Module 6.2

How to run a test using a workflow manager

Workflow manager

Workflow manager

Manages experiments across different platforms

Automatic handling of job submission, dependencies, and error recovery.

Improves workflow scalability and traceability.

EDITO entrypoint to HPC clusters

AUTOSUBMIT

Think workflow!

When solving big computational problems, it is common to think of your top-level **main function** as a **workflow** that is run through a **workflow manager**.

Think workflow!

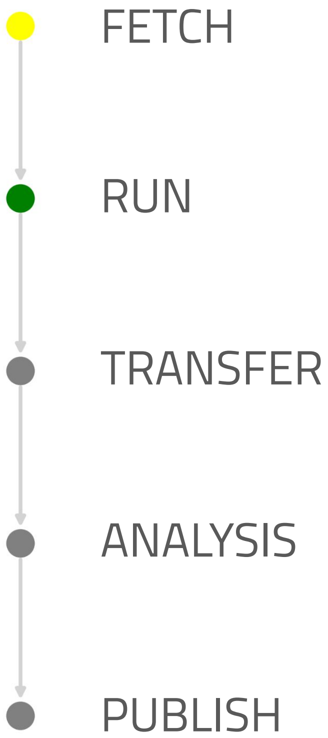
When solving big computational problems, it is common to think of your top-level **main function** as a **workflow** that is run through a **workflow manager**.

```python
def fetch_inputs():
    ...
def run_simulation():
    ...
def transfer_data():
    ...
def analyze_data():
    ...
def publish_paper():
    ...
def main():
    fetch_inputs()
    run_simulation()
    transfer_data()
    analyze_data()
    publish_paper()
if __name__ == "__main__":
    main()
```

Think workflow!

When solving big computational problems, it is common to think of your top-level **main function** as a **workflow** that is run through a **workflow manager**.

```python
def fetch_inputs():
    ...
def run_simulation():
    ...
def transfer_data():
    ...
def analyze_data():
    ...
def publish_paper():
    ...
def main():
    fetch_inputs()
    run_simulation()
    transfer_data()
    analyze_data()
    publish_paper()
if __name__ == "__main__":
    main()
```



FETCH

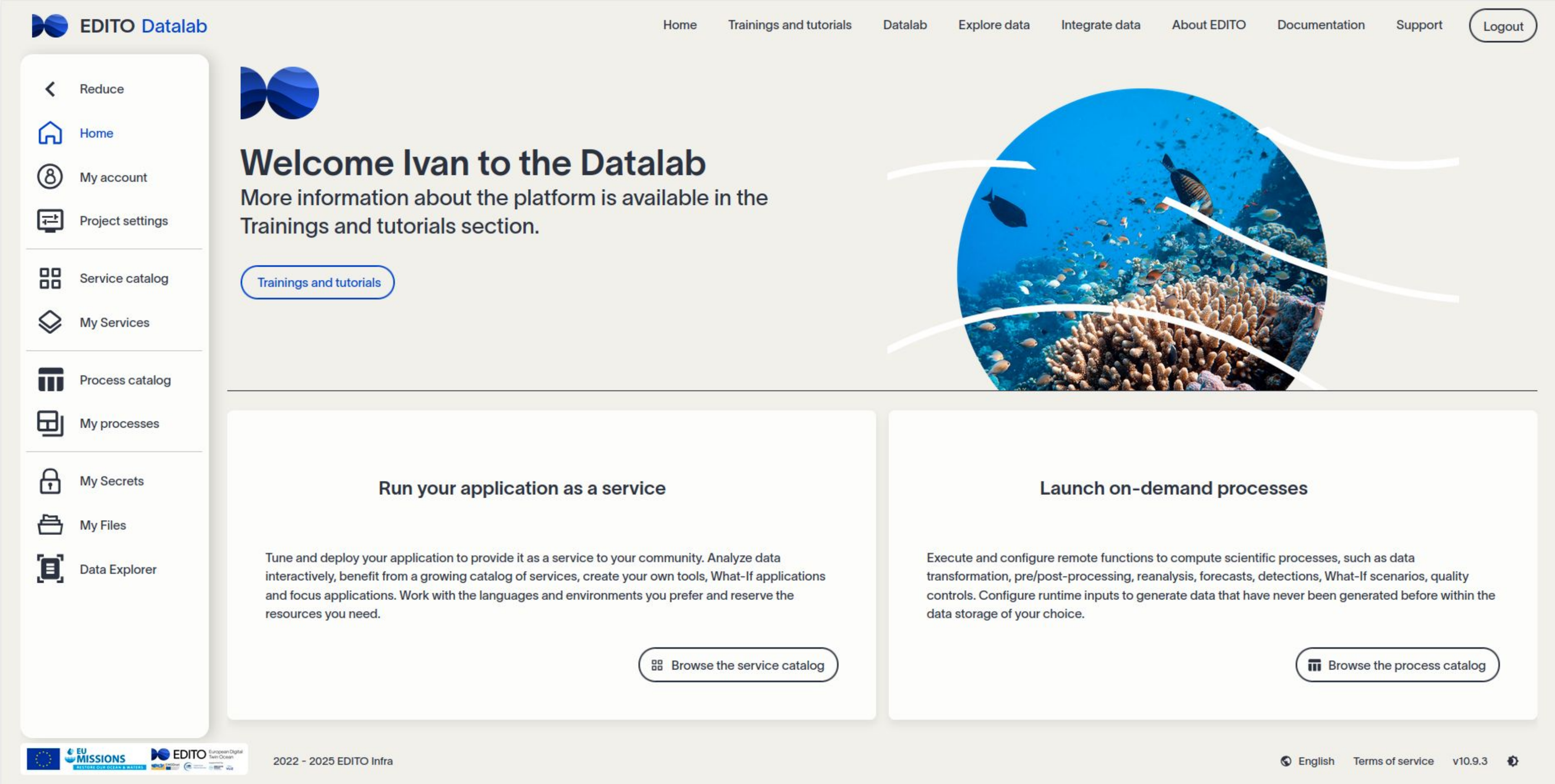
RUN

TRANSFER

ANALYSIS

PUBLISH

EDITO Ecosystem



Welcome Ivan to the Datalab

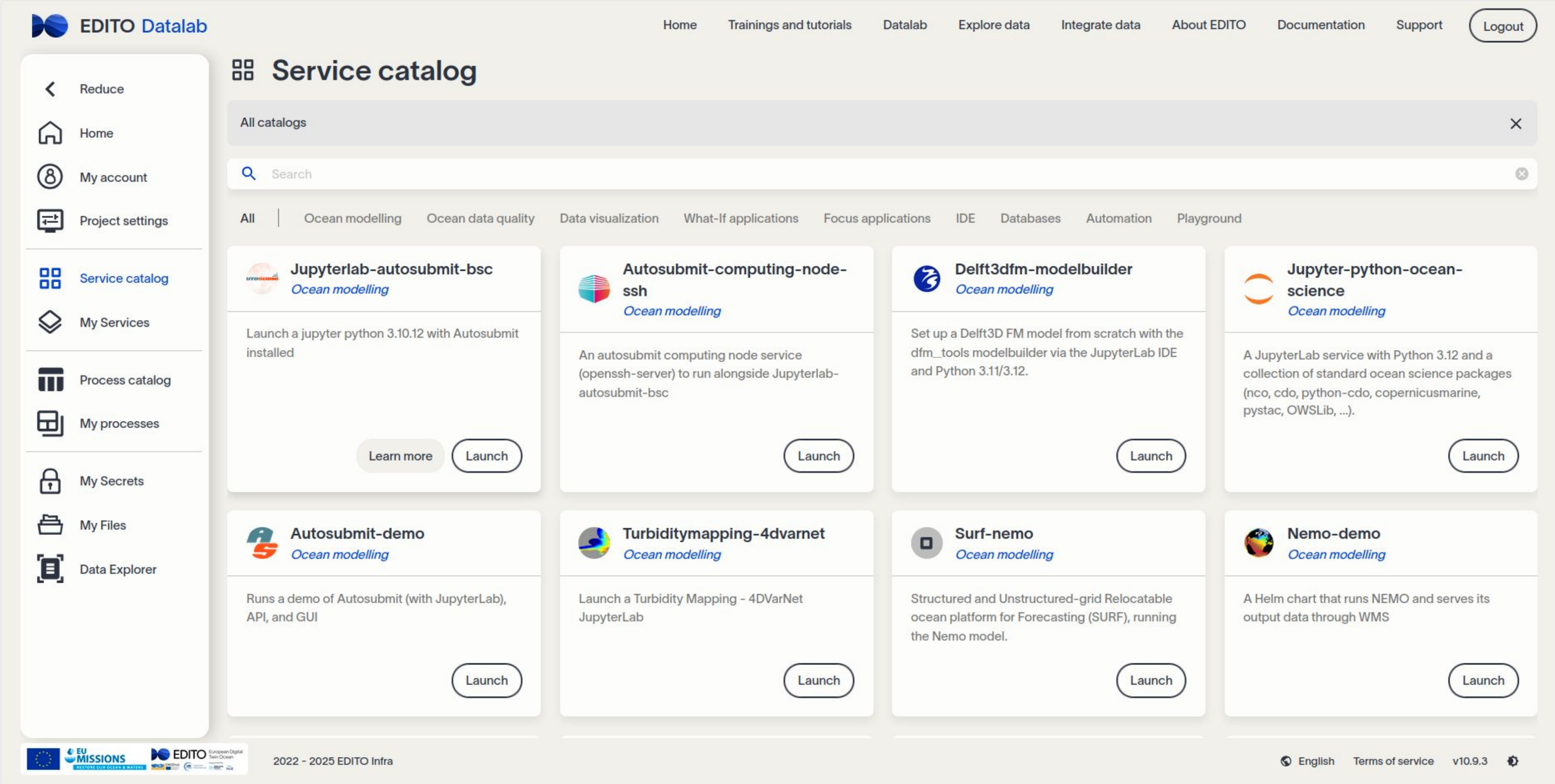
More information about the platform is available in the Trainings and tutorials section.

Run your application as a service

Tune and deploy your application to provide it as a service to your community. Analyze data interactively, benefit from a growing catalog of services, create your own tools, What-If applications and focus applications. Work with the languages and environments you prefer and reserve the resources you need.

Launch on-demand processes

Execute and configure remote functions to compute scientific processes, such as data transformation, pre/post-processing, reanalysis, forecasts, detections, What-If scenarios, quality controls. Configure runtime inputs to generate data that have never been generated before within the data storage of your choice.



Service catalog

Jupyterlab-autosubmit-bsc
Ocean modelling

Launch a jupyter python 3.10.12 with Autosubmit installed

Autosubmit-computing-node-ssh
Ocean modelling

An autosubmit computing node service (openssh-server) to run alongside Jupyterlab-autosubmit-bsc

Delft3dfm-modelbuilder
Ocean modelling

Set up a Delft3D FM model from scratch with the dfm_tools modelbuilder via the JupyterLab IDE and Python 3.11/3.12.

Jupyter-python-ocean-science
Ocean modelling

A JupyterLab service with Python 3.12 and a collection of standard ocean science packages (nco, cdo, python-cdo, copernicusmarine, pystac, OWSLib, ...).

Autosubmit-demo
Ocean modelling

Runs a demo of Autosubmit (with JupyterLab), API, and GUI

Turbiditymapping-4dvarnet
Ocean modelling

Launch a Turbidity Mapping - 4DVarNet JupyterLab

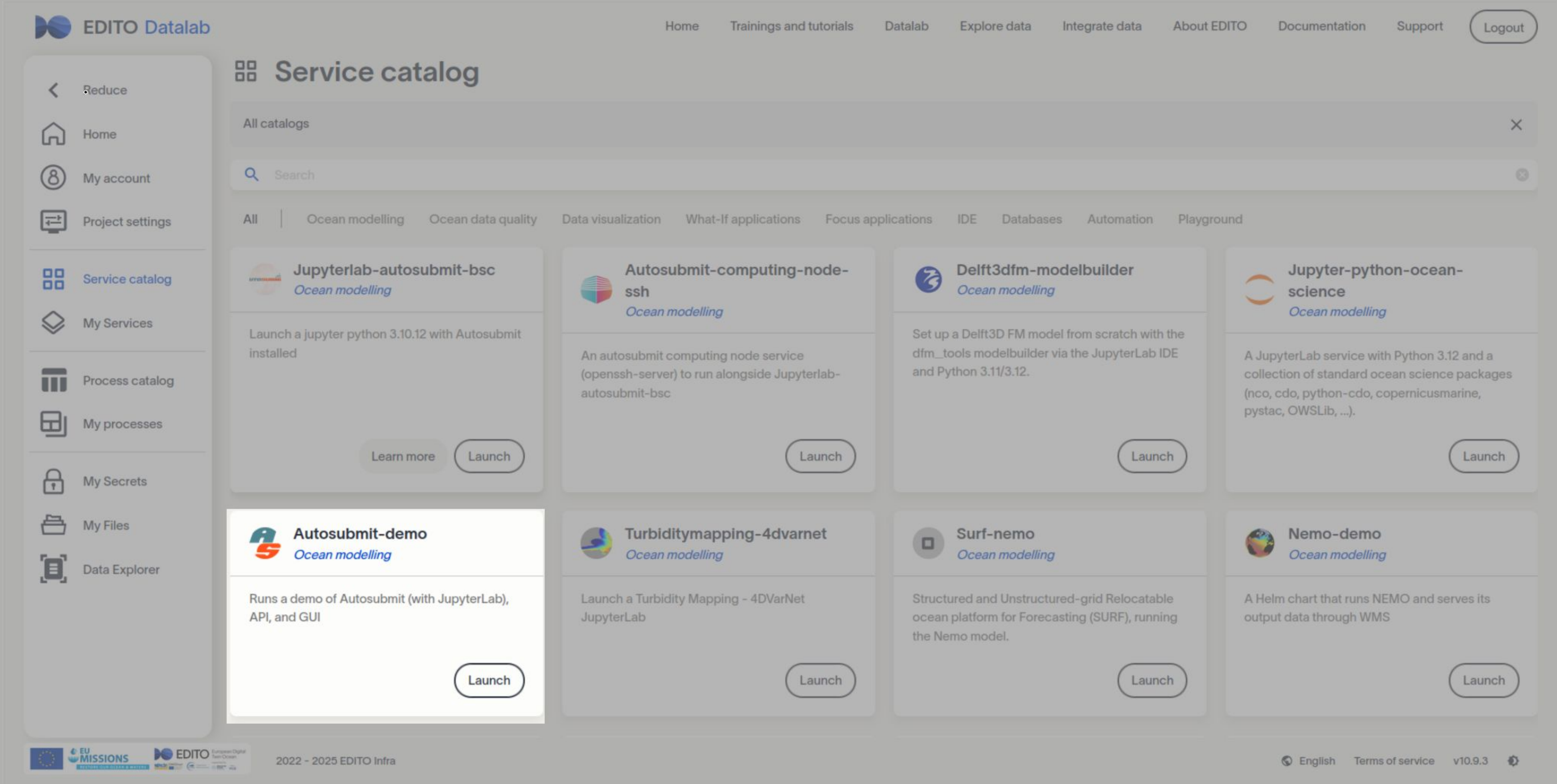
Surf-nemo
Ocean modelling

Structured and Unstructured-grid Relocatable ocean platform for Forecasting (SURF), running the Nemo model.

Nemo-demo
Ocean modelling

A Helm chart that runs NEMO and serves its output data through WMS

2022 - 2025 EDITO Infra



Service catalog

Jupyterlab-autosubmit-bsc
Ocean modelling
Launch a jupyter python 3.10.12 with Autosubmit installed

Autosubmit-computing-node-ssh
Ocean modelling
An autosubmit computing node service (openssh-server) to run alongside Jupyterlab-autosubmit-bsc

Delft3dfm-modelbuilder
Ocean modelling
Set up a Delft3D FM model from scratch with the dfm_tools modelbuilder via the JupyterLab IDE and Python 3.11/3.12.

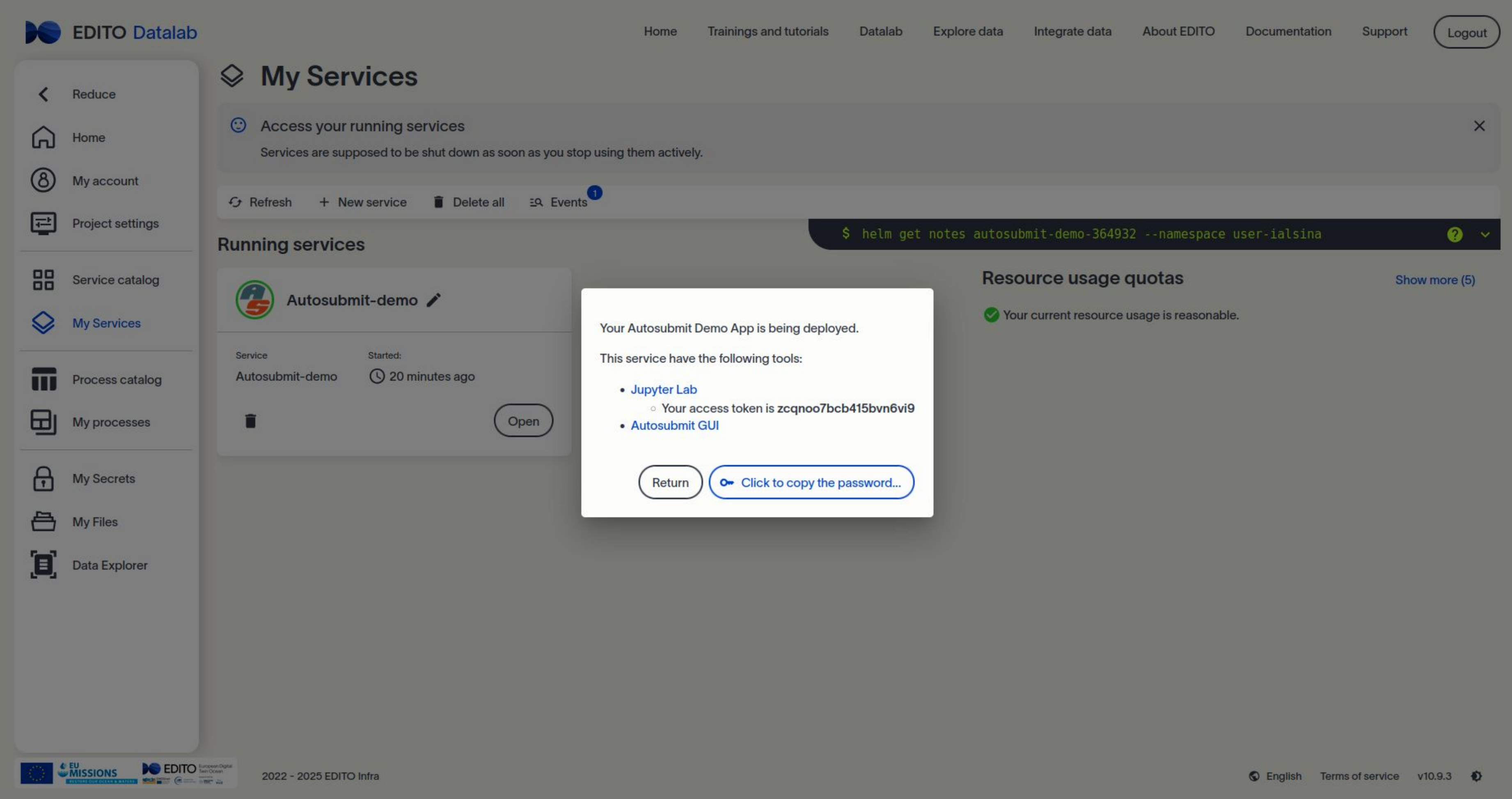
Jupyter-python-ocean-science
Ocean modelling
A JupyterLab service with Python 3.12 and a collection of standard ocean science packages (nco, cdo, python-cdo, copernicusmarine, pystac, OWSLib, ...).

Autosubmit-demo
Ocean modelling
Runs a demo of Autosubmit (with JupyterLab), API, and GUI

Turbiditymapping-4dvarnet
Ocean modelling
Launch a Turbidity Mapping - 4DVarNet JupyterLab

Surf-nemo
Ocean modelling
Structured and Unstructured-grid Relocatable ocean platform for Forecasting (SURF), running the Nemo model.

Nemo-demo
Ocean modelling
A Helm chart that runs NEMO and serves its output data through WMS



My Services

Access your running services

Services are supposed to be shut down as soon as you stop using them actively.

```
$ helm get notes autosubmit-demo-364932 --namespace user-ialsina
```

Running services

Autosubmit-demo

Service: Autosubmit-demo

Started: 20 minutes ago

Resource usage quotas

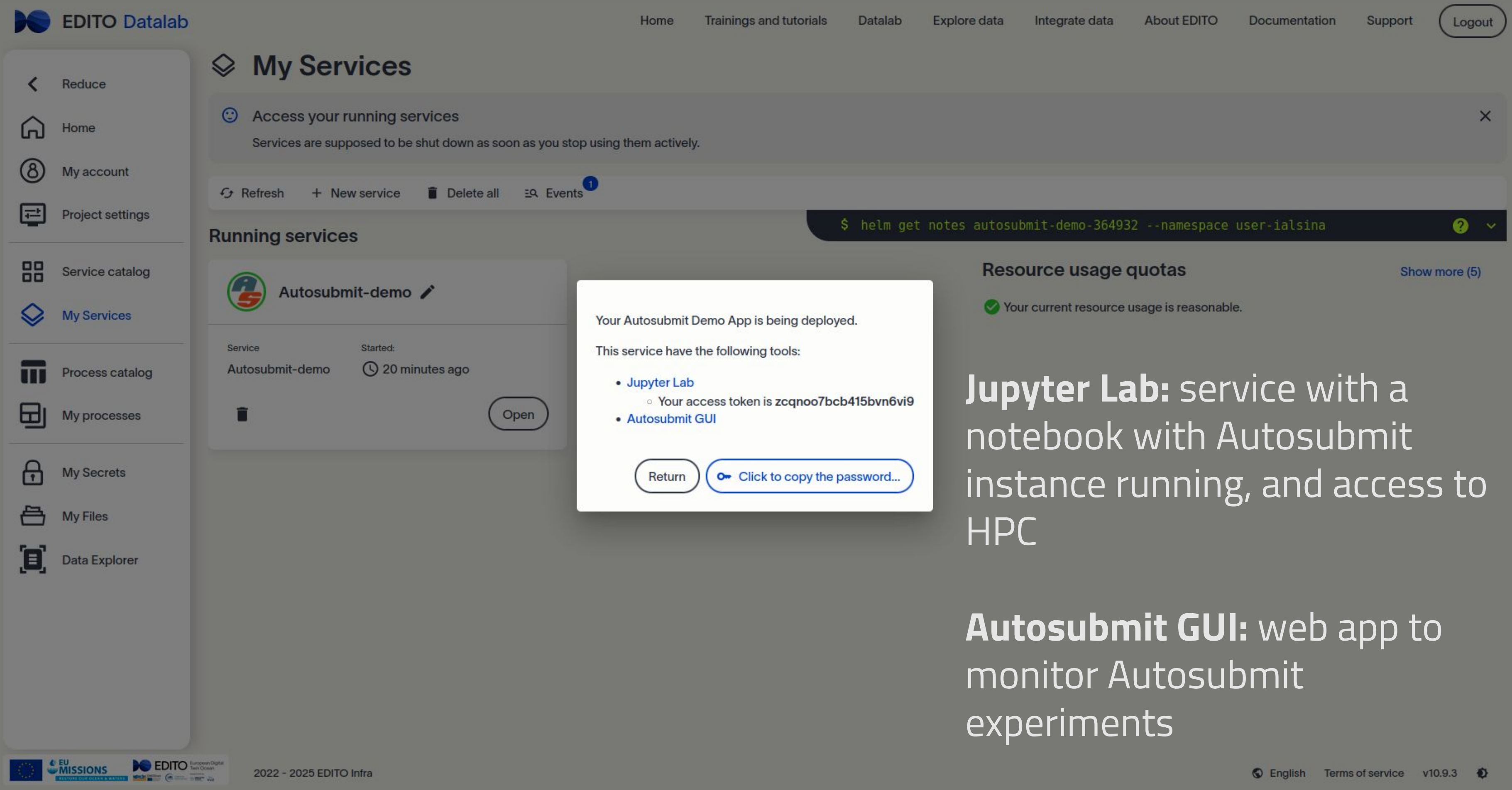
Your current resource usage is reasonable.

Your Autosubmit Demo App is being deployed.

This service have the following tools:

- Jupyter Lab
  - Your access token is **zcqnoo7bcb415bvn6vi9**
- Autosubmit GUI

2022 - 2025 EDITO Infra



My Services

Access your running services

Services are supposed to be shut down as soon as you stop using them actively.

```
$ helm get notes autosubmit-demo-364932 --namespace user-ialsina
```

Running services

Autosubmit-demo

Service: Autosubmit-demo

Started: 20 minutes ago

Your Autosubmit Demo App is being deployed.

This service have the following tools:

- Jupyter Lab
  - Your access token is **zcqnoo7bcb415bvn6vi9**
- Autosubmit GUI

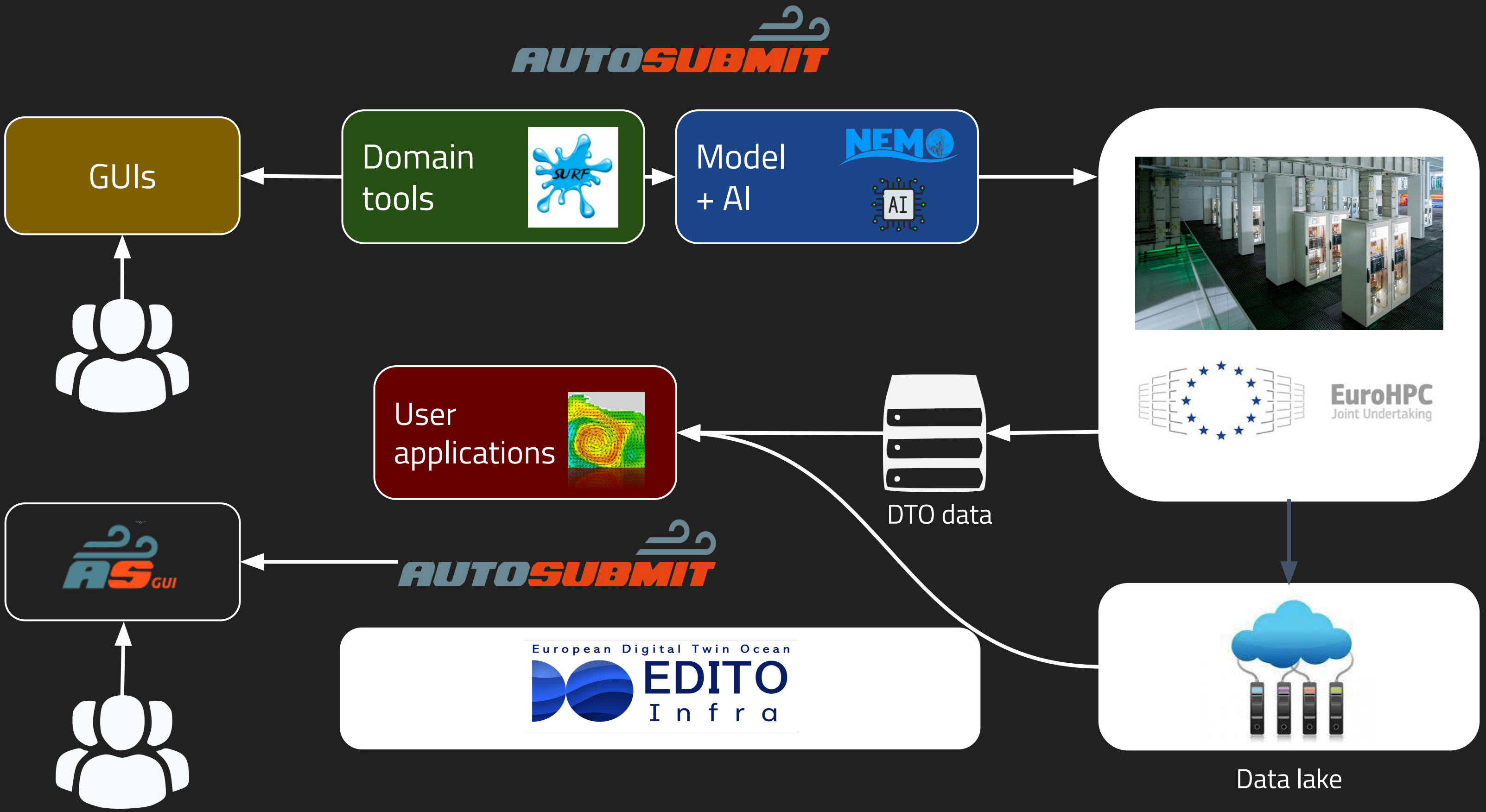
Resource usage quotas

Your current resource usage is reasonable.

2022 - 2025 EDITO Infra

Jupyter Lab: service with a notebook with Autosubmit instance running, and access to HPC

Autosubmit GUI: web app to monitor Autosubmit experiments



GUIs

Domain tools

Model + AI

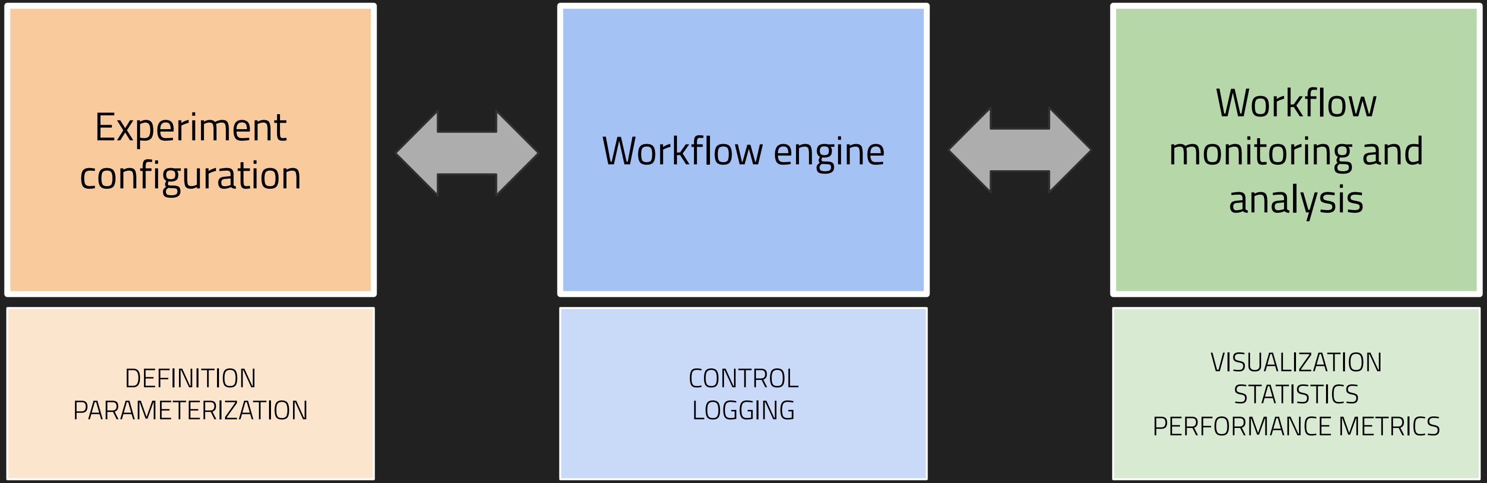
User applications

DTO data

Data lake

Autosubmit

AUTOSUBMIT



Experiment configuration

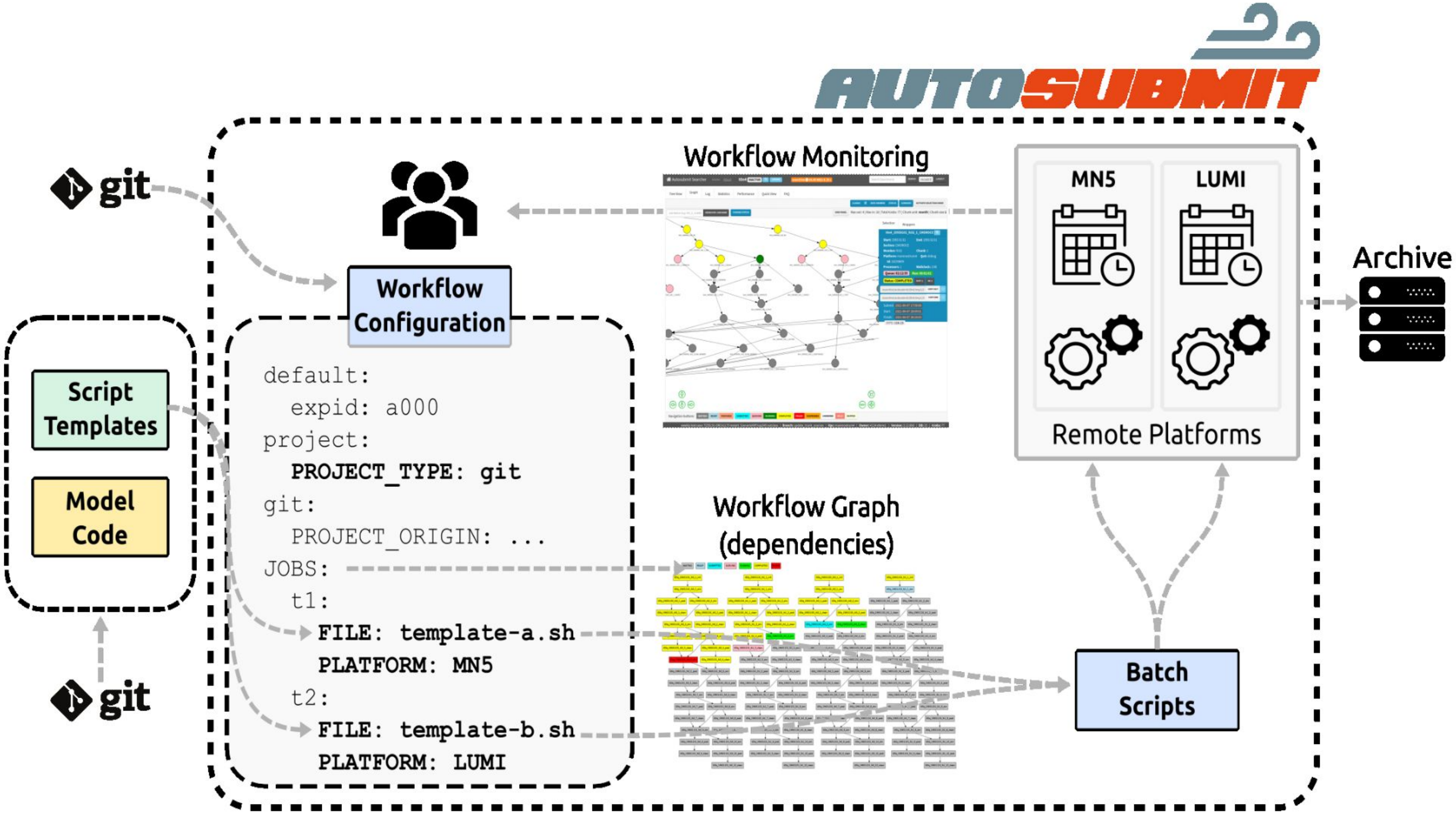
DEFINITION
PARAMETERIZATION

Workflow engine

CONTROL
LOGGING

Workflow monitoring and analysis

VISUALIZATION
STATISTICS
PERFORMANCE METRICS



AUTOSUBMIT

git

Workflow Configuration

```
default:
  expid: a000
project:
  PROJECT_TYPE: git
git:
  PROJECT_ORIGIN: ...
JOBS:
  t1:
   FILE: template-a.sh
   PLATFORM: MN5
  t2:
   FILE: template-b.sh
   PLATFORM: LUMI
```

Script Templates

Model Code

git

Workflow Monitoring

Workflow Graph (dependencies)

MN5

LUMI

Remote Platforms

Batch Scripts

Archive

AUTOSUBMIT

BSC

European Commission

esiwace

is-enes

CMIP Coupled Model Intercomparison Project | WCRP

Destination Earth

EDITO

Declarative

YAML

Validation

Inheritance

Imports

Declarative

- YAML
- Validation
- Inheritance
- Imports

Consistent

- Start Dates
- Members
- Chunks
- Job Splits
- Variables

AUTOSUBMIT

Declarative

YAML

Validation

Inheritance

Imports

Consistent

Start Dates

Members

Chunks

Job Splits

Variables

Composable

Standardised IDs

Create – Copy

Migrate – Monitor

Workflow ⤹ model code

RO-Crate

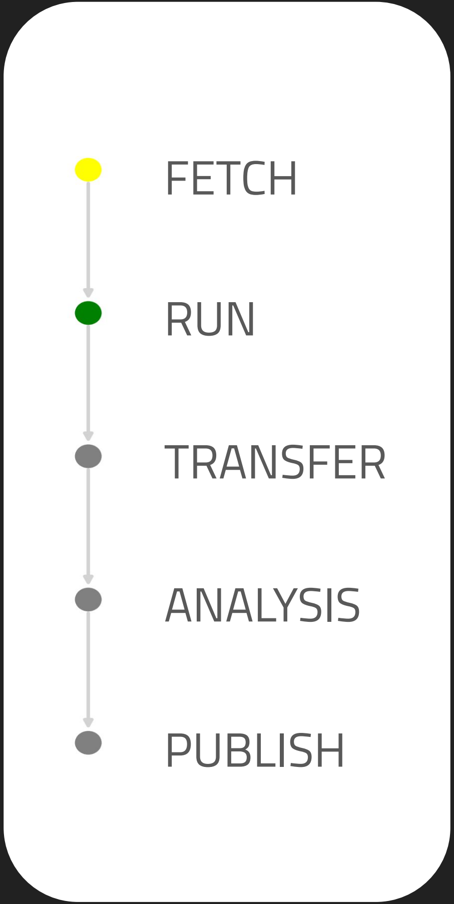
Configuration

Workflow Dependencies

jobs.yml

```
JOBS:
 FETCH:
  FILE: Fetch.sh
 RUN:
  FILE: Run.sh
  DEPENDENCIES: FETCH
 TRANSFER:
  FILE: Transfer.sh
  DEPENDENCIES: RUN
 ANALYSIS:
  FILE: Chunk.py
  DEPENDENCIES: TRANSFER
 PUBLISH:
  FILE: Publish.sh
  DEPENDENCIES: ANALYSIS
```

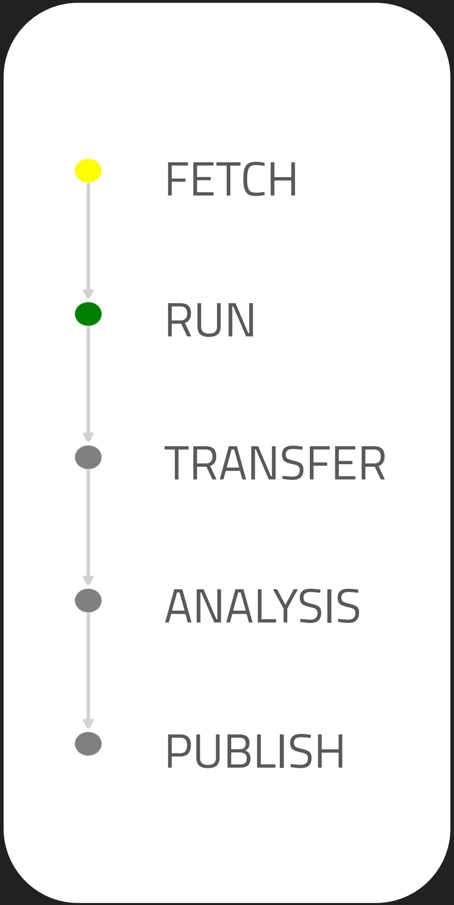
YAML



Workflow Dependencies

jobs.yml

```
JOBS:
 FETCH:
  FILE: Fetch.sh
  PLATFORM: LOCAL
 RUN:
  FILE: Run.sh
  DEPENDENCIES: FETCH
  PLATFORM: REMOTE
 TRANSFER:
  FILE: Transfer.sh
  DEPENDENCIES: RUN
  PLATFORM: REMOTE
 ANALYSIS:
  FILE: Chunk.py
  DEPENDENCIES: TRANSFER
  PLATFORM: LOCAL
 PUBLISH:
  FILE: Publish.sh
  DEPENDENCIES: ANALYSIS
  PLATFORM: LOCAL
```



FETCH

RUN

TRANSFER

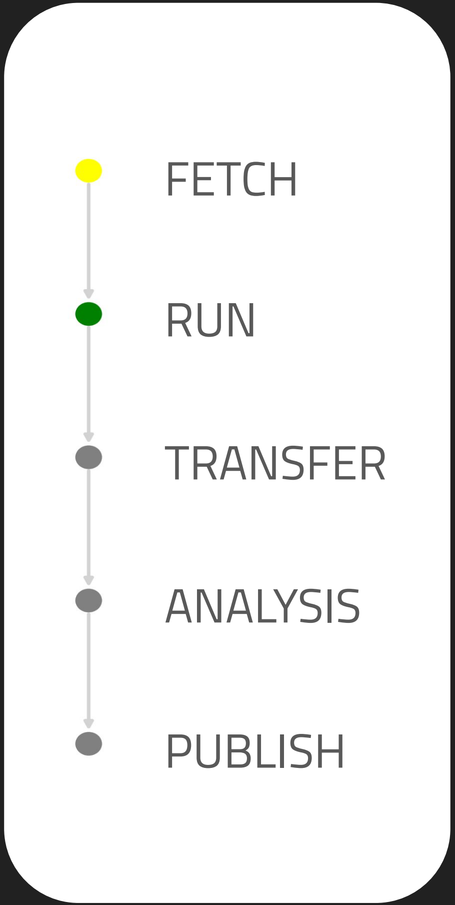
ANALYSIS

PUBLISH

Workflow Platforms

~/platforms.yml

```
PLATFORMS:
  REMOTE:
    USER: my-remote-user
```



FETCH

RUN

TRANSFER

ANALYSIS

PUBLISH

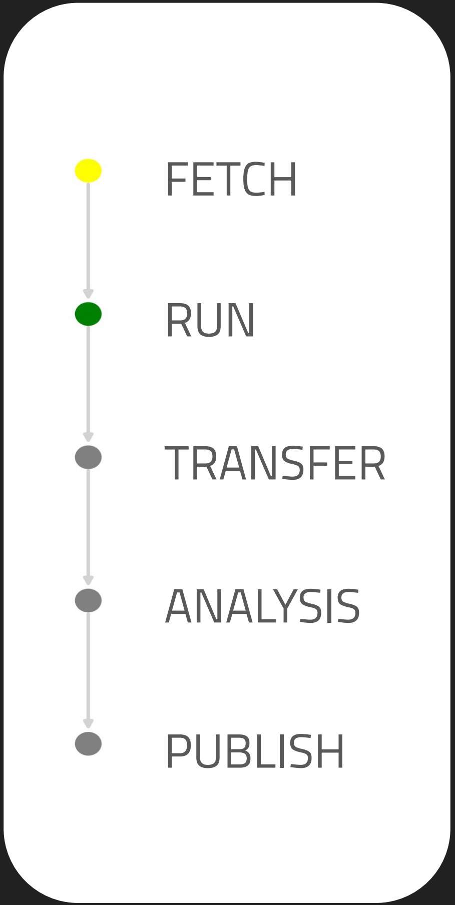
Workflow Platforms

~/platforms.yml

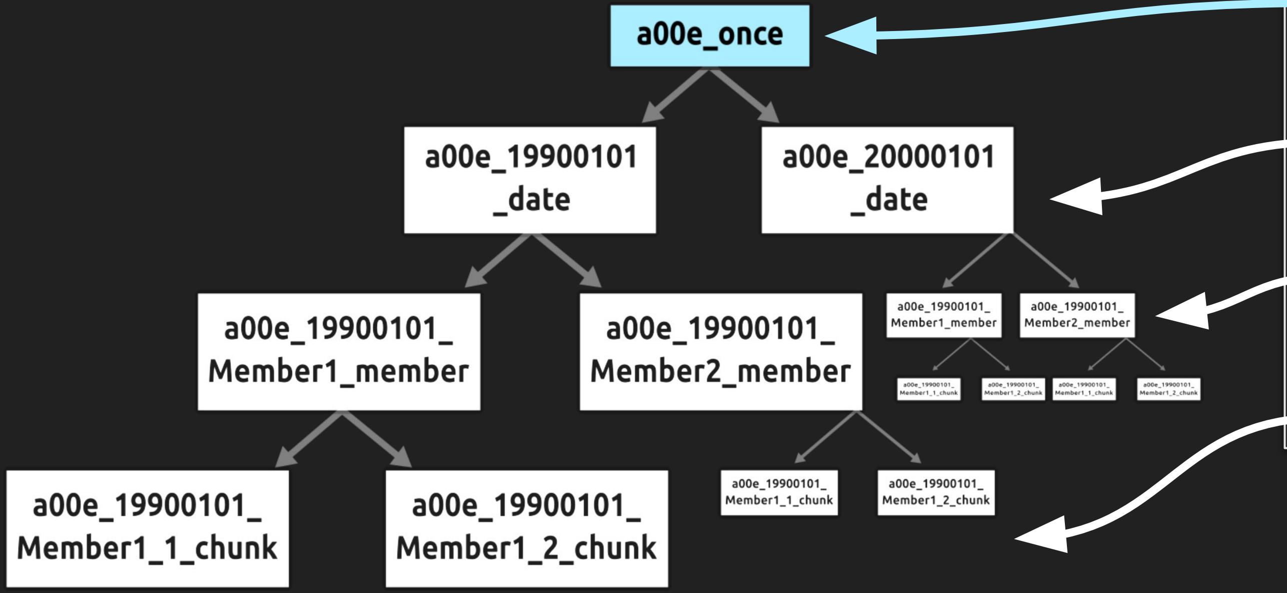
```
PLATFORMS:
  REMOTE:
    USER: my-remote-user
```

~/.ssh/config

```
PLATFORMS:
Host mn5login
        HostName glogin1.bsc.es
        User my-remote-user
        IdentityFile ~/.ssh/id_rsa
        ForwardX11 yes
```

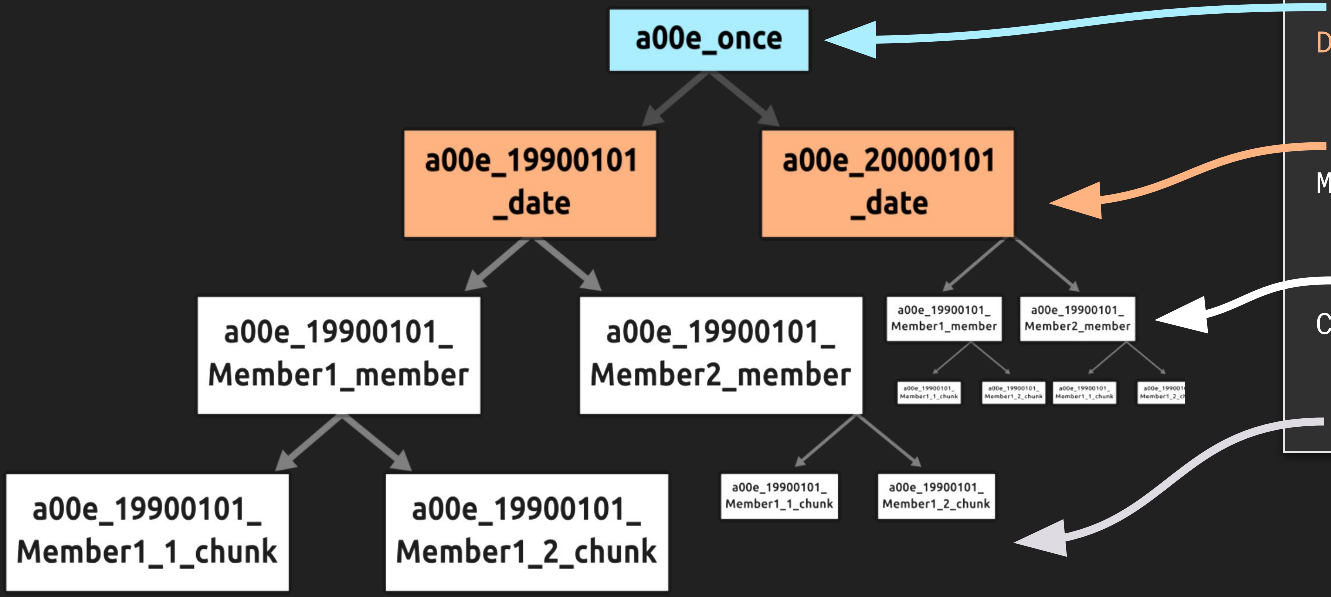


Workflow Hierarchy


a00e_once
a00e_19900101 _date
a00e_20000101 _date
a00e_19900101_ Member1_member
a00e_19900101_ Member2_member
a00e_19900101_ Member1_member
a00e_19900101_ Member2_member
a00e_19900101_ Member1_1_chunk
a00e_19900101_ Member1_2_chunk
a00e_19900101_ Member1_1_chunk
a00e_19900101_ Member1_2_chunk
a00e_19900101_ Member1_1_chunk
a00e_19900101_ Member1_2_chunk
a00e_19900101_ Member1_1_chunk
a00e_19900101_ Member1_2_chunk

```
JOBS:
 ONCE:
  FILE: Once.sh
  RUNNING: once
 DATE:
  FILE: date.sh
  DEPENDENCIES: once
  RUNNING: date
 MEMBER:
  FILE: Member.sh
  DEPENDENCIES: date
  RUNNING: member
 CHUNK:
  FILE: Chunk.py
  DEPENDENCIES: member
  RUNNING: chunk
```

Workflow Hierarchy



a00e_once

a00e_19900101 _date

a00e_20000101 _date

a00e_19900101_ Member1_member

a00e_19900101_ Member2_member

a00e_19900101_ Member1_member

a00e_19900101_ Member2_member

a00e_19900101_ Member1_1_chunk

a00e_19900101_ Member1_2_chunk

a00e_19900101_ Member1_1_chunk

a00e_199001 Member1_2_c

a00e_19900101_ Member1_1_chunk

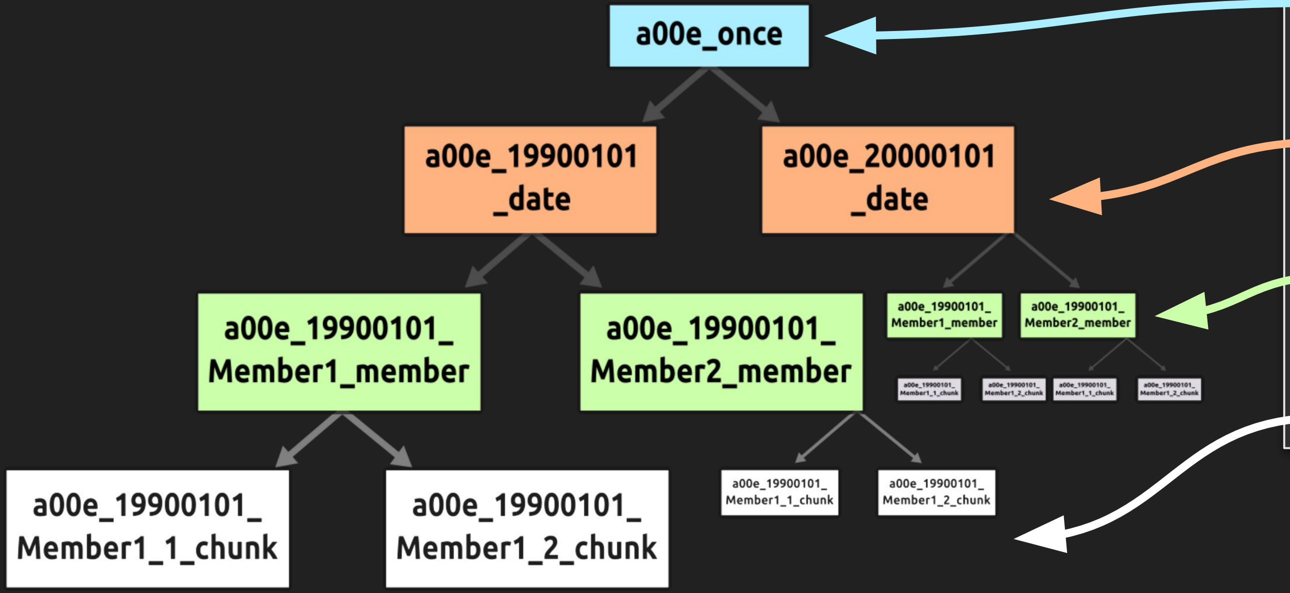
a00e_19900101_ Member1_2_chunk

a00e_19900101_ Member1_1_chunk

a00e_19900101_ Member1_2_chunk

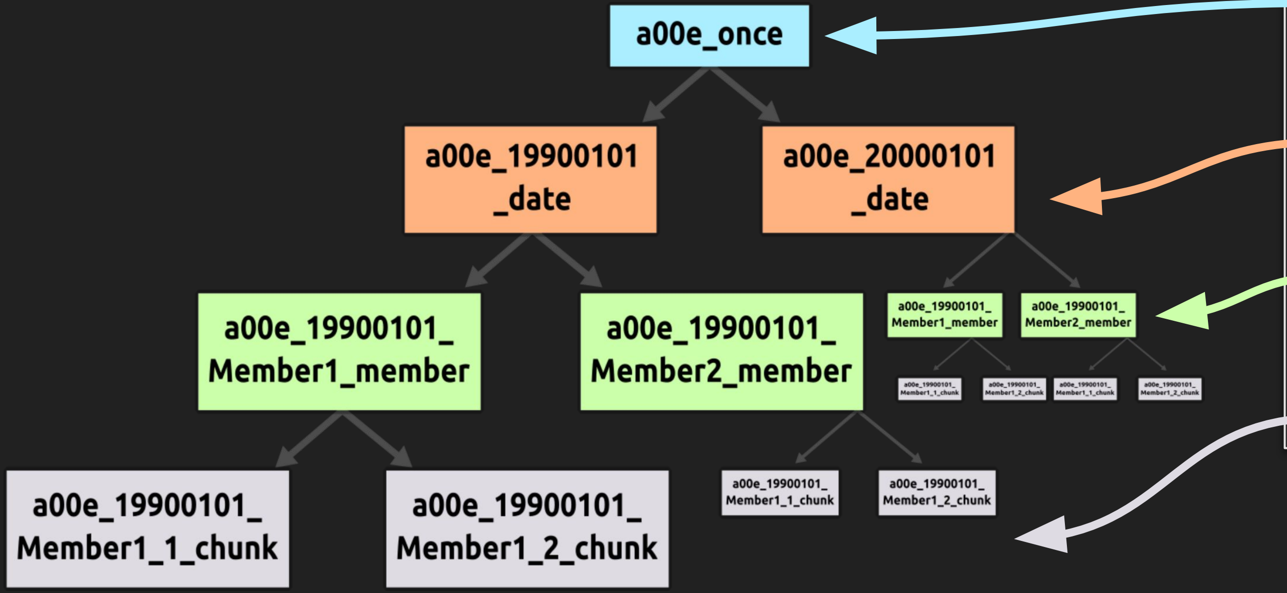
```
JOBS:
 ONCE:
  FILE: Once.sh
  RUNNING: once
 DATE:
  FILE: date.sh
  DEPENDENCIES: once
  RUNNING: date
 MEMBER:
  FILE: Member.sh
  DEPENDENCIES: date
  RUNNING: member
 CHUNK:
  FILE: Chunk.py
  DEPENDENCIES: member
  RUNNING: chunk
```

Workflow Hierarchy


a00e_once
a00e_19900101 _date
a00e_20000101 _date
a00e_19900101_ Member1_member
a00e_19900101_ Member2_member
a00e_19900101_ Member1_member
a00e_19900101_ Member2_member
a00e_19900101_ Member1_1_chunk
a00e_19900101_ Member1_2_chunk
a00e_19900101_ Member1_1_chunk
a00e_19900101_ Member1_2_chunk
a00e_19900101_ Member1_1_chunk
a00e_19900101_ Member1_2_chunk
a00e_19900101_ Member1_1_chunk
a00e_19900101_ Member1_2_chunk

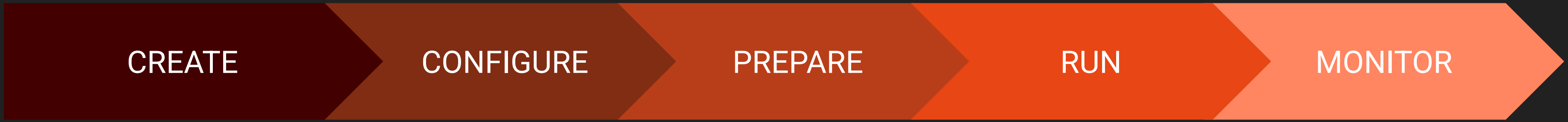
```
JOBS:
 ONCE:
  FILE: Once.sh
  RUNNING: once
 DATE:
  FILE: date.sh
  DEPENDENCIES: once
  RUNNING: date
 MEMBER:
  FILE: Member.sh
  DEPENDENCIES: date
  RUNNING: member
 CHUNK:
  FILE: Chunk.py
  DEPENDENCIES: member
  RUNNING: chunk
```

Workflow Hierarchy


a00e_once
a00e_19900101_date
a00e_20000101_date
a00e_19900101_Member1_member
a00e_19900101_Member2_member
a00e_19900101_Member1_member
a00e_19900101_Member2_member
a00e_19900101_Member1_1_chunk
a00e_19900101_Member1_2_chunk
a00e_19900101_Member1_1_chunk
a00e_19900101_Member1_2_chunk
a00e_19900101_Member1_1_chunk
a00e_19900101_Member1_2_chunk
a00e_19900101_Member1_1_chunk
a00e_19900101_Member1_2_chunk

```
JOBS:
 ONCE:
  FILE: Once.sh
  RUNNING: once
 DATE:
  FILE: date.sh
  DEPENDENCIES: once
  RUNNING: date
 MEMBER:
  FILE: Member.sh
  DEPENDENCIES: date
  RUNNING: member
 CHUNK:
  FILE: Chunk.py
  DEPENDENCIES: member
  RUNNING: chunk
```

Running



CREATE

CONFIGURE

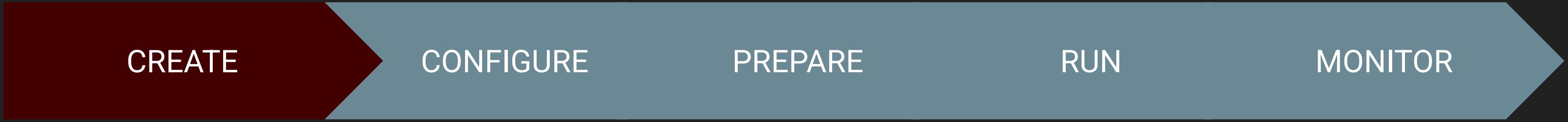
PREPARE

RUN

MONITOR

CREATE NEW EXPERIMENT

```
autosubmit expid -H "local" -d "My new experiment"
```



CREATE

CONFIGURE

PREPARE

RUN

MONITOR

CREATE NEW EXPERIMENT

```
autosubmit expid -H "local" -d "My new experiment"
```

COPY EXISTING EXPERIMENT

```
autosubmit expid -H "local" -d "My new copy" --copy a00x
```

CREATE CONFIGURE PREPARE RUN MONITOR

CREATE NEW EXPERIMENT

```
autosubmit expid -H "local" -d "My new experiment"
```

COPY EXISTING EXPERIMENT

```
autosubmit expid -H "local" -d "My new copy" --copy a00x
```

OUTPUT:

```
Experiment a01a created
```

CREATE
CONFIGURE
PREPARE
RUN
MONITOR

```
AUTOSUBMIT_DATA/a01a/
                    ├ conf/   Experiment configuration
                    │
                    ├ proj/   Project (workflow scripts, config., ...)
                    │
                    ├ plot/   Visualizations
                    │
                    ├ tmp/    Logs, templates
                    │
                    └ pkl/    Workflow database
```

CREATE

CONFIGURE

PREPARE

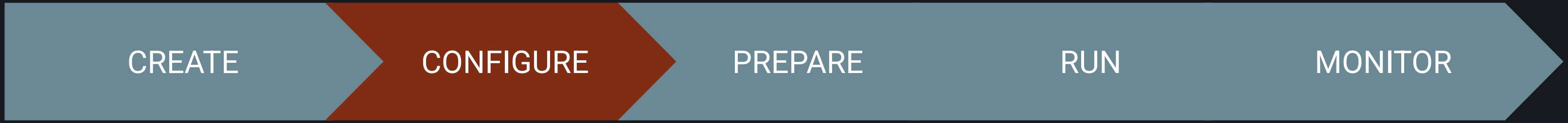
RUN

MONITOR

```
AUTOSUBMIT_DATA/a01a/
```

In Autosubmit, the project is a code base external to the experiment configuration that contains the workflow logic as external resources that must be copied.

```
├ conf/    Experiment configuration
│
├ proj/    Project (workflow scripts, config., ...)
│
├ plot/    Visualizations
│
├ tmp/     Logs, templates
│
└ pkl/     Workflow database
```



```
AUTOSUBMIT_DATA/a01a/
├ conf/    Experiment configuration
│
├ proj/    Project (workflow scripts, config., ...)
│
├ plot/    Visualizations
│
├ tmp/     Logs, templates
│
└ pkl/     Workflow database
```

In Autosubmit, the **project is a code base external to the experiment configuration** that contains the workflow logic as external resources that must be copied.

configuration files
namelists
YAML files
bash scripts
Python code
...

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

AUTOSUBMIT_DATA/a01a/conf/minimal.yml

```
CONFIG:
 AUTOSUBMIT_VERSION: “4.x.x”
DEFAULT:
 HPCARCH: “local”                        # -H “local”
PROJECT:
 PROJECT_TYPE: “local”                    # or git
 PROJECT_DESTINATION: “”
```

CREATE

CONFIGURE

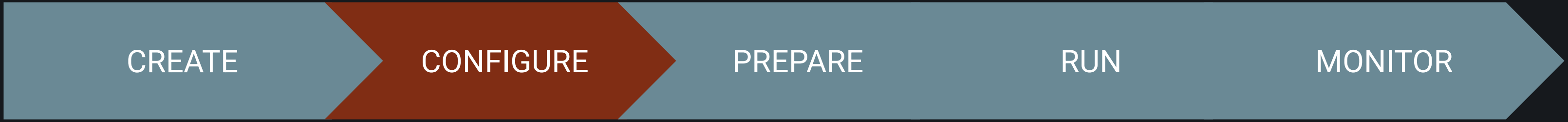
PREPARE

RUN

MONITOR

AUTOSUBMIT_DATA/a01a/conf/minimal.yml

```
CONFIG:
 AUTOSUBMIT_VERSION: “4.x.x”
DEFAULT:
 HPCARCH: “local”                        # -H “local”
PROJECT:
 PROJECT_TYPE: “git”
 PROJECT_DESTINATION: “”
GIT:                                     # Because PROJECT_TYPE is “git”
 PROJECT_ORIGIN: “https://gitlab.bsc.es/...”
 PROJECT_BRANCH: “v1.2.3”
 PROJECT_COMMIT: “123456…”
```



CREATE

CONFIGURE

PREPARE

RUN

MONITOR

/appl/AS/AUTOSUBMIT_DATA/a01a/conf/minimal.yml

```
CONFIG:
 AUTOSUBMIT_VERSION: “4.x.x”
DEFAULT:
 EXPID: $expid                           # From `autosubmit expid`
 HPCARCH: “local”                        # -H “local”
 CUSTOM_CONFIG: “%PROJDIR%/”             # Other files?
PROJECT:
 PROJECT_TYPE: “git”
 PROJECT_DESTINATION: “”
GIT:                                     # Because PROJECT_TYPE is “git”
 PROJECT_ORIGIN: “https://gitlab.bsc.es/...”
 PROJECT_BRANCH: “v1.2.3”
 PROJECT_COMMIT: “123456…”
```

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

/appl/AS/AUTOSUBMIT_DATA/a01a/conf/minimal.yml

Autosubmit will try to load other files located at the **CUSTOM_CONFIG** value.

```
CONFIG:
 AUTOSUBMIT_VERSION: “4.x.x”
DEFAULT:
 EXPID: $expid                               # From `autosubmit expid`
 HPCARCH: “local”                            # -H “local”
 CUSTOM_CONFIG: “%PROJDIR%/”                 # Other files?
PROJECT:
 PROJECT_TYPE: “git”                         # or local, svn
 PROJECT_DESTINATION: “”
GIT:                                         # Because PROJECT_TYPE is “git”
 PROJECT_ORIGIN: “https://gitlab.bsc.es/...”
 PROJECT_BRANCH: “v1.2.3”
 PROJECT_COMMIT: “123456…”
```

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

/appl/AS/AUTOSUBMIT_DATA/a01a/conf/minimal.yml

Autosubmit will try to load other files located at the **CUSTOM_CONFIG** value.

%PROJDIR% is a placeholder variable. Autosubmit replaces this by a value from the workflow configuration.

```
CONFIG:
 AUTOSUBMIT_VERSION: “4.x.x”
DEFAULT:
 EXPID: $expid                           # From `autosubmit expid`
 HPCARCH: “local”                        # -H “local”
 CUSTOM_CONFIG: “%PROJDIR%/”             # Other files?
PROJECT:
 PROJECT_TYPE: “git”                     # or local, svn
 PROJECT_DESTINATION: “”
GIT:                                     # Because PROJECT_TYPE is “git”
 PROJECT_ORIGIN: “https://gitlab.bsc.es/...”
 PROJECT_BRANCH: “v1.2.3”
 PROJECT_COMMIT: “123456…”
```

CREATE
CONFIGURE
PREPARE
RUN
MONITOR

```
autosubmit create a01a
```

The first time, it copies the project content into

```
AUTOSUBMIT_DATA/a01a/proj/
```

Generates the experiment workflow

Generates

```
AUTOSUBMIT_DATA/a01a/plot/a01a_${timestamp}.pdf
```

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

autosubmit run a01a

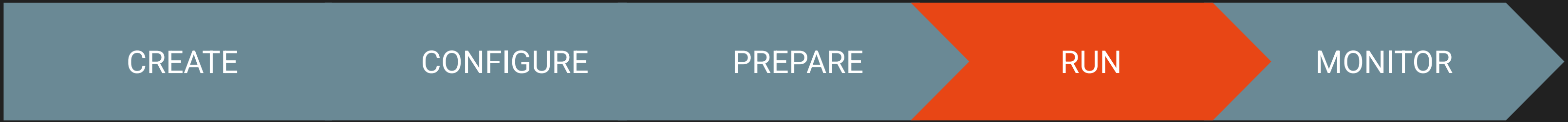
Runs the experiment.

Generates template logs for traceability

```
AUTOSUBMIT_DATA/a01a/tmp/LOG_a01a/
```

Expected output:

```
[local] Connection successful to host localhost
[local] Correct user privileges for host localhost
```



CREATE

CONFIGURE

PREPARE

RUN

MONITOR

```
autosubmit run a01a
```

NO HANG-UP MODE

```
nohup autosubmit monitor a01a &
```

Visualize console output:

```
tail -f nohup.out
```

```
autosubmit monitor a01a
```

CREATE

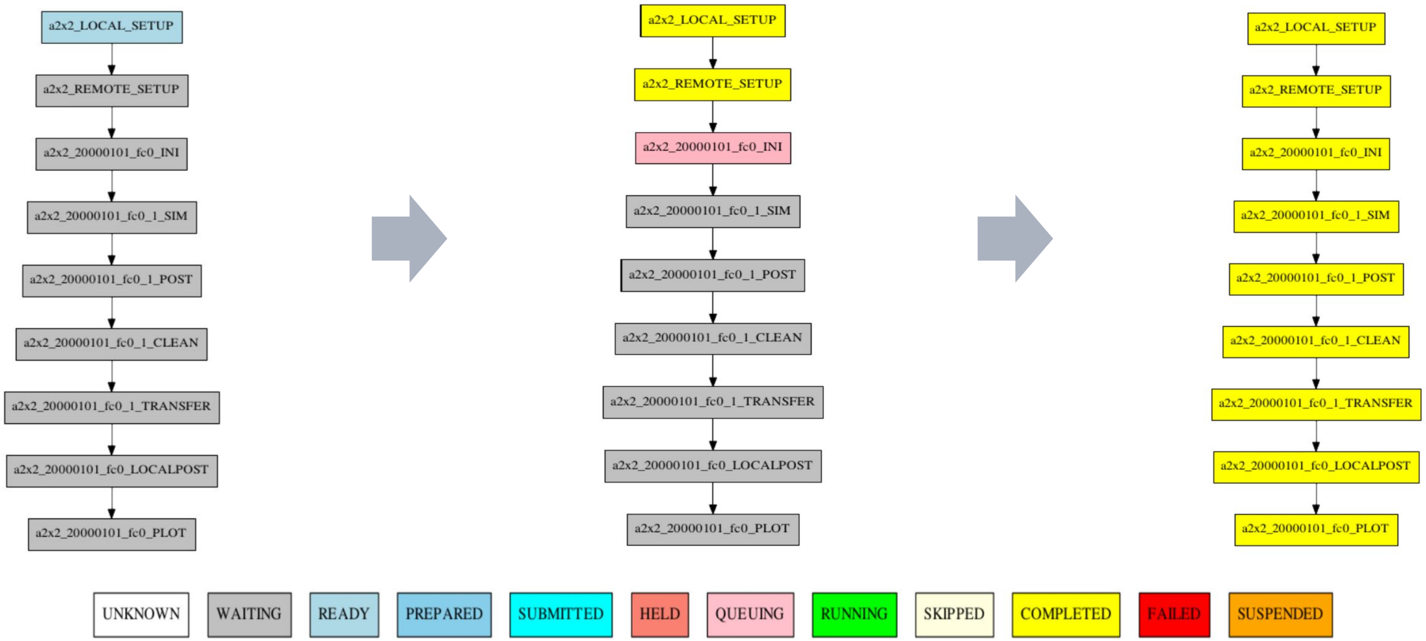
CONFIGURE

PREPARE

RUN

MONITOR

autosubmit monitor a01a



a2x2_LOCAL_SETUP
a2x2_REMOTE_SETUP
a2x2_20000101_fc0_INI
a2x2_20000101_fc0_1_SIM
a2x2_20000101_fc0_1_POST
a2x2_20000101_fc0_1_CLEAN
a2x2_20000101_fc0_1_TRANSFER
a2x2_20000101_fc0_LOCALPOST
a2x2_20000101_fc0_PLOT

a2x2_LOCAL_SETUP
a2x2_REMOTE_SETUP
a2x2_20000101_fc0_INI
a2x2_20000101_fc0_1_SIM
a2x2_20000101_fc0_1_POST
a2x2_20000101_fc0_1_CLEAN
a2x2_20000101_fc0_1_TRANSFER
a2x2_20000101_fc0_LOCALPOST
a2x2_20000101_fc0_PLOT

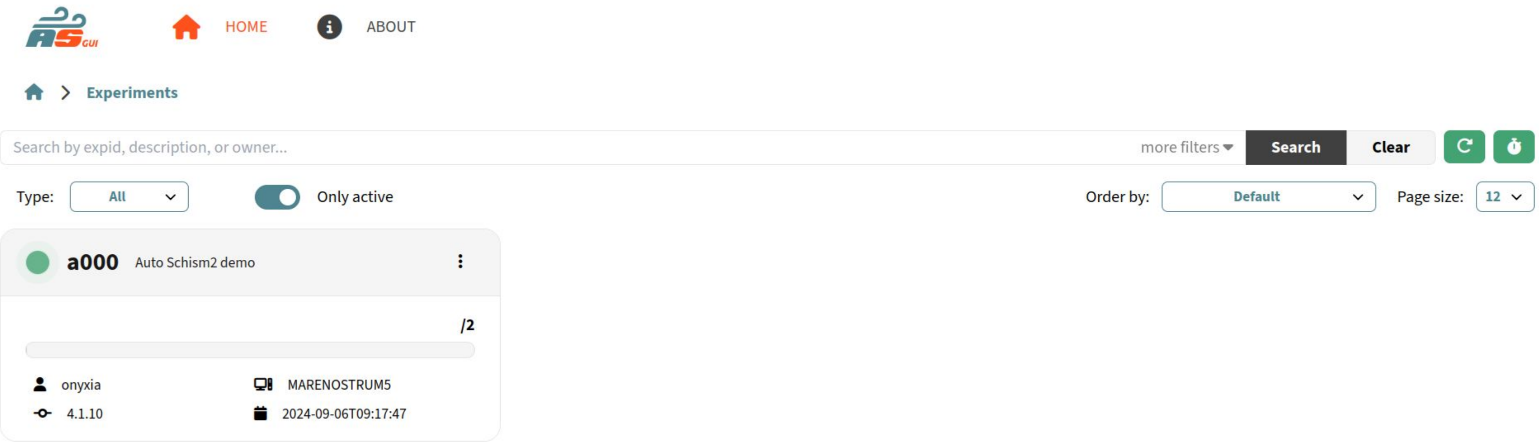
a2x2_LOCAL_SETUP
a2x2_REMOTE_SETUP
a2x2_20000101_fc0_INI
a2x2_20000101_fc0_1_SIM
a2x2_20000101_fc0_1_POST
a2x2_20000101_fc0_1_CLEAN
a2x2_20000101_fc0_1_TRANSFER
a2x2_20000101_fc0_LOCALPOST
a2x2_20000101_fc0_PLOT

UNKNOWN WAITING READY PREPARED SUBMITTED HELD QUEUING RUNNING SKIPPED COMPLETED FAILED SUSPENDED

CREATE CONFIGURE PREPARE RUN MONITOR

Monitoring

Autosubmit GUI



Type: All

Only active

Order by: Default

Page size: 12

a000 Auto Schism2 demo

/2

onyxia

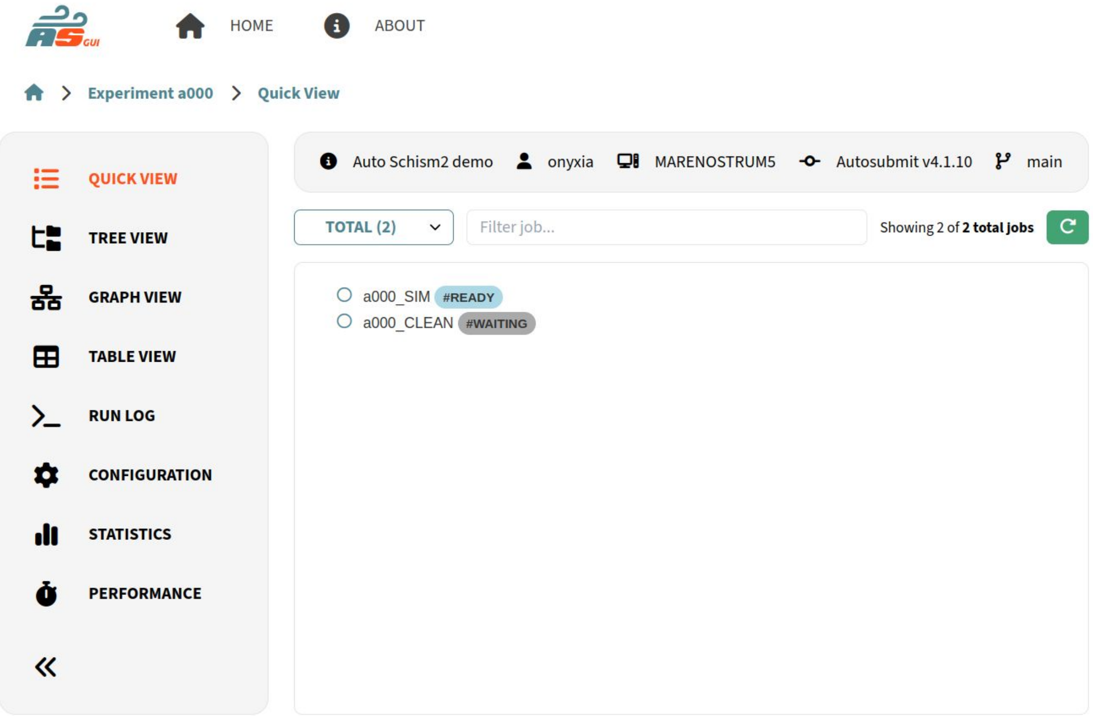
MARENOSTRUM5

4.1.10

2024-09-06T09:17:47

Autosubmit GUI

QUICK VIEW



Auto Schism2 demo · onyxia · MARENOSTRUM5 · Autosubmit v4.1.10 · main

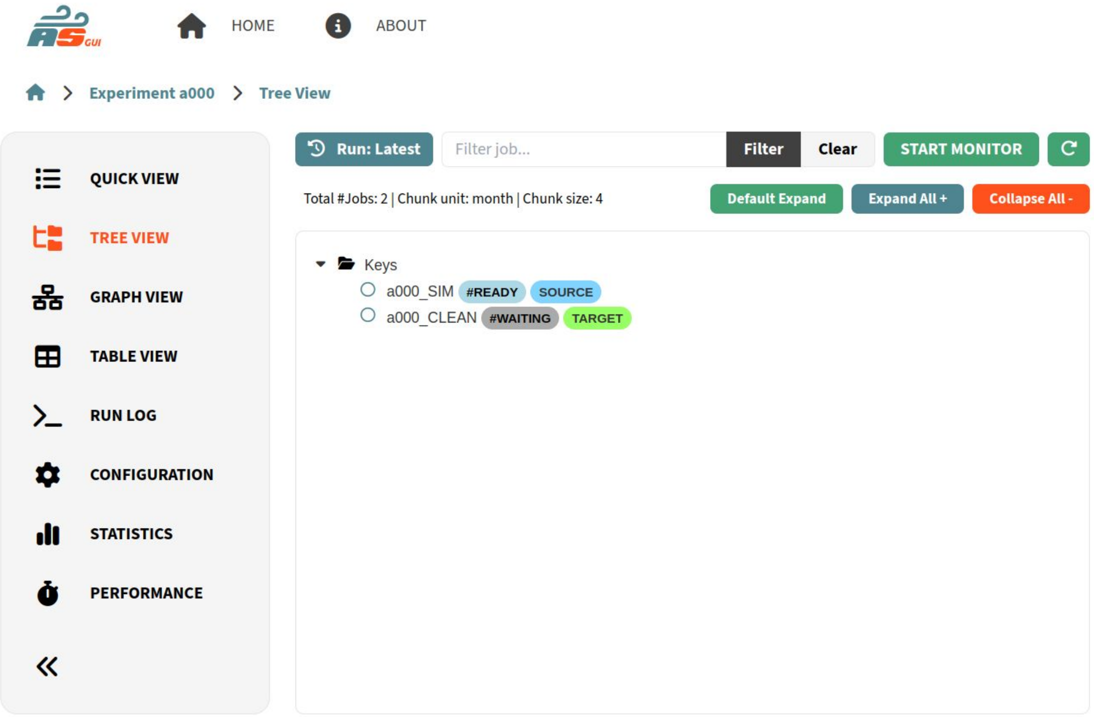
Showing 2 of **2 total jobs**

○ a000_SIM #READY

○ a000_CLEAN #WAITING

Autosubmit GUI

TREE VIEW



Total #Jobs: 2 | Chunk unit: month | Chunk size: 4

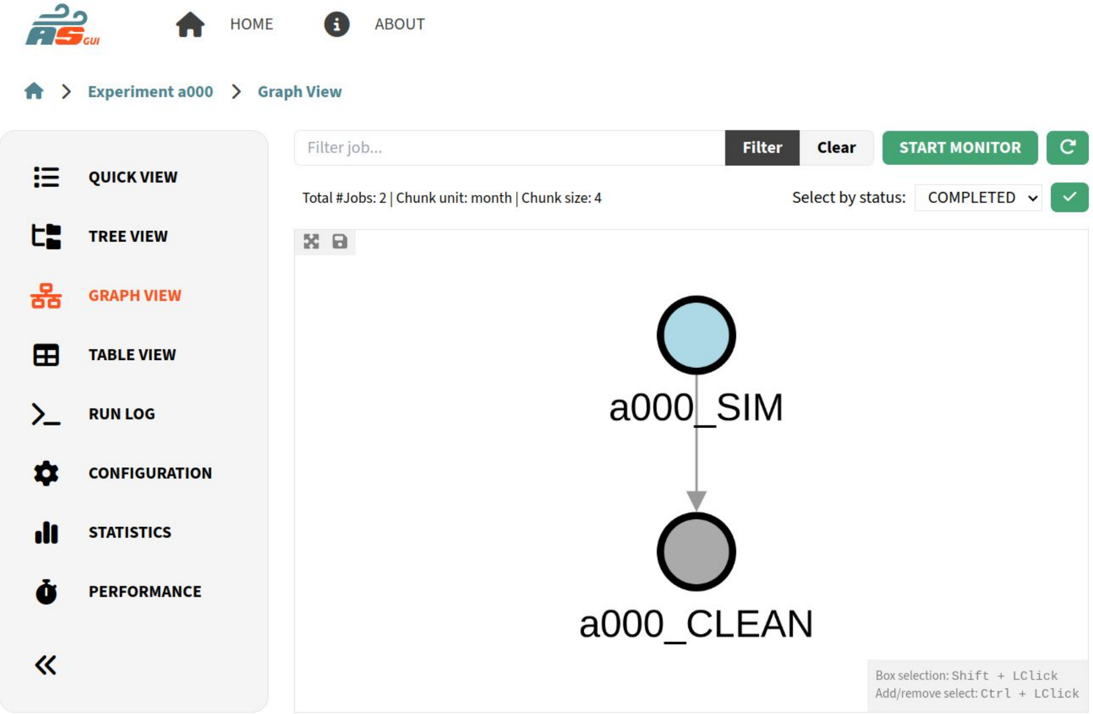
Keys

○ a000_SIM #READY SOURCE

○ a000_CLEAN #WAITING TARGET

Autosubmit GUI

GRAPH VIEW



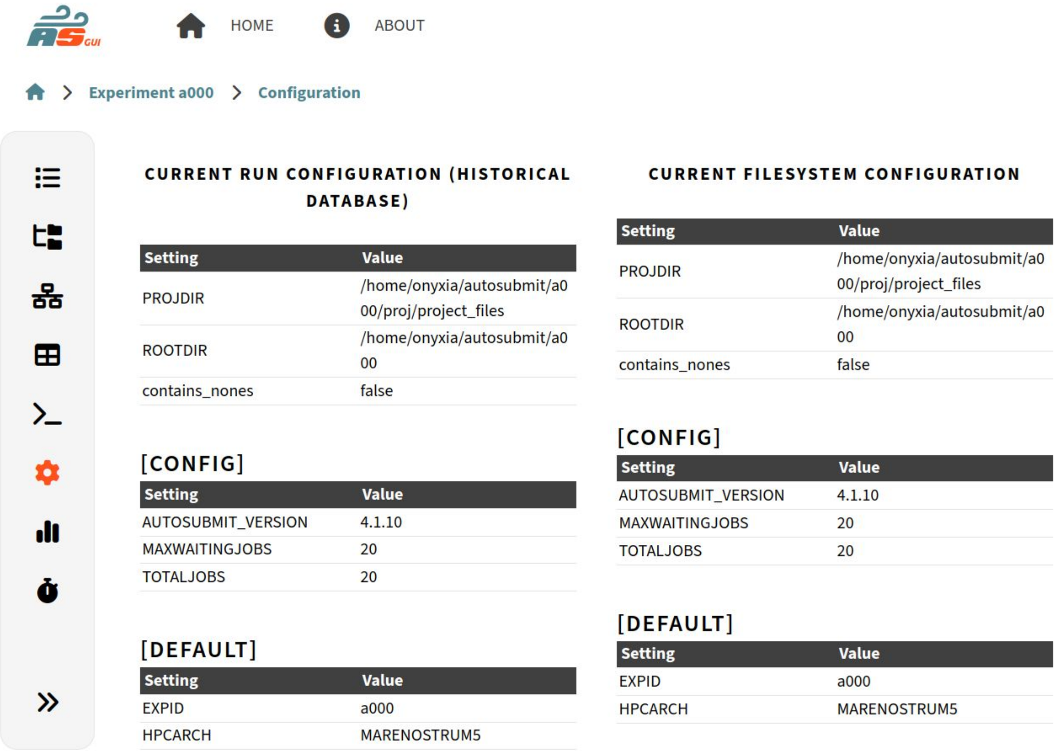
Total #Jobs: 2 | Chunk unit: month | Chunk size: 4

Select by status: COMPLETED

Box selection: Shift + LClick
Add/remove select: Ctrl + LClick

Autosubmit GUI

CONFIGURATION



Experiment a000 > Configuration

CURRENT RUN CONFIGURATION (HISTORICAL DATABASE)

Setting	Value
PROJDIR	/home/onyxia/autosubmit/a000/proj/project_files
ROOTDIR	/home/onyxia/autosubmit/a000
contains_nones	false

[CONFIG]

Setting	Value
AUTOSUBMIT_VERSION	4.1.10
MAXWAITINGJOBS	20
TOTALJOBS	20

[DEFAULT]

Setting	Value
EXPID	a000
HPCARCH	MARENOSTRUM5

CURRENT FILESYSTEM CONFIGURATION

Setting	Value
PROJDIR	/home/onyxia/autosubmit/a000/proj/project_files
ROOTDIR	/home/onyxia/autosubmit/a000
contains_nones	false

[CONFIG]

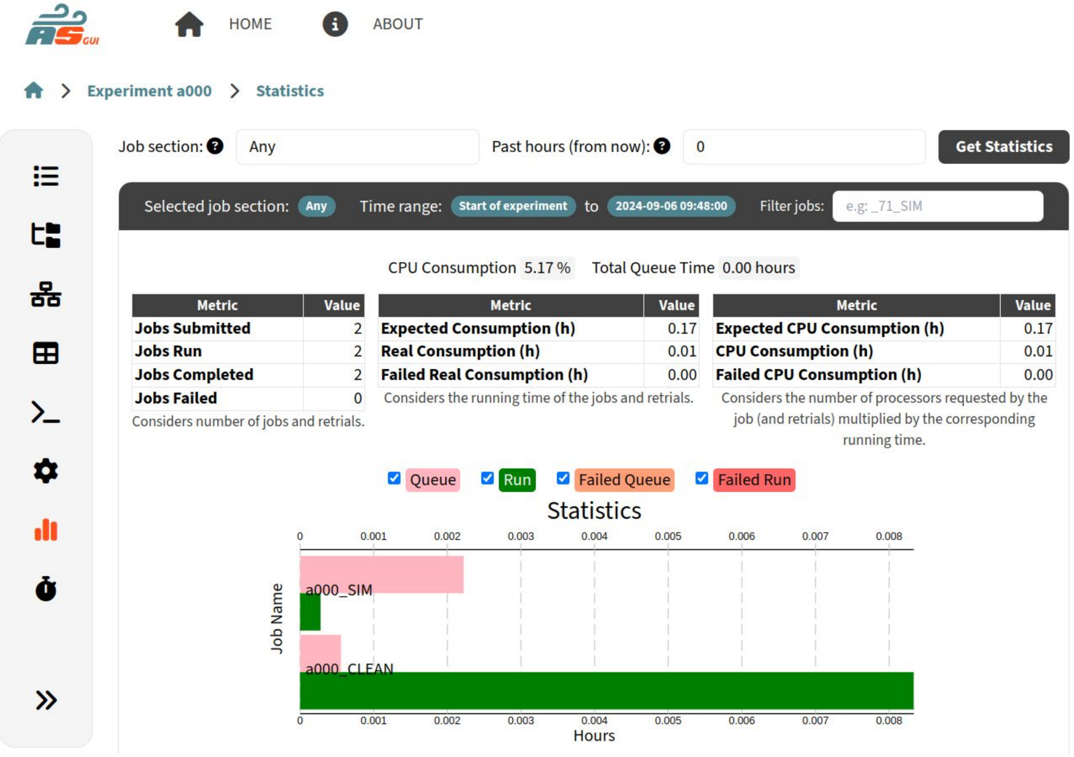
Setting	Value
AUTOSUBMIT_VERSION	4.1.10
MAXWAITINGJOBS	20
TOTALJOBS	20

[DEFAULT]

Setting	Value
EXPID	a000
HPCARCH	MARENOSTRUM5

Autosubmit GUI

STATISTICS



Job section: Any

Past hours (from now): 0

Selected job section: Any Time range: Start of experiment to 2024-09-06 09:48:00 Filter jobs: e.g: _71_SIM

CPU Consumption 5.17 % Total Queue Time 0.00 hours

Metric	Value
Jobs Submitted	2
Jobs Run	2
Jobs Completed	2
Jobs Failed	0

Considers number of jobs and retrials.

Metric	Value
Expected Consumption (h)	0.17
Real Consumption (h)	0.01
Failed Real Consumption (h)	0.00

Considers the running time of the jobs and retrials.

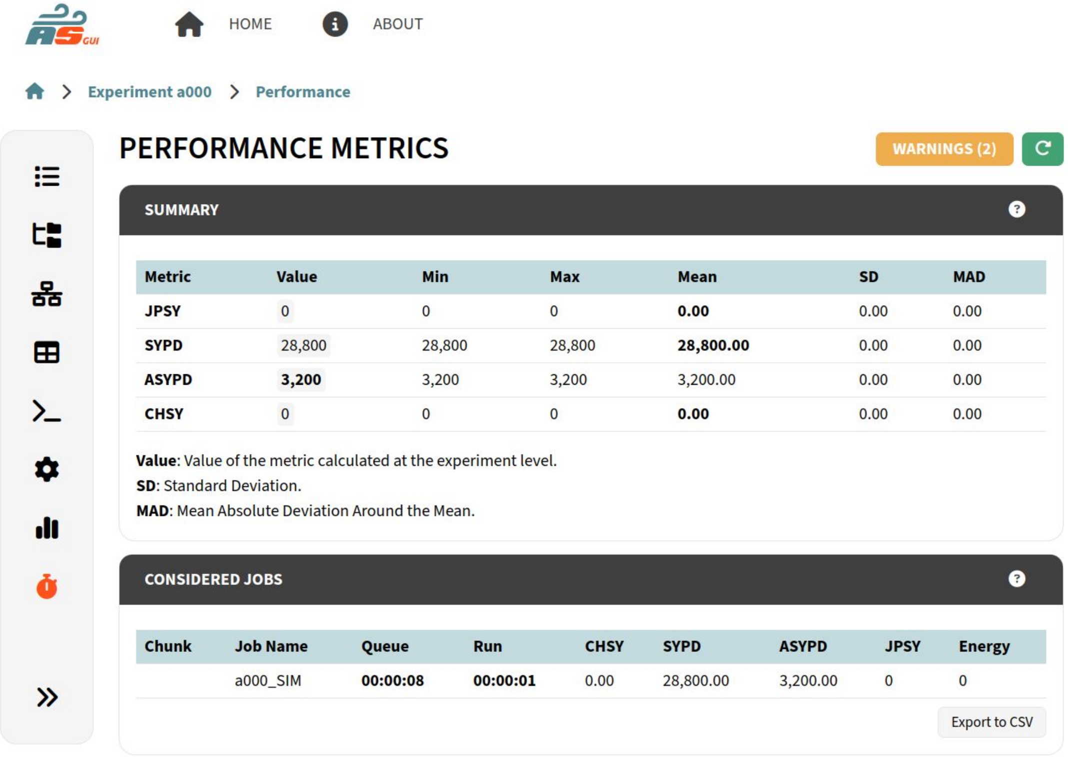
Metric	Value
Expected CPU Consumption (h)	0.17
CPU Consumption (h)	0.01
Failed CPU Consumption (h)	0.00

Considers the number of processors requested by the job (and retrials) multiplied by the corresponding running time.

☑ Queue ☑ Run ☑ Failed Queue ☑ Failed Run

Autosubmit GUI

PERFORMANCE



PERFORMANCE METRICS

SUMMARY

Metric	Value	Min	Max	Mean	SD	MAD
JPSY	0	0	0	0.00	0.00	0.00
SYPD	28,800	28,800	28,800	28,800.00	0.00	0.00
ASYPD	3,200	3,200	3,200	3,200.00	0.00	0.00
CHSY	0	0	0	0.00	0.00	0.00

Value: Value of the metric calculated at the experiment level.
SD: Standard Deviation.
MAD: Mean Absolute Deviation Around the Mean.

CONSIDERED JOBS

Chunk	Job Name	Queue	Run	CHSY	SYPD	ASYPD	JPSY	Energy
	a000_SIM	00:00:08	00:00:01	0.00	28,800.00	3,200.00	0	0

Export to CSV

Autosubmit API

A *comprehensive waypoint* to the Autosubmit persistence layer

Autosubmit API

```
GET /v3/expinfo/a75t
```

Autosubmit API

`GET /v3/expinfo/a75t`

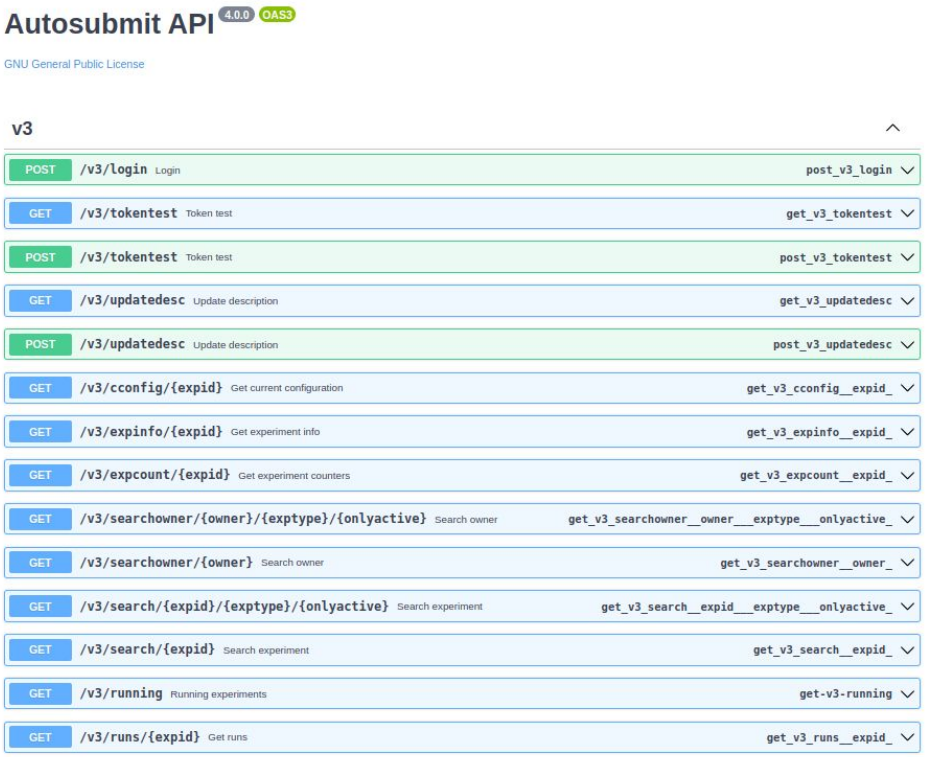
```
{
   "branch": "FORCeS-RaFSIPv2-DURF-MELUXINA",
   "chunk_size": 12,
   "chunk_unit": "month",
   "completed_jobs": 801,
   "db_historic_version": 18,
   "description": "[DURF] EC-Earth3-FORCeS AMIP-IFS-TM5 1850-2000 histSST BASELINE",
   "error": false,
   "error_message": "",
   "expid": "a75t",
   "hpc": "marenostrum5",
   "model": "https://earth.bsc.es/gitlab/es/auto-ecearth3.git",
   "owner": "mariag",
   "owner_id": 7464,
   "path": "/esarchive/autosubmit/a75t",
   "pkl_timestamp": 1718884227,
   "running": true,
   "time_last_access": "2024-06-20 02:04:43",
   "time_last_mod": "2024-05-09 13:40:29",
   "total_jobs": 1056,
   "updateTime": 10,
   "version": "3.15.14"
}
```

Autosubmit API

RESTful Interface

Open API Specification

Swagger



Autosubmit API 4.0.0 OAS3

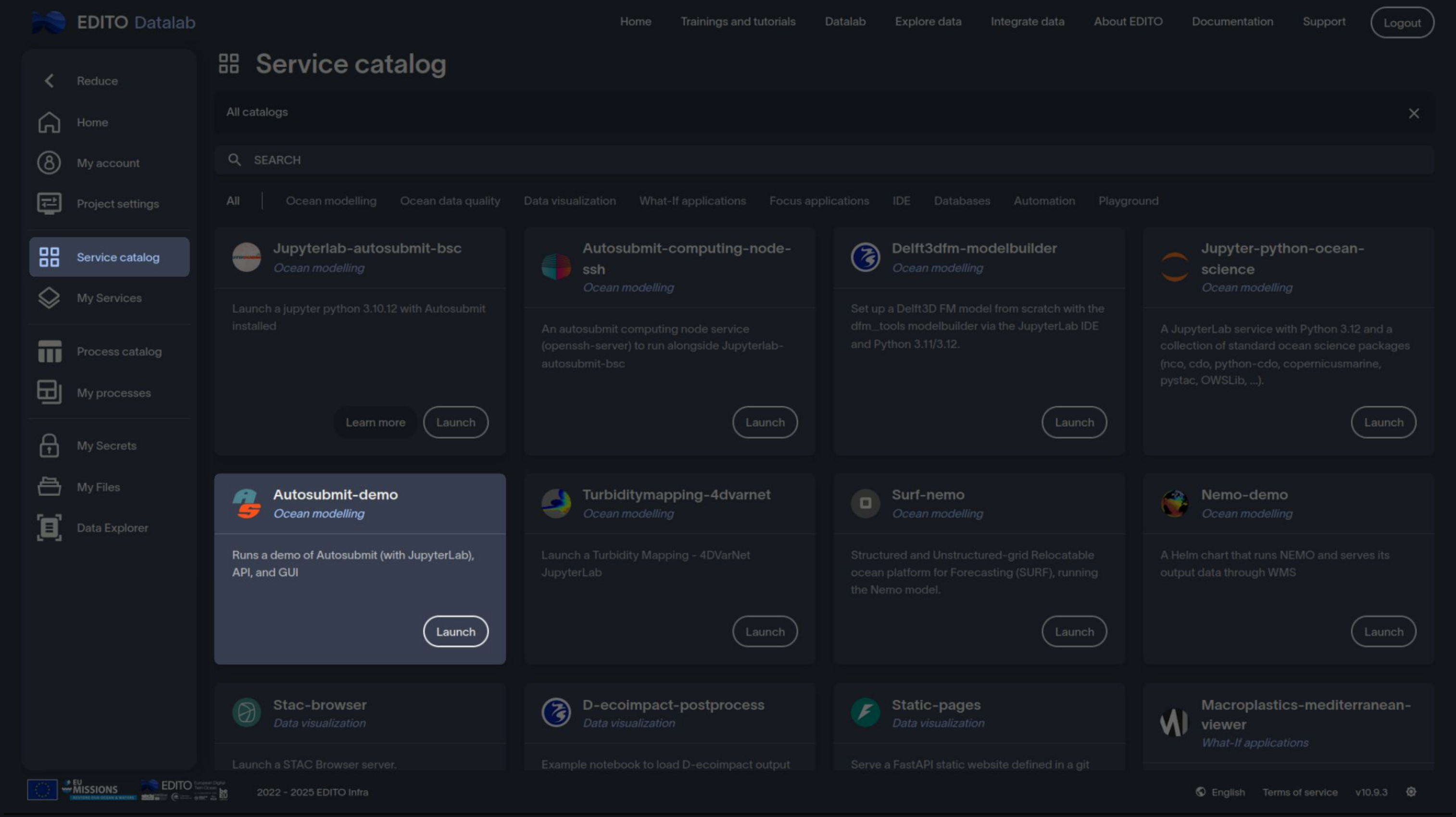
GNU General Public License

v3

POST	/v3/login	Login	post_v3_login
GET	/v3/tokentest	Token test	get_v3_tokentest
POST	/v3/tokentest	Token test	post_v3_tokentest
GET	/v3/updatedesc	Update description	get_v3_updatedesc
POST	/v3/updatedesc	Update description	post_v3_updatedesc
GET	/v3/cconfig/{expid}	Get current configuration	get_v3_cconfig__expid_
GET	/v3/expinfo/{expid}	Get experiment info	get_v3_expinfo__expid_
GET	/v3/expcount/{expid}	Get experiment counters	get_v3_expcount__expid_
GET	/v3/searchowner/{owner}/{exptype}/{onlyactive}	Search owner	get_v3_searchowner__owner___exptype___onlyactive_
GET	/v3/searchowner/{owner}	Search owner	get_v3_searchowner__owner_
GET	/v3/search/{expid}/{exptype}/{onlyactive}	Search experiment	get_v3_search__expid___exptype___onlyactive_
GET	/v3/search/{expid}	Search experiment	get_v3_search__expid_
GET	/v3/running	Running experiments	get-v3-running
GET	/v3/runs/{expid}	Get runs	get_v3_runs__expid_

Hands On

Step 1. Find the Autosubmit-demo service card in the Service Catalog



EDITO Datalab

Home | Trainings and tutorials | Datalab | Explore data | Integrate data | About EDITO | Documentation | Support | Logout

Reduce
Home
My account
Project settings
Service catalog
My Services
Process catalog
My processes
My Secrets
My Files
Data Explorer

Service catalog

All catalogs

SEARCH

All | Ocean modelling | Ocean data quality | Data visualization | What-If applications | Focus applications | IDE | Databases | Automation | Playground

Jupyterlab-autosubmit-bsc
Ocean modelling
Launch a jupyter python 3.10.12 with Autosubmit installed
Learn more | Launch

Autosubmit-computing-node-ssh
Ocean modelling
An autosubmit computing node service (openssh-server) to run alongside Jupyterlab-autosubmit-bsc
Launch

Delft3dfm-modelbuilder
Ocean modelling
Set up a Delft3D FM model from scratch with the dfm_tools modelbuilder via the JupyterLab IDE and Python 3.11/3.12.
Launch

Jupyter-python-ocean-science
Ocean modelling
A JupyterLab service with Python 3.12 and a collection of standard ocean science packages (nco, cdo, python-cdo, copernicusmarine, pystac, OWSLib, ...).
Launch

Autosubmit-demo
Ocean modelling
Runs a demo of Autosubmit (with JupyterLab), API, and GUI
Launch

Turbiditymapping-4dvarnet
Ocean modelling
Launch a Turbidity Mapping - 4DVarNet JupyterLab
Launch

Surf-nemo
Ocean modelling
Structured and Unstructured-grid Relocatable ocean platform for Forecasting (SURF), running the Nemo model.
Launch

Nemo-demo
Ocean modelling
A Helm chart that runs NEMO and serves its output data through WMS
Launch

Stac-browser
Data visualization
Launch a STAC Browser server.

D-ecoimpact-postprocess
Data visualization
Example notebook to load D-ecoimpact output

Static-pages
Data visualization
Serve a FastAPI static website defined in a git

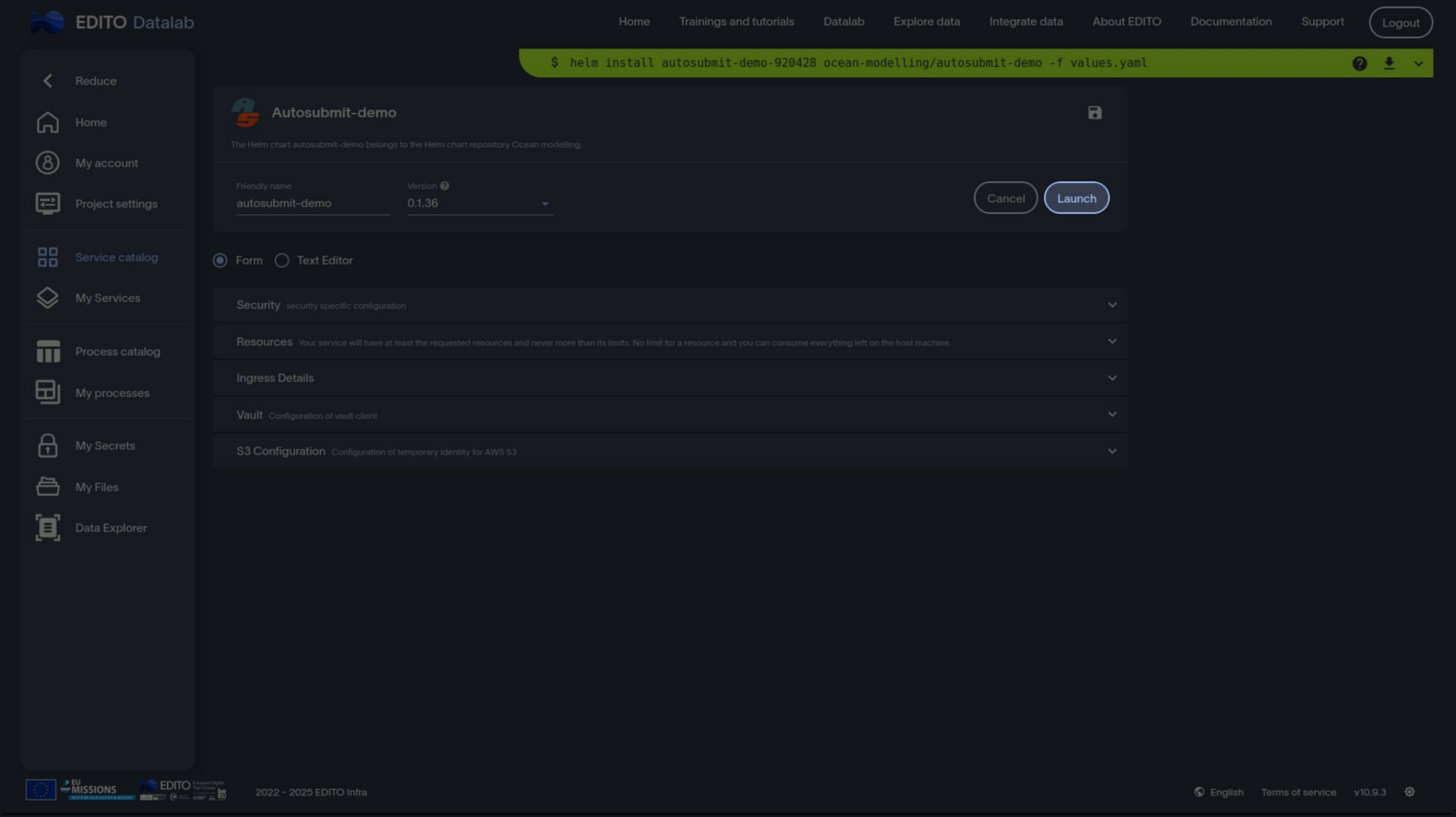
Macroplastics-mediterranean-viewer
What-If applications

EU MISSIONS | EDITO

2022 - 2025 EDITO Infra

English | Terms of service | v10.9.3

Step 2. Review your configuration (if desired) and click Launch.



```
$ helm install autosubmit-demo-920428 ocean-modelling/autosubmit-demo -f values.yaml
```

Autosubmit-demo

The Helm chart autosubmit-demo belongs to the Helm chart repository Ocean modelling.

Friendly name: autosubmit-demo

Version: 0.1.36

● Form ○ Text Editor

Security security specific configuration

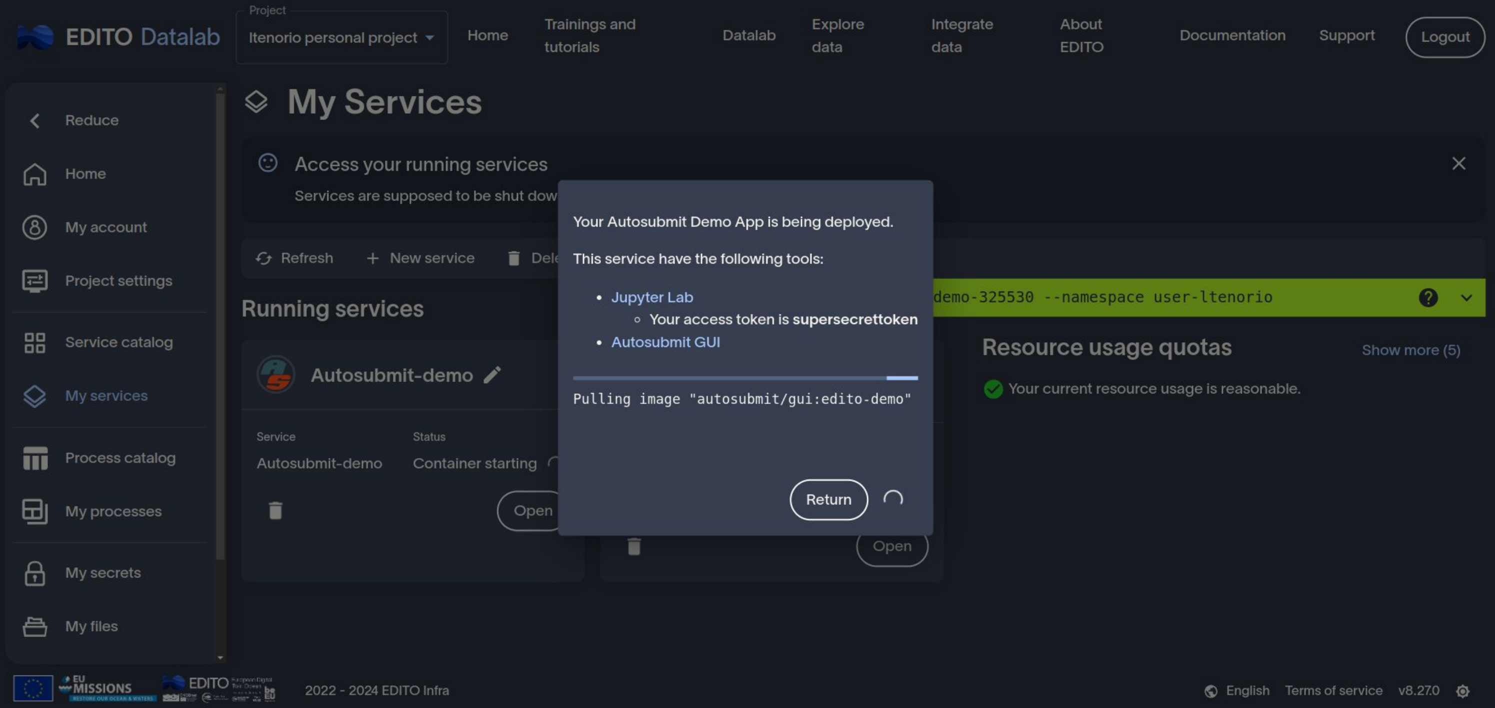
Resources Your service will have at least the requested resources and never more than its limits. No limit for a resource and you can consume everything left on the host machine.

Ingress Details

Vault Configuration of vault client

S3 Configuration Configuration of temporary identity for AWS S3

Step 3. Wait until your new Autosubmit personal service is deployed



My Services

Access your running services

Services are supposed to be shut dow

Running services

Autosubmit-demo

Service: Autosubmit-demo

Status: Container starting

Your Autosubmit Demo App is being deployed.

This service have the following tools:

- Jupyter Lab
    - Your access token is **supersecrettoken**
- Autosubmit GUI

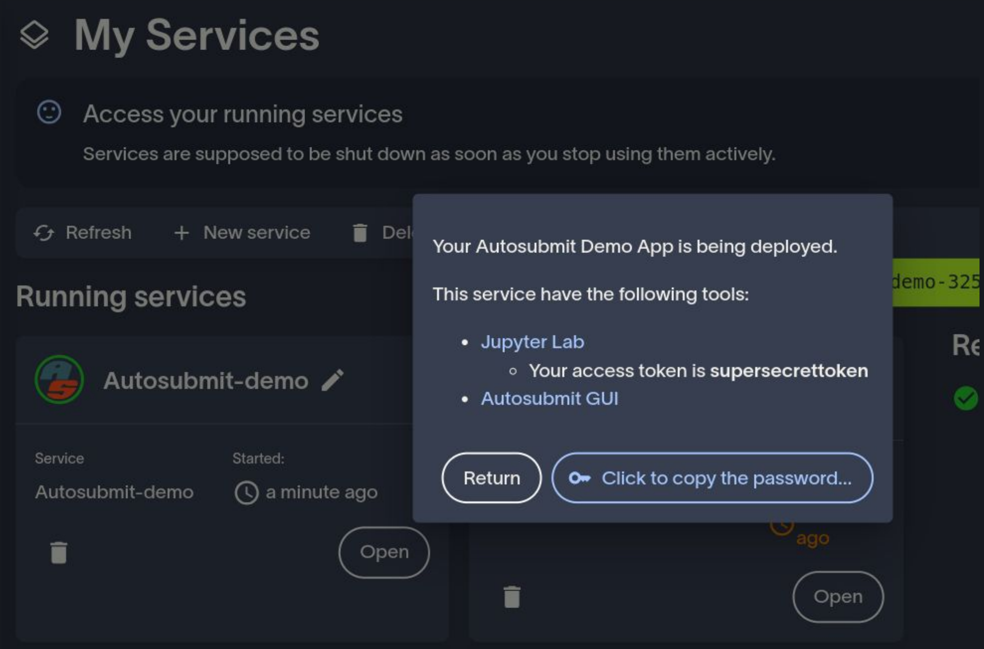
```
Pulling image "autosubmit/gui:edito-demo"
```

```
demo-325530 --namespace user-ltenorio
```

Resource usage quotas

Your current resource usage is reasonable.

2022 - 2024 EDITO Infra



My Services
Access your running services
Services are supposed to be shut down as soon as you stop using them actively.
Refresh
New service
Del
Running services
Autosubmit-demo
Service
Autosubmit-demo
Started:
a minute ago
Open
demo-325
Re
ago
Open

Your Autosubmit Demo App is being deployed.

This service have the following tools:

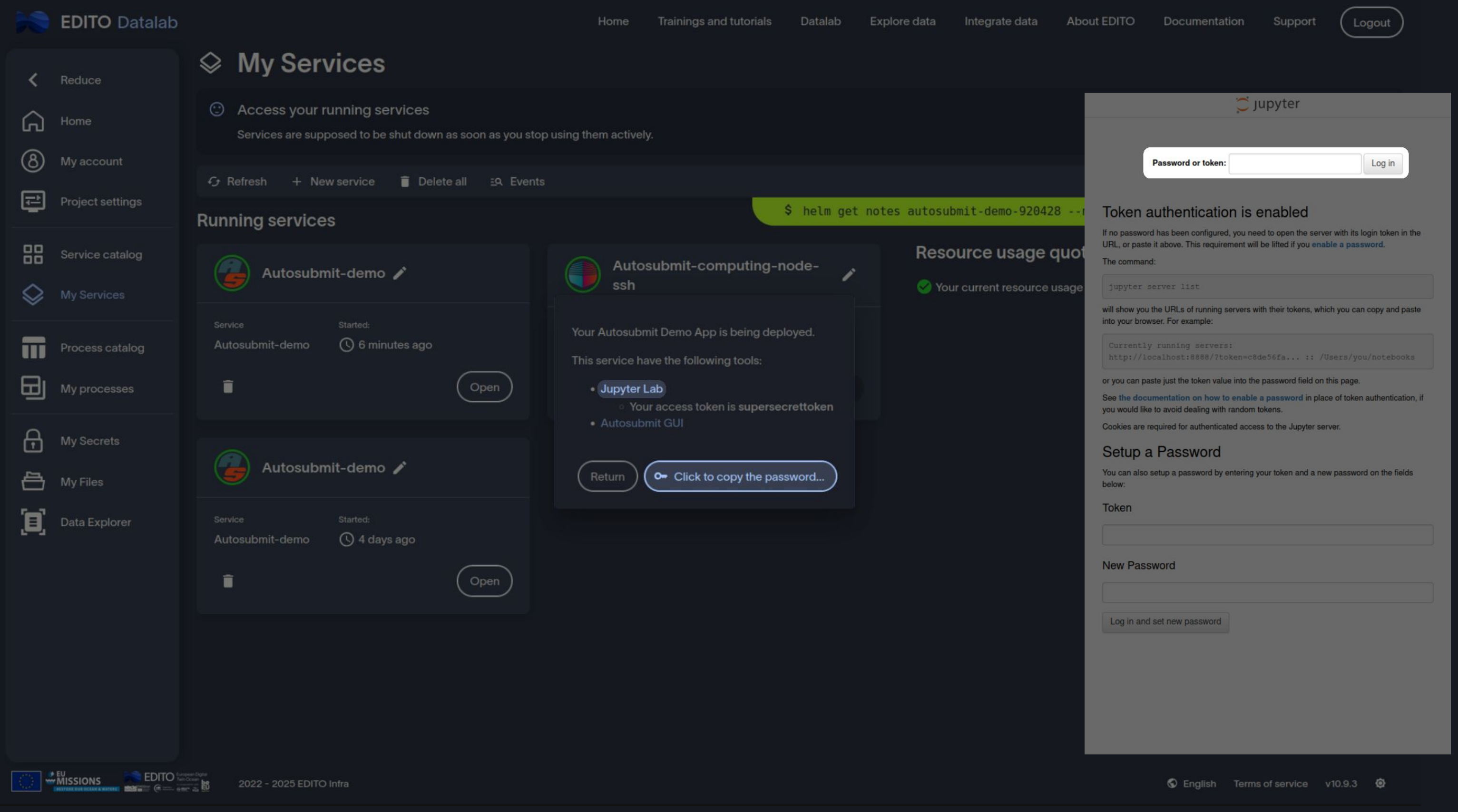
- Jupyter Lab
  - Your access token is **supersecrettoken**
- Autosubmit GUI

Return
Click to copy the password...

Jupyter Lab: service with a notebook with Autosubmit instance running, and access to HPC

Autosubmit GUI: web app to monitor Autosubmit experiments

Step 4. On Jupyter Lab, you’ll have to use your access token from the previous page



Access your running services

Services are supposed to be shut down as soon as you stop using them actively.

```
$ helm get notes autosubmit-demo-920428 --r
```

Service: Autosubmit-demo — Started: 6 minutes ago

Service: Autosubmit-demo — Started: 4 days ago

Your current resource usage

Your Autosubmit Demo App is being deployed.

This service have the following tools:

- Jupyter Lab
  - Your access token is supersecrettoken
- Autosubmit GUI

2022 - 2025 EDITO Infra

jupyter

Password or token: ▭ Log in

Token authentication is enabled

If no password has been configured, you need to open the server with its login token in the URL, or paste it above. This requirement will be lifted if you enable a password.

The command:

```
jupyter server list
```

will show you the URLs of running servers with their tokens, which you can copy and paste into your browser. For example:

```
Currently running servers:
http://localhost:8888/?token=c8de56fa... :: /Users/you/notebooks
```

or you can paste just the token value into the password field on this page.

See the documentation on how to enable a password in place of token authentication, if you would like to avoid dealing with random tokens.

Cookies are required for authenticated access to the Jupyter server.

Setup a Password

You can also setup a password by entering your token and a new password on the fields below:

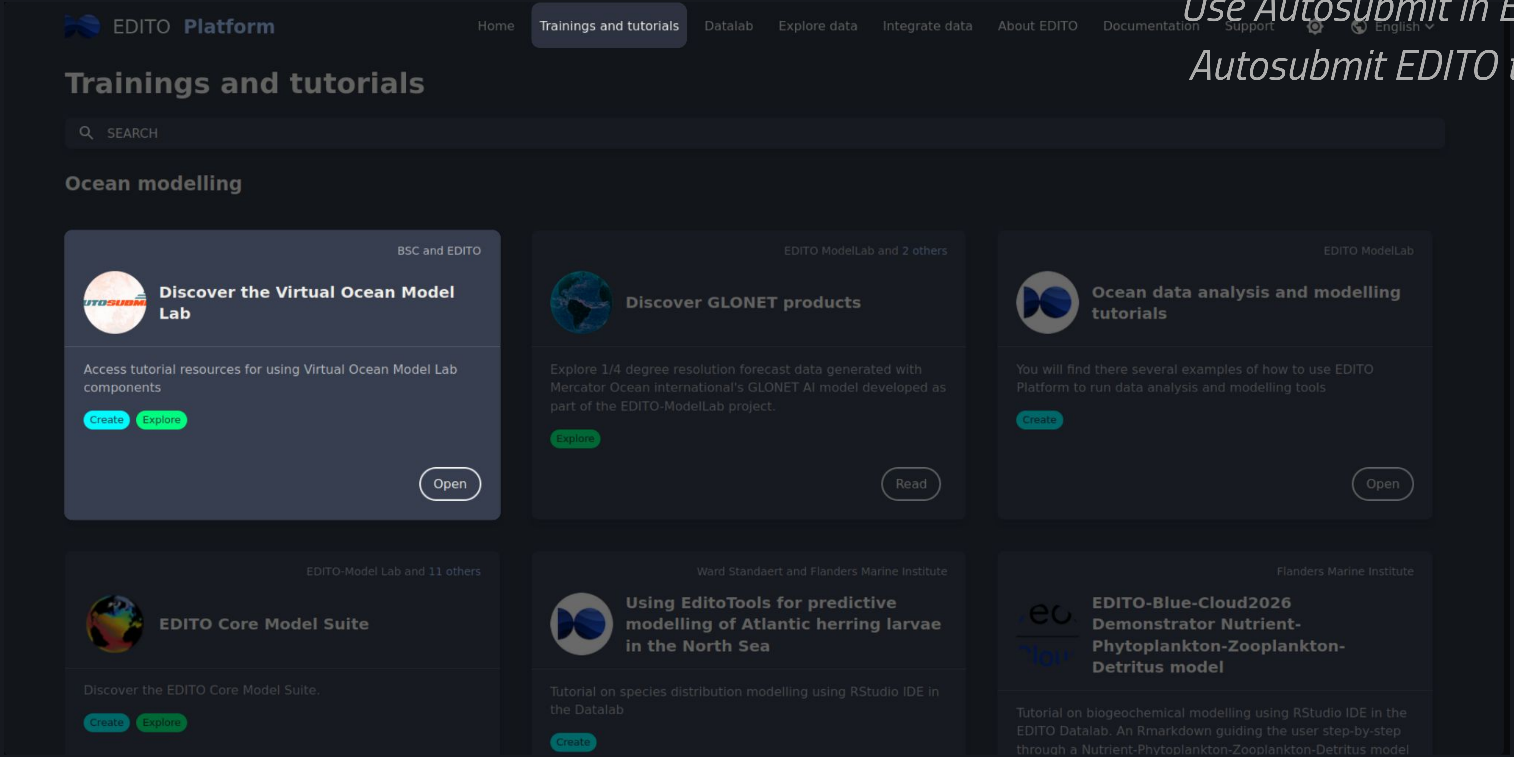
Token

New Password

Log in and set new password

Step 5. Find the "Autosubmit EDITO training"

Trainings and Tutorials >
Discover the Virtual Ocean Model Lab >
Use Autosubmit in EDITO >
Autosubmit EDITO training



Trainings and tutorials

Ocean modelling

BSC and EDITO

Discover the Virtual Ocean Model Lab

Access tutorial resources for using Virtual Ocean Model Lab components

Create Explore

Open

EDITO ModelLab and 2 others

Discover GLONET products

Explore 1/4 degree resolution forecast data generated with Mercator Ocean international's GLONET AI model developed as part of the EDITO-ModelLab project.

Explore

Read

EDITO ModelLab

Ocean data analysis and modelling tutorials

You will find there several examples of how to use EDITO Platform to run data analysis and modelling tools

Create

Open

EDITO-Model Lab and 11 others

EDITO Core Model Suite

Discover the EDITO Core Model Suite.

Create Explore

Ward Standaert and Flanders Marine Institute

Using EditoTools for predictive modelling of Atlantic herring larvae in the North Sea

Tutorial on species distribution modelling using RStudio IDE in the Datalab

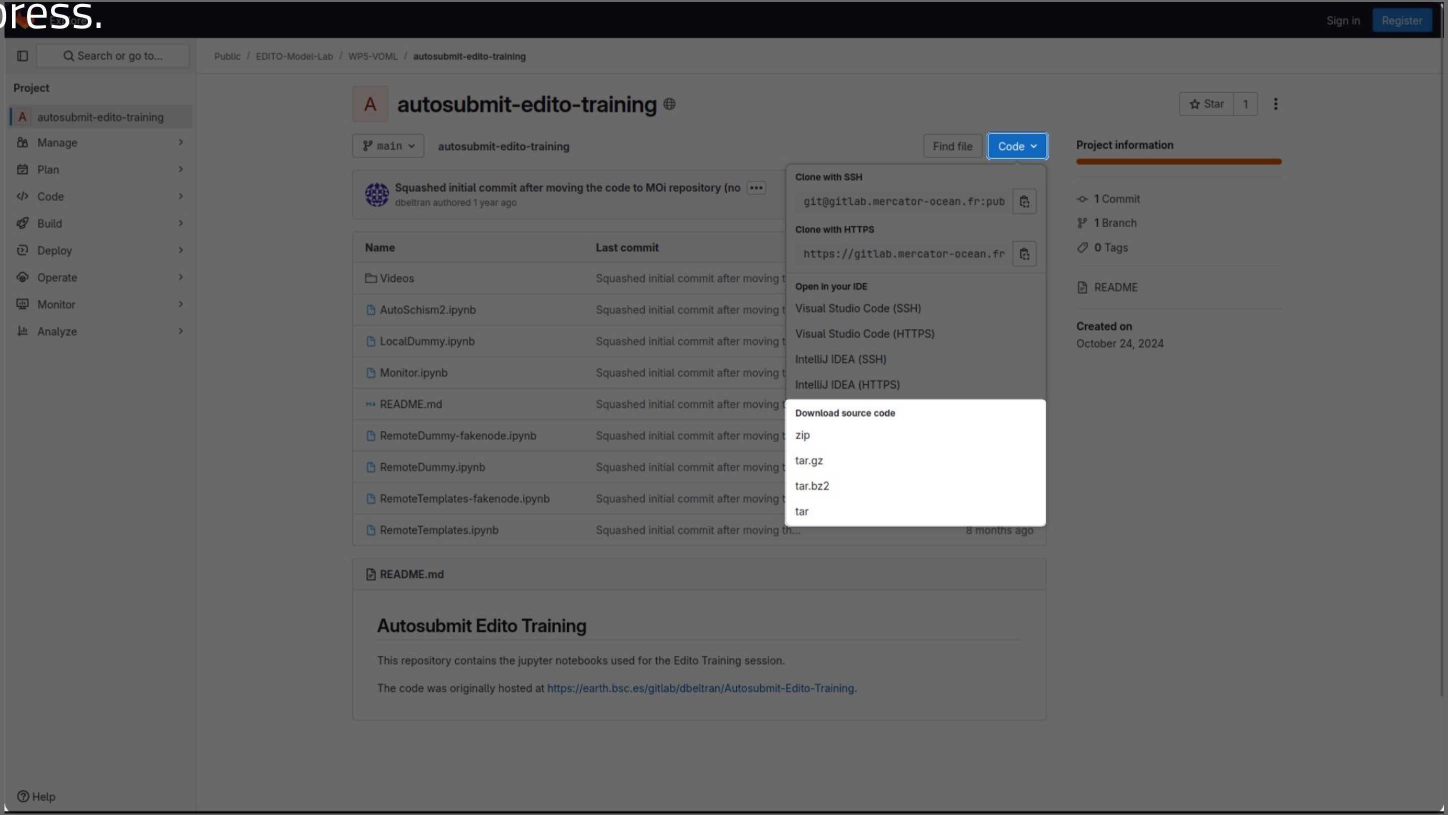
Create

Flanders Marine Institute

EDITO-Blue-Cloud2026 Demonstrator Nutrient-Phytoplankton-Zooplankton-Detritus model

Tutorial on biogeochemical modelling using RStudio IDE in the EDITO Datalab. An Rmarkdown guiding the user step-by-step through a Nutrient-Phytoplankton-Zooplankton-Detritus model

Step 6. Download the repository code in your preferred compression format and decompress.



autosubmit-edito-training

Squashed initial commit after moving the code to MOi repository (no
dbeltran authored 1 year ago

Clone with SSH

```
git@gitlab.mercator-ocean.fr:pub
```

Clone with HTTPS

```
https://gitlab.mercator-ocean.fr
```

Name	Last commit
Videos	Squashed initial commit after moving t
AutoSchism2.ipynb	Squashed initial commit after moving t
LocalDummy.ipynb	Squashed initial commit after moving t
Monitor.ipynb	Squashed initial commit after moving t
README.md	Squashed initial commit after moving t
RemoteDummy-fakenode.ipynb	Squashed initial commit after moving t
RemoteDummy.ipynb	Squashed initial commit after moving t
RemoteTemplates-fakenode.ipynb	Squashed initial commit after moving t
RemoteTemplates.ipynb	Squashed initial commit after moving th... 8 months ago

Created on
October 24, 2024

README.md

Autosubmit Edito Training

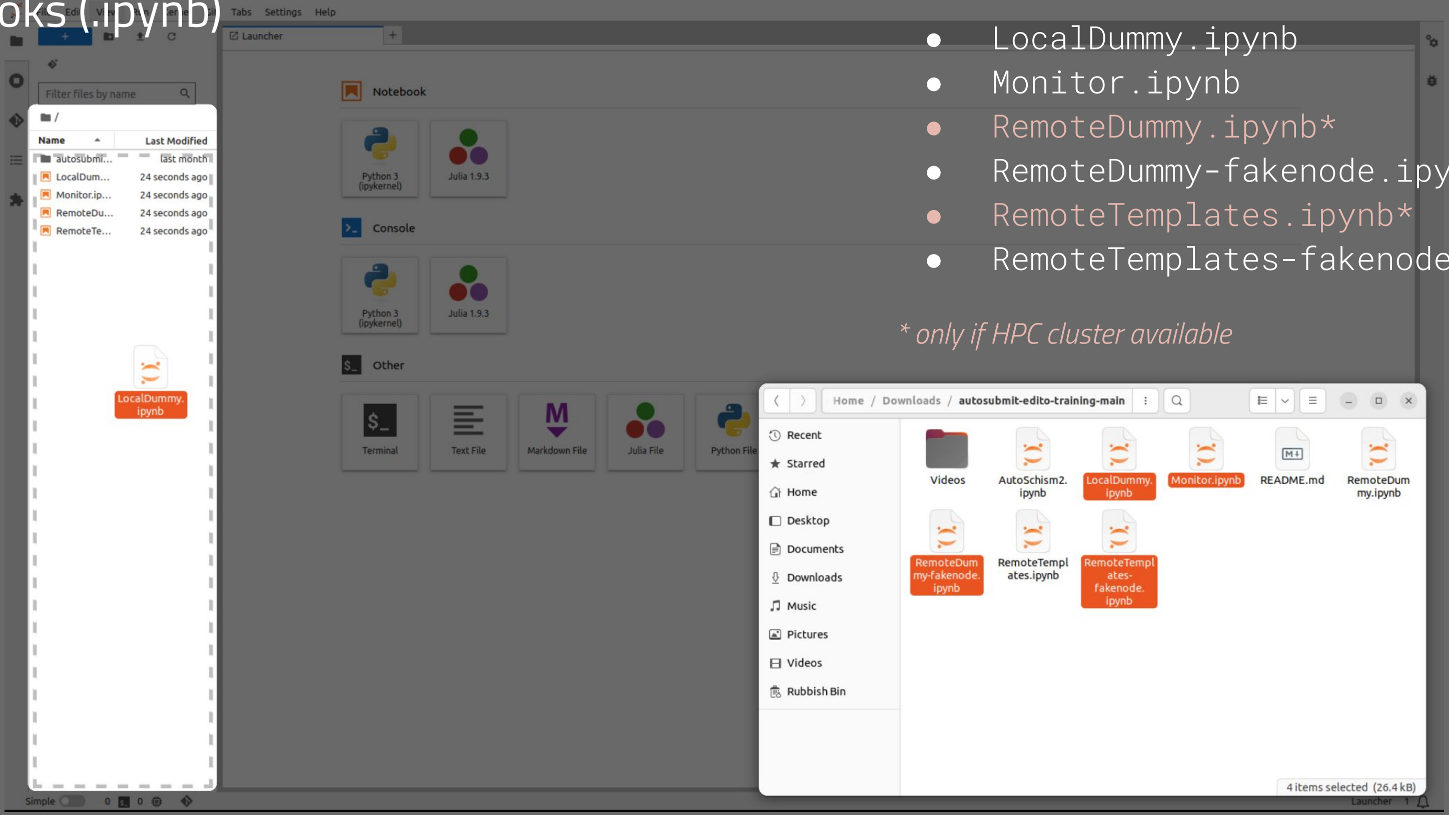
This repository contains the jupyter notebooks used for the Edito Training session.

The code was originally hosted at https://earth.bsc.es/gitlab/dbeltran/Autosubmit-Edito-Training.

Step 7. Go to the Jupyter Lab open in Step 4, drag and drop the compressed Jupyter Notebooks (.ipynb)

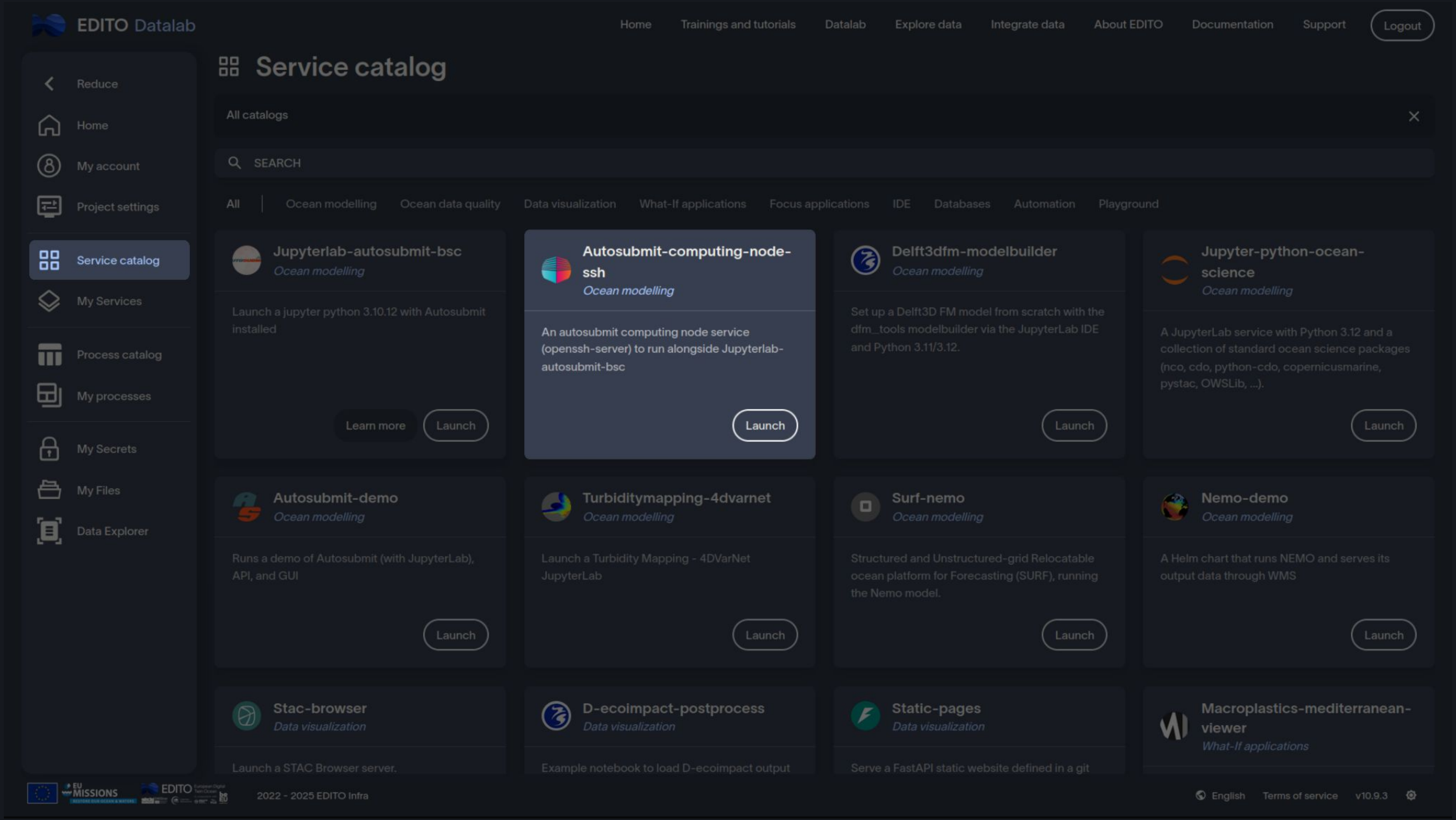
- LocalDummy.ipynb
- Monitor.ipynb
- RemoteDummy.ipynb*
- RemoteDummy-fakenode.ipynb
- RemoteTemplates.ipynb*
- RemoteTemplates-fakenode.ipynb

** only if HPC cluster available*



Name	Last Modified
autosubmi…	last month
LocalDum…	24 seconds ago
Monitor.ip…	24 seconds ago
RemoteDu…	24 seconds ago
RemoteTe…	24 seconds ago

Step 8. In the Service Catalog, launch the *Autosubmit-computing-node-ssh*



Service catalog

Jupyterlab-autosubmit-bsc
Ocean modelling
Launch a jupyter python 3.10.12 with Autosubmit installed

Autosubmit-computing-node-ssh
Ocean modelling
An autosubmit computing node service (openssh-server) to run alongside Jupyterlab-autosubmit-bsc

Delft3dfm-modelbuilder
Ocean modelling
Set up a Delft3D FM model from scratch with the dfm_tools modelbuilder via the JupyterLab IDE and Python 3.11/3.12.

Jupyter-python-ocean-science
Ocean modelling
A JupyterLab service with Python 3.12 and a collection of standard ocean science packages (nco, cdo, python-cdo, copernicusmarine, pystac, OWSLib, ...).

Autosubmit-demo
Ocean modelling
Runs a demo of Autosubmit (with JupyterLab), API, and GUI

Turbiditymapping-4dvarnet
Ocean modelling
Launch a Turbidity Mapping - 4DVarNet JupyterLab

Surf-nemo
Ocean modelling
Structured and Unstructured-grid Relocatable ocean platform for Forecasting (SURF), running the Nemo model.

Nemo-demo
Ocean modelling
A Helm chart that runs NEMO and serves its output data through WMS

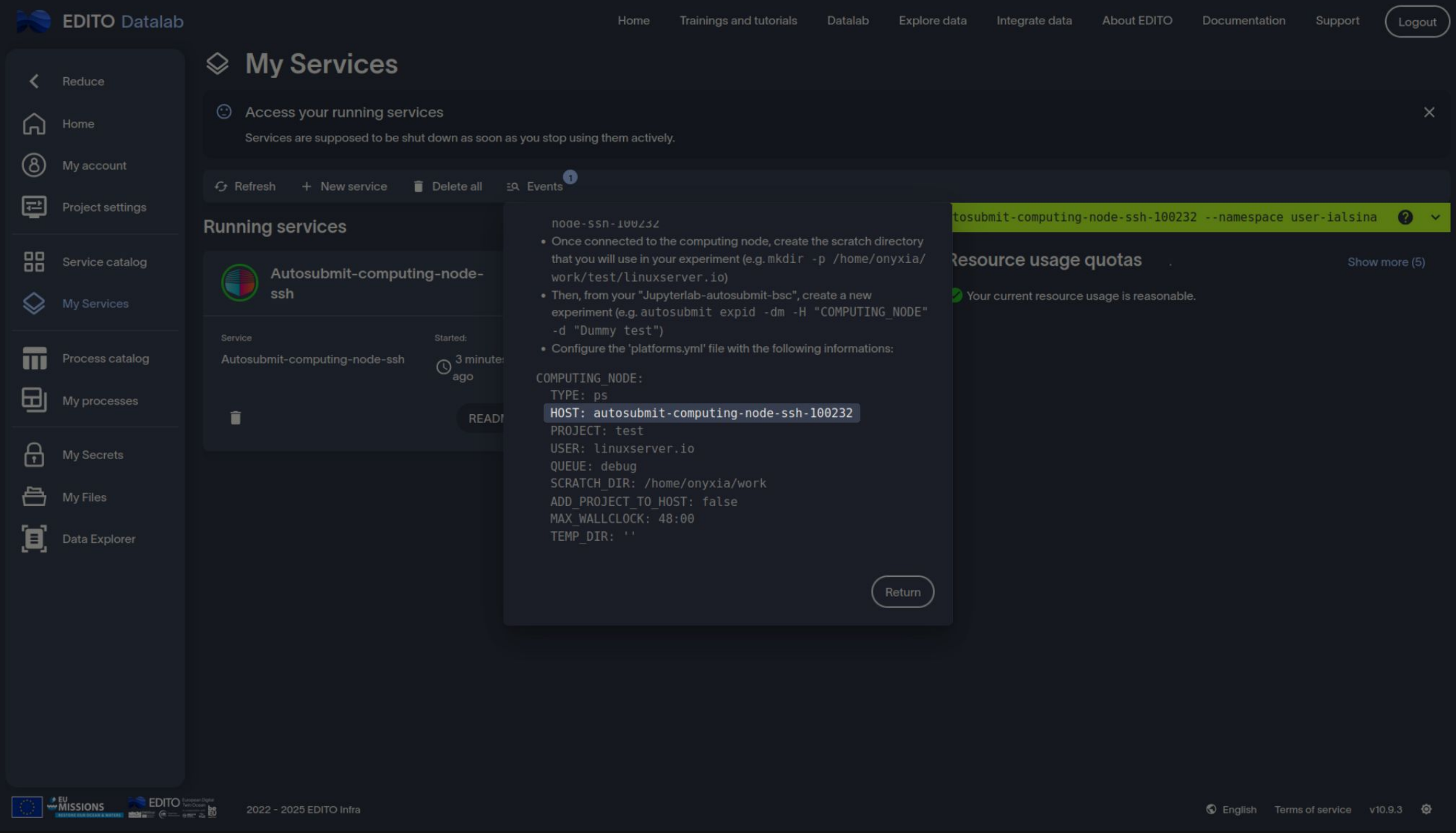
Stac-browser
Data visualization
Launch a STAC Browser server.

D-ecoimpact-postprocess
Data visualization
Example notebook to load D-ecoimpact output

Static-pages
Data visualization
Serve a FastAPI static website defined in a git

Macroplastics-mediterranean-viewer
What-If applications

Step 9. Once the service is launched, take note of the *HOST* key.



My Services

Access your running services

Services are supposed to be shut down as soon as you stop using them actively.

Running services

Autosubmit-computing-node-ssh

Service: Autosubmit-computing-node-ssh

Started: 3 minute ago

```
node-ssh-100232
```

- Once connected to the computing node, create the scratch directory that you will use in your experiment (e.g. `mkdir -p /home/onyxia/work/test/linuxserver.io`)
- Then, from your "Jupyterlab-autosubmit-bsc", create a new experiment (e.g. `autosubmit expid -dm -H "COMPUTING_NODE" -d "Dummy test"`)
- Configure the 'platforms.yml' file with the following informations:

```
COMPUTING_NODE:
  TYPE: ps
  HOST: autosubmit-computing-node-ssh-100232
  PROJECT: test
  USER: linuxserver.io
  QUEUE: debug
  SCRATCH_DIR: /home/onyxia/work
  ADD_PROJECT_TO_HOST: false
  MAX_WALLCLOCK: 48:00
  TEMP_DIR: ''
```

```
tosubmit-computing-node-ssh-100232 --namespace user-ialsina
```

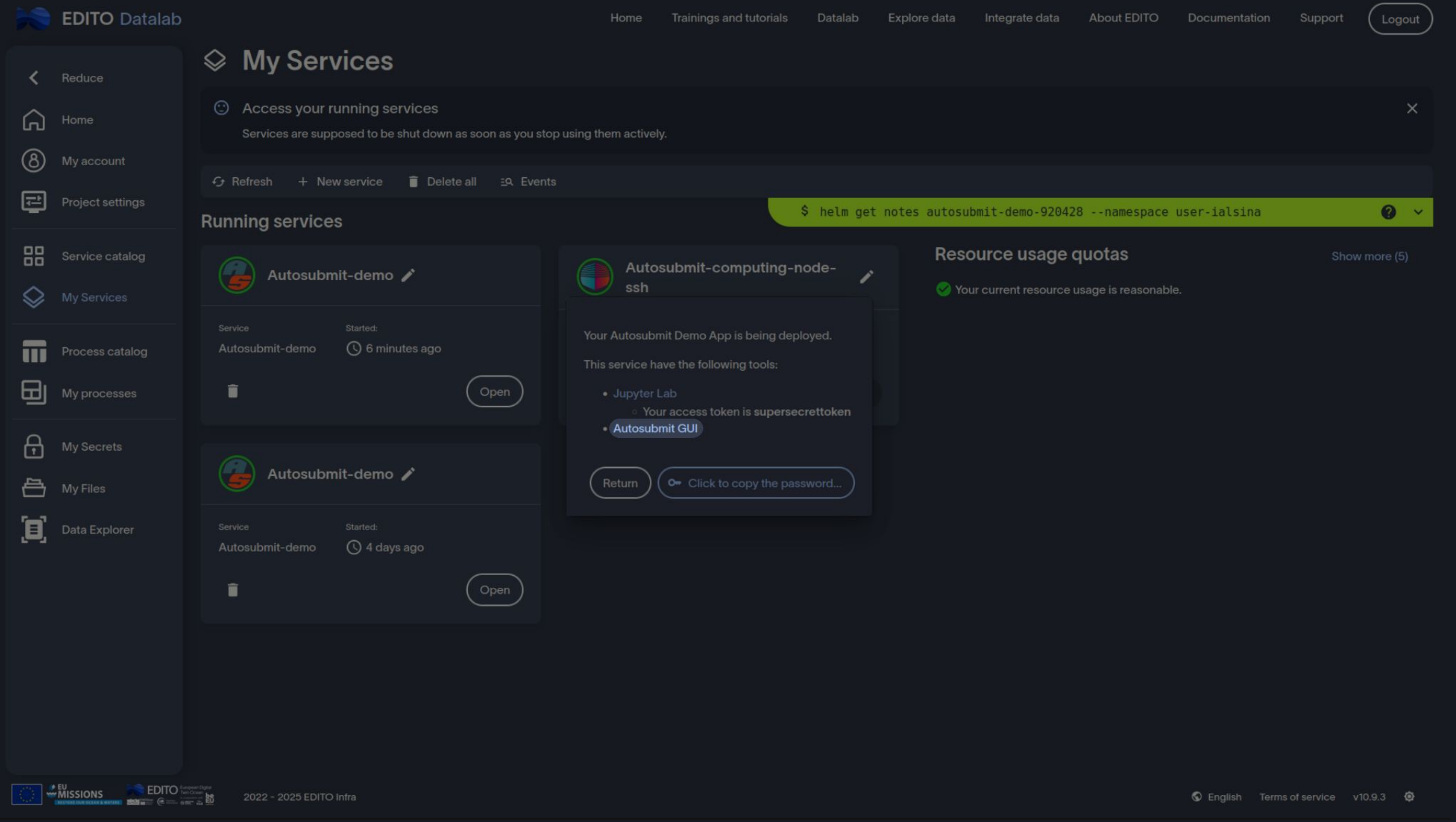
Resource usage quotas

Your current resource usage is reasonable.

2022 - 2025 EDITO Infra

Step 10. Follow the Jupyter Notebooks, changing the remote user, EDITO user and hostname where needed.

Step 11. Go back to the Autosubmit-demo service, and open the Autosubmit GUI.



My Services

Access your running services

Services are supposed to be shut down as soon as you stop using them actively.

```
$ helm get notes autosubmit-demo-920428 --namespace user-ialsina
```

Running services

Autosubmit-demo

Service: Autosubmit-demo

Started: 6 minutes ago

Autosubmit-demo

Service: Autosubmit-demo

Started: 4 days ago

Autosubmit-computing-node-ssh

Your Autosubmit Demo App is being deployed.

This service have the following tools:

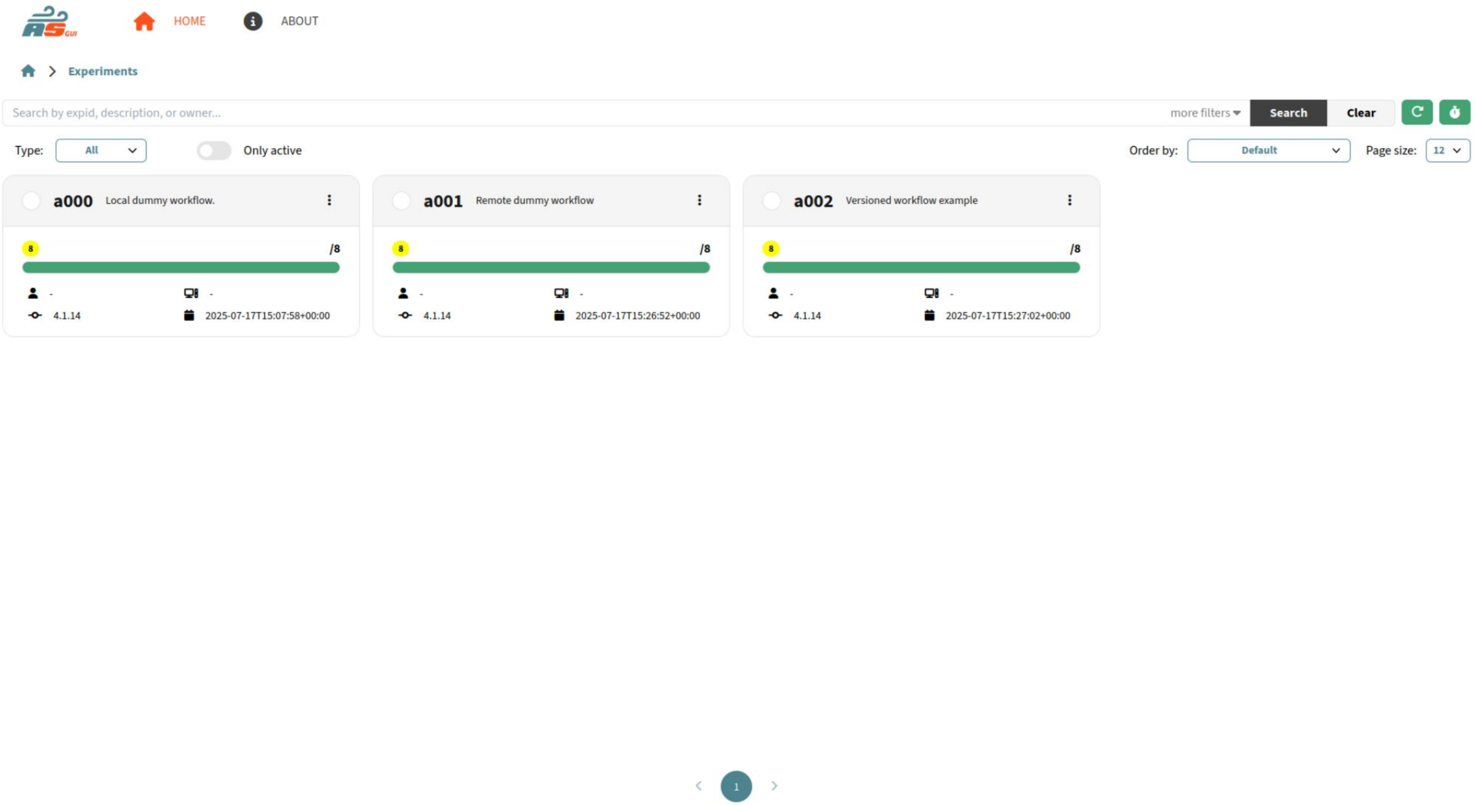
- Jupyter Lab
  - Your access token is supersecrettoken
- Autosubmit GUI

Resource usage quotas

Your current resource usage is reasonable.

2022 - 2025 EDITO Infra

Step 12. Explore your active experiments. Notice the *"Only active"* switch.



Search by expid, description, or owner...

Type: All

Only active

Order by: Default

Page size: 12

a000 Local dummy workflow.
8 /8
-
-
4.1.14
2025-07-17T15:07:58+00:00

a001 Remote dummy workflow
8 /8
-
-
4.1.14
2025-07-17T15:26:52+00:00

a002 Versioned workflow example
8 /8
-
-
4.1.14
2025-07-17T15:27:02+00:00

1

Useful links

- Autosubmit (AS)
  - Pip install and docs: https://autosubmit.readthedocs.io/en/master/installation/index.html
  - Repository: https://github.com/BSC-ES/autosubmit-gui
  - Contact: support-autosubmit@bsc.es
- Autosubmit GUI
  - Public mockup: https://earth.bsc.es/gitlab/wuruchi/autosubmitreact
  - Demo: https://autosubmitgui.bsc.es/presentation/