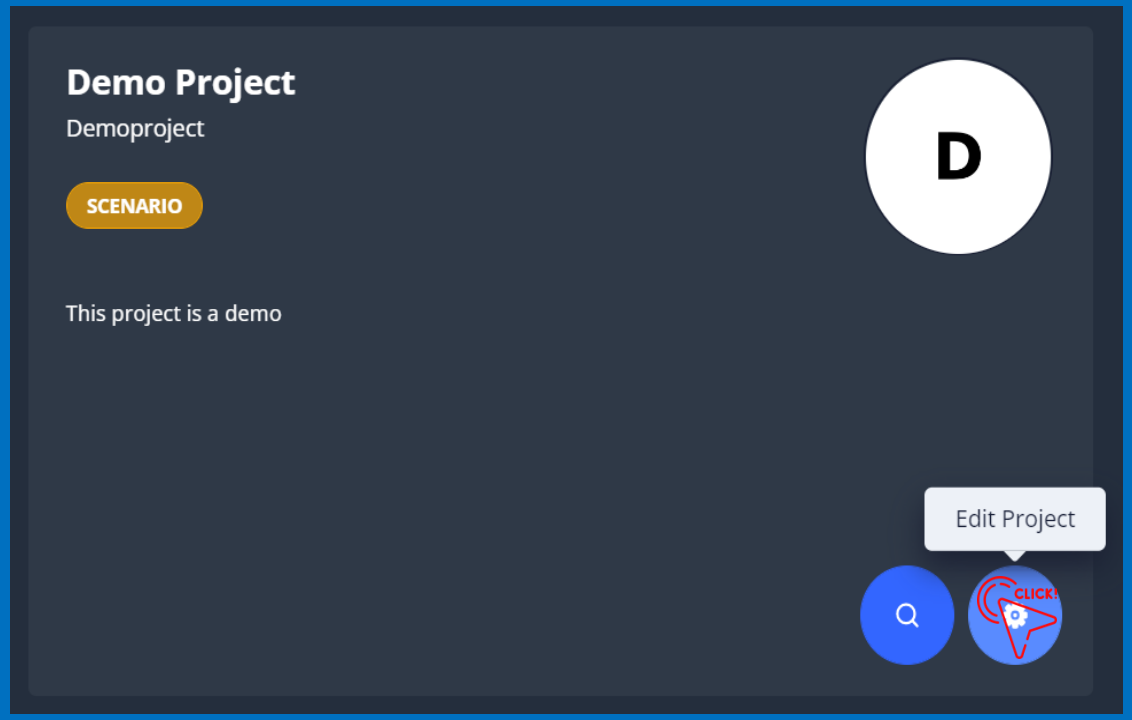
Co-financed by the Connecting Europe Facility of the European Union

File Manager

4. Place the mouse in the project card

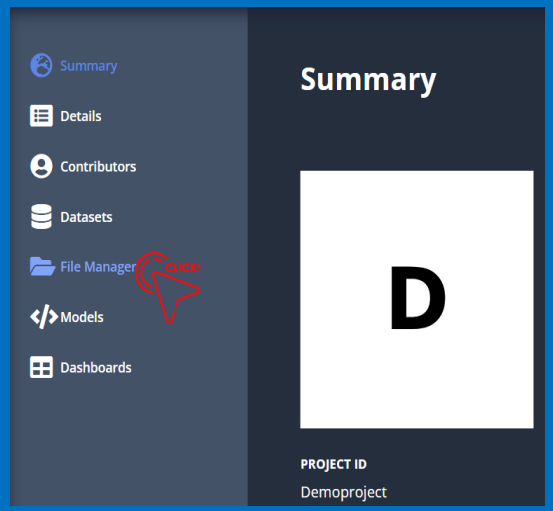


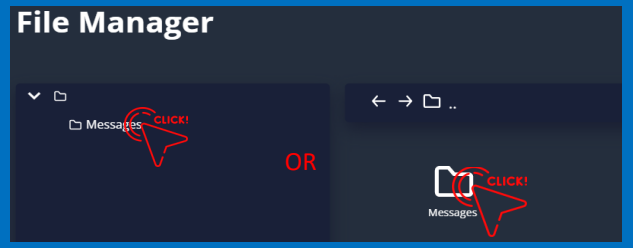
5. Click on the edit project button



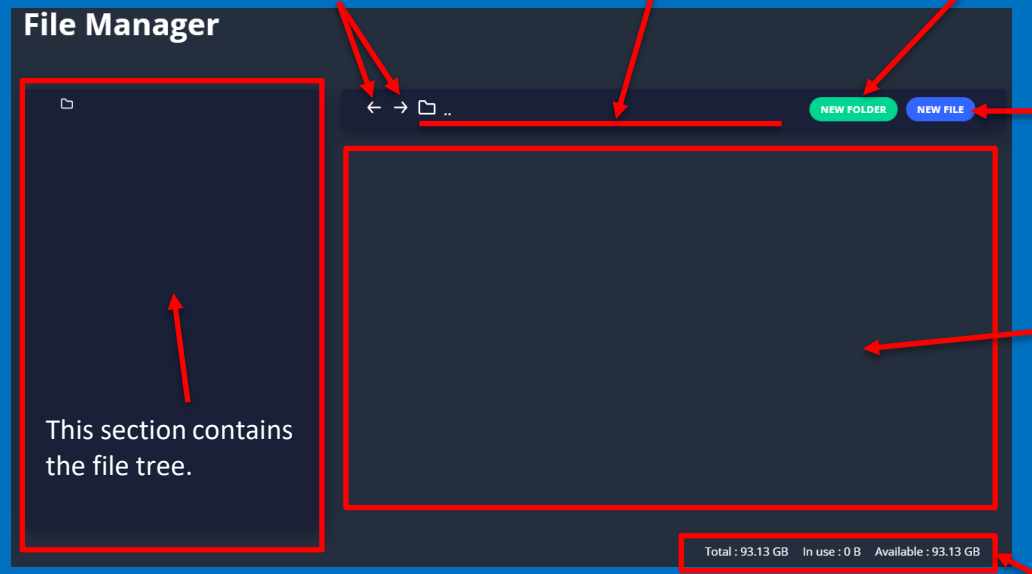
File Manager

6. Click on the File Manager section





To navigate to a folder click on the folder from the file tree section or click on the folder.



Click here to navigate to the previous or the next folder

In this section there is the path of the current folder

Click here to add a new folder

Click here to add a new file.

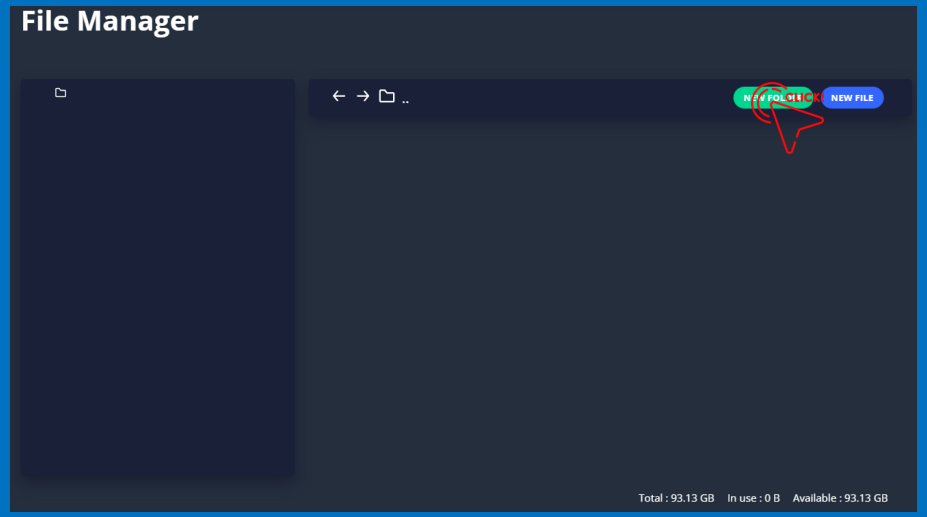
This section contains the file tree.

In this section will appear all files and folders contained in the current path

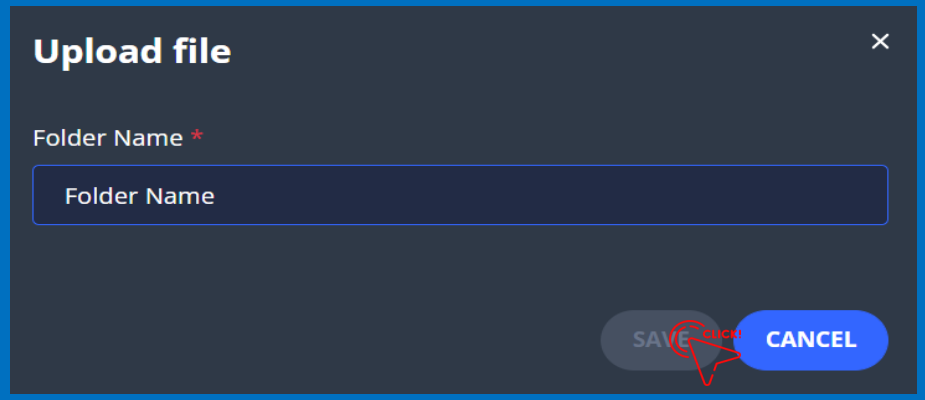
It shows how much space you have in total, in use and availabe.

File Manager – Add new folder

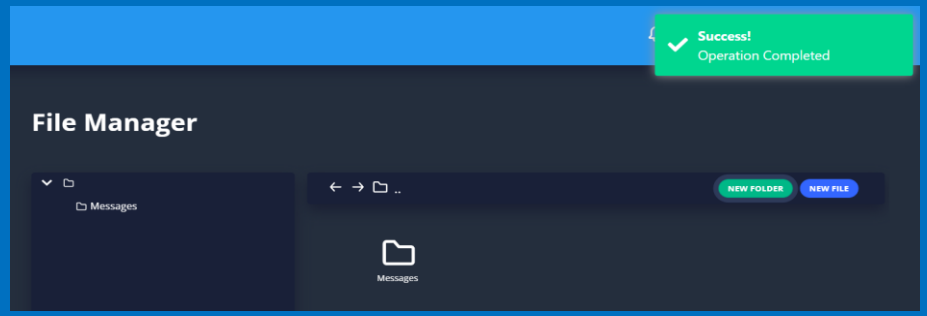
7. Click on the new folder button



8. Fill the form and click on save

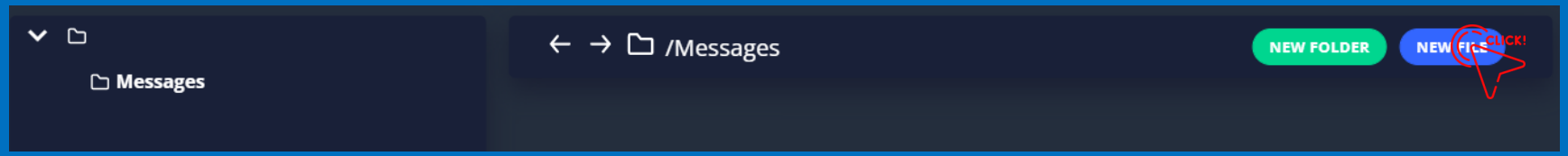


You will receive a notification that the folder is created like the one shown.

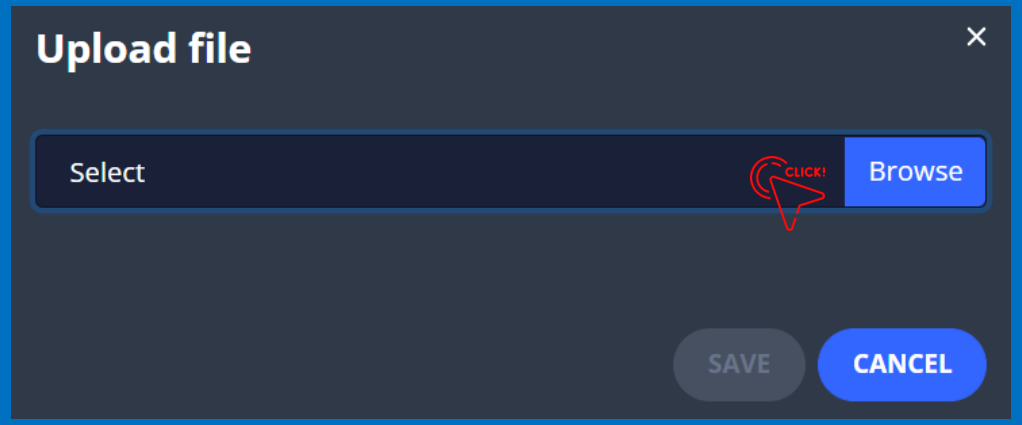


File Manager – Add new file

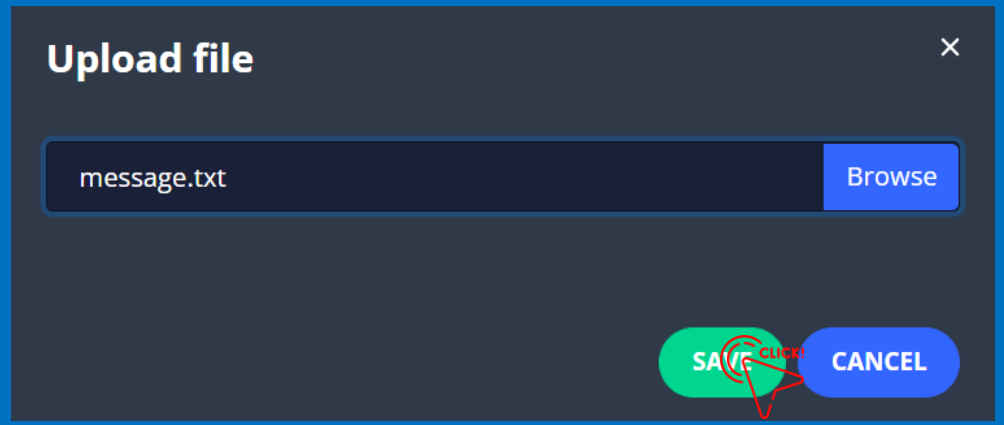
9. Click on the new file button



10. Select the file

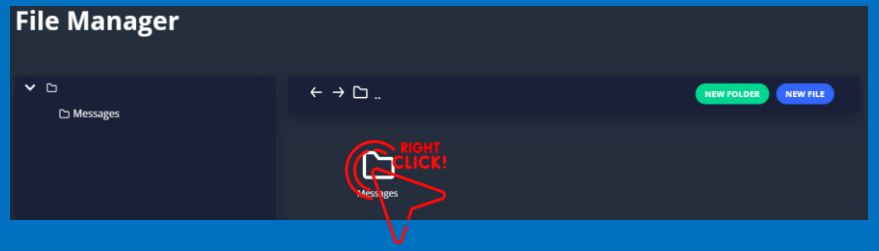


11. Click on save

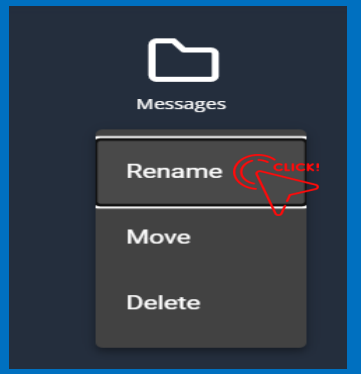


File Manager
Rename a folder/file

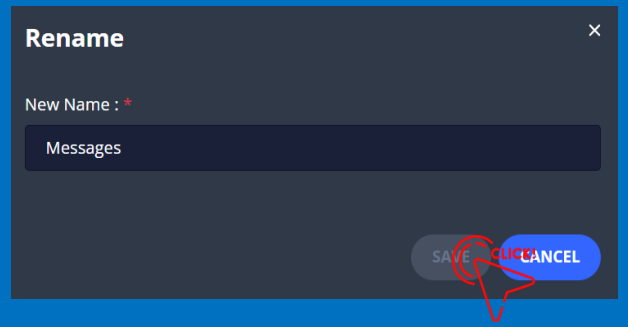
1. To rename a folder/file right click on it



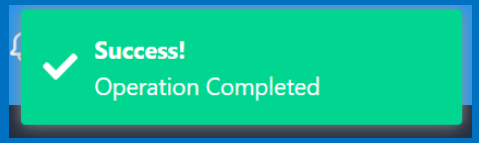
2. Click on the Rename button



3. Fill the form and click on save

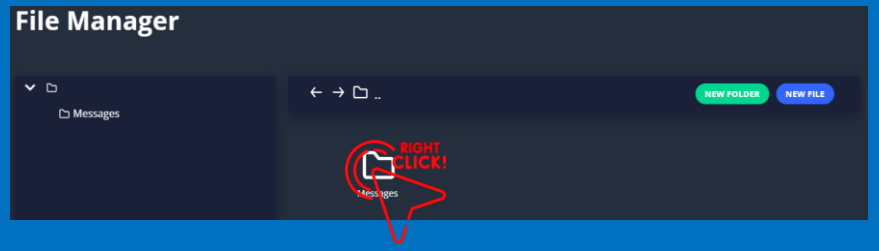


A notification will appear that the folder/file has been renamed like the one shown.

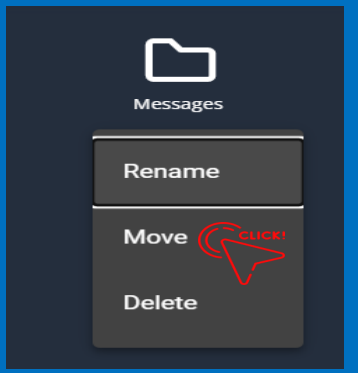


File Manager
Move a folder/file

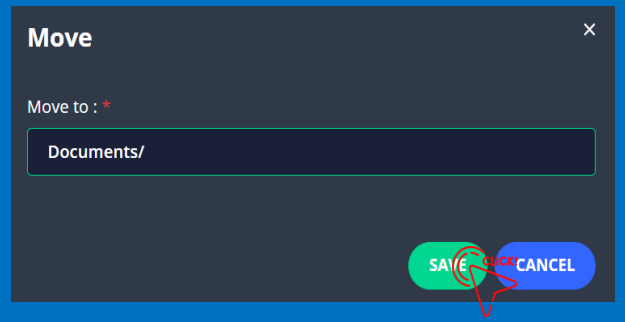
1. To rename a folder/file right click on it



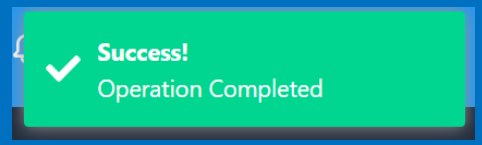
2. Click on the Move button



3. Fill the form and click on save

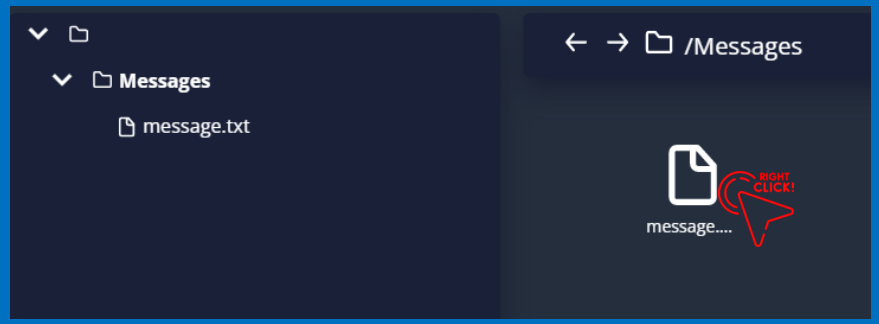


A notification will appear that the folder/file has been moved like the one shown.

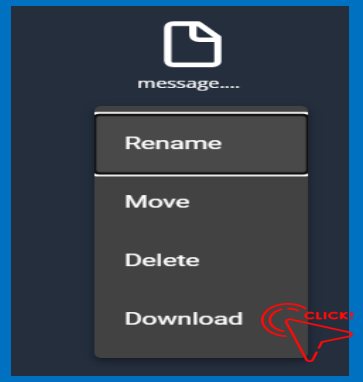


File Manager – Download a file

1. To download a file right click on it



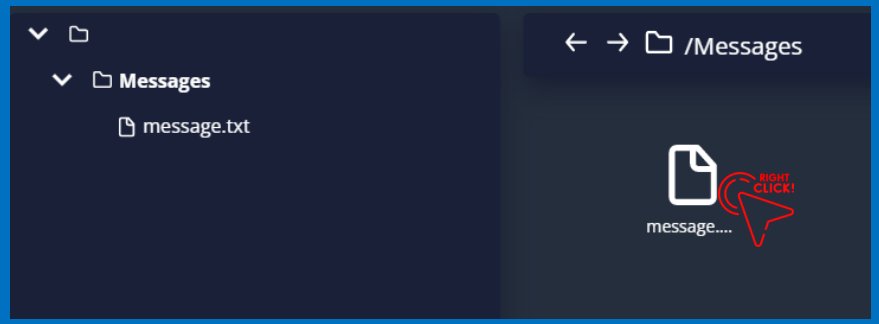
2. Click on the Download button



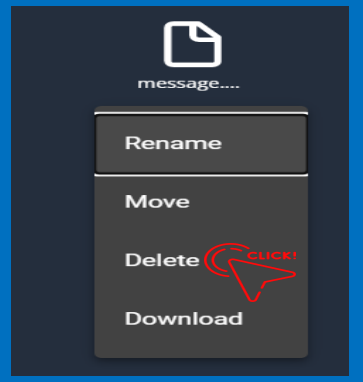
Co-financed by the Connecting Europe Facility of the European Union

File Manager – Delete a folder/file

1. To delete a folder right click on it

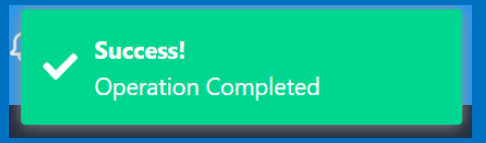


2. Click on the Delete button



WARNING!!! If you delete a folder everything inside it will be deleted

A notification will appear that the folder has been deleted like the one shown.



Model

A model represents a container of all the scripts needed for a single execution.

Scripts can be organized in folders and have a point of entry by default.

Model execution can be scheduled or run manually.

Data read and write operations are allowed only by using the embedded library that manages security and grants access to datasets.

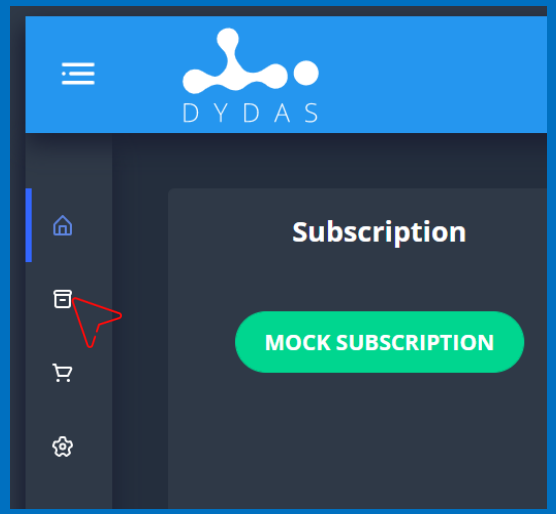
Both read and write operations can only be performed on datasets that are part of the project, using the schema defined on each one.

Projects can contain multiple models.

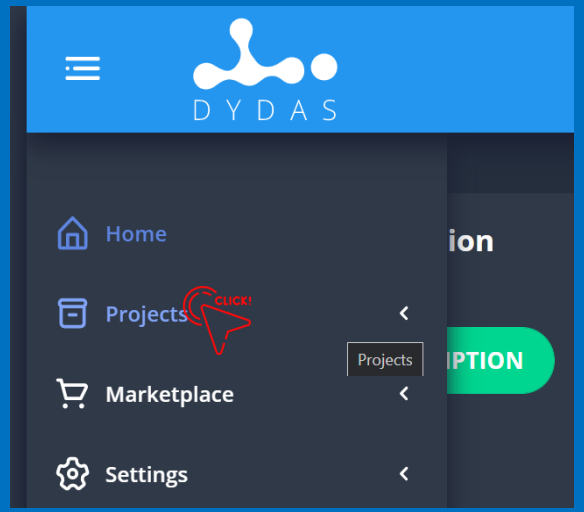
DYDAS supports Python scripts only in this version.

Model creation

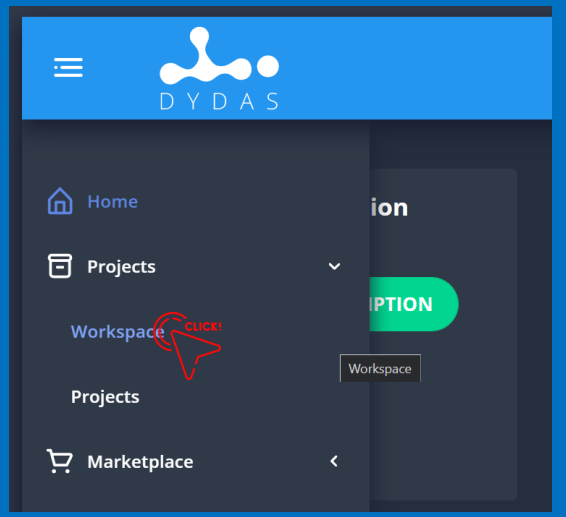
1. Place the mouse in the project icon



2. Click on the projects section



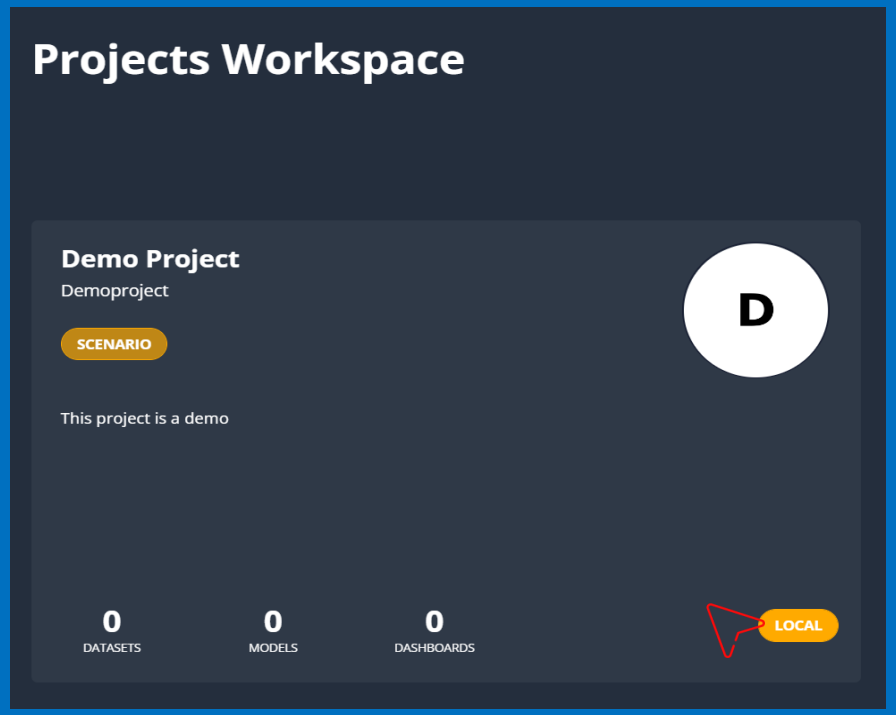
3. Click on the workspace section



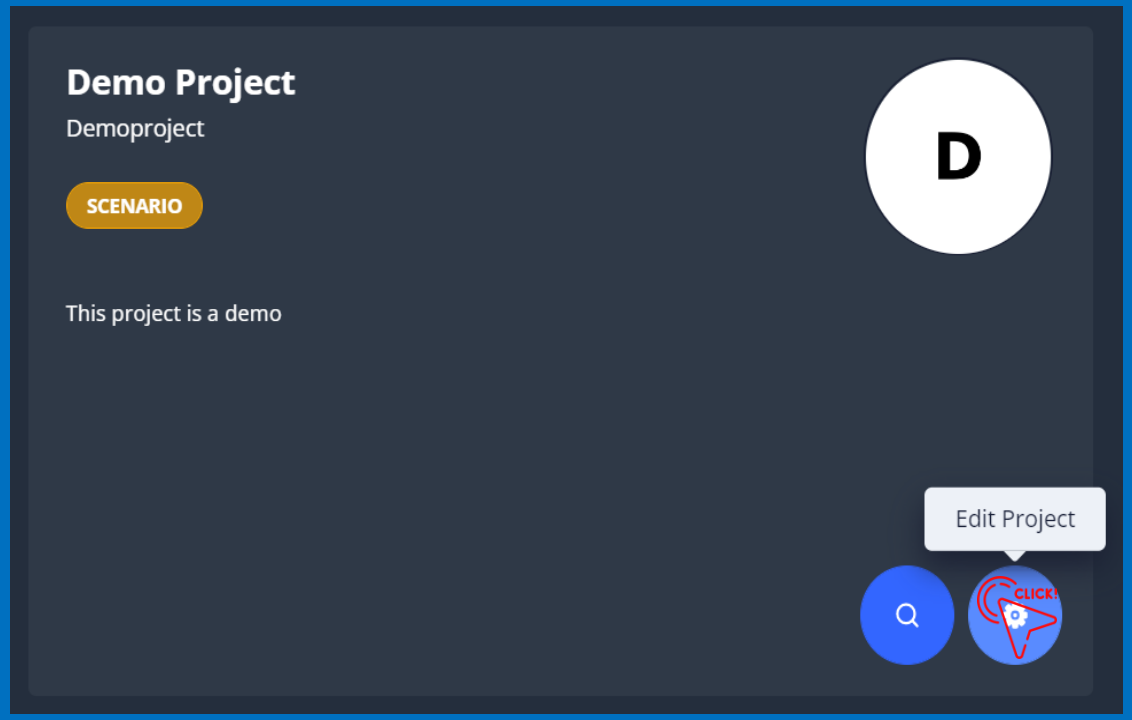
Co-financed by the Connecting Europe Facility of the European Union

Model creation

4. Place the mouse in the project card

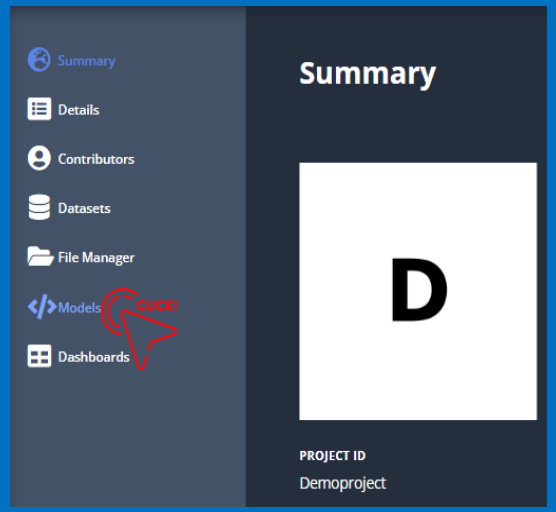


5. Click on the edit project button

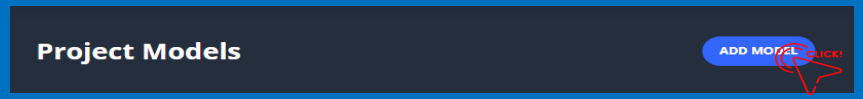


Model creation

6. Click on the models section

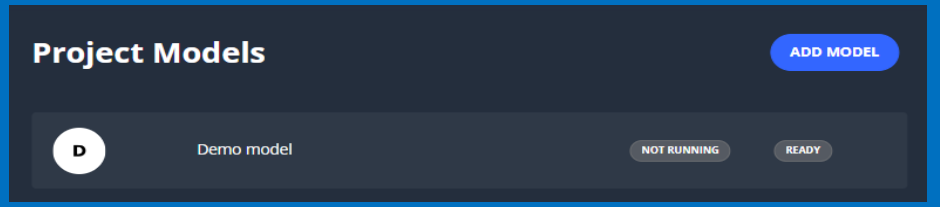


7. Click on the add model button



The model is now created. You will be redirected to the project model list containing the new model.

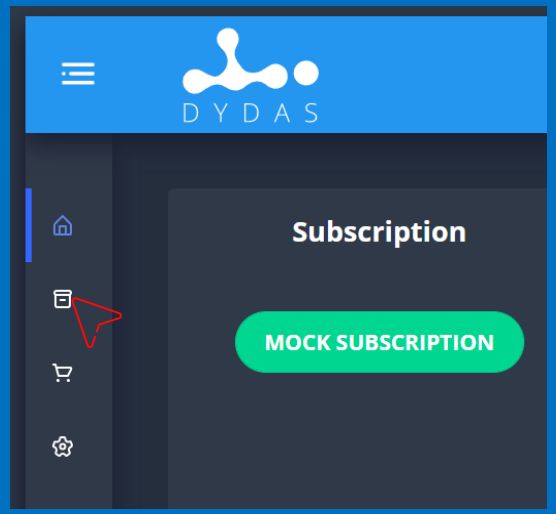
8. The model lab will appear. Fill the form and click on save.



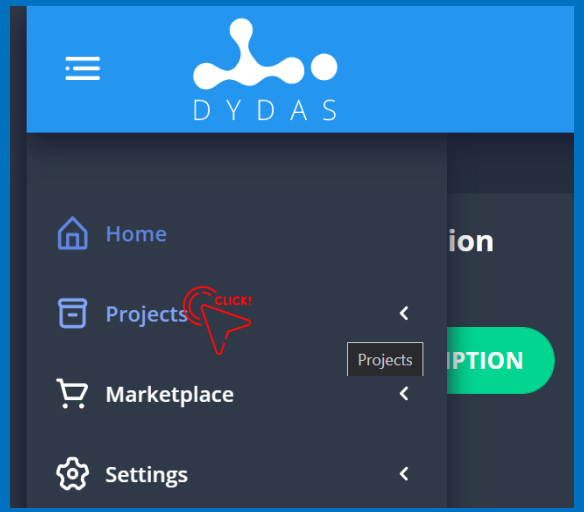
Co-financed by the Connecting Europe Facility of the European Union

Model script editor

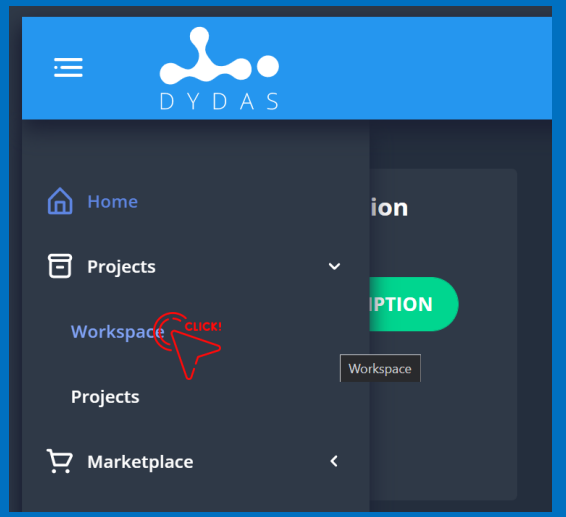
1. Place the mouse in the project icon



2. Click on the projects section



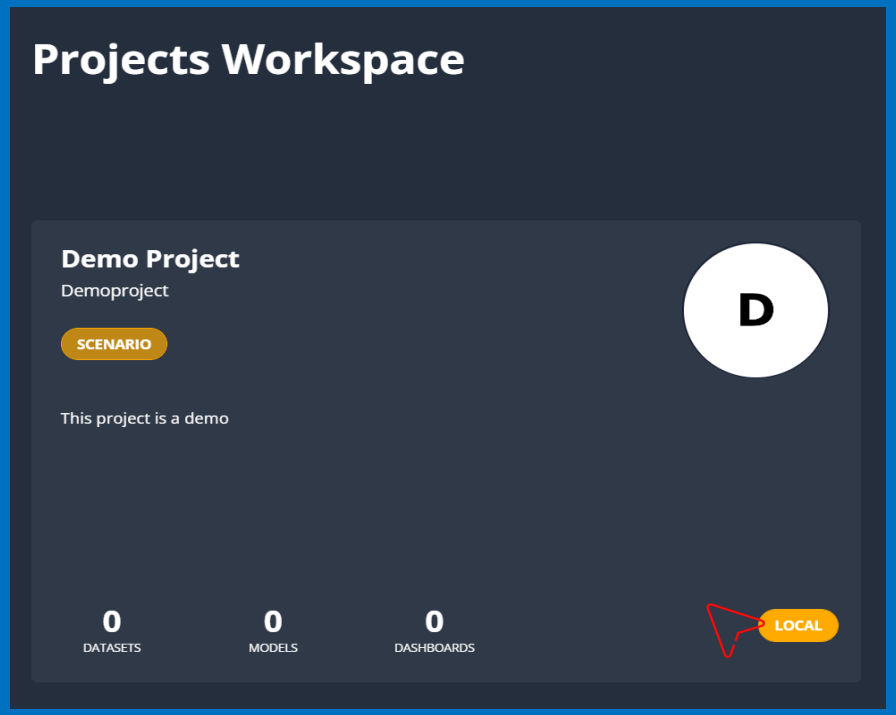
3. Click on the workspace section



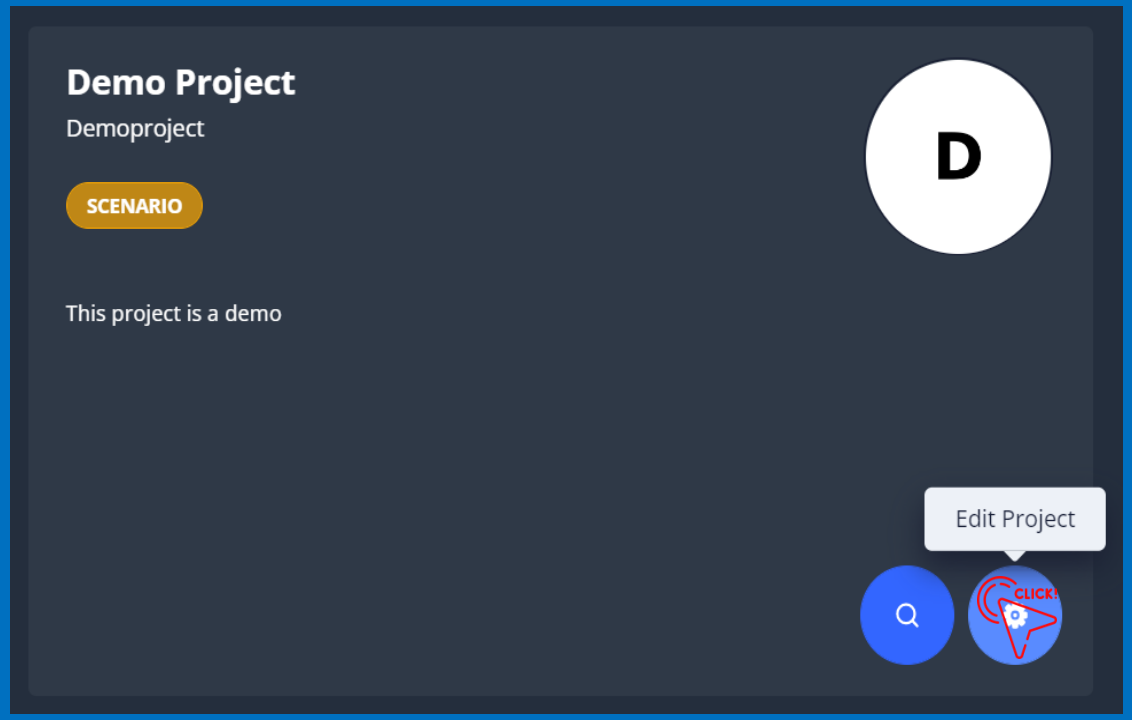
Co-financed by the Connecting Europe Facility of the European Union

Model script editor

4. Place the mouse in the project card

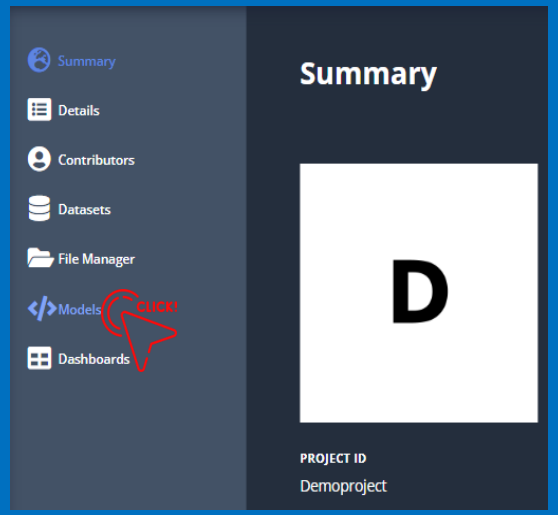


5. Click on the edit project button

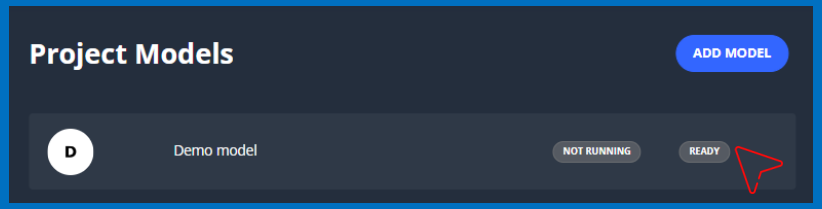


Model script editor

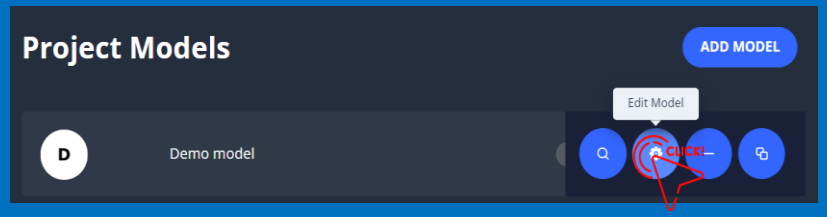
6. Click on the models section



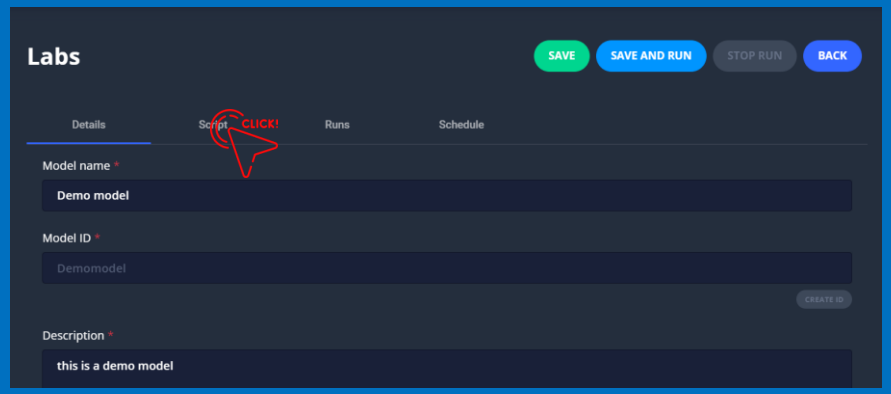
7. Hover the mouse over the model



8. Click on the edit model button.



9. Click on the script section



Co-financed by the Connecting Europe Facility of the European Union

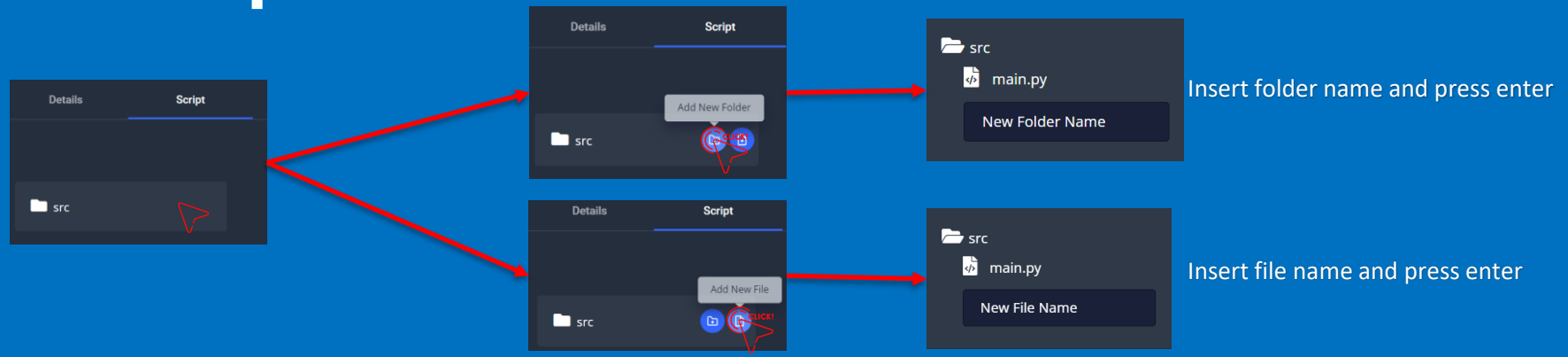
Model script editor

To the left you can find the source of the model.

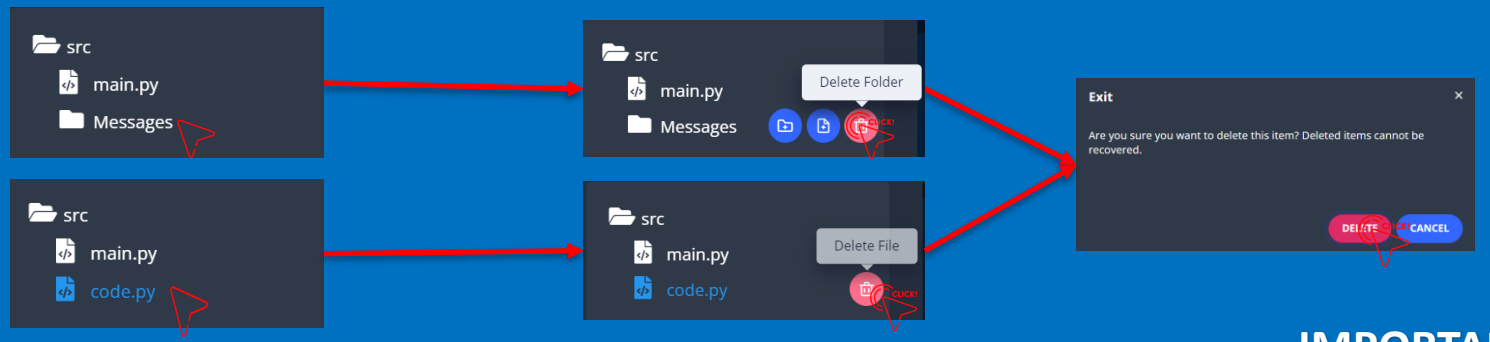
To let the model be run by the system inside the src there should be a main.py that imports the DYDAS api.

Model script editor

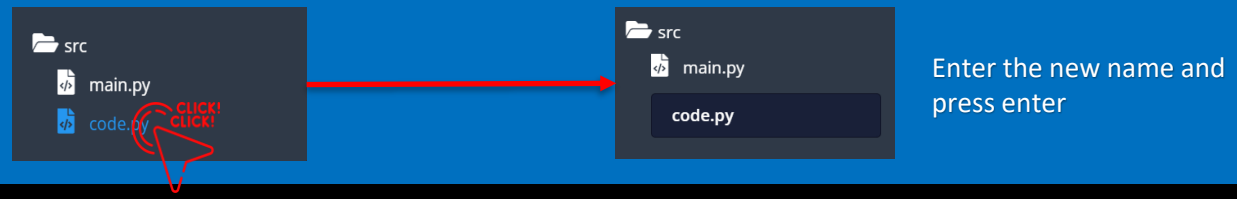
To add a file or a folder hover the mouse over the folder



To delete a file or a folder hover the mouse over the folder or the file



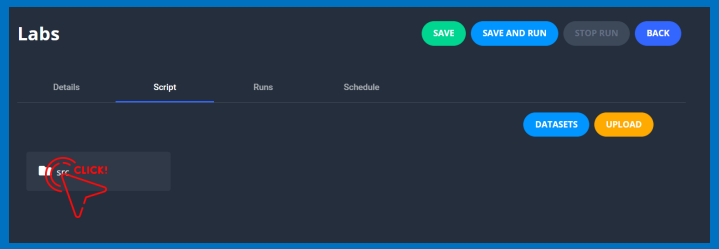
To rename a file or a folder double click on it



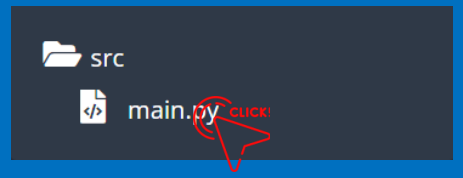
IMPORTANT!!! All these steps have no effect until the model is saved.

Model script editor

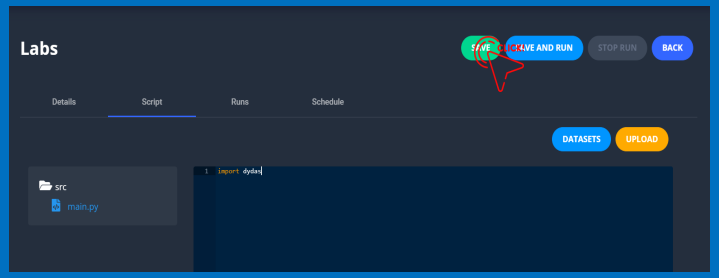
10. Click on the src folder



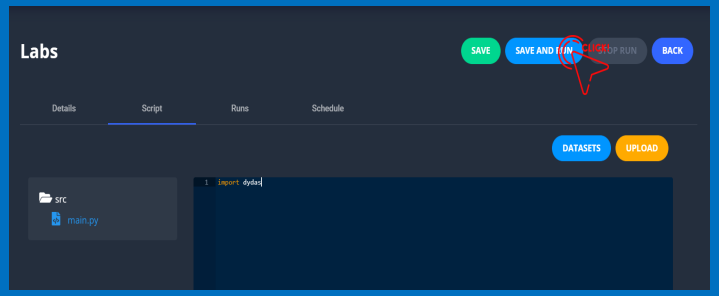
11. Click on the main.py file



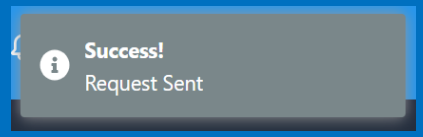
12. A file editor will appear where you can edit the file. To save it click on the save button



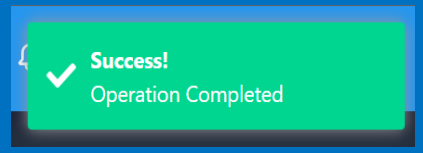
13. To run the model click on the save and run button



When this message appears the model is saved.



A request to save and run the model is then sent and a message appears like the one shown



Model script – packages

Python packages and version:

Package	Version
absl-py	0.15.0
affine	2.3.1
asciitree	0.3.3
astunparse	1.6.3
attrs	21.4.0
cached-property	1.5.2
cachetools	4.2.4
certifi	2021.10.8
charset-normalizer	2.0.12
clang	5.0
click	8.1.3
click-plugins	1.1.1
cligj	0.7.2
cycler	0.11.0
dydas	1.0.0
fasteners	0.17.3
Fiona	1.8.21
flatbuffers	1.12
fonttools	4.33.3
gast	0.4.0
GDAL	3.4.2
geopandas	0.10.2
google-auth	1.35.0

Package	Version
google-auth-oauthlib	0.4.6
google-pasta	0.2.0
grpcio	1.46.1
h5py	3.1.0
idna	3.3
importlib-metadata	4.11.3
joblib	1.1.0
JPype1	1.1.2
keras	2.6.0
Keras-Preprocessing	1.1.2
kiwisolver	1.4.2
llvmlite	0.38.0
Markdown	3.3.7
matplotlib	3.5.2
munch	2.5.0
networkx	2.6.3
numba	0.55.1
numcodecs	0.9.1
numpy	1.19.5
oauthlib	3.2.0
opt-einsum	3.3.0
packaging	21.3
pandas	1.3.5

Package	Version
Pillow	9.1.0
pip	21.2.4
protobuf	3.20.1
py4j	0.10.7
pyasn1	0.4.8
pyasn1-modules	0.2.8
pynndescent	0.5.6
pyparsing	3.0.9
pyproj	3.2.1
pyspark	2.3.2
python-dateutil	2.8.2
pytz	2022.1
rasterio	1.2.10
requests	2.27.1
requests-oauthlib	1.3.1
rioxarray	0.9.1
rsa	4.8
Rtree	1.0.0
scikit-learn	1.0.2
scipy	1.7.3
Shapely	1.8.2
six	1.15.0
setuptools	57.5.0

Package	Version
sklearn	0.0
snuggs	1.4.7
tensorboard	2.6.0
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tensorflow	2.6.2
tensorflow-estimator	2.6.0
termcolor	1.1.0
threadpoolctl	3.1.0
tqdm	4.64.0
typing-extensions	3.7.4.3
umap-learn	0.5.3
urllib3	1.26.9
Werkzeug	2.1.2
wheel	0.37.1
wrapt	1.12.1
xarray	0.20.2
xlrd	2.0.1
zarr	2.11.3
zipp	3.8.0

Model script – DYDAS API

It is possible to interact with project's datasets and files in the file manager from the python code in the model via the DYDAS API package named Dydas (import Dydas).

After you imported the DYDAS package, to use the API call it like a method and store it in a variable like this:

```
import dydas

def main():
    dydas_api = dydas.API()

    …


if __name__ == "__main__":
    main()
```

Model script – DYDAS API

Dataset API:

The allowed operations from models using the API's object of the DYDAS package are the following:

- **Upload / Download dataset**
- **Truncate dataset**

The upload and download functions, are named differently regarding of the type of the dataset (Vector / Raster / Other)

FileManager API:

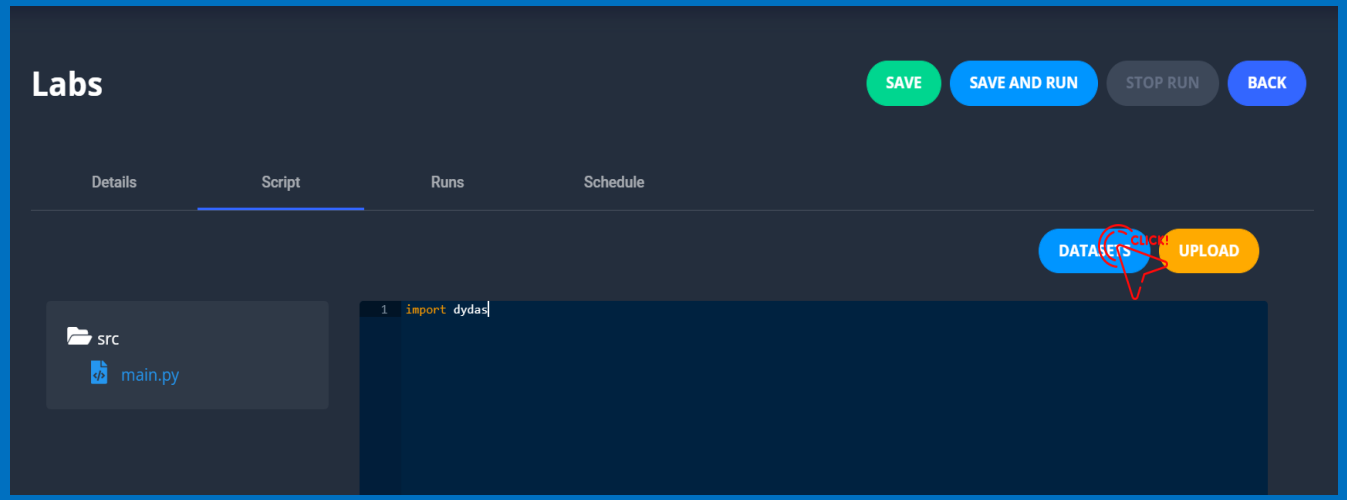
From the models is possible upload / download data from the project's File Manager, a cloud repository where could be storing all kind of files.

The operation allowed from the Models API are:

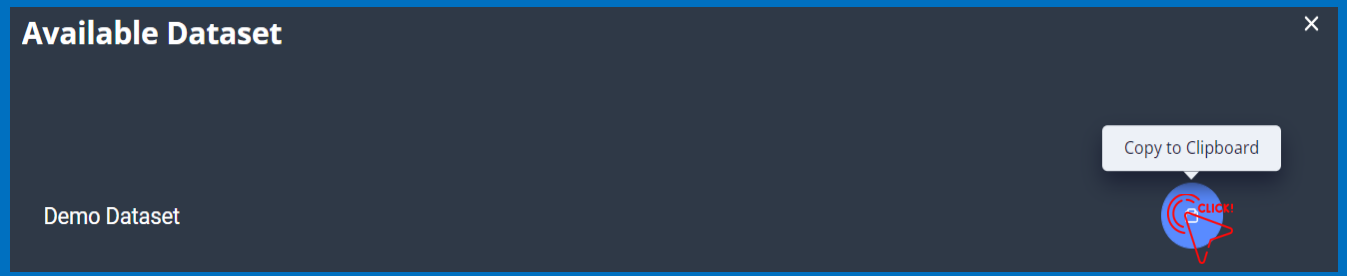
- **List**
- **Upload / Create**
- **Download**
- **Delete**

Model script – Dataset table name

To get the table name (or dataset id) to use it as a parameter in the Dydas APIs, from the model script editor click the datasets button



Click on the copy to clipboard button
The table id is now in your clipboard



Model script – Dataset API

Dydas_api.get_raster_dataset

Used to download raster data from a dataset of type Raster

Parameter	Optional / Required	Description	Data Type
table_name	Required	The full name of the dataset	str
srid	Required	The SRID that the data will have	str
raster_type	Required	The file format of the data	dydas.RasterType
directory	Required	The folder where the data should be saved	pathlib.Path
ecql_filter	Optional	The ECQL filter to apply when retrieving the data	str

Return value: List

The List contains 0+ pathlib.Path, each of those are the position, in the local filesystem where the model is running, of the Raster downloaded data. The parent of each Path is the parameter: “directory” passed at the method.

Model script – Dataset API

Dydas_api.get_vector_dataset

Used to download vector data from a dataset of type Vector

Parameter	Optional / Required	Description	Data Type
table_name	Required	The full name of the dataset	str
srid	Required	The SRID that the data will have	str
vector_type	Required	The file format of the data	dydas.VectorType
directory	Required	The folder where the data should be saved	pathlib.Path
ecql_filter	Optional	The ECQL filter to apply when retrieving the data	str

Return value: pathlib.Path

The returned value is the position, in the local filesystem where the model is running, of the Vector downloaded file. The parent of the Path is the parameter: “directory” passed at the method.

Model script – Dataset API

Dydas_api.get_other_dataset

Used to download simple / timeseries data from a dataset of type Other

Parameter	Optional / Required	Description	Data Type
table_name	Required	The full name of the dataset	str
other_type	Required	The file format of the data	dydas.OtherType
directory	Required	The folder where the data should be saved	pathlib.Path
delimiter	Required	The delimiter between data. E.g. '|' ',' '-' '.'	str
ecql_filter	Optional	The ECQL filter to apply when retrieving the data	str

Return value: pathlib.Path

The returned value is the position, in the local filesystem where the model is running, of the Other downloaded file. The parent of the Path is the parameter: "directory" passed at the method.

Model script – Dataset API

Dydas_api.upload_raster_dataset

Used to upload raster data in a dataset of type Raster

Parameter	Optional / Required	Description	Data Type
table_name	Required	The full name of the dataset	str
image	Required	The raster data to upload	pathlib.Path
raster_type	Required	The file format of the data	dydas.RasterType
table_type	Required	The type of dataset (e.g. geospatial)	dydas.TableType
overwrite	Required	Delete all data in dataset before upload	bool

Return value: None

Model script – Dataset API

Dydas_api.upload_vector_dataset

Used to upload vector data in a dataset of type Vector

Parameter	Optional / Required	Description	Data Type
table_name	Required	The full name of the dataset	str
file	Required	The vector data to upload	pathlib.Path
vector_type	Required	The file format of the data	dydas.VectorType
srid	Required	The SRID of the data to upload	str
overwrite	Required	Delete all data in dataset before upload	bool

Return value: None

Model script – Dataset API

Dydas_api.upload_other_dataset

Used to upload simple / timeseries data in a dataset of type Other

Parameter	Optional / Required	Description	Data Type
table_name	Required	The full name of the dataset	str
file	Required	The simple / timeseries data to upload	pathlib.Path
other_type	Required	The file format of the data	dydas.OtherType
delimiter	Required	The delimiter between data. E.g. '|' ',' '-' '.'	str
overwrite	Required	Delete all data in dataset before upload	bool

Return value: None

Model script – Dataset API

Dydas_api.truncate_dataset

Used to delete all the data in a datastore

Parameter	Optional / Required	Description	Data Type
table_name	Required	The full name of the dataset	str

Return value: bool
Returns True if all the data are eliminated permanently

Model script – FileManager API

Dydas_api.list_files

Used to get a list of all files in a remote path, useful, for example, in order to iterate over them and download one at the time

Parameter	Optional / Required	Description	Data Type
remote_src	Required	The remote path to list files e.g '/myfolder/'	str

Return value: List
The List contains 0+ str, that are the names of the files inside the 'remote_src'

Model script – FileManager API

Dydas_api.upload_file

Used to upload (create if not exists or overwrite) a file from the local filesystem where the model is running, to the remote File Manager

Parameter	Optional / Required	Description	Data Type
src_file	Required	Path of the file in the local filesystem of the model	pathlib.Path
remote_dst	Required	Remote path where upload the file e.g. '/myfolder/myfile.txt'	str
overwrite_if_exists	Required	Overwrite the file if exists or return False	bool

Return value: bool
Return True if the file is correctly upload or False otherwise

Model script – FileManager API

Dydas_api.download_file

Used to download a single file from the File Manager in a Path in the local filesystem of the model

Parameter	Optional / Required	Description	Data Type
dst_file	Required	Path where save the file in the local filesystem of the model	pathlib.Path
remote_src	Required	Remote path of the file to download	str

Return value: bool
Return True if the file is correctly download or False otherwise

Model script – FileManager API

Dydas_api.delete_file

Used to delete a file or folder in the remote File Manager

Parameter	Optional / Required	Description	Data Type
remote_src	Required	Remote path of the file to delete	str

Return value: bool
Return True if the file is deleted correctly

Model script – enumerations

To use the Dataset API, some parameters must be restricted.
For this reason, DYDAS enumerations must be used and are the following:

Dydas.RasterType
GEOTIFF
GRIB
HDFS4
HDFS5
NETCDF

Dydas.VectorType
GEOJSON
SHAPEFILE

Dydas.OtherType
CSV

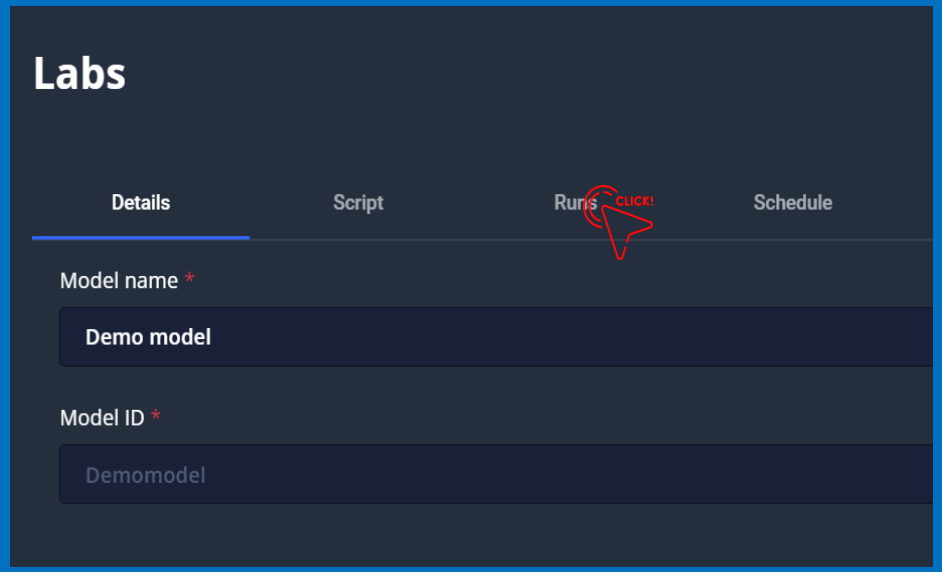
Dydas.TableType
SPACE - GEOSPATIAL
SPACETIME - GEOSPATIALTIMESERIES
TIME - TIMESERIES
OTHER - SIMPLE

Usage example:

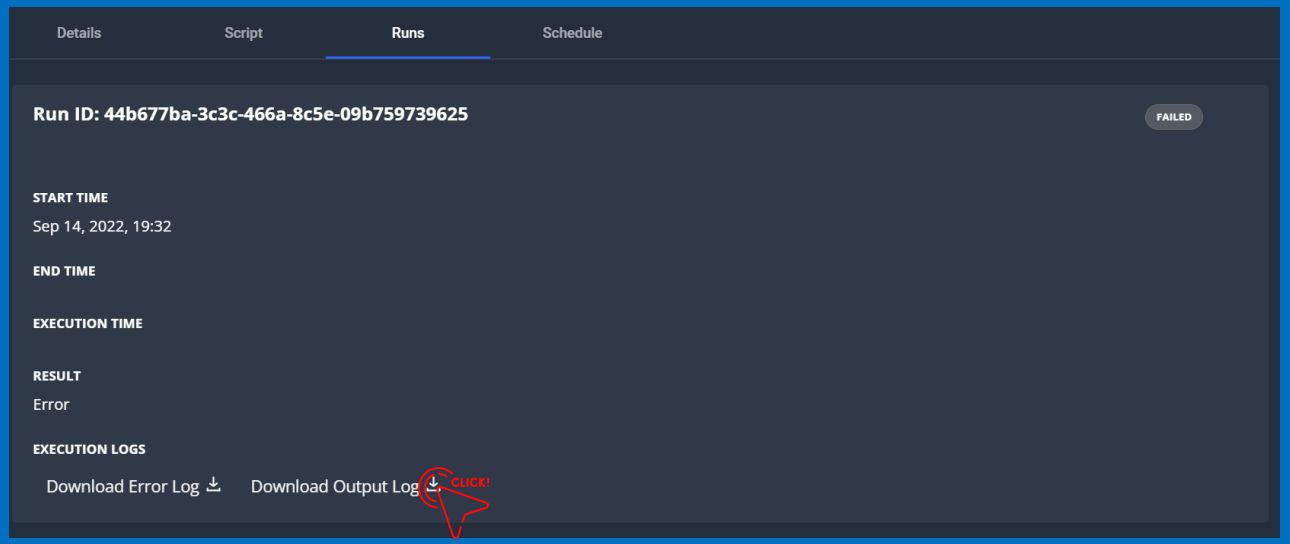
```
dydas.OtherType.CSV
```

Model runs

To check the runs of a model, select the runs section in the labs page



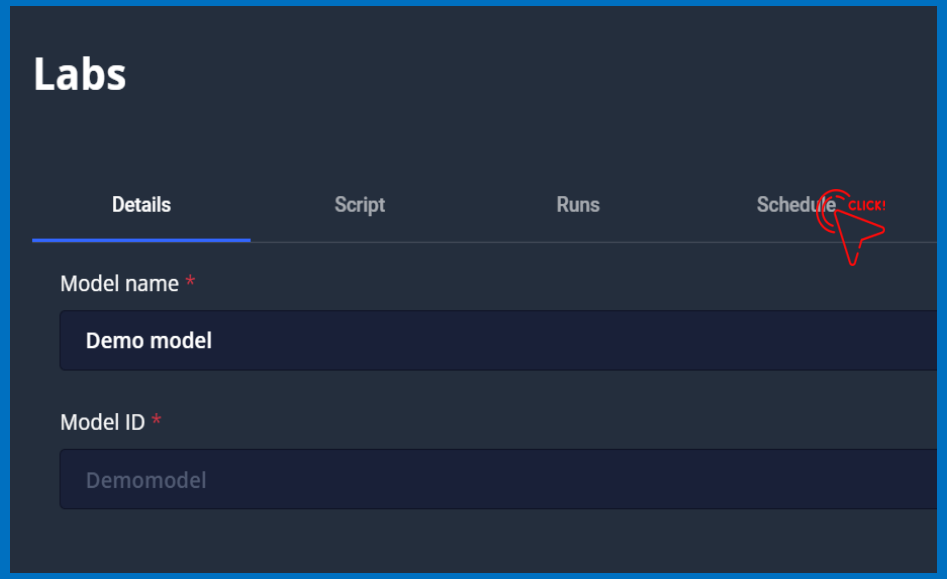
There is a list of the models. In every model: the run id, the starting time, the end time, the execution time, the result and the logs are shown and available for 24h



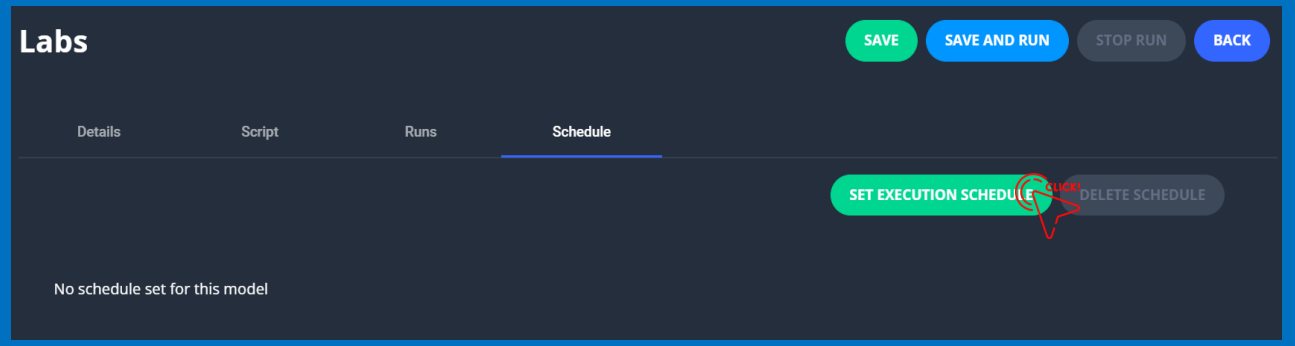
Co-financed by the Connecting Europe Facility of the European Union

Model run schedule

To schedule a model, select the schedule section in the labs page



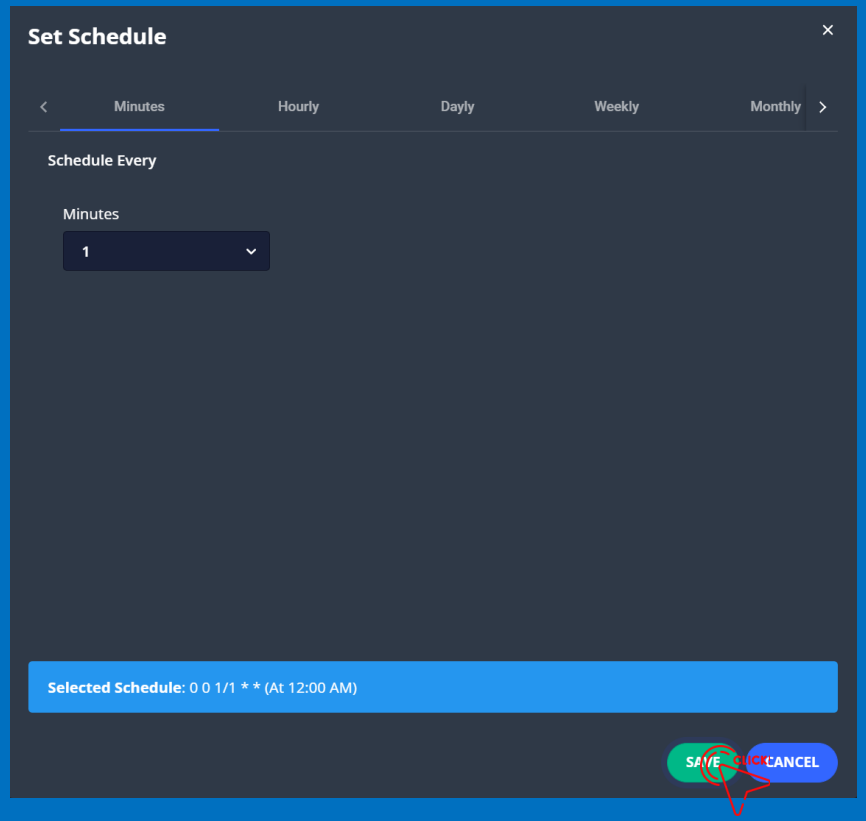
To set a schedule click the set execution schedule button (to delete a schedule press the delete schedule button)



Model run schedule

The model can be scheduled by minutes, hourly, daily, weekly, monthly, yearly, with a custom expression or defining a date for a single execution.

Select the wanted behaviour and click the save button.



Please note that for a project only one model can run at a time

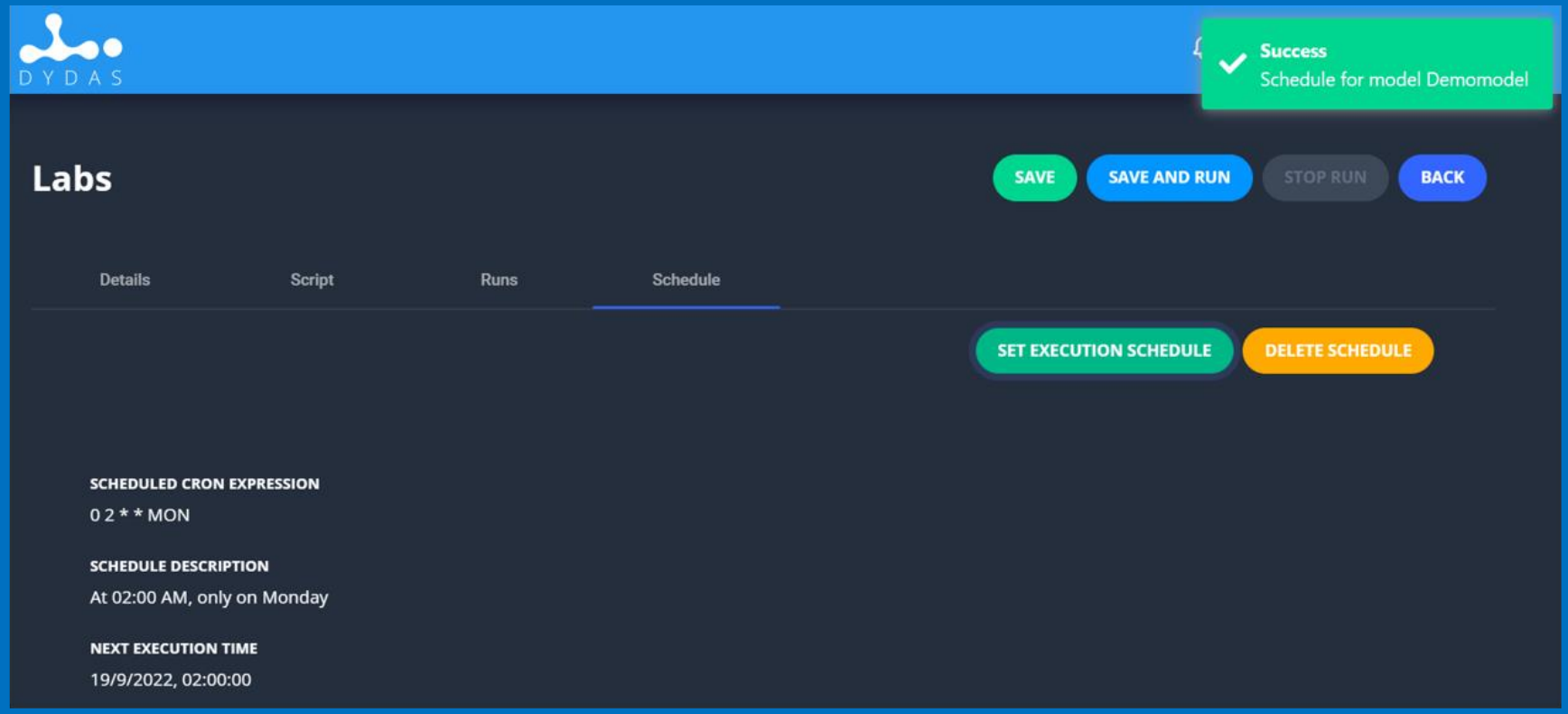
Co-financed by the Connecting Europe Facility of the European Union

Model run schedule

A message that the schedule has been updated will pop up like the one on the right

In this page you can see the cron expression, the schedule description and the next execution time

It is not necessary to save the model to save the execution because the schedule is already updated once you save it.



Dashboard

Dashboards are the containers for configurable views that users can define for a project.

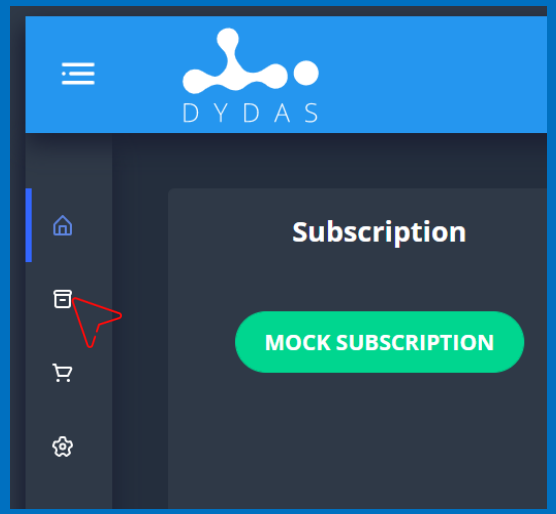
The views can be built using a *drag & drop* approach and are fully customizable both by using selectable options or by using CQL language.

Each analysis tool (**widget**) can be positioned on a single screen and configured separately or bound to centralized filters.

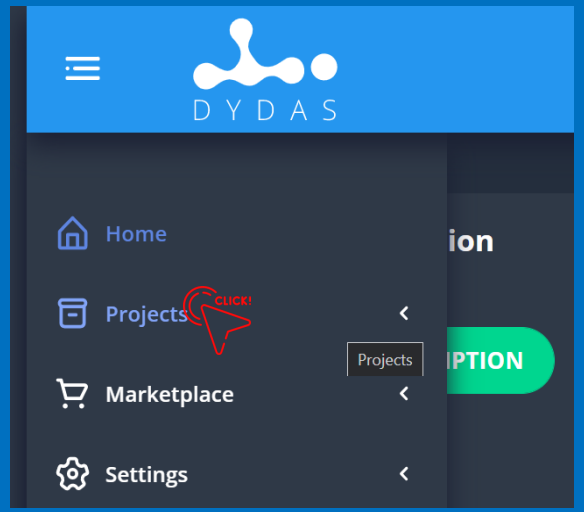
Datasets can only use data that is contained or referenced by the project, on which the user has read permissions.

Dashboard creation

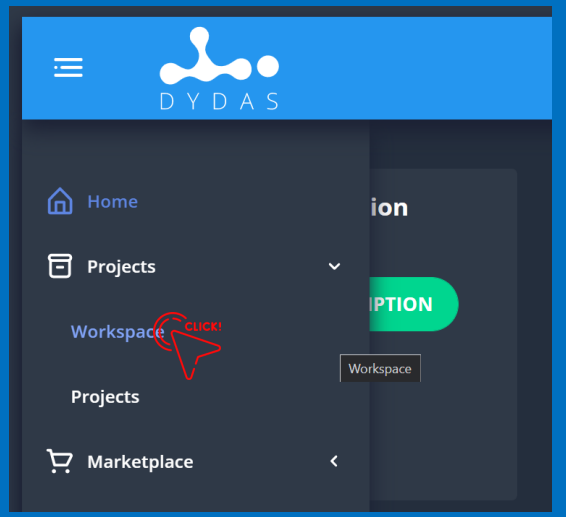
1. Place the mouse in the project icon



2. Click on the projects section



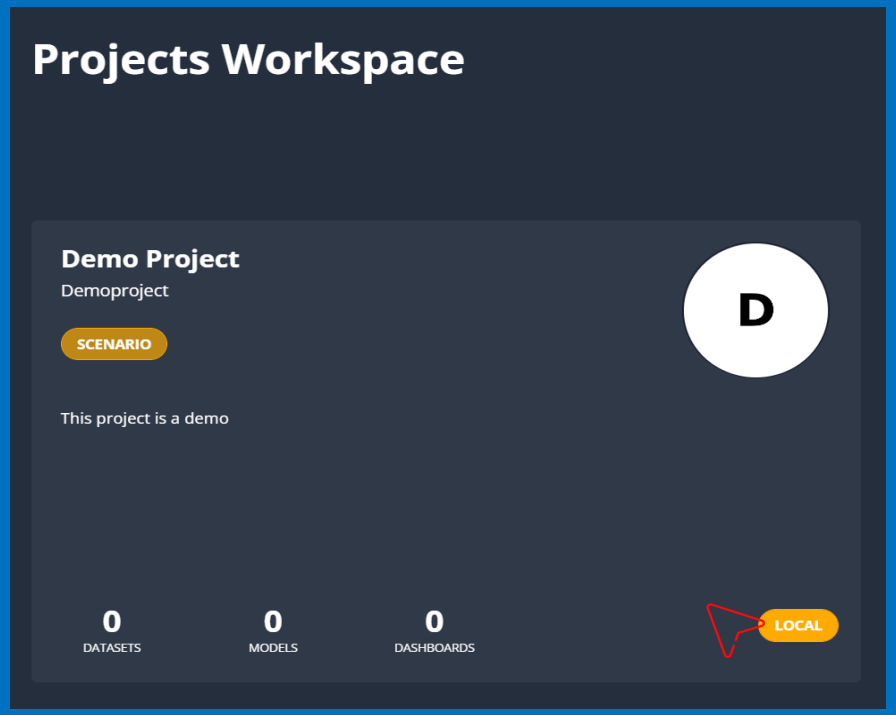
3. Click on the workspace section



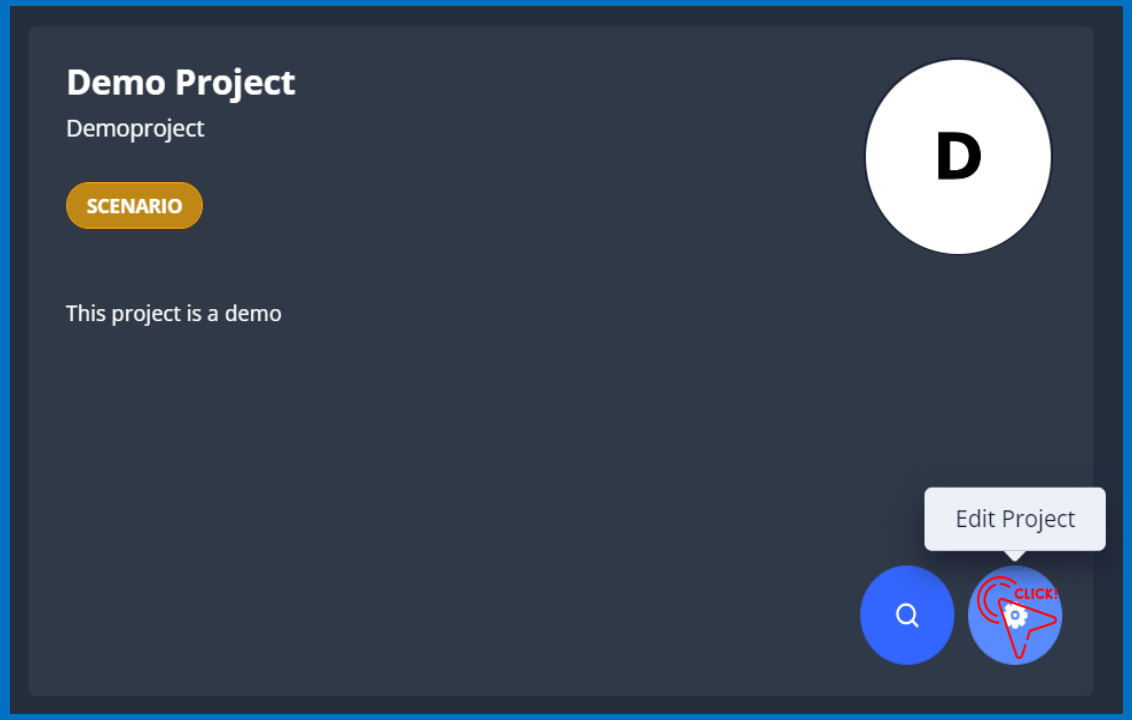
Co-financed by the Connecting Europe Facility of the European Union

Dashboard creation

4. Place the mouse in the project card

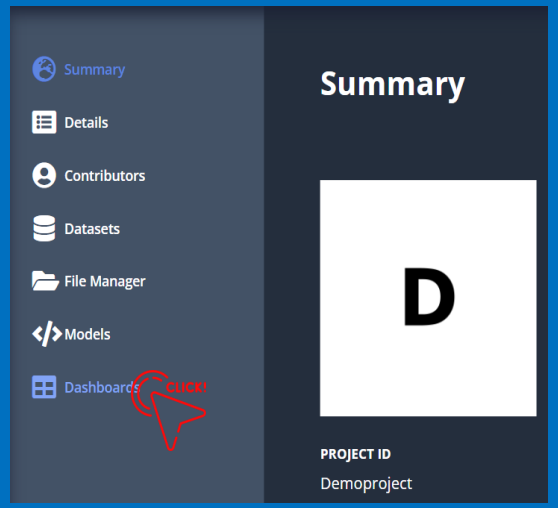


5. Click on the edit project button

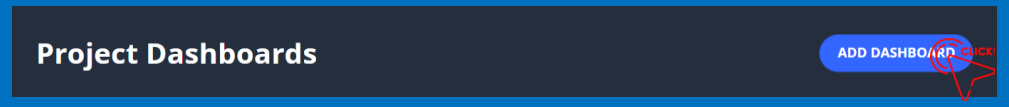


Dashboard creation

6. Click on the dashboard section



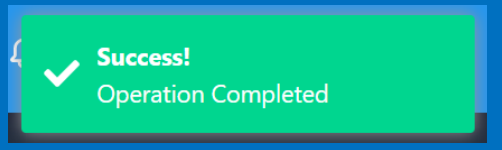
7. Click on the add dashboard button



8. The dashboard editor will appear.

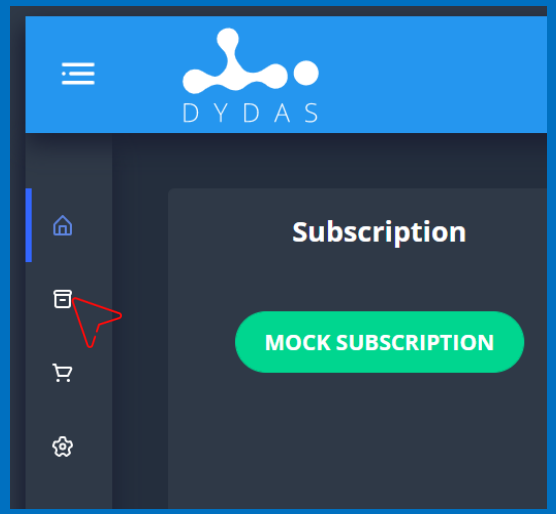
Fill the form and click on save.

A message like the one to the right appears and the dashboard is now created.

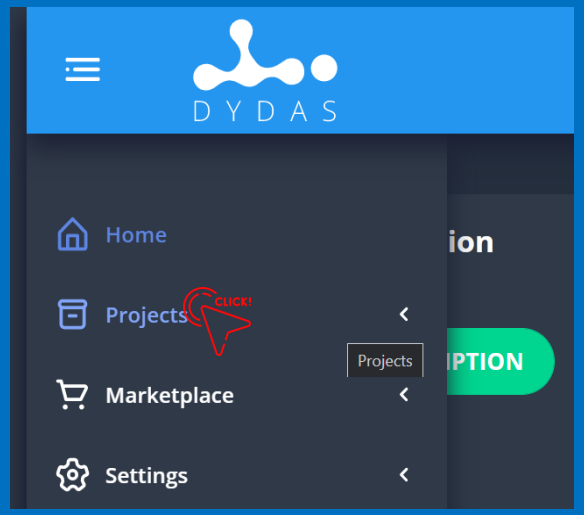


Dashboard edit – widget creation

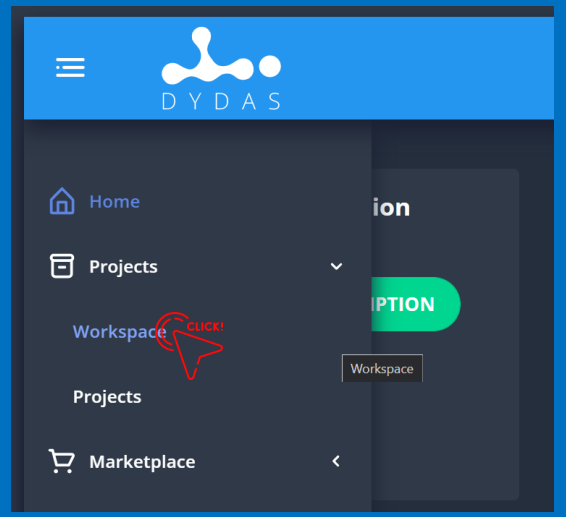
1. Place the mouse in the project icon



2. Click on the projects section

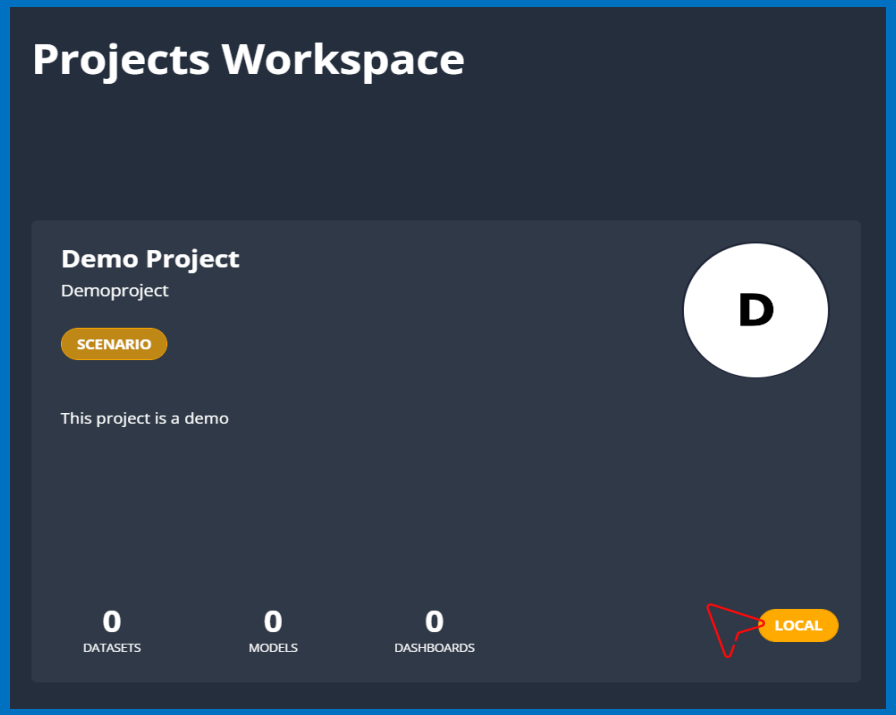


3. Click on the workspace section

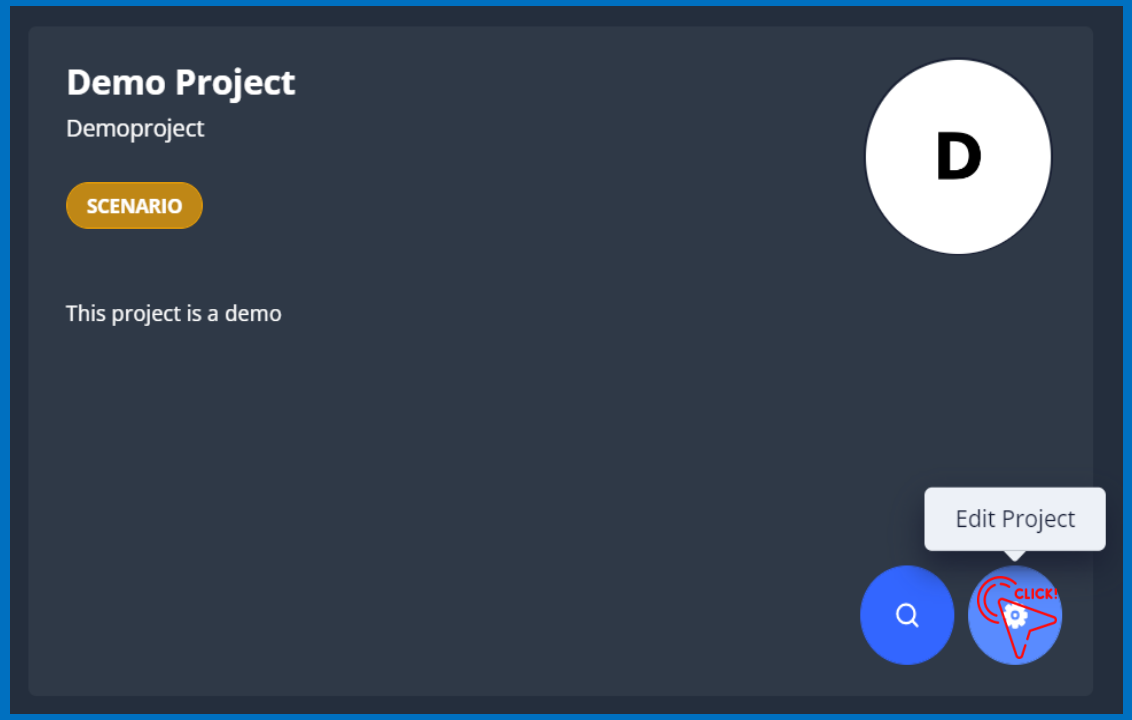


Dashboard edit – widget creation

4. Place the mouse in the project card

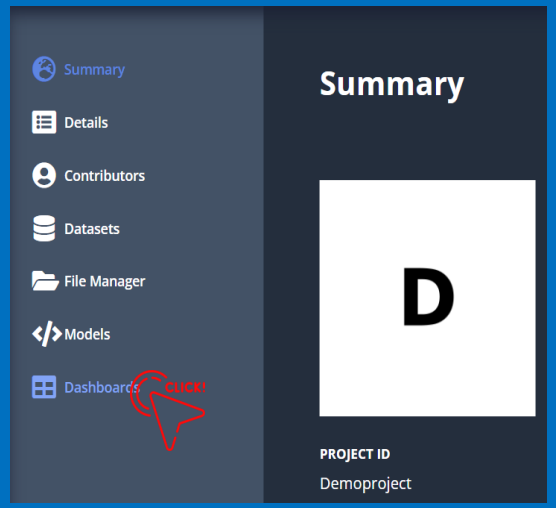


5. Click on the edit project button

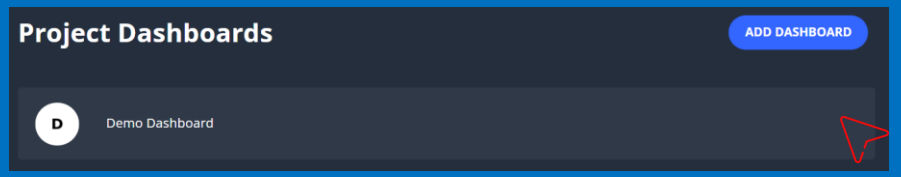


Dashboard edit – widget creation

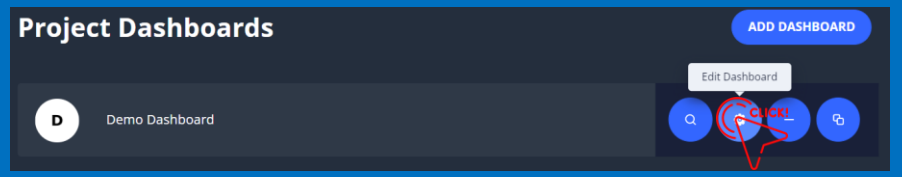
6. Click on the dashboard section



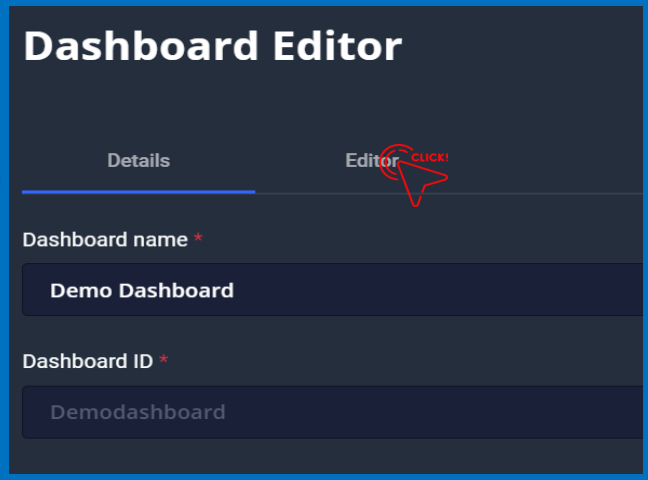
7. Hover the mouse over the dashboard



8. Click on the edit dashboard button.



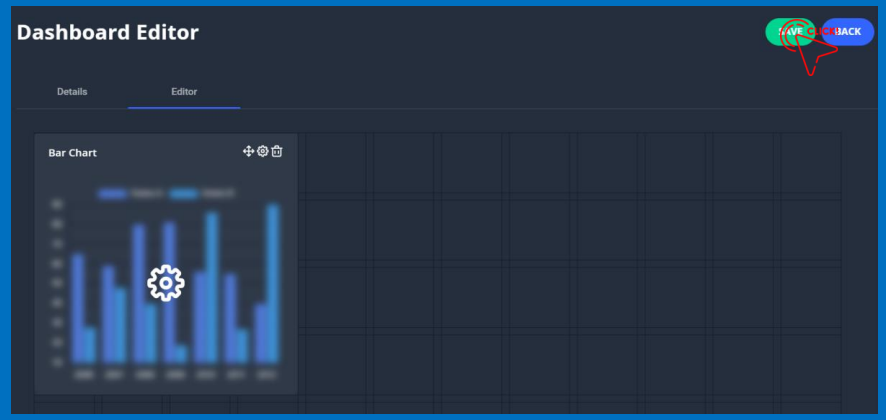
9. Click on the editor section



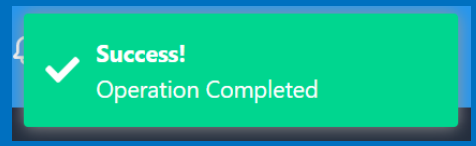
Dashboard edit – widget creation

10. Drag the Barchart in the grid

11. Click on save



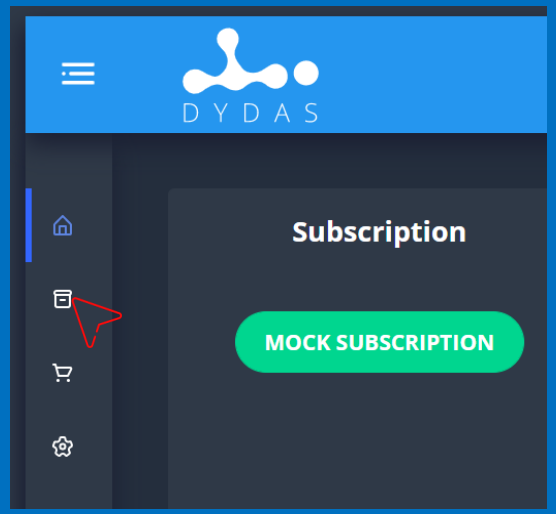
The dashboard is now created. You will be redirected to the dashboard list and a notification like the one shown will appear.



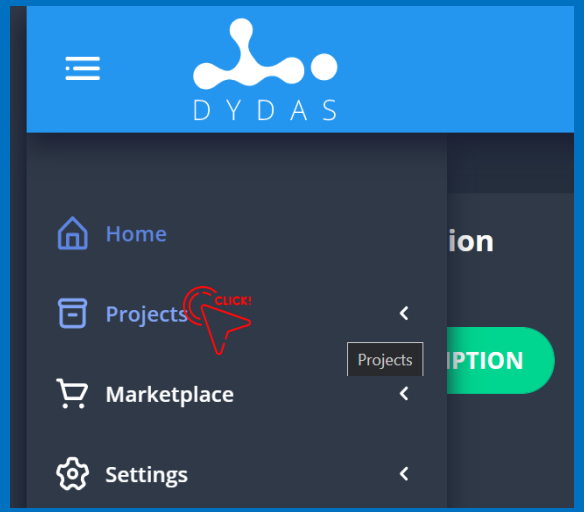
Co-financed by the Connecting Europe Facility of the European Union

Dashboard edit – widget settings

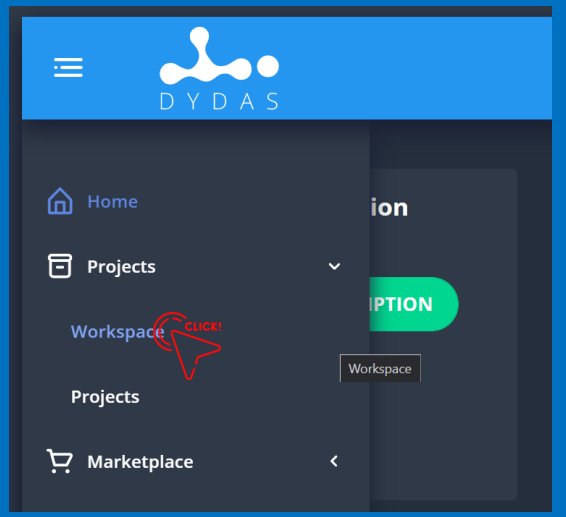
1. Place the mouse in the project icon



2. Click on the projects section

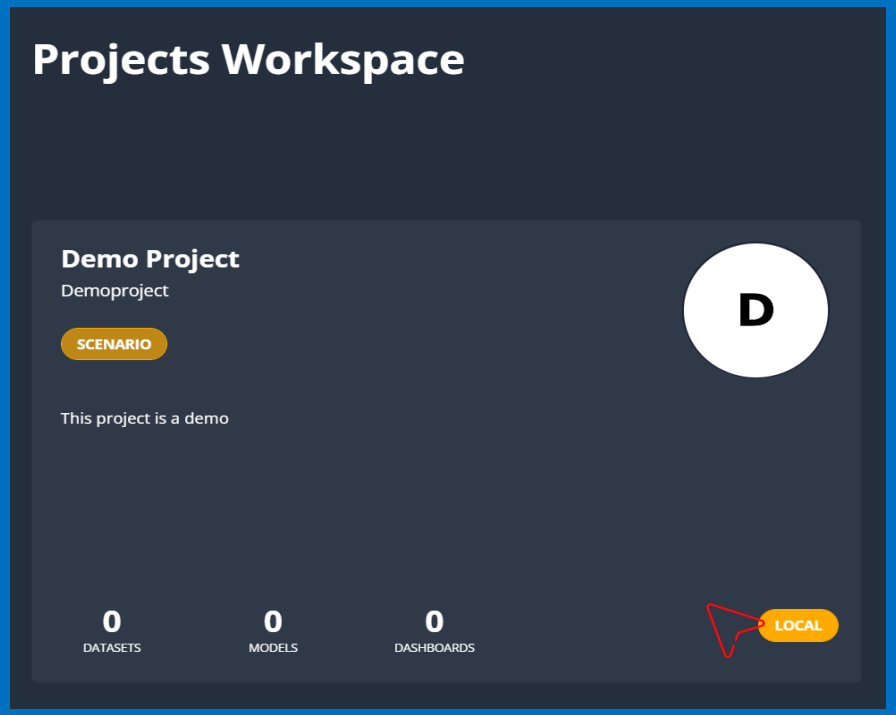


3. Click on the workspace section

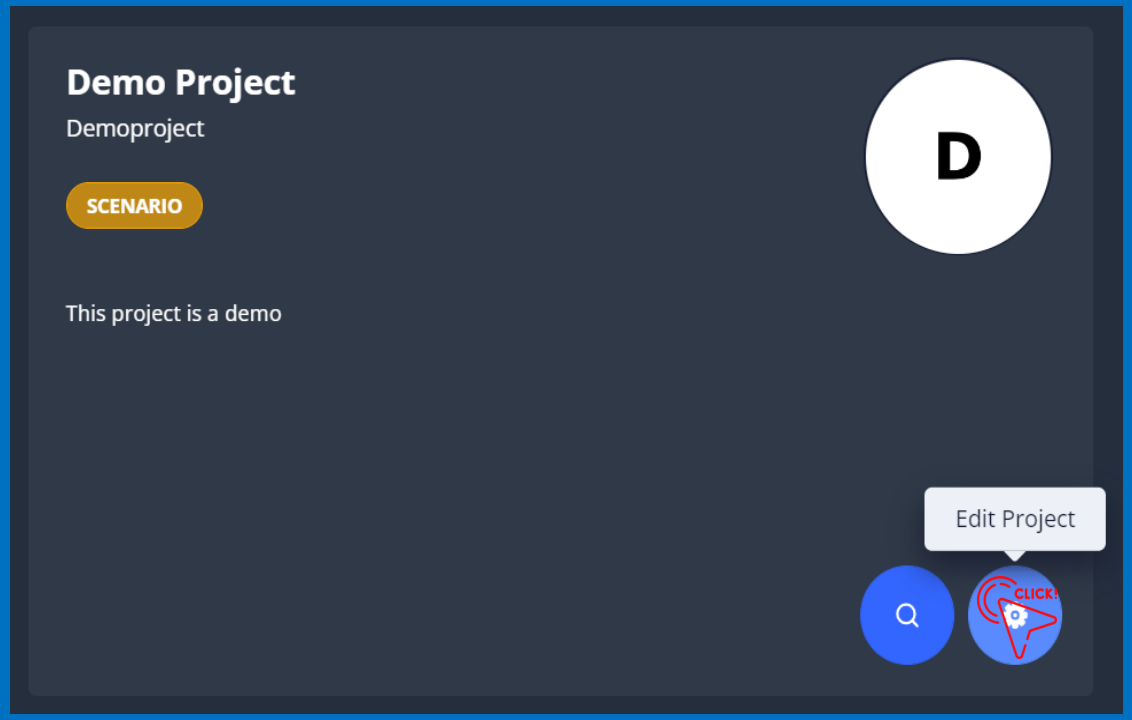


Dashboard edit – widget settings

4. Place the mouse in the project card

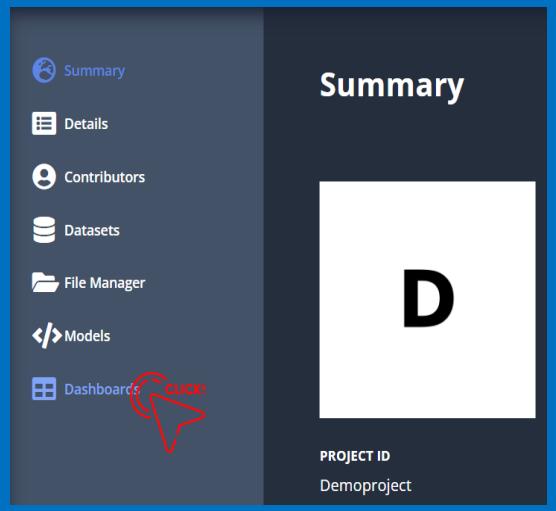


5. Click on the edit project button

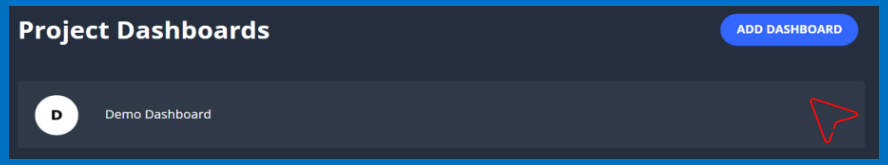


Dashboard edit – widget settings

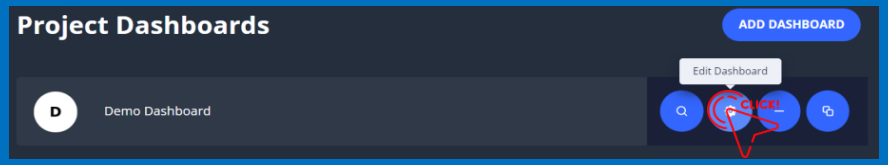
6. Click on the dashboard section.



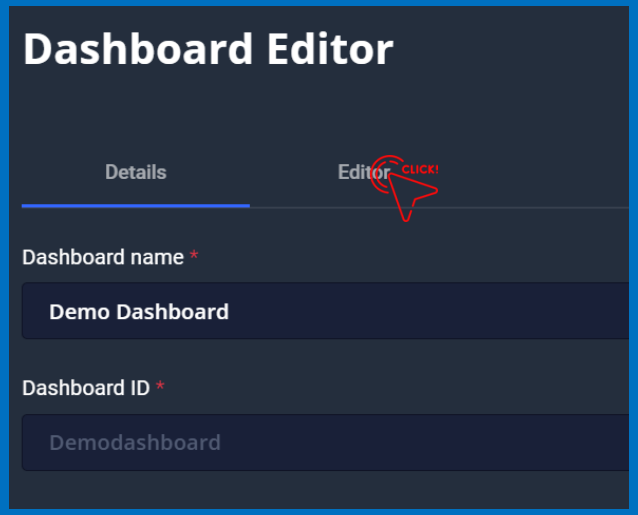
7. Hover the mouse over the dashboard.



8. Click on the edit dashboard button.

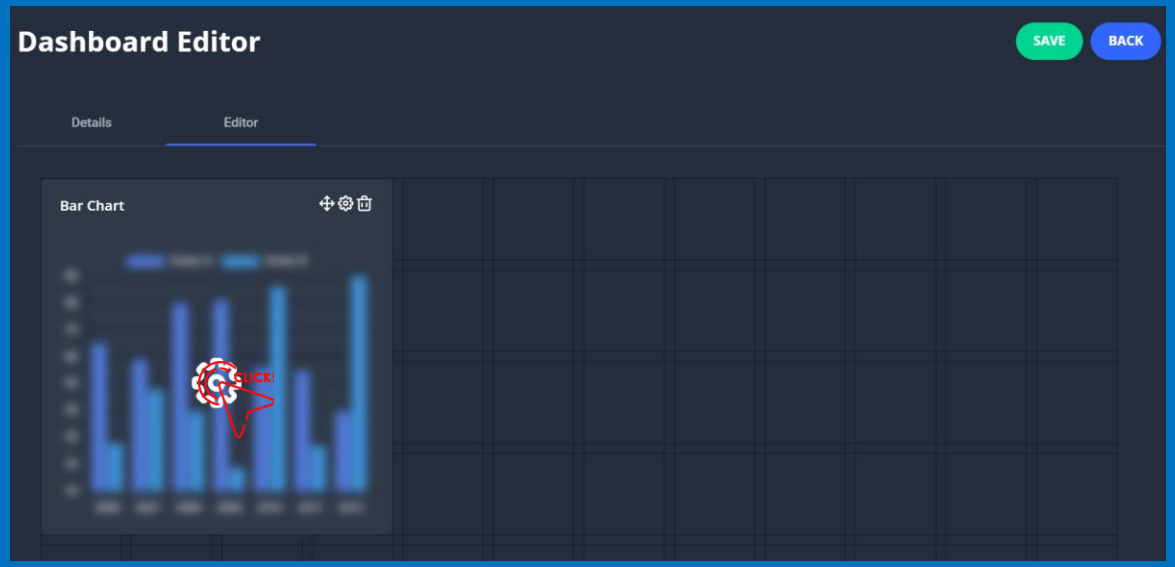


9. Click on the editor section.

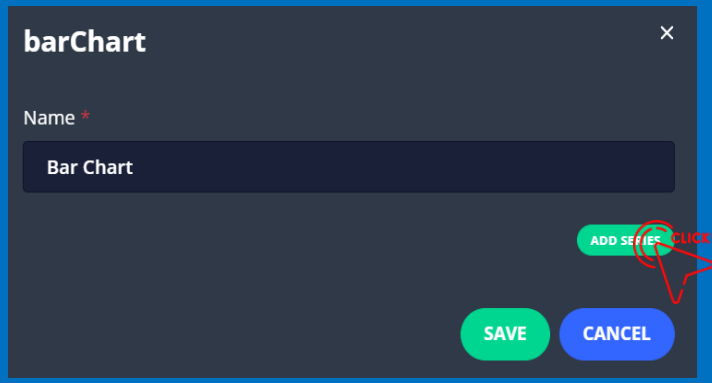


Dashboard edit – widget settings

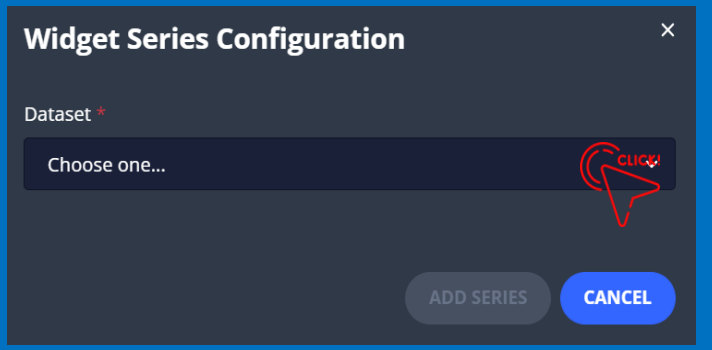
11. Click on the settings button



12. Here you can change the name. Click on add series to connect the widget to a dataset



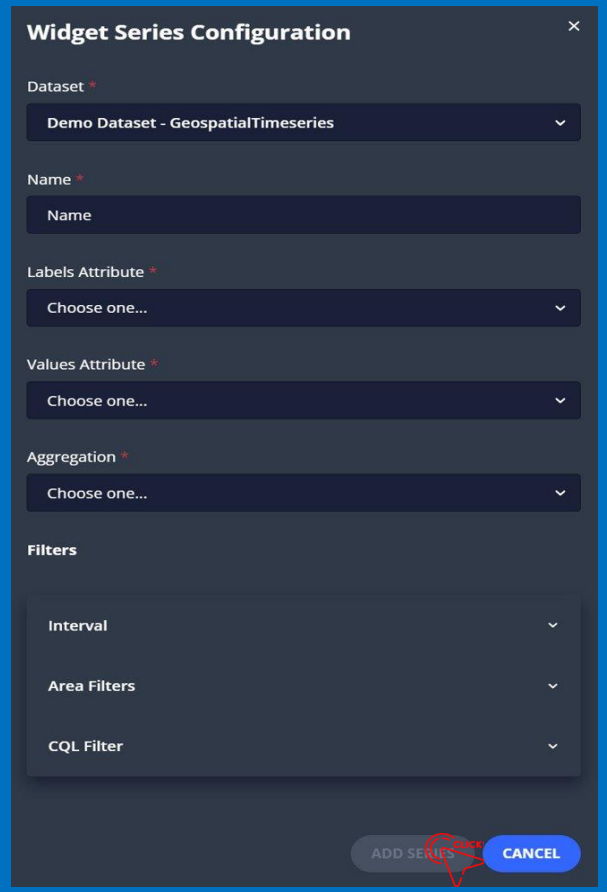
13. Select a dataset



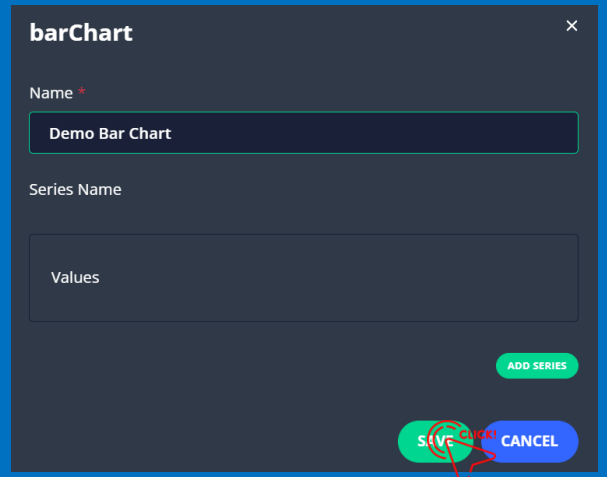
Dashboard edit – widget settings

14. Fill the form and click on the add series button.

(For the filters check the dataset download section)



15. Click on the save widget button



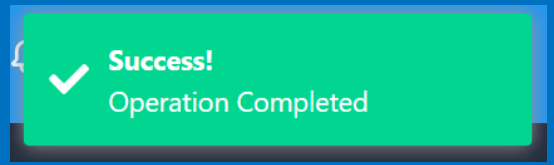
Co-financed by the Connecting Europe Facility of the European Union

Dashboard edit – widget settings

16. Click on the save dashboard button

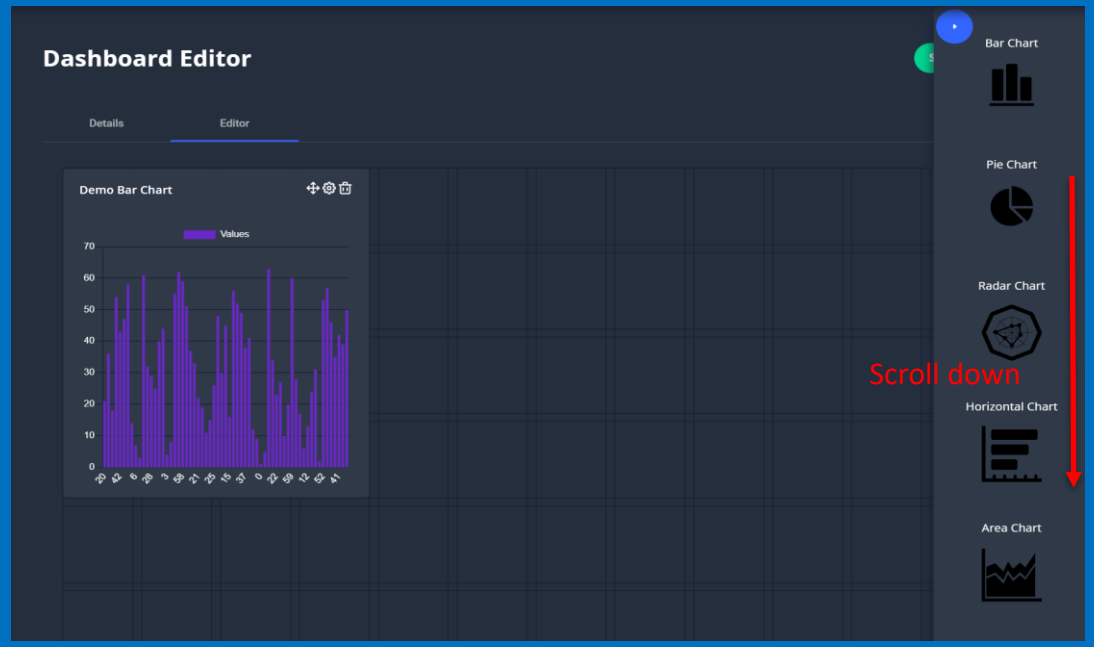
IMPORTANT!!! All the changes to the dashboard and to the widgets will be saved only if the dashboard is saved.

The dashboard is edited, and a notification like the one shown will appear.

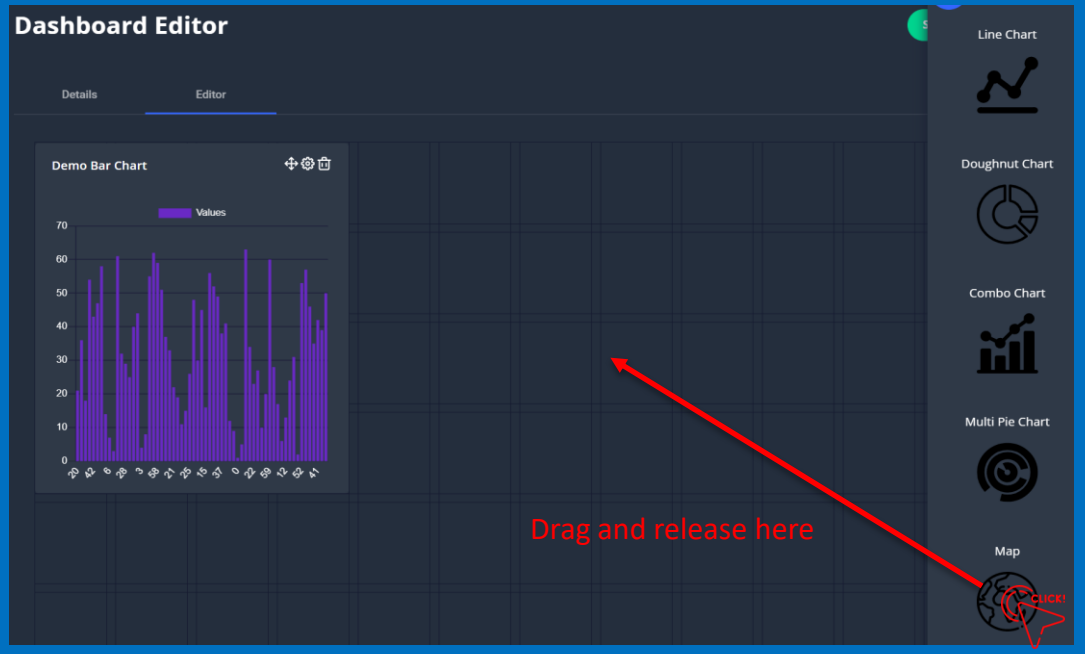


Dashboard edit – map widget

1. In the dashboard editor scroll down in the widget section until the map widget is shown



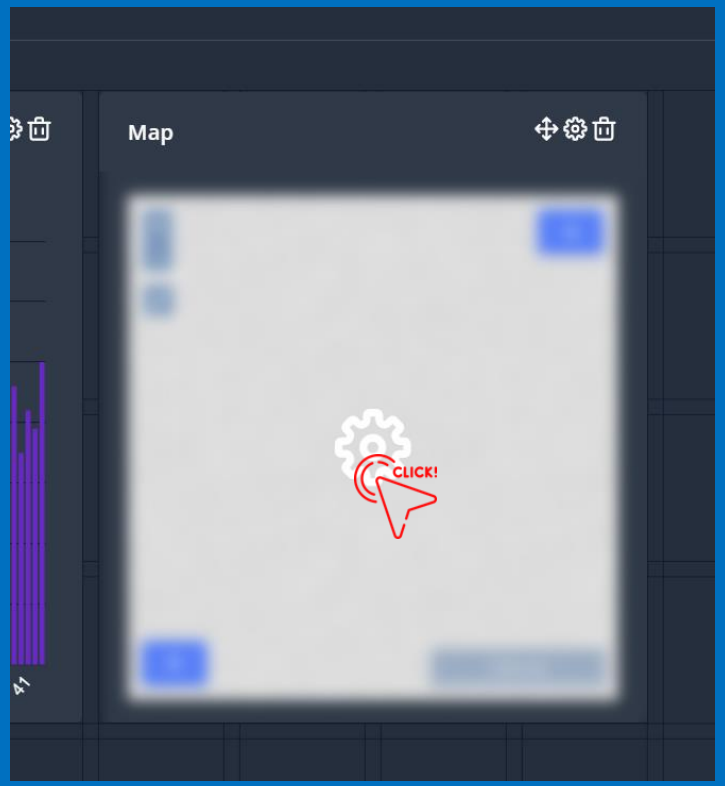
2. Drag the map widget to the grid



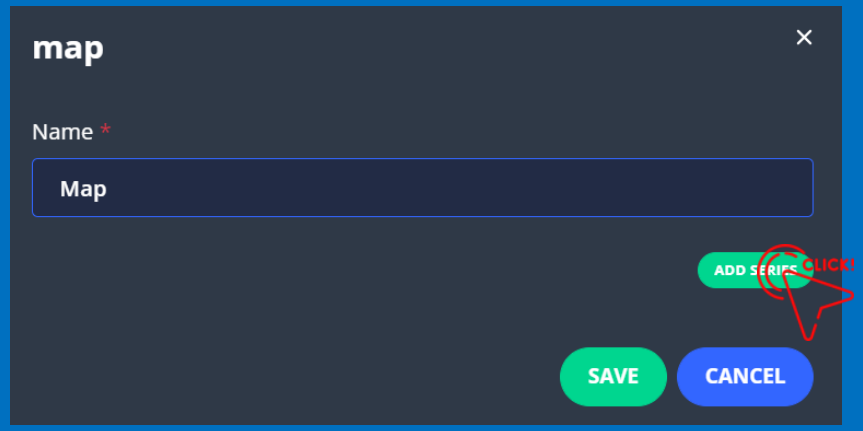
Co-financed by the Connecting Europe Facility of the European Union

Dashboard edit – map widget

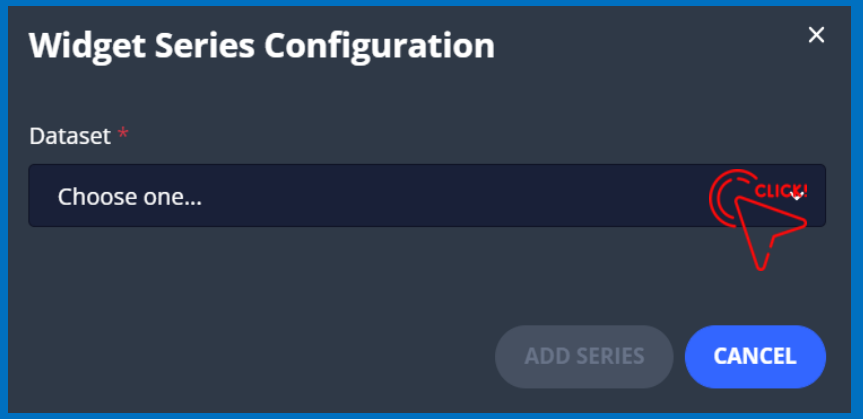
3. Click on the edit widget button



4. Here you can change the name. Click on add series to connect the widget to a dataset

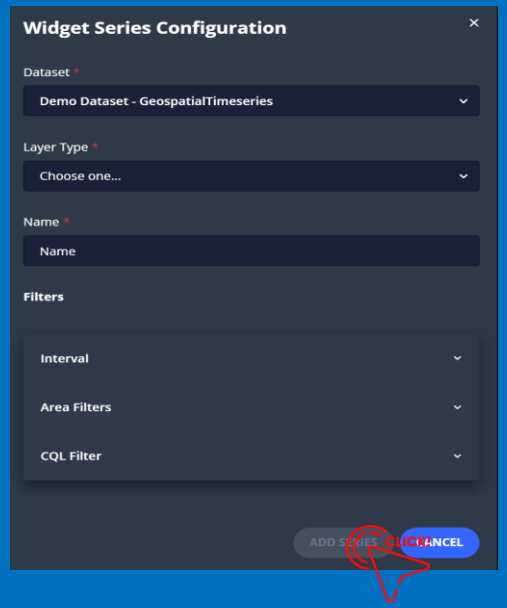


5. Select a dataset

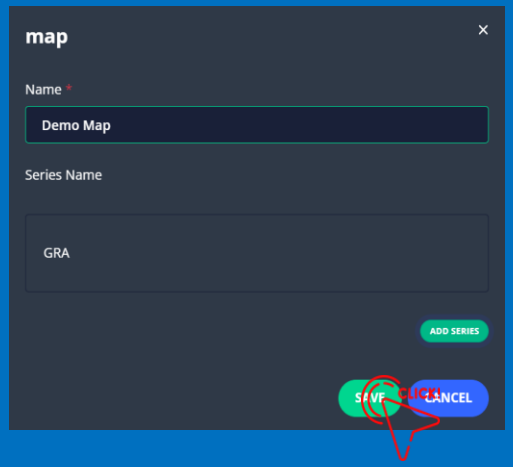


Dashboard edit – map widget

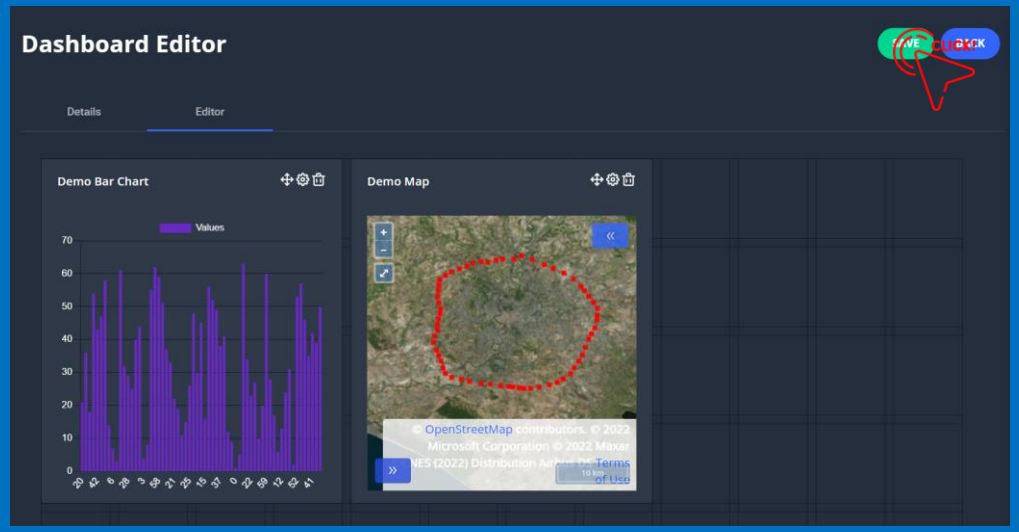
6. Fill the form and click on the add series button



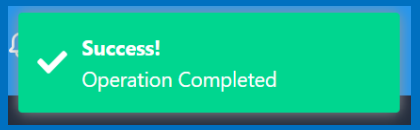
7. Click on the save widget button



8. Click on the save dashboard button



The dashboard is now edited.
You will be redirected to the dashboard list and a notification like the one on the right will appear.



Contributors

Contributors are users that have access to the project and can work/edit the project.

Contributors must be registered users on the platform.

Project contributors can read and edit any section of the project but cannot delete and publish projects and cannot change project owner: these features are reserved to project owner.

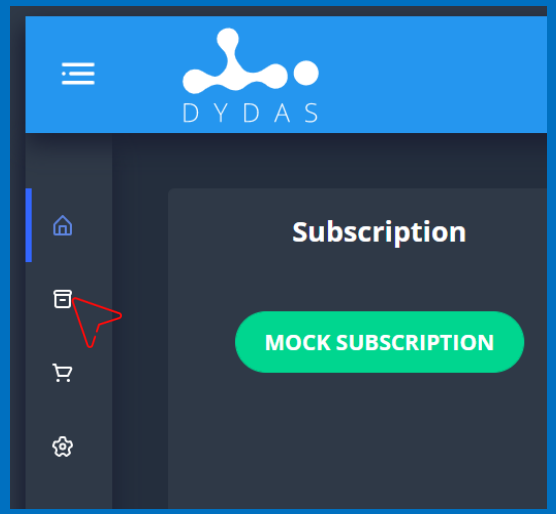
A contributor **can't**:

- Delete a project
- Publish a project
- Associate non open dataset
- Add other contributors

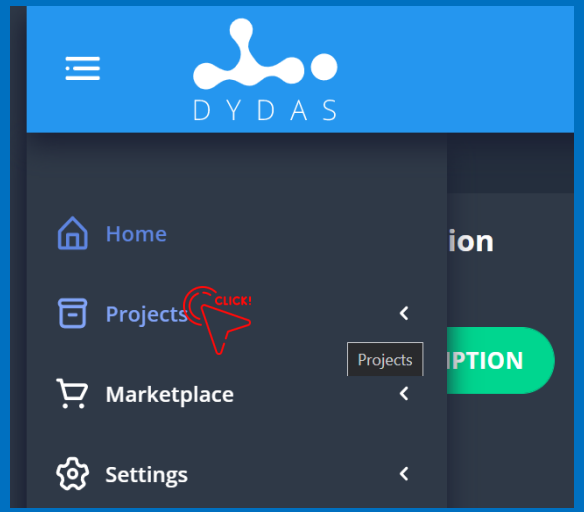
Please Note: Project owner cannot be changed

Contributors – add contributor

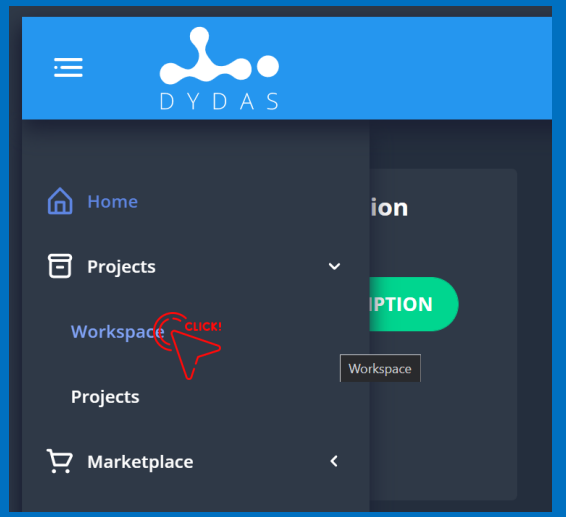
1. Place the mouse in the project icon



2. Click on the projects section

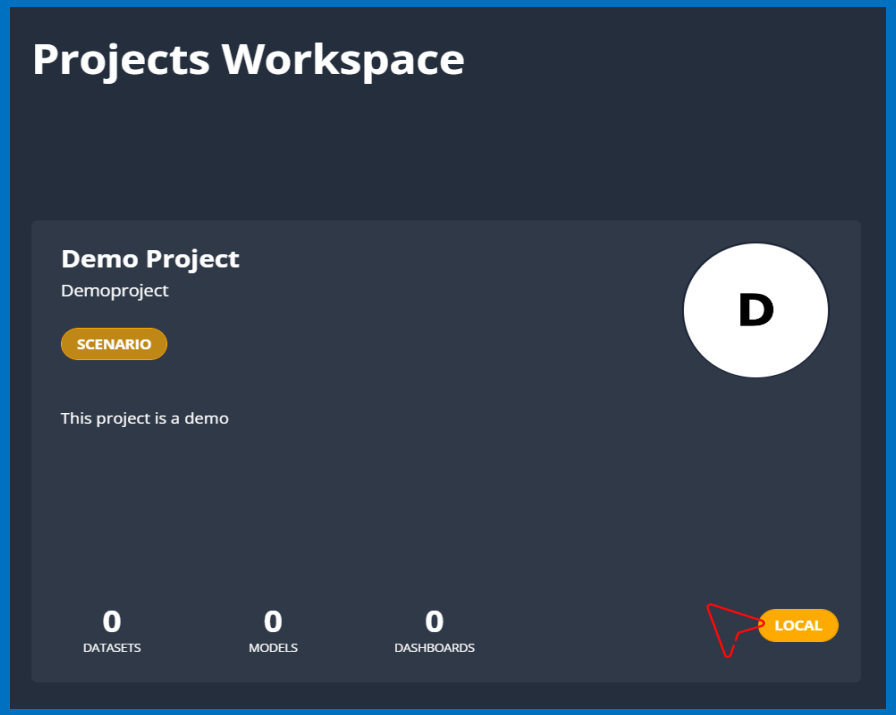


3. Click on the workspace section

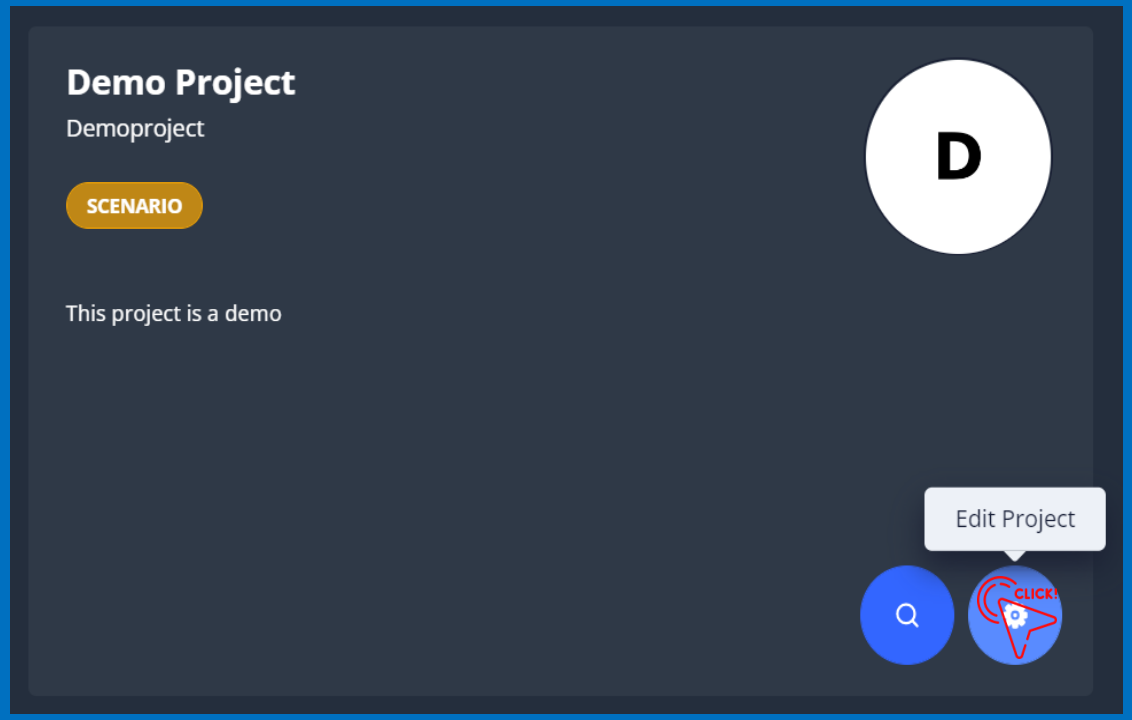


Contributors – add contributor

4. Place the mouse in the project card

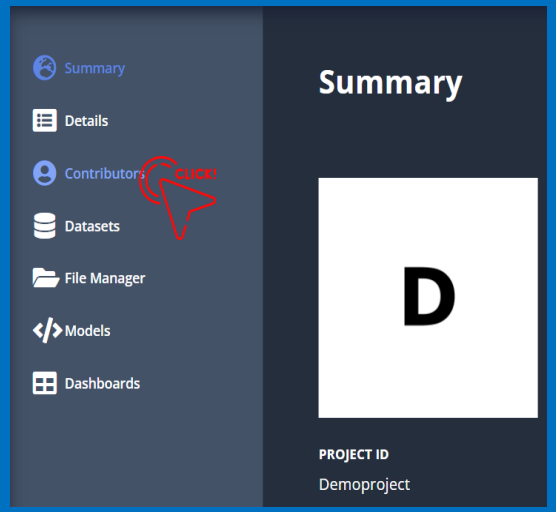


5. Click on the edit project button

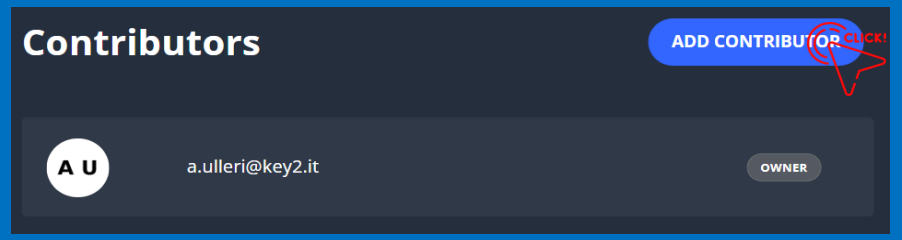


Contributors – add contributor

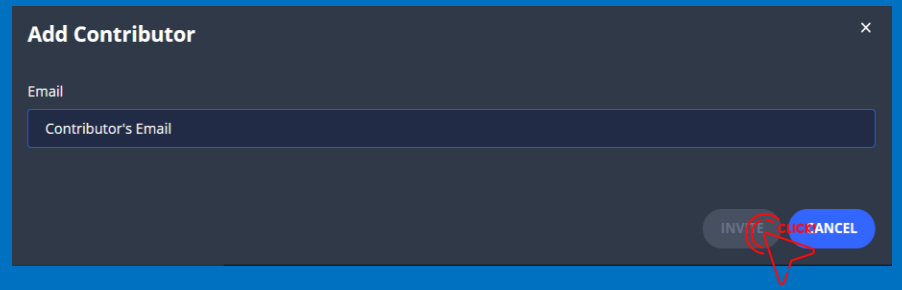
6. Click on the contributor section



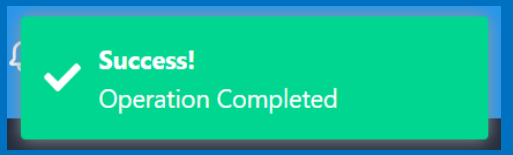
7. Click on the add contributor button



8. Fill the form and click the invite button

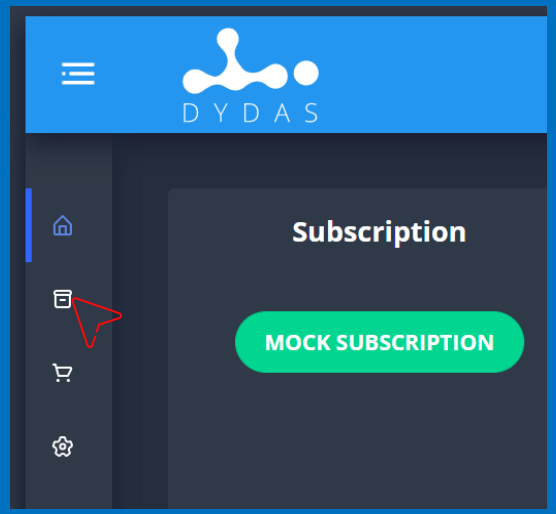


A message like the one to the right appears and the contributor is added to the project

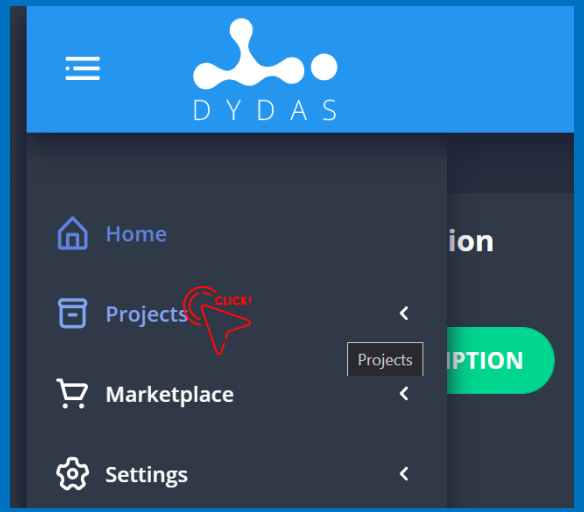


Contributors – delete contributor

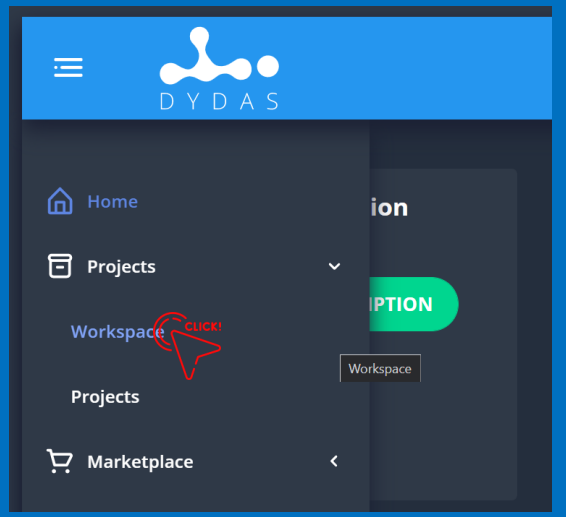
1. Place the mouse in the project icon



2. Click on the projects section

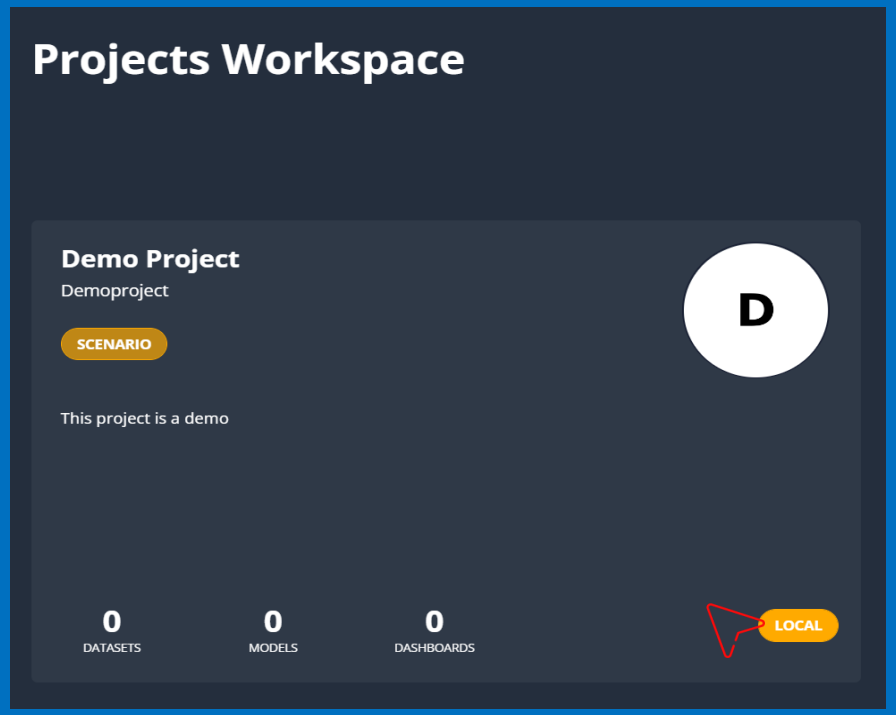


3. Click on the workspace section

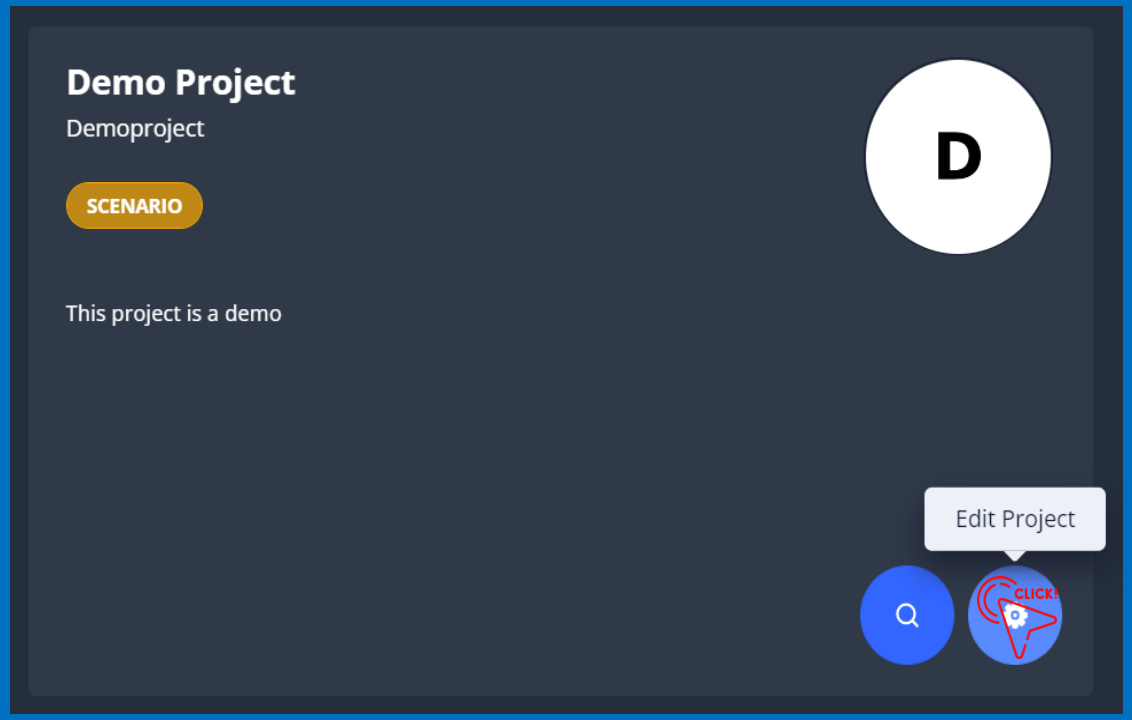


Contributors – delete contributor

4. Place the mouse in the project card

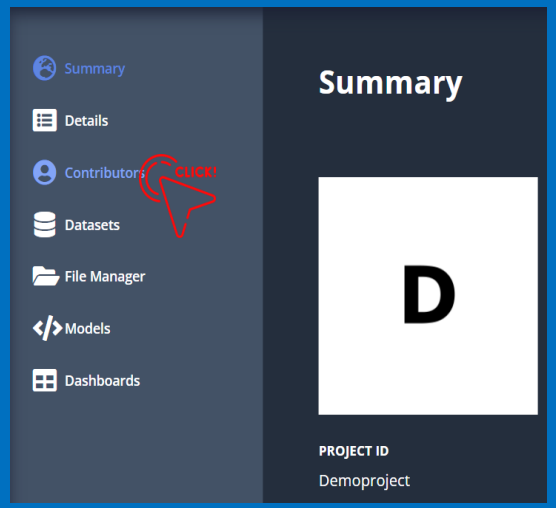


5. Click on the edit project button

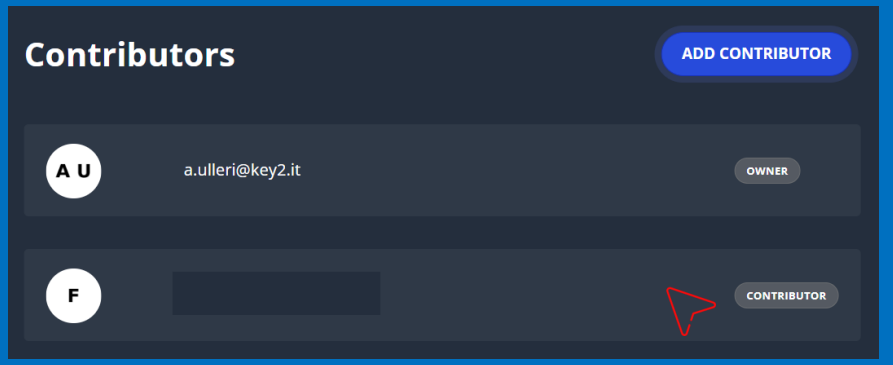


Contributors – delete contributor

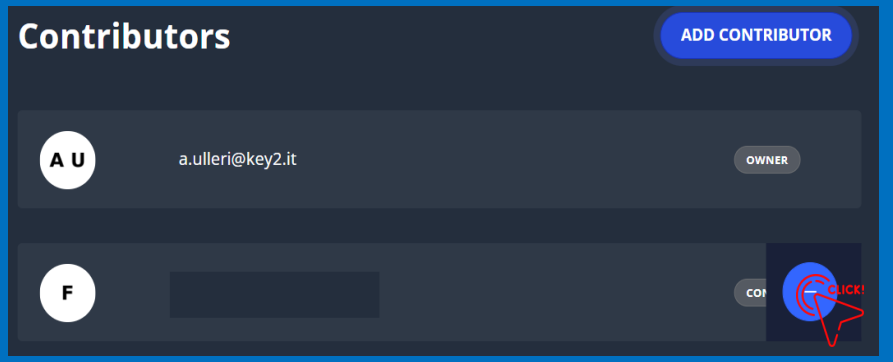
6. Click on the contributor section



7. Hover the mouse over the contributor you want to delete

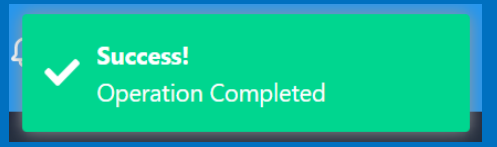


8. Click the delete contributor button



9. Click delete

A message like the one to the right appears and the contributor is deleted from the project



Co-financed by the Connecting Europe Facility of the European Union

Hackathon explained

The data represent a weighted directed graph where each vertex is a crossroad, and the edges are the roads.

Each road, or edge, is directed and is named after the direction.
Example:

Vertex: A,B,
Edges: AB (edge from vertex A to B), BA (edge from vertex B to A) and so on.

Each crossroad (or vertex) has 1 or more semaphores, each semaphore regulates the traffic of 2 roads only, one entering the vertex and one exiting it.
Example:

Vertex: A,B,C
Edges: AB, BA, BC, CB, CA
Semaphore:

Vertex A has 1 semaphore that regulates (CA, AB)
Vertex B has 2 semaphores that regulates (AB, BC), (CB, BA)
Vertex C has 1 semaphore that regulates (BC, CA)

The U inversion is not allowed, so there are not semaphores that regulates (AB, BA), (CB, BC) ecc.

Hackathon explained

Each semaphore can have only 2 colors, **RED** and **GREEN**.

For each vertex, there can only be 1 GREEN semaphore at the time.

Example:

In a vertex, with 4 semaphores we have:

- GREEN, RED, RED, RED
- RED, GREEN, RED, RED
- RED, RED, GREEN, RED
- RED, RED, RED, GREEN

The semaphore changes each 60s.

Each semaphore registers always the number of cars waiting to cross (RED) or the number of cars that crossed the semaphore (GREEN).

Each road has a different length, and the cars will take time to cross that road depending on its length.

It is possible to retrieve all the data in real-time via Context Broker

Context Broker

It is possible to access all the hackathon data via **CRUD** operation.

Anyone can access the data, as long as they are authenticated to the DYDAS Portal.

Any request to retrieve the data is allowed by **GET** method, all the other requests are denied by default (updating, deleting and posting new entities is forbidden).

The only exception regards the subscription: it is indeed possible to create a subscription via **POST** request.

Remember to add the access token in the headers to authenticate.

Orion context broker endpoint:
https://preview.dydas.eu/orion-hackathon/

If additional documentation is required visit:
https://fiware-orion.readthedocs.io/en/1.3.0/user/walkthrough_apiv2/index.html

Context Broker – Access token

The access token should be used to authenticate requests and it should be added in the headers with the key-value pair:

Authorization – Bearer [access token]

To obtain the access token make a POST request to the endpoint https://preview.dydas.eu/api/authorization-server-oauth2/oauth/token

Headers: (Key – Value)

```
Authorization – Basic VVNFUl9DTElFTlRfQVBQOnBhc3N3b3Jk
Content Type – multipart/form-data
```

Body: (Key – Value)

```
grant_type – password
username – [your username]
password – [your password]
```

The response is in JSON format and has the following structure:

```
{
    "access_token": "[access token]",
    "token_type": "bearer",
    "refresh_token": "[refresh token]",
    "expires_in": [expire time],
    "scope": "[whitespace-separated roles]",
    "email": "[username]",
    "jti": "[jwt id]"
}
```

DYDAS

Co-financed by the Connecting Europe Facility of the European Union

DYDAS: OPEN AND PRIVATE DATA FOR THE PUBLIC SECTOR AND INDUSTRIES