HORIZON 2020 - ICT-14-2016-1

AEGIS

Advanced Big Data Value Chains for Public Safety and Personal Security

WP4 – AEGIS Infrastructure Implementation and Rollout

D4.4 - AEGIS Platform - Release 4.00

Version 1.0

Due date: 30.06.2019 **Delivery Date: 30.06.2019**

Author(s): Maurizio Megliola, Ambra De Bonis, Javier Ladiana (GFT), Dimitrios Miltiadou, Konstantinos Perakis, Stamatis Pitsios (UBITECH), Spiros Mouzakitis, Christos Botsikas (NTUA), Fritz Meiners, Fabian Kirstein (Fraunhofer), Marios Phinikettos, George Bikas (SUITE5), Alexandru Ormenisan (KTH)

Editor: Maurizio Megliola (GFT)

Lead Beneficiary of Deliverable: GFT

Dissemination level: PU **Nature of the Deliverable:** Demonstrator

Internal Reviewers: Mahmoud Ismail (KTH), Spiros Mouzakitis (NTUA)

EXPLANATIONS FOR FRONTPAGE

Author(s): Name(s) of the person(s) having generated the Foreground respectively having written the content of the report/document. In case the report is a summary of Foreground generated by other individuals, the latter have to be indicated by name and partner whose employees he/she is. List them alphabetically.

Editor: Only one. As formal editorial name only one main author as responsible quality manager in case of written reports: Name the person and the name of the partner whose employee the Editor is. For the avoidance of doubt, editing only does not qualify for generating Foreground; however, an individual may be an Author - if he has generated the Foreground - as well as an Editor - if he also edits the report on its own Foreground.

Lead Beneficiary of Deliverable: Only one. Identifies name of the partner that is responsible for the Deliverable according to the AEGIS DOW. The lead beneficiary partner should be listed on the frontpage as Authors and Partner. If not, that would require an explanation.

Internal Reviewers: These should be a minimum of two persons. They should not belong to the authors. They should be any employees of the remaining partners of the consortium, not directly involved in that deliverable, but should be competent in reviewing the content of the deliverable. Typically this review includes: Identifying typos, Identifying syntax & other grammatical errors, Altering content, Adding or deleting content.

AEGIS KEY FACTS

Topic:	ICT-14-2016 - Big Data PPP: cross-sectorial and cross-lingual data integration and experimentation
Type of Action:	Innovation Action
Project start:	1 January 2017
Duration:	30 months from **01.01.2017** to **30.06.2019** (Article 3 GA)
Project Coordinator:	Fraunhofer
Consortium:	10 organizations from 8 EU member states

AEGIS PARTNERS

Fraunhofer	Fraunhofer-Gesellschaft zur Förderung der angewandten Forschung e.V.
GFT	GFT Italia SRL
KTH	Kungliga Tekniska högskolan
UBITECH	UBITECH Limited
VIF	Kompetenzzentrum - Das virtuelle Fahrzeug , Forschungsgesellschaft-GmbH
NTUA	National Technical University of Athens - NTUA
EPFL	École polytechnique fédérale de Lausanne
SUITE5	SUITE5 Limited
KONKAT	ANONYMOS ETAIREIA KATASKEVON-TECHNIKON ERGON, EMPORIKON, VIOMICHANIKONKAI NAUTILIAKON EPICHEIRISEON KON'KAT
HDIA	HDI Assicurazioni S.P.A

EXECUTIVE SUMMARY

This deliverable, as part of WP4 which is responsible for the implementation of the AEGIS platform, consists of the fourth and final version of the software package of the AEGIS core platform and its APIs with corresponding supporting documentation on the deployment of each component and usage of the APIs.

This accompanying document describes the implementation of the different AEGIS components of this final prototype release.

Table of Contents

LIST OF FIGURES

ABBREVIATIONS

CO	Confidential, only for members of the Consortium (including the Commission Services)
D	Deliverable
DoW	Description of Work
H2020	Horizon 2020 Programme
FLOSS	Free/Libre Open Source Software
GUI	Graphical User Interface
IPR	Intellectual Property Rights
MGT	Management
MS	Milestone
OS	Open Source
OSS	Open Source Software
O	Other
P	Prototype
PU	Public
PM	Person Month
R	Report
RTD	Research and Development
WP	Work Package
Y2	Year 2

1. Introduction

The present deliverable is released within the context of Workpackage 4 "AEGIS Infrastructure Implementation and Rollout" and is in particular associated with Task 4.4 "Continuous Integration and Testing Activities". Within this task, the third release of the AEGIS platform has been developed.

1.1. Insights from other tasks and deliverables

This deliverable is strictly related to the work done in WP3, more precisely to the work reported in D3.5, regarding two main features: from one hand, the detailed information of the AEGIS components, comprising an overview of its functionalities and positioning in the overall architecture, the technologies used for its implementation and the API (when available) that it exposes in order for other components to interact with. On the other hand, the BPMN diagrams that correspond to the main tasks a user will perform through the AEGIS, as seen from the user perspective, but also outlining component interactions, which provided the basis for the development of the AEGIS prototype. Another important source is the work done in WP2, as reported in D2.1/D2.2, especially the information about the features that must be offered through certain components, clarifying how important parts of the data value chain should be supported in practice, e.g. data policies, data harmonisation, metadata handling, knowledge extraction, and visualisation. Finally, the user perspective when utilising AEGIS, depicted by the usage scenarios reported in D1.2, has also driven the design and development of the AEGIS platform functionalities.

1.2. Structure

Section 2 of this document reports, for each foreseen platform component, the status within the prototype or -in case it is not yet integrated in the prototype- the component concept and application screenshots.

Section 3 concludes the deliverable.

2. AEGIS PLATFORM

In this chapter, we provide the references to the AEGIS project software source code as well as the related documentation about the component deployment and API usage.

Next, for each platform components, its status within the integrated prototype is described, in terms of its functionalities and how it works.

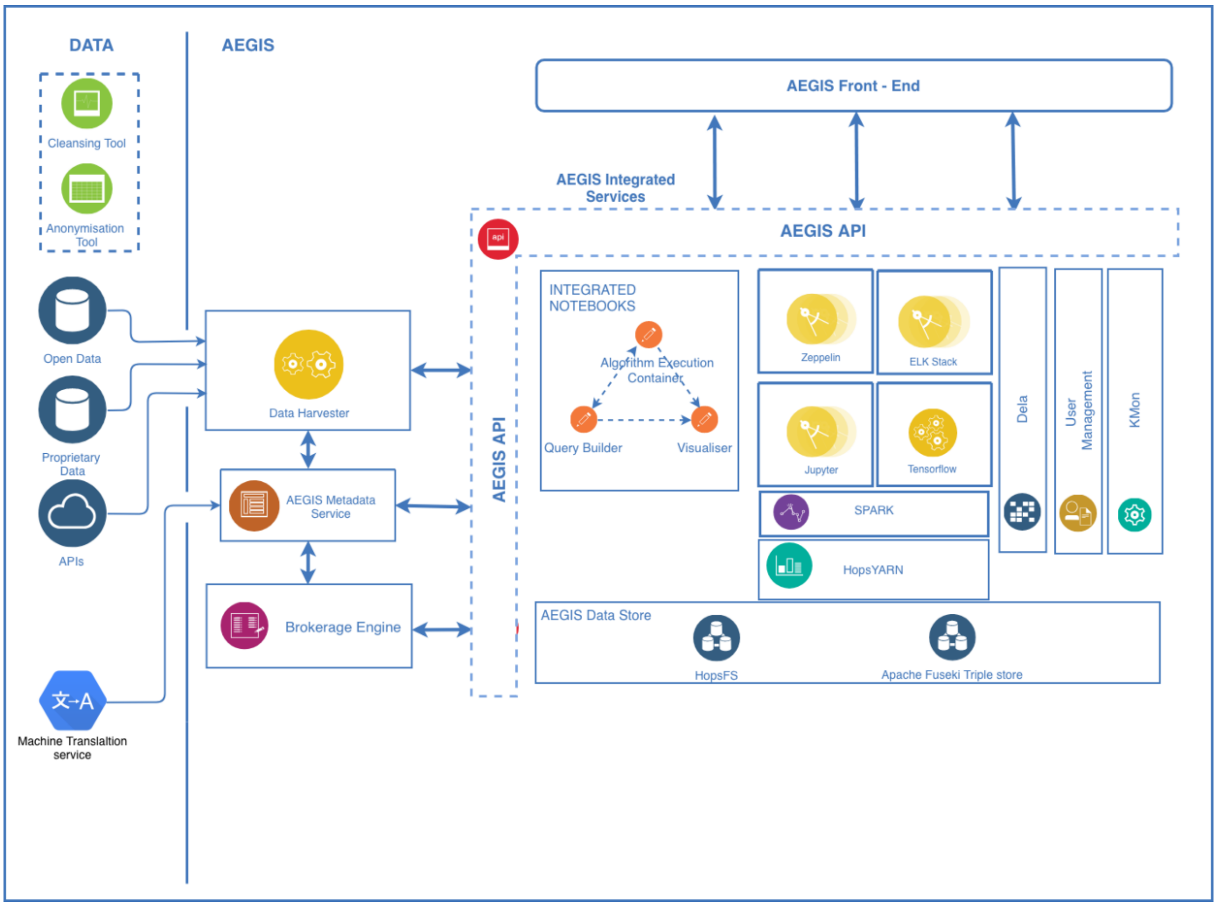


Figure 1: AEGIS platform architecture

2.1. URL

The AEGIS integrated prototype is available at the following URL:

https://platform.aegis-bigdata.eu [1]

The software core packages are available in AEGIS code repository at Github.com, at the following link:

[1] User: test@aegisbigdata.com; Password: Test01. The given AEGIS platform prototype URL will be activated in the first days of July 2019, until then the following URL should be used https://bbc6.sics.se:8181

https://github.com/aegisbigdata

The documentation related to the deployment of each component and usage of the APIs is provided on the project Github wiki, at the following link:

https://github.com/aegisbigdata/documentation/wiki

2.2. Data Harvester and Annotator

The following sections describe the status and implementation details of the Data Harvester and Data Annotator components in the final release of the prototype. The application can be accessed here: https://platform.aegis-bigdata.eu/hopsworks-api/harvester/

Data Harvester Microservices

The Data Harvester constitutes a microservice architecture, consisting out of the following four basic services, which services represent the fundamental harvesting pipeline.

- **Importer** - implements all functionality for retrieving data from a specific data source
- **Transformer** - converts the retrieved data from an importer into the target format of the AEGIS platform
- **Aggregator** - collects converted data from a transformer over a configurable time interval
- **Exporter** - uploads transformed and/or aggregated data to the AEGIS platform

The orchestration architecture used for the AEGIS Harvester follows a "pipeline" pattern, in which data is passed through several services, with each service manipulating the data in some sort. Each service is responsible for exactly one task. This permits a rather generic implementation of each service. The aim is to encourage a separation of concerns in order to enhance reusability, as well as allowing the dynamic scaling in times of high load. The latter is achieved by deploying additional instances of the services demanded most. Once new instances are spawned, request may dynamically be routed to the instance of a service with the least load.

For the final release of the AEGIS platform the following instances of the services have been implemented and orchestrated:

Importer
Seven importers have been implemented as described in the following table:

Data Provider	Description	Interfaces
OpenWeatherMap (https://openweathermap.org/)	Continuous harvesting of European Weather Data	RESTful JSON

European Data Portal - Single Dataset (https://www.europeandataportal.eu/data)	Harvest the resources and metadata of a single dataset	JSON Action API for metadata, Basic HTTP for files
European Data Portal – Multiple Datasets (https://www.europeandataportal.eu/data)	Harvest resources and metadata from multiple datasets by providing search parameters.	JSON Action API for metadata, Basic HTTP for files
AEGIS Event Detector	Push collected event data to the AEGIS platform	RESTful JSON (Push Method)
EM-DAT The International Disaster Database (https://www.emdat.be/)	Collect disaster data	HTML Crawling Basic HTTP for files
Generic Dataset Creation	Creates CSV datasets and metadata in the AEGIS platform based on tabular data provided as JSON.	RESTful JSON
Generic Dataset Upload	Comprehensive interface for uploading file and metadata to the AEGIS platform. It includes support for bulk upload.	RESTful JSON

Transformer
A basic transformer is available, allowing to provide scripts (JavaScript) for creating custom transformations from source to target. In addition, a basic CSV-to-CSV transformer was implemented.

Aggregator
One aggregator for accumulating CSV data over a specified time interval was developed.

Exporter
One exporter for uploading CSV files to the AEGIS platform was implemented.

To name a concrete example: Weather data may be imported hourly from an external service as JSON, which is then transformed into CSV with values being converted into the metric system. The results are then aggregated for 24 hours and the final file exported to an analytics platform.

Data Harvester Technology

In order to reuse as much as code as possible, the microservices introduced in the previous section have also been written in the Java programming language. However, this does not mean that future services must also be written using the same language, as each service is designed to be an independent piece of software. To aid with the creation of each service the framework Eclipse Vert.x[2] has been chosen. Apart from propagating an asynchronous programming paradigm ensuring a high availability and throughput, Eclipse Vert.x also provides a lot of built in support for managing microservices. Concepts that come to use are circuit breakers, health checks, and performance metrics.

The final services are deployed via containerization, in which each service is isolated from its host and other service's operating systems. The services are thereby packaged separately as *images*. The technology used for building these images is called Docker[3]. Aside from enhancing security this method also ensures platform independency, which means that these images (and consequently the services in question) will run on any infrastructure providing a Docker runtime. The concept of containerization also greatly helps with the dynamic launching of service instances mentioned previously.

To handle the frictionless interaction between various microservices their APIs have been defined using the OpenAPI 3[4] specification, which allows the precise definition of RESTful interfaces in a commonly understood format. The previously mentioned Eclipse Vert.x framework supports loading an OpenAPI document and exposing the endpoints defined, including request validation.

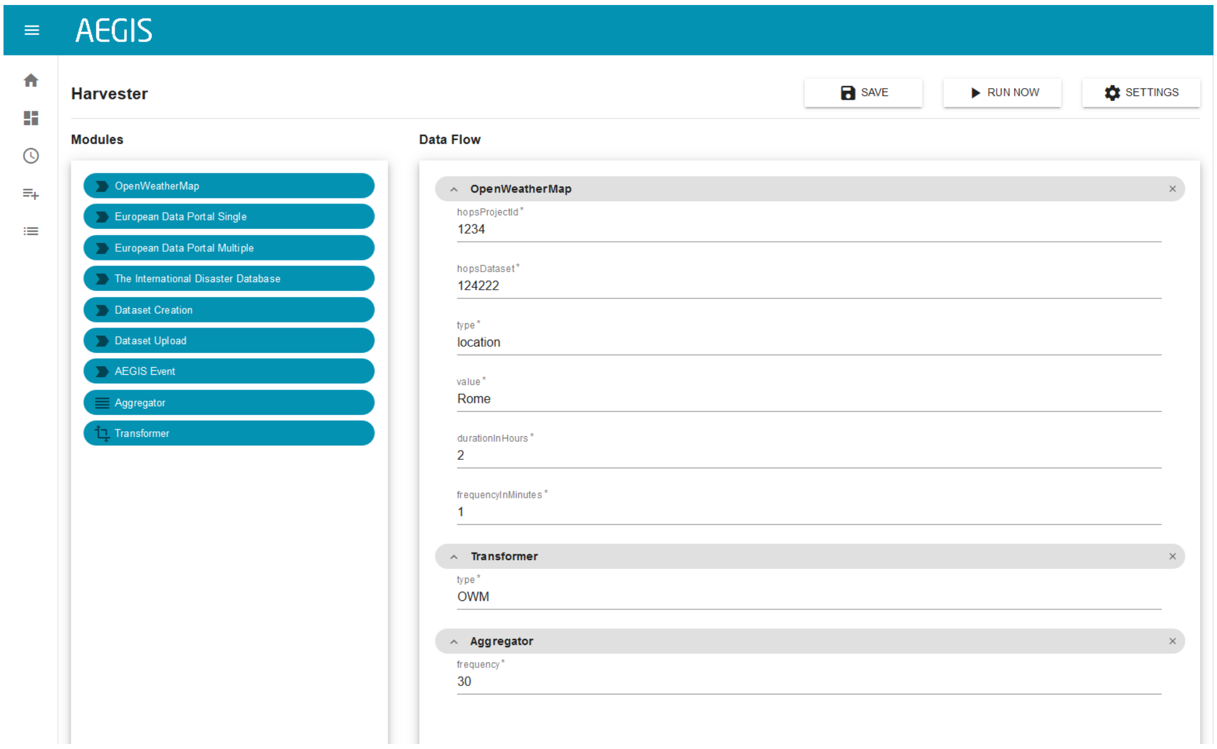
Data Harvester Frontend

The new approach for implementing the Data Harvester requires a specialised frontend, which allows the orchestration, configuration and execution of a specific harvesting process (pipe). The first basic version of this frontend was developed for the third release of the AEGIS platform. Figure 2 shows the main page of the application.

[2] https://vertx.io/

[3] https://www.docker.com/

[4] https://github.com/OAI/OpenAPI-Specification



OpenWeatherMap

hopsProjectId*: 1234
hopsDataset*: 124222
type*: location
value*: Rome
durationInHours*: 2
frequencyInMinutes*: 1

Transformer

type*: OWM

Aggregator

frequency*: 30

Figure 2: Mainpage of the Harvester Frontend

Data Annotator

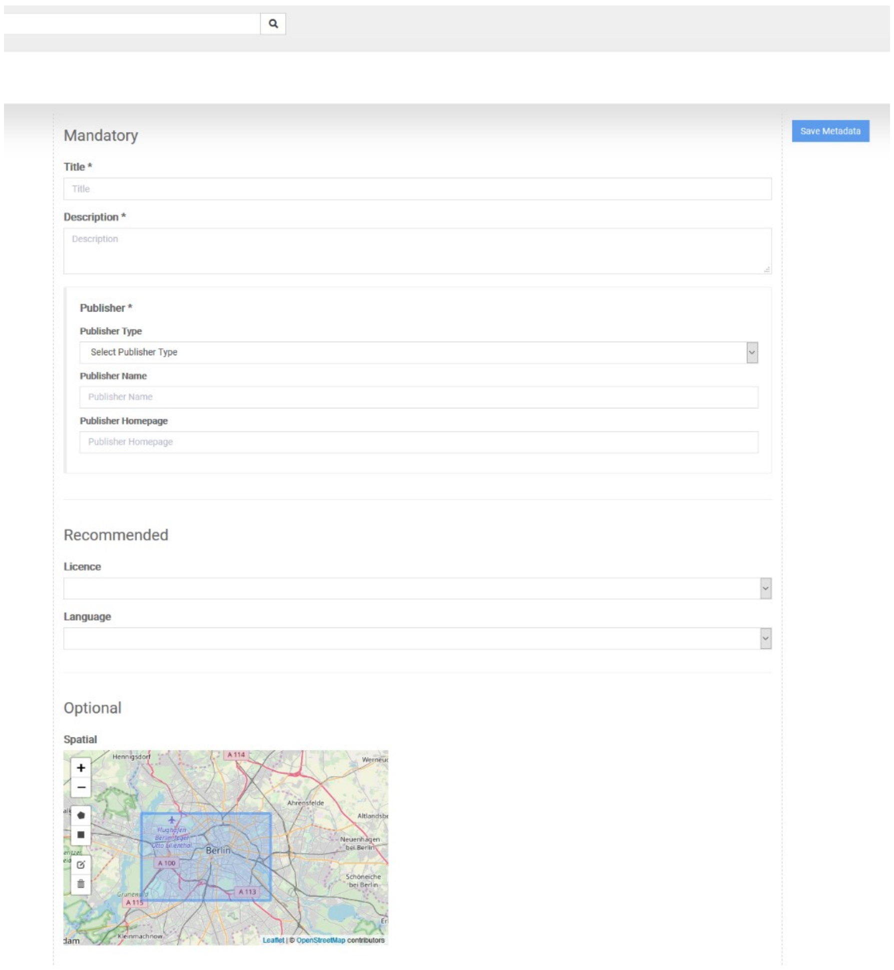
The Data Annotator (Metadata) was integrated into the AEGIS platform core frontend, by extending the AngularJS frontend. It allows the simple provision of detailed and semantic valuable metadata for projects, data sets and files in the AEGIS platform. The Data Annotator provides metadata forms and schemas for the three organisational levels of the AEGIS platform and is based on the DCAT-AP specification:

Project - The metadata corresponds to the DCAT-AP class *Catalogue*

Dataset - The metadata corresponds to the DCAT-AP class *Dataset* and is extended by additional properties to represent the marketplace functionality, e.g. the price.

File - The metadata corresponds to the DCAT-AP class *Distribution* and is extended by semantic descriptions for tabular data based on the AEGIS vocabulary and ontology. Each file/distribution is linked to a dataset.

The Data Annotator uses the AEGIS Metadata Service for storing and managing the information. The following figures show the actual annotation interfaces:



Mandatory

Title *

Title

Description *

Description

Publisher *

Publisher Type

Select Publisher Type

Publisher Name

Publisher Name

Publisher Homepage

Publisher Homepage

Recommended

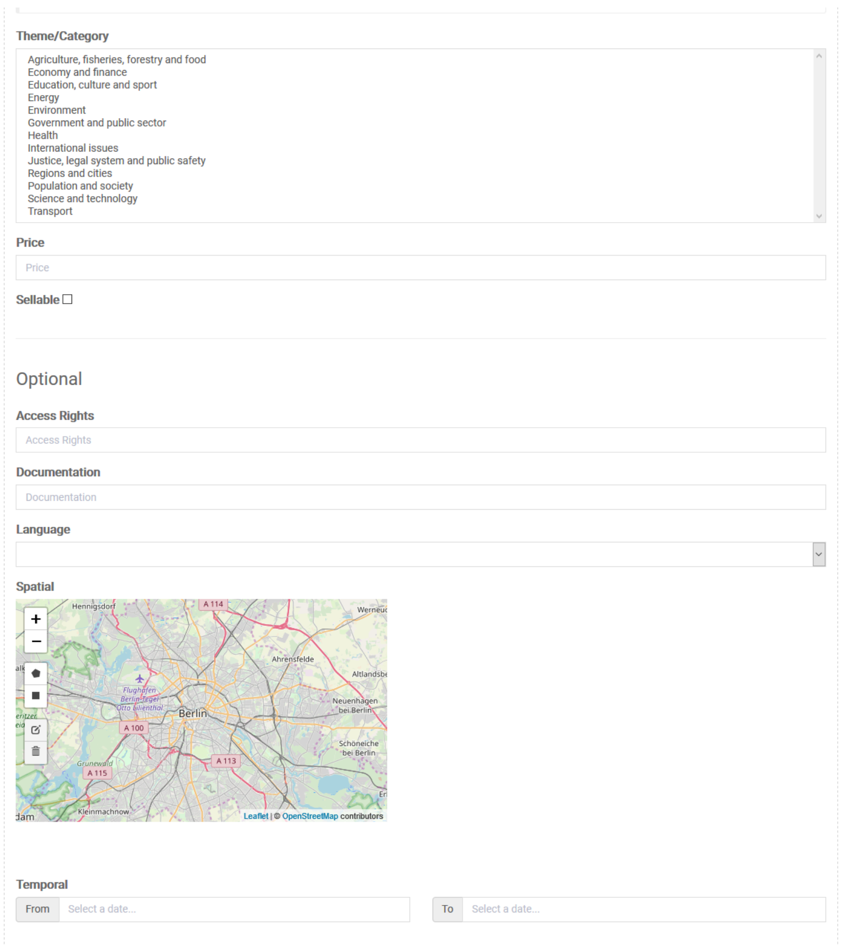
Licence

Language

Optional

Spatial

Figure 3: Metadata for a Project



Theme/Category

Agriculture, fisheries, forestry and food
Economy and finance
Education, culture and sport
Energy
Environment
Government and public sector
Health
International issues
Justice, legal system and public safety
Regions and cities
Population and society
Science and technology
Transport

Price

Price

Sellable ☐

Optional

Access Rights

Access Rights

Documentation

Documentation

Language

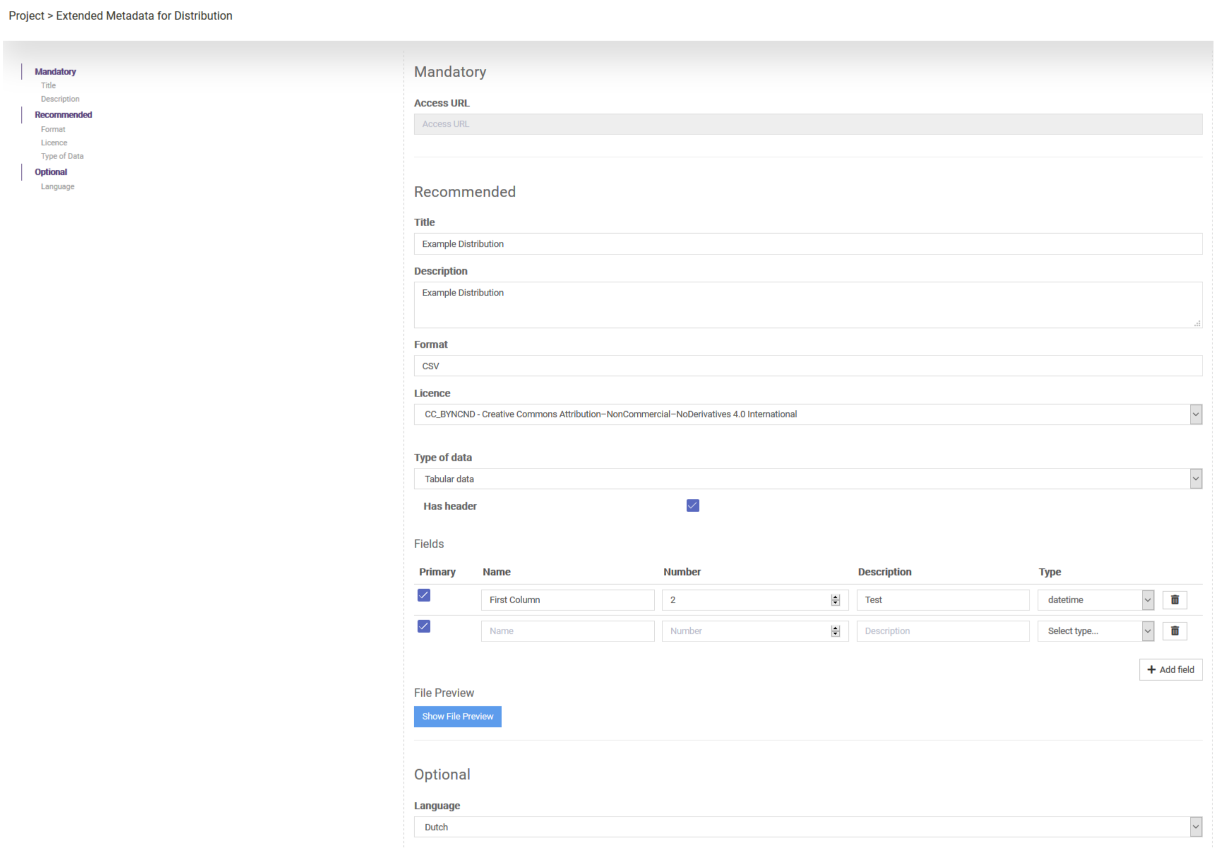
Spatial

Temporal

From Select a date...

To Select a date...

Figure 4: Excerpt of Annotator for a Dataset


Project > Extended Metadata for Distribution

Mandatory
Title
Description
Recommended
Format
Licence
Type of Data
Optional
Language

Mandatory

Access URL: Access URL

Recommended

Title: Example Distribution

Description: Example Distribution

Format: CSV

Licence: CC_BYNCND - Creative Commons Attribution–NonCommercial–NoDerivatives 4.0 International

Type of data: Tabular data

Has header ✓

Fields

Primary	Name	Number	Description	Type
✓	First Column	2	Test	datetime
✓	Name	Number	Description	Select type...

Optional

Language: Dutch

Figure 5: Data Annotator for Files

2.3. Cleansing Tool

Within the context of the AEGIS platform, the data cleansing tasks, as it have been documented also in the deliverables of WP3, are implemented following a two-fold approach: (a) an offline cleansing tool, residing where the data are located, that is offering a variety of cleansing processes and functionalities covering all the crucial and required data cleansing aspects with regard to data validation, data cleansing and data completion that can applied to any datasets prior to importing them in the AEGIS platform and (b) an online cleansing tool suitable for data cleansing and manipulation tasks that are executed during the data query creation process, capable of addressing certain simple cleansing tasks that are time computationally intense leveraging the computational power of the AEGIS platform.

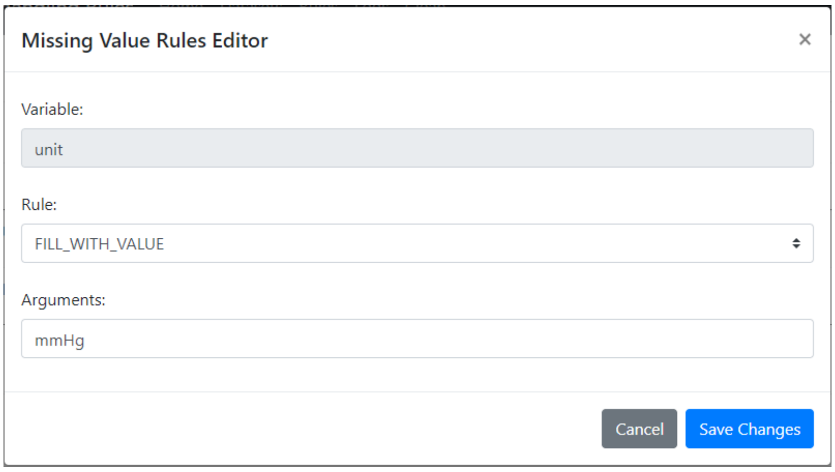
With regard to the online cleansing tool, the described functionalities are the offered functionalities that are incorporated inside the Query Builder component and will be described in the corresponding section (See 2.8).

The AEGIS offline cleansing tool is designed and implemented with the aim of facilitating the users to accomplish the required cleansings tasks. For this reason, the tool is easily customisable and adaptable to the user's needs taking into consideration the variety of the heterogeneous sources that will be imported in the AEGIS platform. Following the design that is documented in the deliverables of WP3, the offline cleansing tool is implemented as a standalone application written in Python and following the microservices architecture using Flask microframework[5] and a set of libraries such as Pandas[6] and NumPy[7].

The offline cleansing tool offers an extended list of rules for data validation, data cleansing and missing data handling to the user that can be set according the user's needs and the nature of the data. Besides the set of basic and advanced rules that tool is offering, additional rules are included in the list based on the feedback received from the AEGIS stakeholders. The tool is offering an intuitive user interface where the user is able to review the results of the execution of the cleansing tasks. Furthermore, the user interface of the tool has been optimised in order to offer a novel user experience to the users.

The offline cleansing tool is also offering a REST-API with the appropriate endpoints facilitating the uploading of the dataset that will be used in the cleansing process and the execution of the cleansing tasks. The REST-API documentation is also available using the Swagger framework. In addition to this, the tool provides a user interface with an upload form suitable for the cleansing of CSV and XLSX files. Additionally, the tool is optimised in order to handle large datasets efficiently. Also, the offline cleansing tool can be connected with the Harvester, in order to upload the cleaned datasets directly into the platform. However, this is supported only via the exposed REST API and not through the UI. Finally, the tool is available also as a Docker image, facilitating the easy installation on the premises of the user.

The following figures illustrate some indicative examples from the execution of the offline cleansing tool.


Missing Value Rules Editor

Variable: unit

Rule: FILL_WITH_VALUE

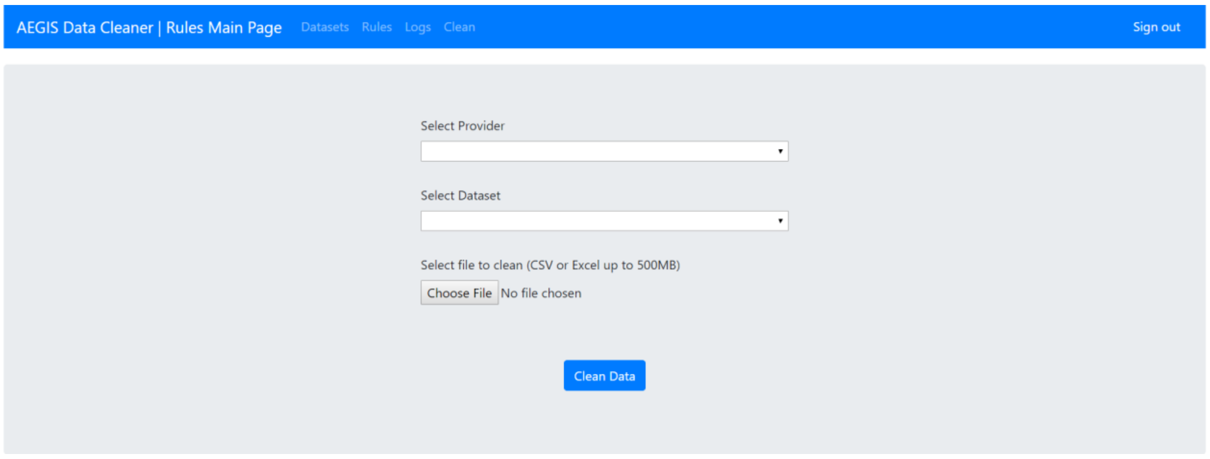
Arguments: mmHg

Figure 6: Offline cleansing tool rules editing

[5] http://flask.pocoo.org/

[6] https://pandas.pydata.org/

[7] http://www.numpy.org/



AEGIS Data Cleaner | Rules Main Page Datasets Rules Logs Clean Sign out

Select Provider

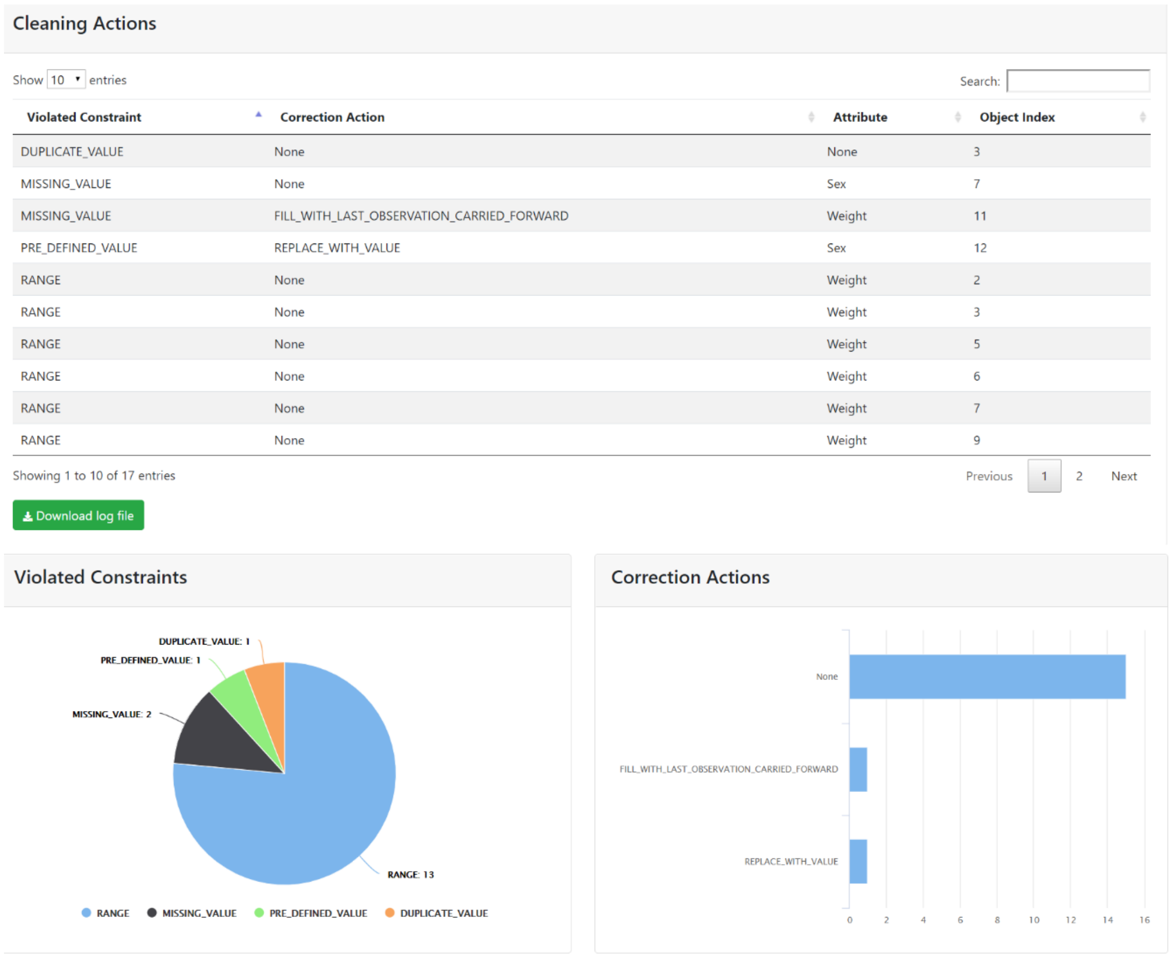
Select Dataset

Select file to clean (CSV or Excel up to 500MB)

Choose File No file chosen

Clean Data

Figure 7: Offline cleansing tool upload form for CSV files



Cleaning Actions

Show 10 entries Search:

Violated Constraint	Correction Action	Attribute	Object Index
DUPLICATE_VALUE	None	None	3
MISSING_VALUE	None	Sex	7
MISSING_VALUE	FILL_WITH_LAST_OBSERVATION_CARRIED_FORWARD	Weight	11
PRE_DEFINED_VALUE	REPLACE_WITH_VALUE	Sex	12
RANGE	None	Weight	2
RANGE	None	Weight	3
RANGE	None	Weight	5
RANGE	None	Weight	6
RANGE	None	Weight	7
RANGE	None	Weight	9

Showing 1 to 10 of 17 entries Previous 1 2 Next

Download log file

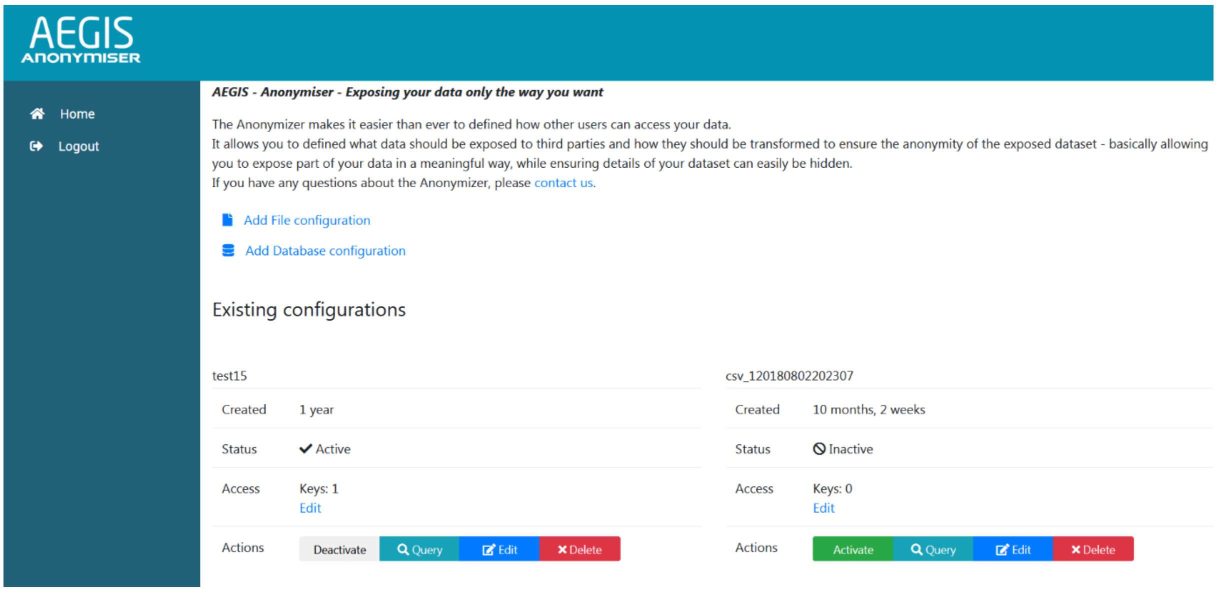
Figure 8: Offline cleansing tool results overview

2.4. Anonymisation Tool

The anonymisation tool is an extensible, schema-agnostic plugin that allows real-time efficient data anonymisation. The anonymisation tool has been utilized for offline, private usage but offers the ability to output the anonymized data through a secured, web API. The anonymization either syncs with private database servers or imports files from the filesystem and executes anonymisation functions on datasets of various sizes with little or no overhead. The purpose of the anonymisation is to allow the exploitation of the raw data in the system by accounting for privacy concerns and legal limitations.

For the final release, the UI was rebranded to match the AEGIS platform and several bugs were fixed.

The first screen of the tool allows the user to setup or edit an existing, saved configuration. Each configuration is basically a separate anonymization project to various database back-ends or text files, with different executed anonymization functions. By creating a new configuration, the system will prompt the user to connect to the private database backend or select the file to open and select the entities / tables to anonymise.



AEGIS - Anonymiser - Exposing your data only the way you want

The Anonymizer makes it easier than ever to defined how other users can access your data.
It allows you to defined what data should be exposed to third parties and how they should be transformed to ensure the anonymity of the exposed dataset - basically allowing you to expose part of your data in a meaningful way, while ensuring details of your dataset can easily be hidden.
If you have any questions about the Anonymizer, please contact us.

Add File configuration

Add Database configuration

Existing configurations

test15

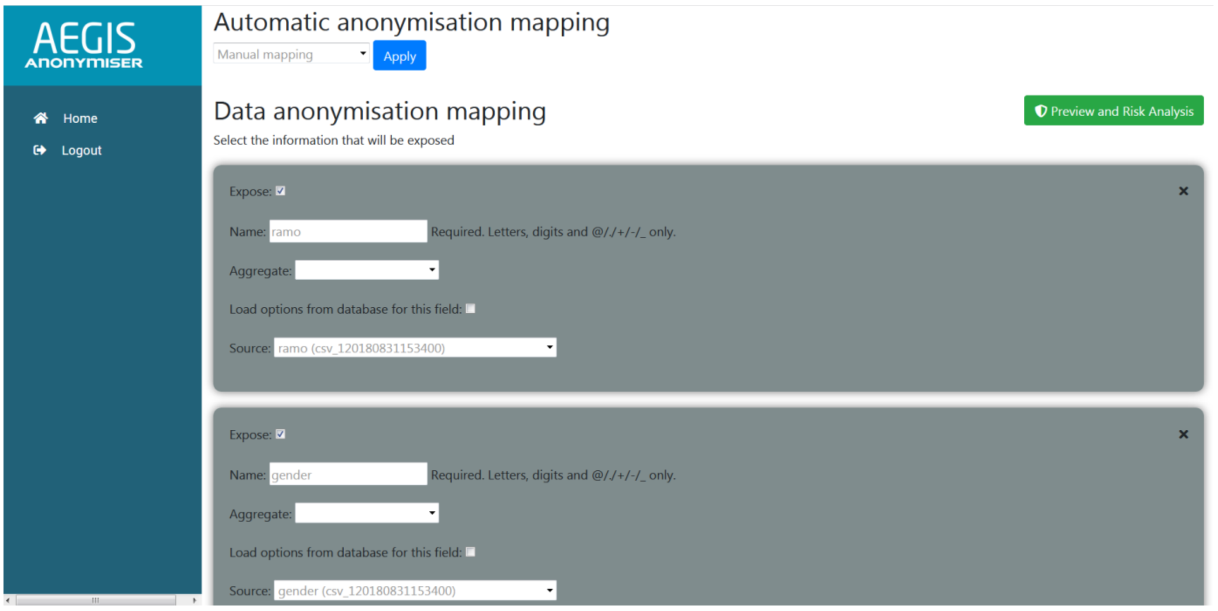
Created	1 year
Status	✓ Active
Access	Keys: 1 Edit
Actions	Deactivate · Query · Edit · Delete

csv_120180802202307

Created	10 months, 2 weeks
Status	Inactive
Access	Keys: 0 Edit
Actions	Activate · Query · Edit · Delete

Figure 9: Anonymiser configuration screen

The system then prompts the user to select which fields from the data source to expose to the anonymised set, as well as the anonymisation function to be performed.



Automatic anonymisation mapping

Manual mapping

Data anonymisation mapping

Select the information that will be exposed

Expose: ☑

Name: ramo Required. Letters, digits and @/./+/-/_ only.

Aggregate:

Load options from database for this field: ☐

Source: ramo (csv_120180831153400)

Expose: ☑

Name: gender Required. Letters, digits and @/./+/-/_ only.

Aggregate:

Load options from database for this field: ☐

Source: gender (csv_120180831153400)

Figure 10: Anonymiser data selection

The anonymisation system comes with a list of predefined anonymisation functions that can be used directly (e.g. city from an exact address, range of values from an integer), as well as a list of aggregation functions (e.g. average).

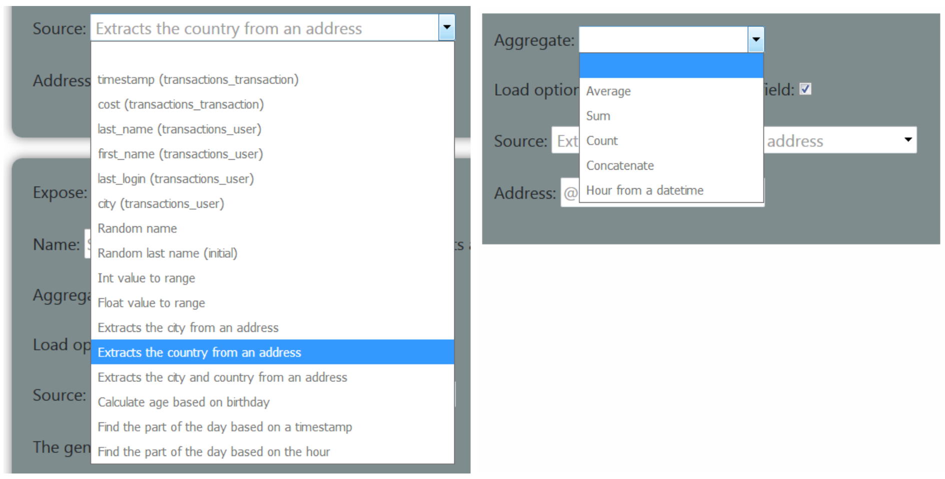
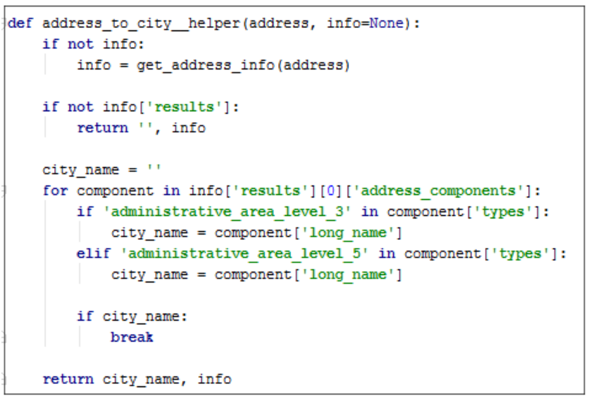

Source: Extracts the country from an address
Address
Expose:
Name:
Aggreg
Load op
Source:
The gen
timestamp (transactions_transaction)
cost (transactions_transaction)
last_name (transactions_user)
first_name (transactions_user)
last_login (transactions_user)
city (transactions_user)
Random name
Random last name (initial)
Int value to range
Float value to range
Extracts the city from an address
Extracts the country from an address
Extracts the city and country from an address
Calculate age based on birthday
Find the part of the day based on a timestamp
Find the part of the day based on the hour
Aggregate:
Load option
ield: ☑
Source: Ext
address
Address: @
Average
Sum
Count
Concatenate
Hour from a datetime

Figure 11: Anonymiser indicative functions

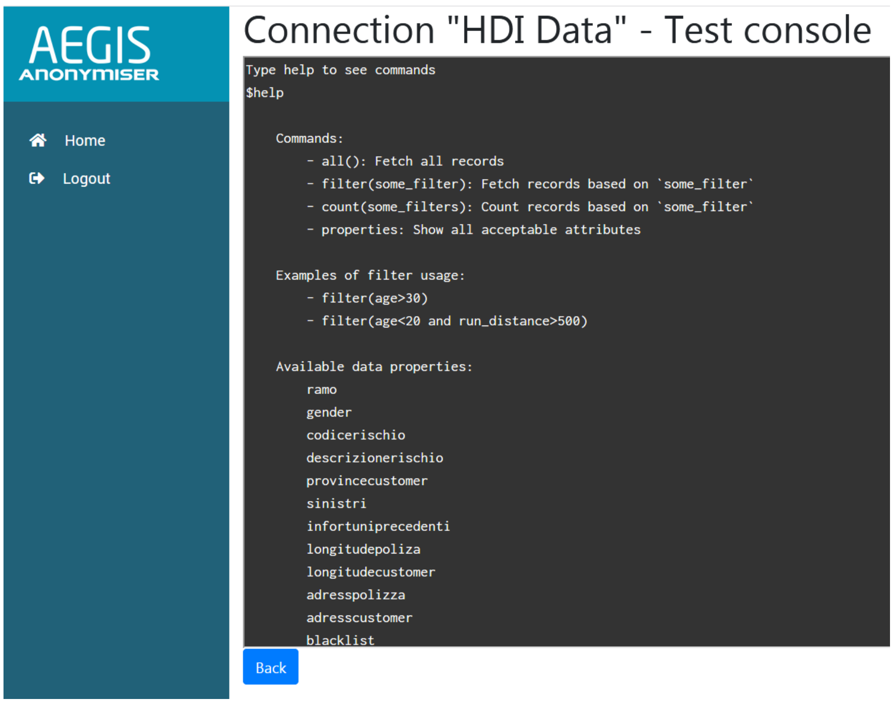
The tool can be easily extended with any new, custom anonymisation functions defined by the user in a python module.



```
def address_to_city__helper(address, info=None):
    if not info:
        info = get_address_info(address)

    if not info['results']:
        return '', info

    city_name = ''
    for component in info['results'][0]['address_components']:
        if 'administrative_area_level_3' in component['types']:
            city_name = component['long_name']
        elif 'administrative_area_level_5' in component['types']:
            city_name = component['long_name']

        if city_name:
            break

    return city_name, info
```

Figure 12: Anonymiser custom user-defined function

The user is able to execute queries and test the anonymised output through the integrated console of the tool as seen in the following screenshot.

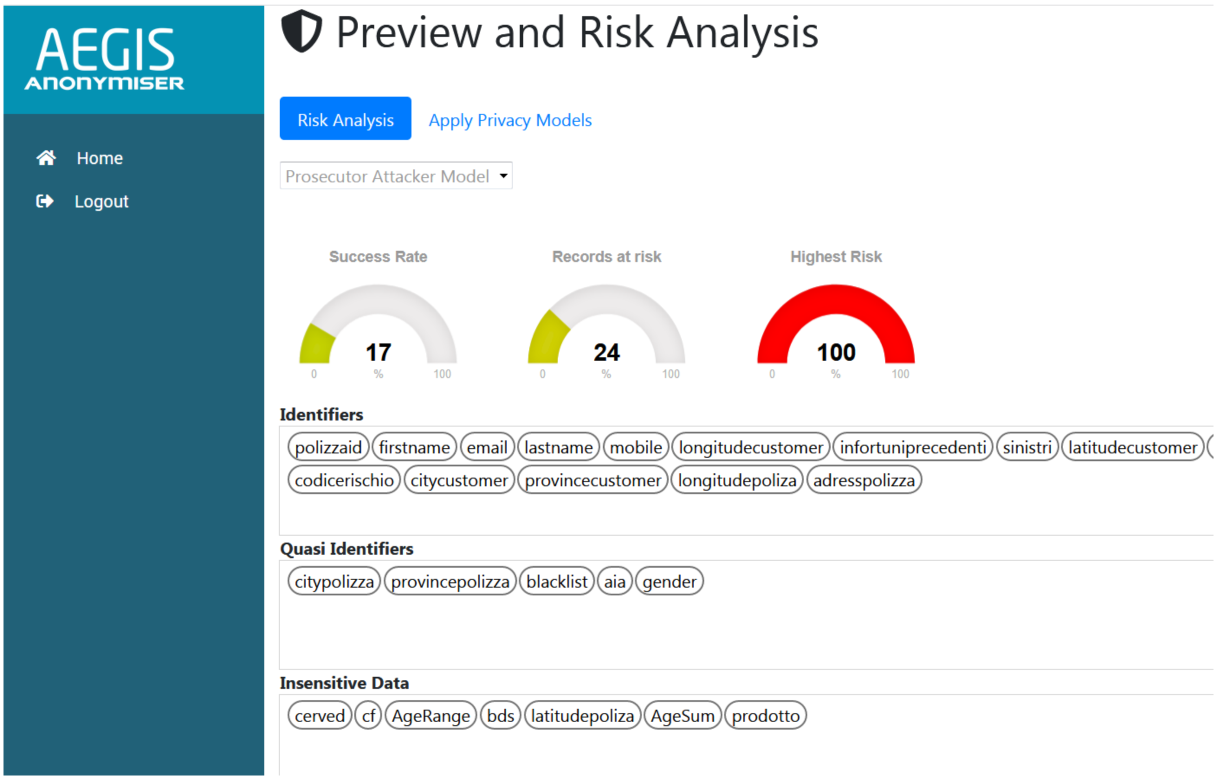


Connection "HDI Data" - Test console

```
Type help to see commands
$help

    Commands:
        - all(): Fetch all records
        - filter(some_filter): Fetch records based on `some_filter`
        - count(some_filters): Count records based on `some_filter`
        - properties: Show all acceptable attributes

    Examples of filter usage:
        - filter(age>30)
        - filter(age<20 and run_distance>500)

    Available data properties:
        ramo
        gender
        codicerischio
        descrizionerischio
        provincecustomer
        sinistri
        infortuniprecedenti
        longitudepoliza
        longitudecustomer
        adresspolizza
        adresscustomer
        blacklist
```

Figure 13: Anonymiser output

By clicking Preview and Risk analysis the user is able to assess the risk levels of his anonymization mapping as shown in the following interface:



Preview and Risk Analysis

Prosecutor Attacker Model

Identifiers

polizzaid firstname email lastname mobile longitudecustomer infortuniprecedenti sinistri latitudecustomer codicerischio citycustomer provincecustomer longitudepoliza adresspolizza

Quasi Identifiers

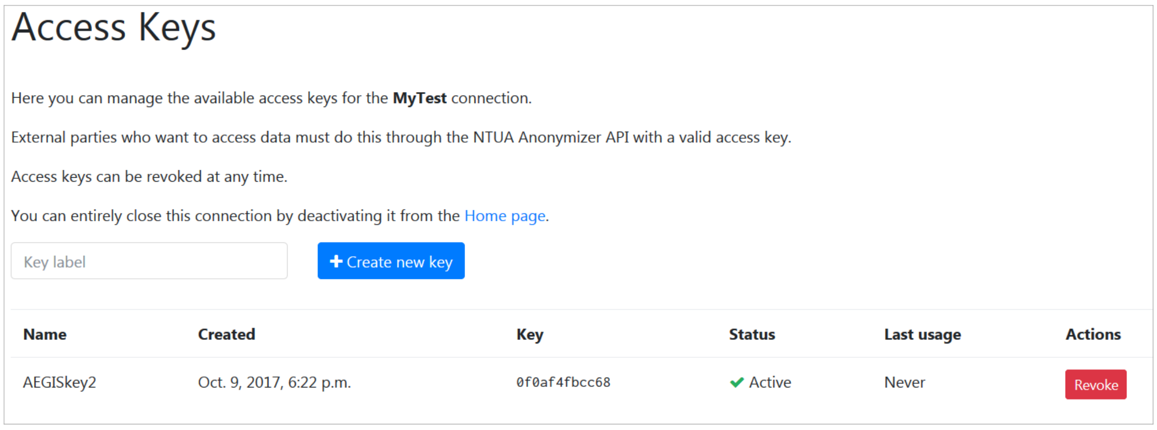
citypolizza provincepolizza blacklist aia gender

Insensitive Data

cerved cf AgeRange bds latitudepoliza AgeSum prodotto

Figure 14: Preview and Risk analysis

The anonymised (output) data can be exposed through API to external parties in a secure way through the provision of access keys.



Access Keys

Here you can manage the available access keys for the **MyTest** connection.

External parties who want to access data must do this through the NTUA Anonymizer API with a valid access key.

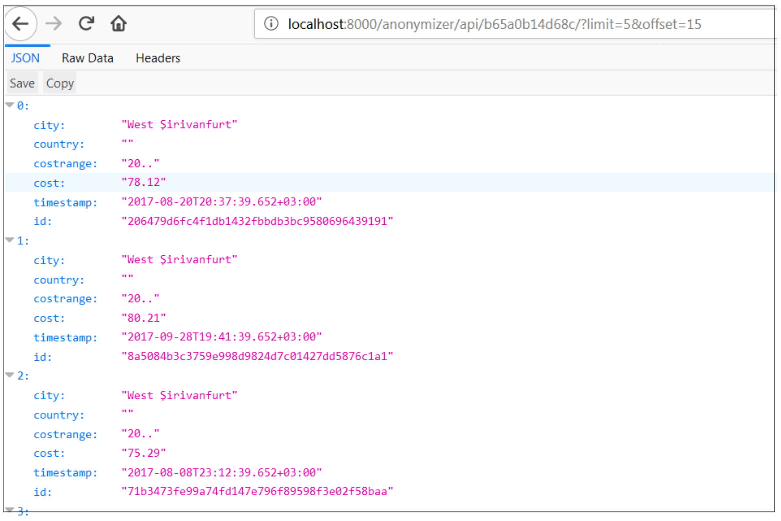
Access keys can be revoked at any time.

You can entirely close this connection by deactivating it from the Home page.

Name	Created	Key	Status	Last usage	Actions
AEGISkey2	Oct. 9, 2017, 6:22 p.m.	0f0af4fbcc68	✓ Active	Never	Revoke

Figure 15: Anonymiser exposed API

Users can access the anonymised data in JSON format through API provided by the anonymization tool using their private access keys.


localhost:8000/anonymizer/api/b65a0b14d68c/?limit=5&offset=15
JSON
Raw Data
Headers
Save
Copy

```
0:
    city:        "West Şirivanfurt"
    country:     ""
    costrange:   "20.."
    cost:        "78.12"
    timestamp:   "2017-08-20T20:37:39.652+03:00"
    id:          "206479d6fc4f1db1432fbbdb3bc9580696439191"
1:
    city:        "West Şirivanfurt"
    country:     ""
    costrange:   "20.."
    cost:        "80.21"
    timestamp:   "2017-09-28T19:41:39.652+03:00"
    id:          "8a5084b3c3759e998d9824d7c01427dd5876c1a1"
2:
    city:        "West Şirivanfurt"
    country:     ""
    costrange:   "20.."
    cost:        "75.29"
    timestamp:   "2017-08-08T23:12:39.652+03:00"
    id:          "71b3473fe99a74fd147e796f89598f3e02f58baa"
3:
```

Figure 16: Anonymiser API response

2.5. Brokerage Engine

The AEGIS Brokerage engine has been deployed inside the cluster machines of the AEGIS platform to overcome security and interconnection constrains that were present in previous deployments. With this configuration, the other future nodes of the network can be installed in the AEGIS clusters to be deployed.

The main API methods that were described in the D3.x deliverables have been revised accordingly, and have been implemented using Java, in order to be able for the Brokerage Engine to interconnect to the underlying HOPS engine and invocations will be done at the UI level, to ensure that the calls to the blockchain remain independent from the underlying infrastructure, to allow for better maintenance and updating procedures in both the HOPS backend, and the blockchain network infrastructure.

Those developed calls are the following:

User Management in the Brokerage Engine

- Create new user - Creates a user with user id (uid in the code snippet below) john.doe@example.com , having an initial balance of 1000 tokens:

```
curl -XPOST https://localhost:3000/api/User -H 'Accept: application/json' -d '{ "$class": "eu.aegis.User", "uid": "john.doe@example.com", "balance": "1000.0" }'
```

- Get all users

```
curl -XGET https://localhost:3000/api/User
```

- Get specific user's details - To get a specific user's information, add the uid at the end of the url. i.e.:

```
curl -XGET https://localhost:3000/api/User/john.doe@example.com
```

- Load tokens to a user's account - To load 200 tokens to (user with uid) john.doe@example.com :

```
curl -XGET https://localhost:3000/api/LoadBalance -H 'Accept: application/json' -d '{
"user": "resource:eu.aegis.User#john.doe@example.com", "amount": 200 }'
```

Asset Management

- Create a new asset To create an asset, you need to provide an asset id (aid below), its type (can be any of: Dataset , Microservice , Algorithm , Analysis , Visualization , Other), the cost of the asset, its status (can be Free , Paid , Subscription or Private - i.e. not for sale) and its owner (the uid of the user that owns the dataset).

```
curl -XPOST https://localhost:3000/api/AEGISAsset -H 'Accept: application/json' -d '{
"$class": "eu.aegis.AEGISAsset", "aid": "1", "assetType": "Dataset", "cost": 100,
"status": "Paid", "owner": "marios@suite5.eu" }'
```

- Get all assets

```
curl -XGET https://localhost:3000/api/AEGISAsset
```

- Get details on a specific asset - As with user, we use the aid as a parameter

```
curl -XGET https://localhost:3000/api/AEGISAsset/1
```

- Change the status of an asset - To change the status of asset with aid=1 to Private , use:

```
curl -XPOST https://localhost:3000/api/ChangeAssetStatus -H 'Accept:
application/json' -d '{

"relatedAsset": "eu.aegis.AEGISAsset#1", "newStatus": "Private" }'
```

- Change the cost of an asset - To change the status of asset with aid=1 to Private , use:

```
curl -XPOST https://localhost:3000/api/ChangeAssetCost -H 'Accept: application/json'
d '{"relatedAsset": "eu.aegis.AEGISAsset#1", "newCost": "200" }'
```

Building Contracts

- Create a new (draft) contract - To create a contract, you need to provide a transaction id (tid), the desired exclusivity (can be None , Subscription or Lifetime), the amount (to be) paid, the status of the contract (always use Draft at this stage), a text containing the terms of the contract (optional), the buyer and the seller (using their uid s) and the asset to be sold (use its aid).

```
curl -XPOST https://localhost:3000/api/Contract -H 'Accept: application/json' -d '{
"$class": "eu.aegis.Contract", "tid": "123", "exclusivity": "None", "amountPaid":
"100.0", "status": "Draft", "text": "string", "seller": "marios@suite5.eu", "buyer":
"john.doe@example.com", "relatedAsset": "eu.aegis.AEGISAsset#1" }'
```

- Get all contracts

```
curl -XGET https://localhost:3000/api/Contract
```

- Get all contracts involving assets bought/sold by someone - To get all contracts with a specific buyer/seller we need to filter the contracts. This is done by appending a (url encoded) filter in the url. For example, to get all contracts involving john.doe@example.com as a buyer, the filter should be:

```
{"where": {"buyer": {"eq":"resource:eu.aegis.User#john.doe@example.com"}}}
```

As a seller:

```
{"where": {"seller": {"eq":"resource:eu.aegis.User#john.doe@example.com"}}}

curl -XGET https://localhost:3000/api/Contract?filter={...}
```

The url encoded example of contracts with assets bought by john.doe@example.com is

```
curl -XGET
https://localhost:3000/api/Contract?filter=%7B%22where%22%3A%20%7B%22buy
er%
22%3A%20%7B%22eq%22%3A%22resource%3Aeu.aegis.User%23john.doe%40exa
mple.com%22%7D%7D%7D
```

- Validate a contract - To validate (switch it from Draft it Active):

```
curl -XPOST https://localhost:3000/api/ValidateContract -H 'Accept: application/json'
-d '{ "contract": "eu.aegis.Contract#123" }'
```

where 123 is the tid (as set in the draft contract creation).

2.6. AEGIS Integrated Services

The AEGIS platform provides data management and processing, user management, and service monitoring through the use of Hopsworks integrated services. Hopsworks introduces the notion of Project, Dataset, and User to enable multi-tenancy within the context of data management. Data processing includes data parallel processing services such as MapReduce, Spark, and Flink, as well as interactive analytics using notebooks such as Jupyter. Full-text search capability is offered by the included ELK stack component (Elasticsearch). Real time analytics is enabled by the use of the included Kafka service. A Dataset is a directory in the underlying file system (HopsFS) that could contains as many files and directories, and it could be shared between projects. A Project is a collection of datasets, notebooks, users, and jobs (Flink, Spark, Flink). A User in AEGIS can create multiple project, share projects with other users, create datasets, and upload/download/preview data in his/her dataset. In the following subsections, we show the main components provided by AEGIS.

Users API

A new user of the platform is required to first register with AEGIS, by providing: first name, last name, email, password and a security question/answer as we can see in the figure below.

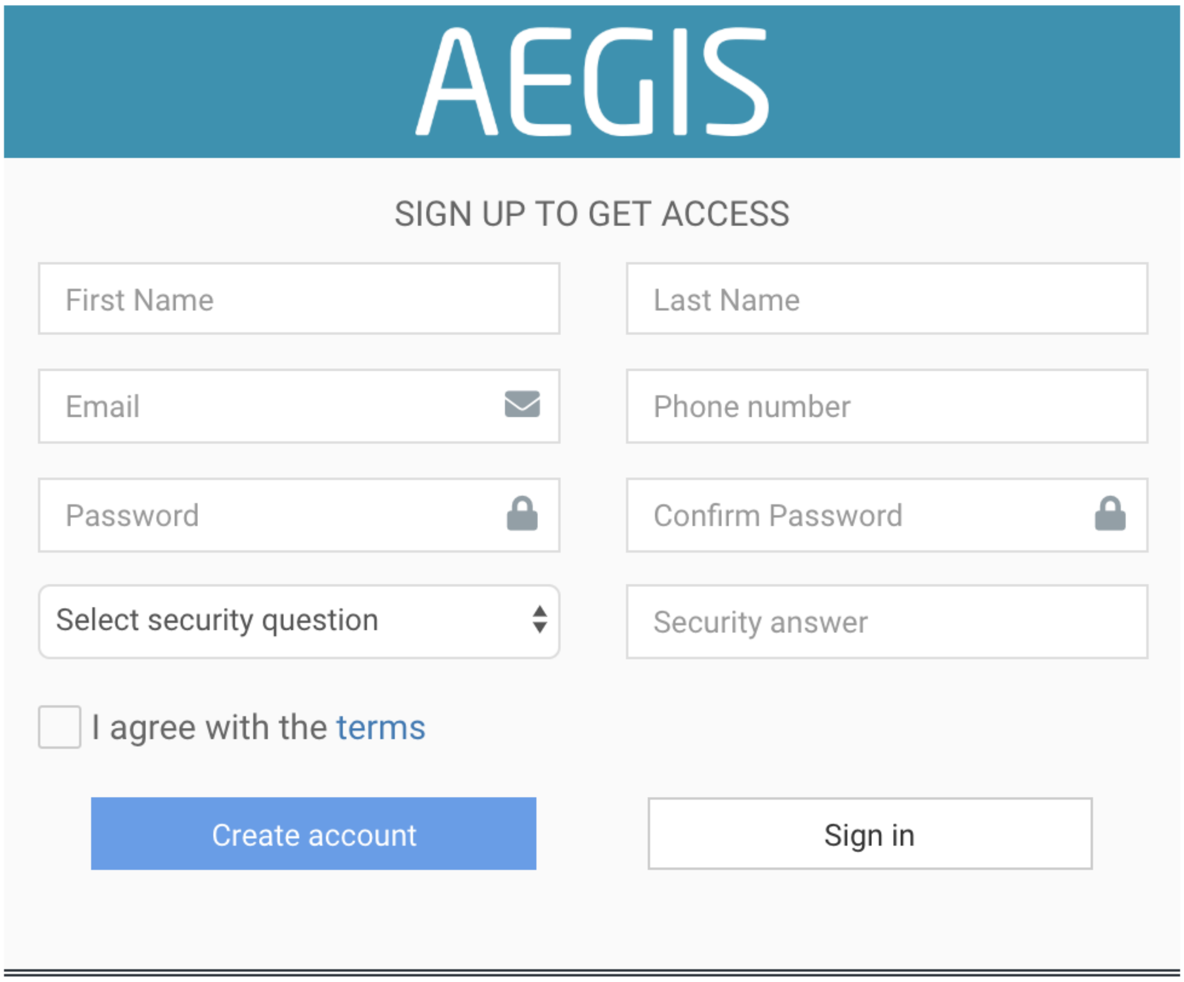


Figure 17: Registration page

Once the user creates an account, an AEGIS administrator can search and validate the new account with the appropriate roles using the admin user interface. Once the account is active, the user can log in with his/her credentials on the AEGIS platform.

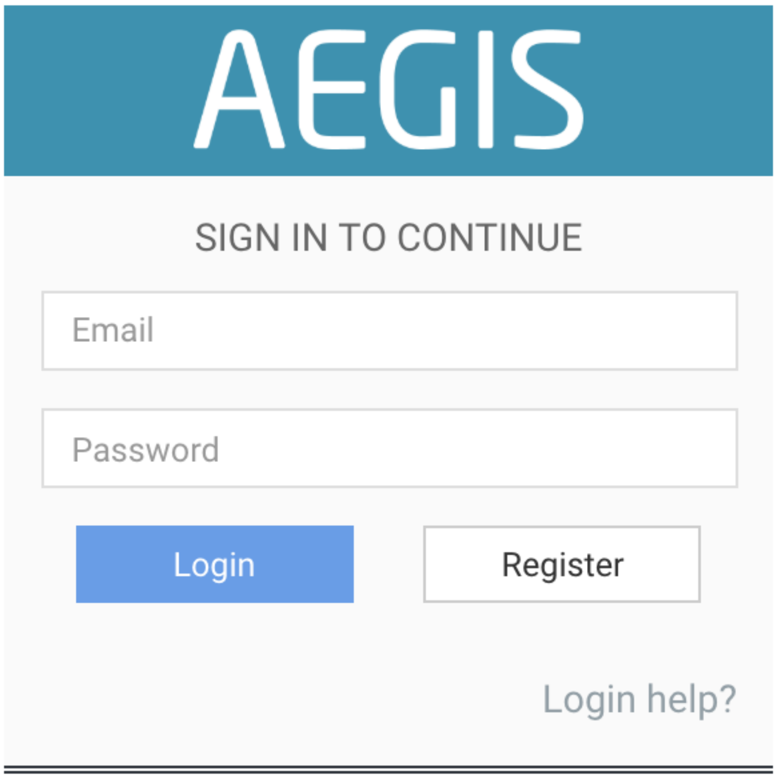


Figure 18: Login page

Projects and Datasets

Once the user is logged in, he/she can start using the AEGIS platform by first creating a project by clicking on the New Project button as shown in Figure 19.

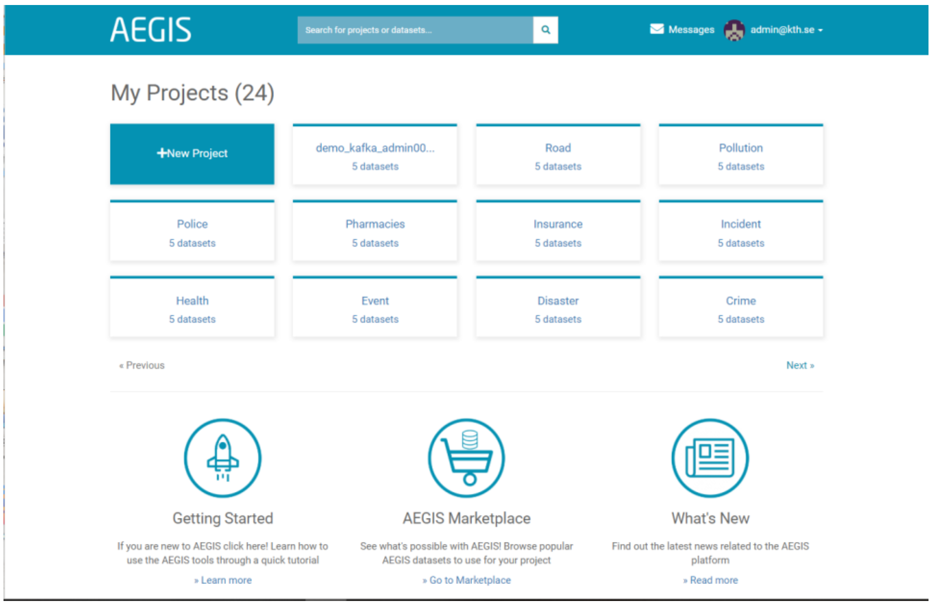


Figure 19: Projects page

Then, the user can navigate to the project 'MyProject' where a list of existing datasets is provided, as shown.

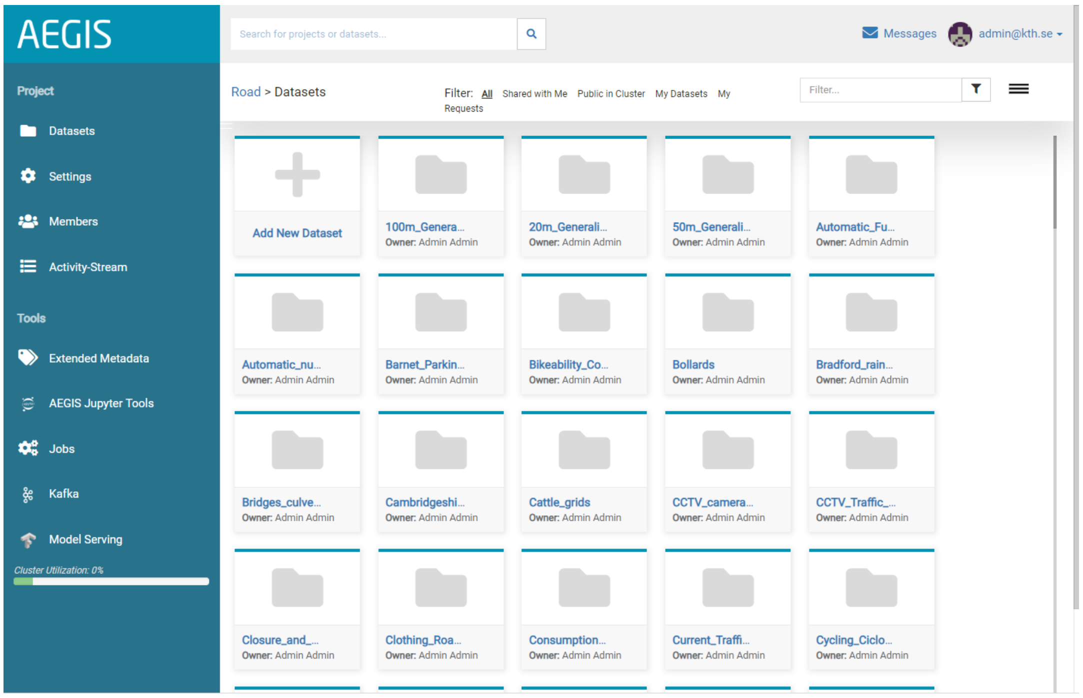

AEGIS
Search for projects or datasets...
Messages
admin@kth.se
Project
Datasets
Settings
Members
Activity-Stream
Tools
Extended Metadata
AEGIS Jupyter Tools
Jobs
Kafka
Model Serving
Cluster Utilization: 0%
Road > Datasets
Filter: All Shared with Me Public in Cluster My Datasets My Requests
Filter...
Add New Dataset
100m_Genera... Owner: Admin Admin
20m_Generali... Owner: Admin Admin
50m_Generali... Owner: Admin Admin
Automatic_Fu... Owner: Admin Admin
Automatic_nu... Owner: Admin Admin
Barnet_Parkin... Owner: Admin Admin
Bikeability_Co... Owner: Admin Admin
Bollards Owner: Admin Admin
Bradford_rain... Owner: Admin Admin
Bridges_culve... Owner: Admin Admin
Cambridgeshi... Owner: Admin Admin
Cattle_grids Owner: Admin Admin
CCTV_camera... Owner: Admin Admin
CCTV_Traffic_... Owner: Admin Admin
Closure_and_... Owner: Admin Admin
Clothing_Roa... Owner: Admin Admin
Consumption... Owner: Admin Admin
Current_Traffi... Owner: Admin Admin
Cycling_Ciclo... Owner: Admin Admin

Figure 20: Datasets page

Inside the project, on the left bar there are tools that the users can use to interact/process their data such as AEGIS Jupyter tools. To start interacting with these tools, the users first create their own datasets, then upload their data inside. Inside the dataset, the users can create folders, upload files, and then navigate through the created structure.

Jobs

The AEGIS platform allows users to run their MapReduce, Spark, and Flink jobs on the platform using the Jobs service. To run a job, the users need to first upload their job files (.jar, .py, .ipynb) to a dataset. Then, they can add a new job by clicking on the New job button as shown in the following figure.

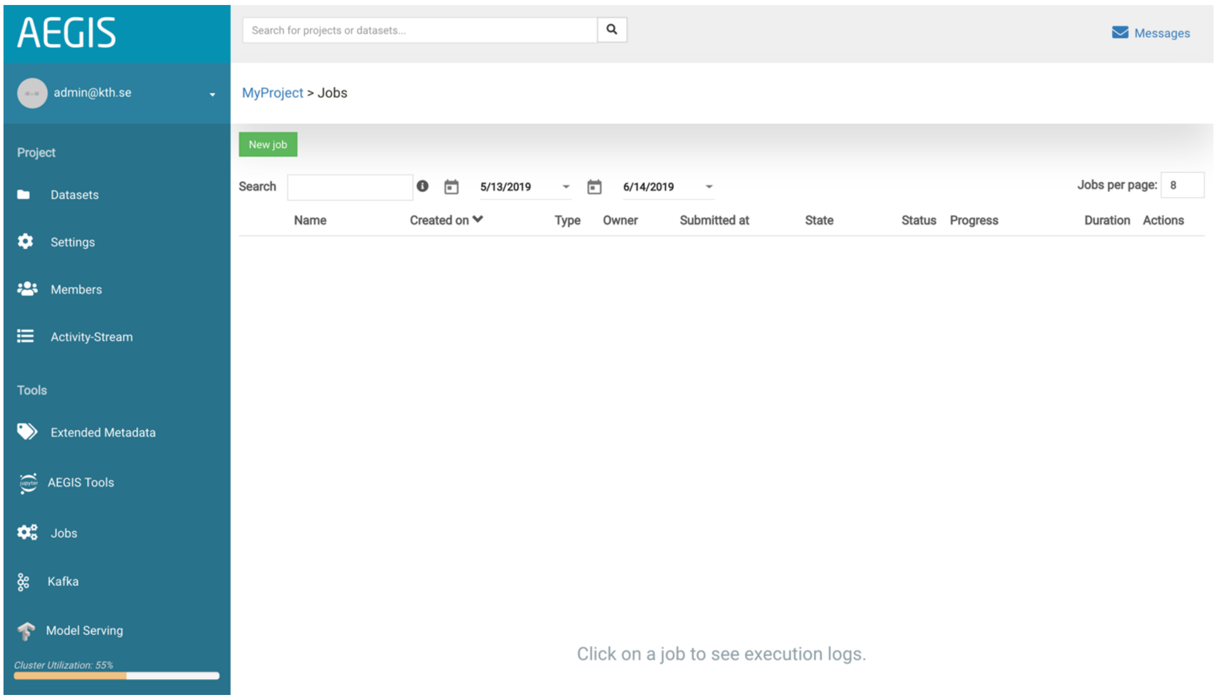
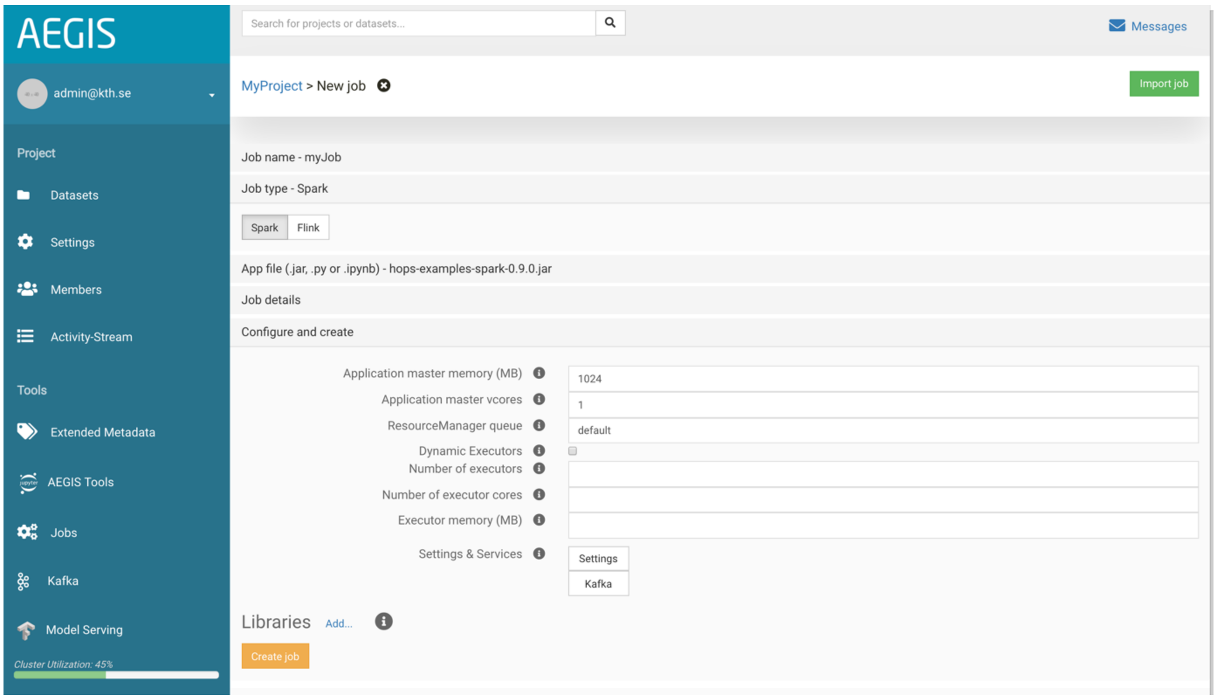


Figure 21: Jobs page (1)

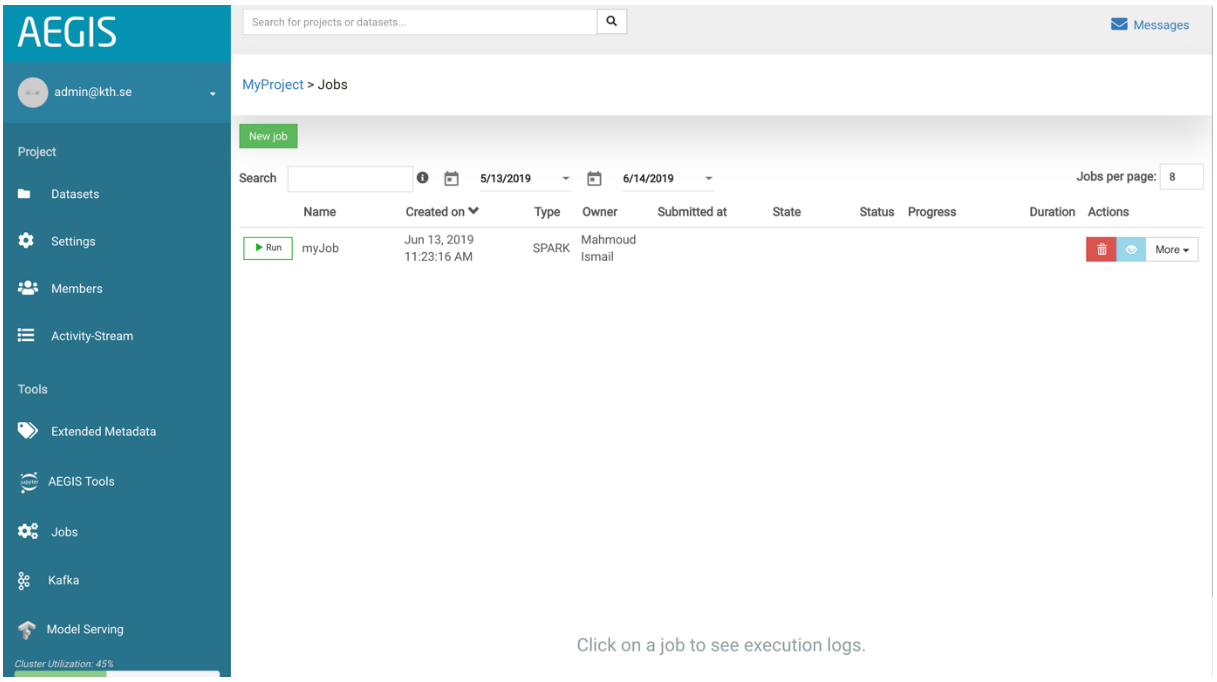


Job name - myJob
Job type - Spark
Spark Flink
App file (.jar, .py or .ipynb) - hops-examples-spark-0.9.0.jar
Job details
Configure and create

Application master memory (MB)	1024
Application master vcores	1
ResourceManager queue	default
Dynamic Executors	☐
Number of executors	
Number of executor cores	
Executor memory (MB)	
Settings & Services	Settings Kafka

Figure 22: Jobs page (2)

Then once the job is created, the users can run, stop, delete, copy, export, and edit the job. Also, they can look at the execution logs listed at the end of the page.



	Name	Created on	Type	Owner	Submitted at	State	Status	Progress	Duration	Actions
Run	myJob	Jun 13, 2019 11:23:16 AM	SPARK	Mahmoud Ismail						More

Figure 23: Jobs page (3)

Kafka

The AEGIS platform provides Kafka service for real time analytics. The user interface allows users to create schemas for their topics, then create associated topics that will be used by applications for writing/reading from/to these topics. The user can create a Kafka topic by clicking on the “New Topic” button as shown in the following figure.

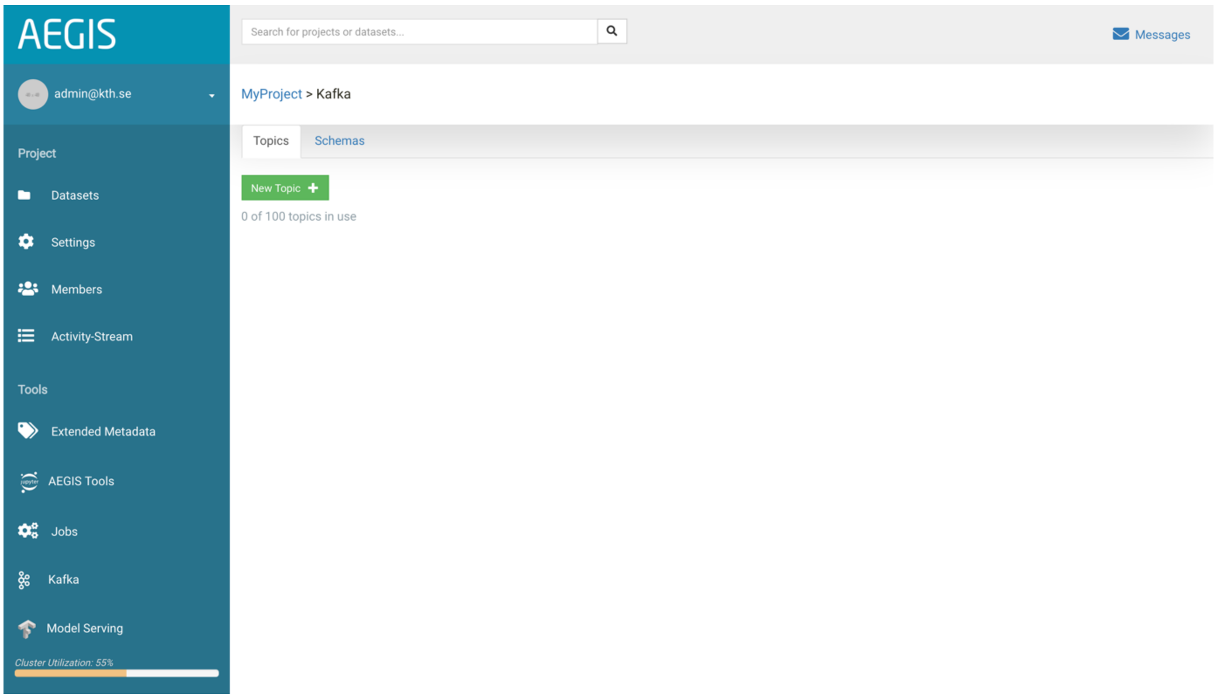
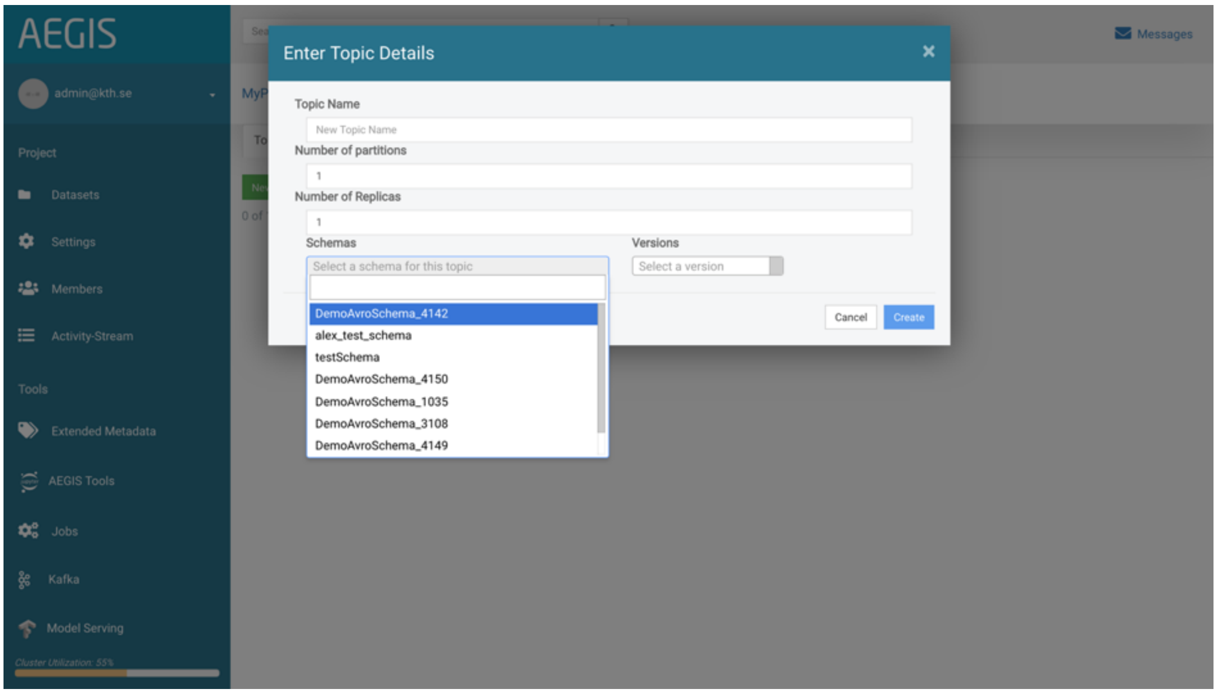


Figure 24: Kafka main page

After clicking on the new topic button, the user is required to fill the topic details such as topic name, number of partitions, number of replicas, and the topic schema. The user can select a schema from the existing schema in the platform or create his own schema first by navigating to the schemas tab.

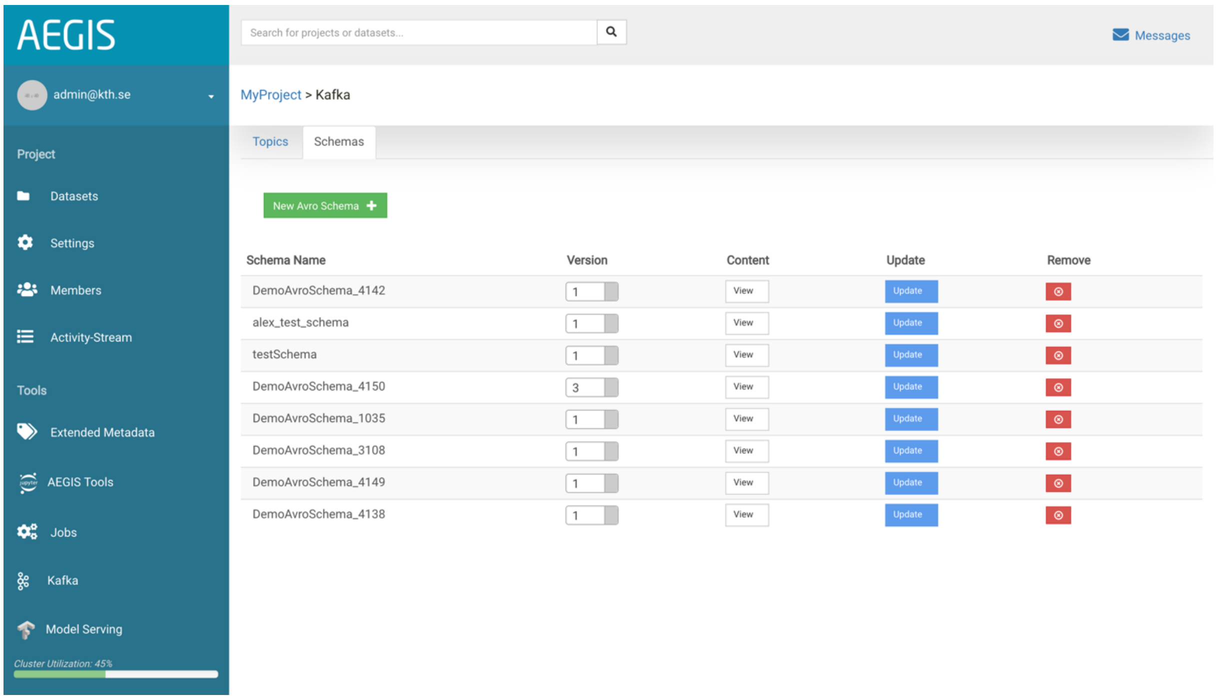

Enter Topic Details

Topic Name: New Topic Name
Number of partitions: 1
Number of Replicas: 1
Schemas: Select a schema for this topic
Versions: Select a version

- DemoAvroSchema_4142
- alex_test_schema
- testSchema
- DemoAvroSchema_4150
- DemoAvroSchema_1035
- DemoAvroSchema_3108
- DemoAvroSchema_4149

Figure 25: New Kafka Topic

The user can create his/her own schema by clicking on the new avro schema button and then provide the schema's details.



AEGIS

Search for projects or datasets...

Messages

admin@kth.se

MyProject > Kafka

Project: Datasets, Settings, Members, Activity-Stream

Tools: Extended Metadata, AEGIS Tools, Jobs, Kafka, Model Serving

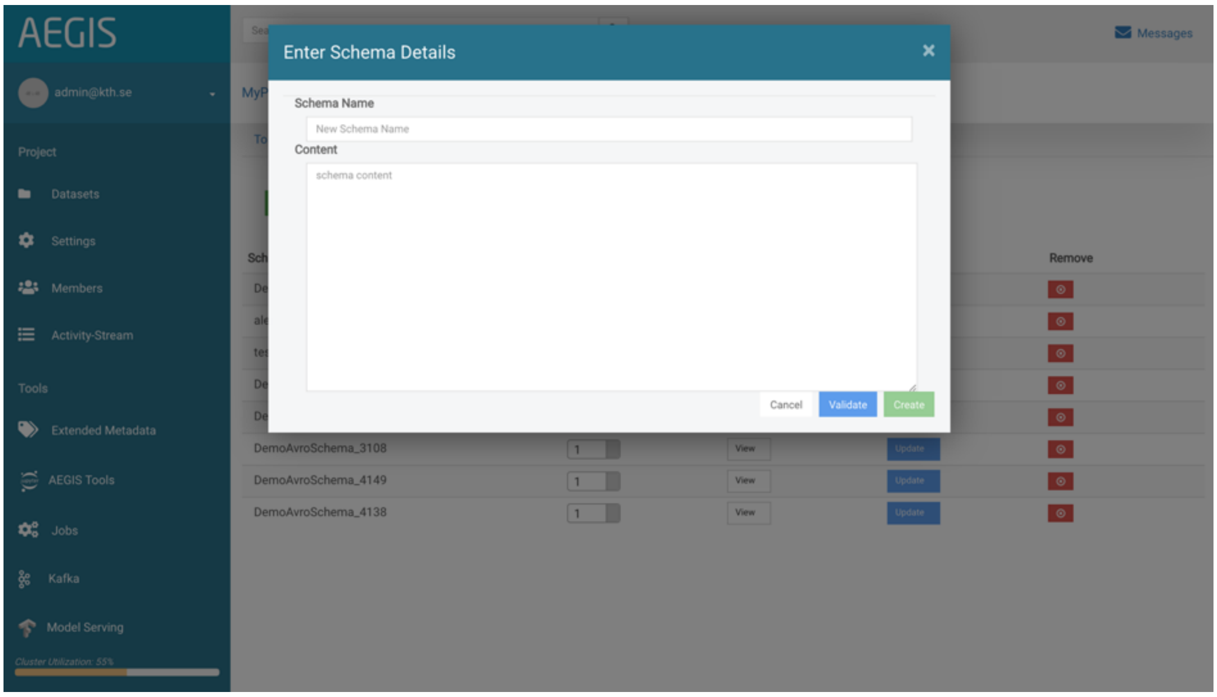
Cluster Utilization: 45%

Topics | Schemas

New Avro Schema

Schema Name	Version	Content	Update	Remove
DemoAvroSchema_4142	1	View	Update	⊗
alex_test_schema	1	View	Update	⊗
testSchema	1	View	Update	⊗
DemoAvroSchema_4150	3	View	Update	⊗
DemoAvroSchema_1035	1	View	Update	⊗
DemoAvroSchema_3108	1	View	Update	⊗
DemoAvroSchema_4149	1	View	Update	⊗
DemoAvroSchema_4138	1	View	Update	⊗

Figure 26: Kafka page (1)



Enter Schema Details

Schema Name: New Schema Name

Content: schema content

Cancel | Validate | Create

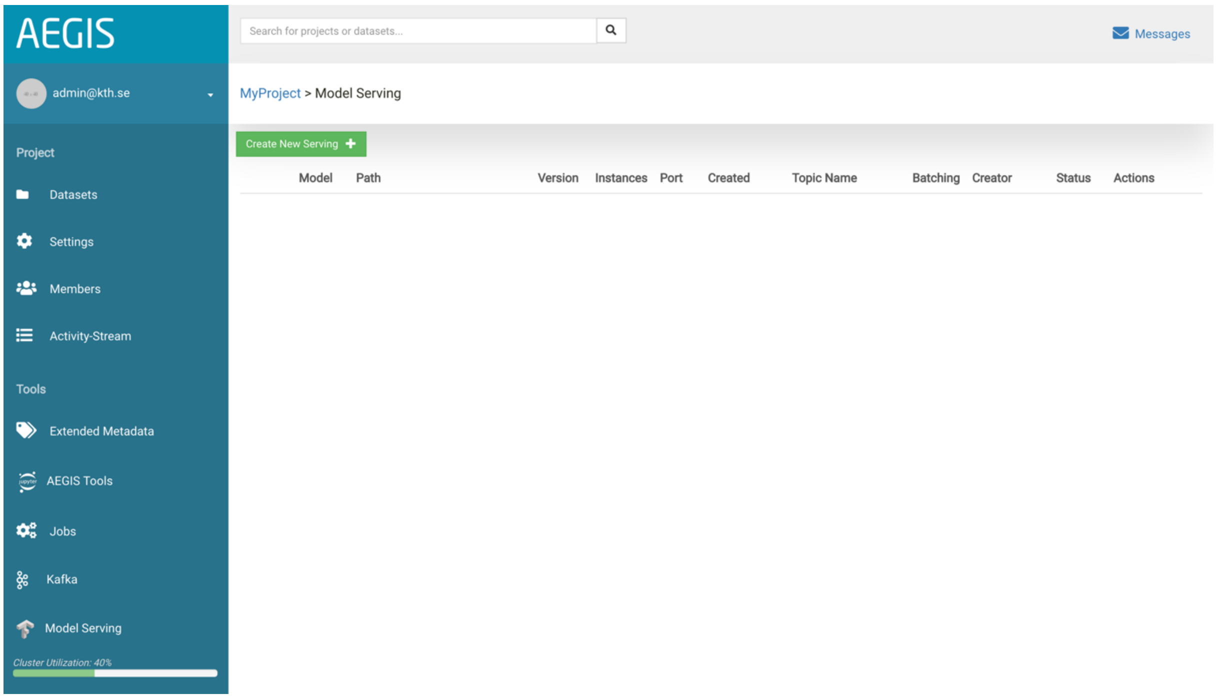
Figure 27: New Kafka Schema

After creating the topic and the schema, the users can then build and run their own applications that interact with the Kafka service using the hops-util (https://github.com/logicalclocks/hops-util) for Java/Scala applications and hops-util-py (https://github.com/logicalclocks/hops-util-py) for Python applications. The hops-util and hops-util-py libraries abstract away all the configuration boilerplate code such as Kafka endpoints, topics etc allowing users to easily build their applications.

Tensor Flow

TensorFlow is an ecosystem for developing deep learning models, containing all the tools from building to deployment. TensorFlow has 3 main components:

- TensorFlow(API) -contains the API's to define the models and train the models with the data
- TensorBoard -helps to analyse, visualize, and debug TensorFlow graphs.
- TensorFlow Serving -helps to deploy the pre-trained models.



Model	Path	Version	Instances	Port	Created	Topic Name	Batching	Creator	Status	Actions

Figure 28: TensorFlow Main page

The AEGIS platform provides a model serving service to deploy pre trained models using the Tensorflow serving. The user can create a new serving by clicking on the “Create New Serving” button and then providing a model name, version, and a path as shown in the following figure.

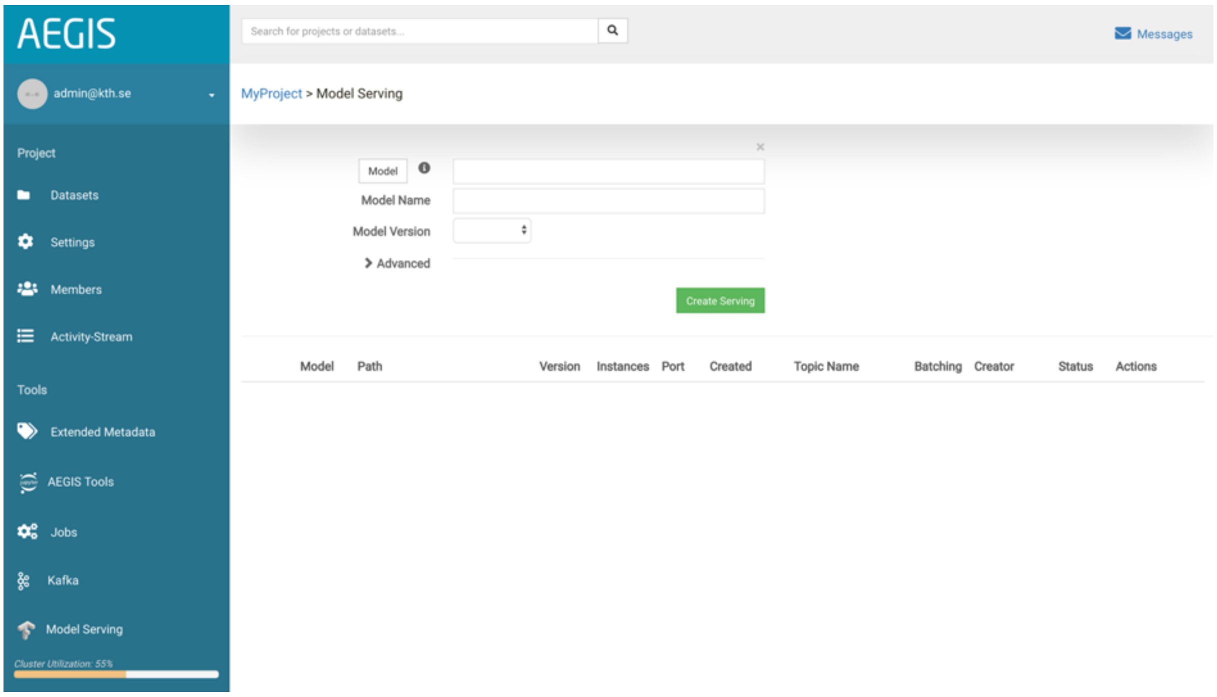
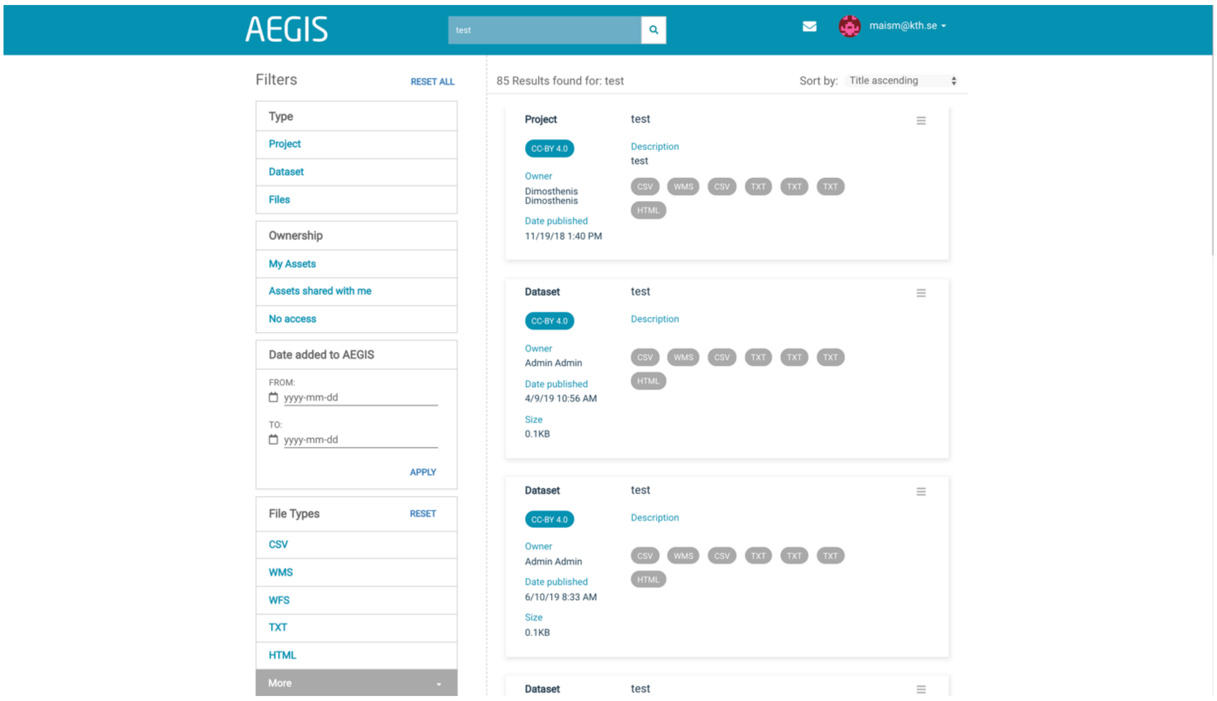


Figure 29: New Model Serving

Full-text Search

Elasticsearch, one of the ELK stack components, is used to provide full text search capabilities to explore projects and datasets within the AEGIS platform. The search space available to each of the users depends on the context from which the user searches. For example, within the context of the home page, the search space includes public datasets, projects, and private datasets. When inside the project, the scope of the search is reduced to the project's datasets including the shared datasets from other projects. After doing the search, the users can filter the search results using the filters on the right side.



85 Results found for: test

Project	test
CC-BY 4.0	Description: test
Owner	Dimosthenis Dimosthenis
Date published	11/19/18 1:40 PM

Dataset	test
CC-BY 4.0	Description
Owner	Admin Admin
Date published	4/9/19 10:56 AM
Size	0.1KB

Dataset	test
CC-BY 4.0	Description
Owner	Admin Admin
Date published	6/10/19 8:33 AM
Size	0.1KB

Dataset test

Figure 30: Full text search page

2.7. Metadata Service

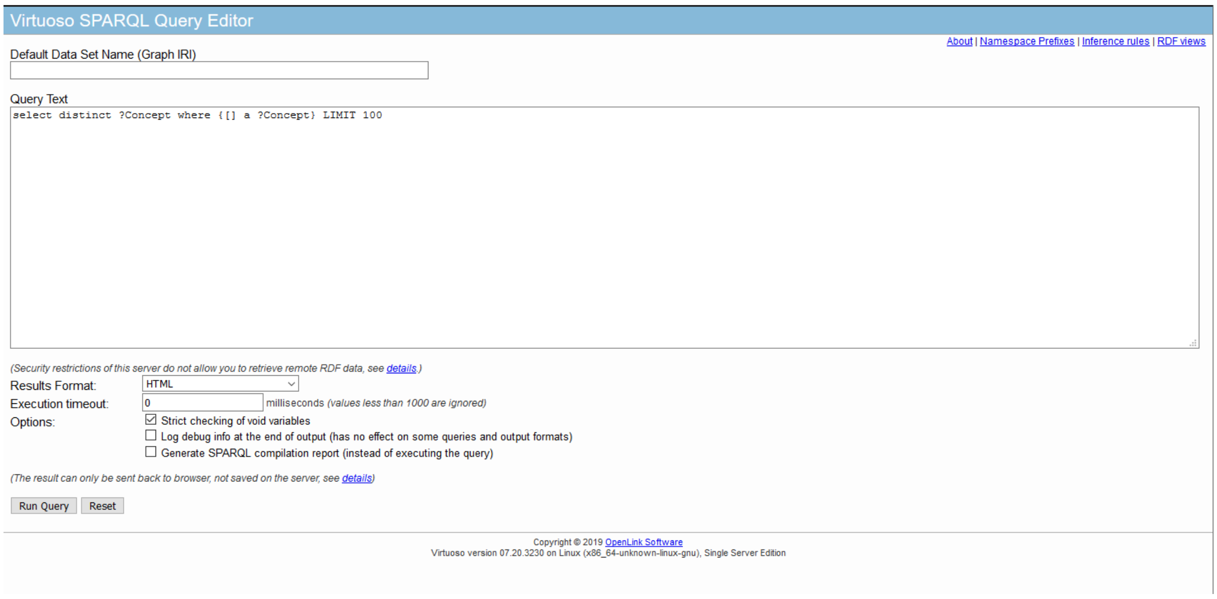
The Metadata Service is responsible for storing the metadata associated with a particular dataset within the AEGIS platform. This metadata poses the foundation of the processing of the data within the AEGIS platform. It is based on the AEGIS ontology and vocabulary.

Triplestore

The foundation of the Metadata Service is the Virtuoso Triplestore. It can be directly accessed here:

https://platform.aegis-bigdata.eu/hopsworks-api/virtuoso/

It offers multiple standardised Linked Data interfaces, like SPARQL or the Graph Store HTTP Protocol. These interfaces can be utilised from other components or users of the AEGIS platform. Figure 31 shows the SPARQL interface of Virtuoso.



Virtuoso SPARQL Query Editor

About | Namespace Prefixes | Inference rules | RDF views

Default Data Set Name (Graph IRI)

Query Text

```
select distinct ?Concept where {[] a ?Concept} LIMIT 100
```

(Security restrictions of this server do not allow you to retrieve remote RDF data, see details.)

Results Format: HTML

Execution timeout: 0 milliseconds *(values less than 1000 are ignored)*

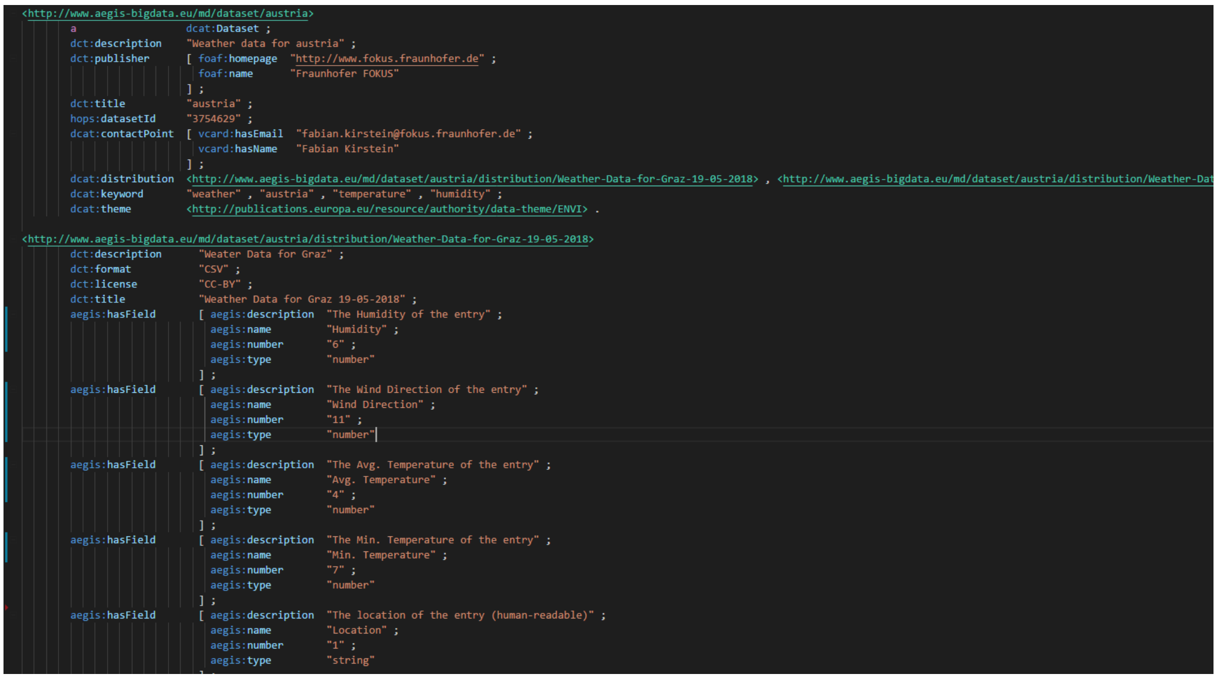
Options:
- [x] Strict checking of void variables
- [ ] Log debug info at the end of output (has no effect on some queries and output formats)
- [ ] Generate SPARQL compilation report (instead of executing the query)

(The result can only be sent back to browser, not saved on the server, see details)

Copyright © 2019 OpenLink Software
Virtuoso version 07.20.3230 on Linux (x86_64-unknown-linux-gnu), Single Server Edition

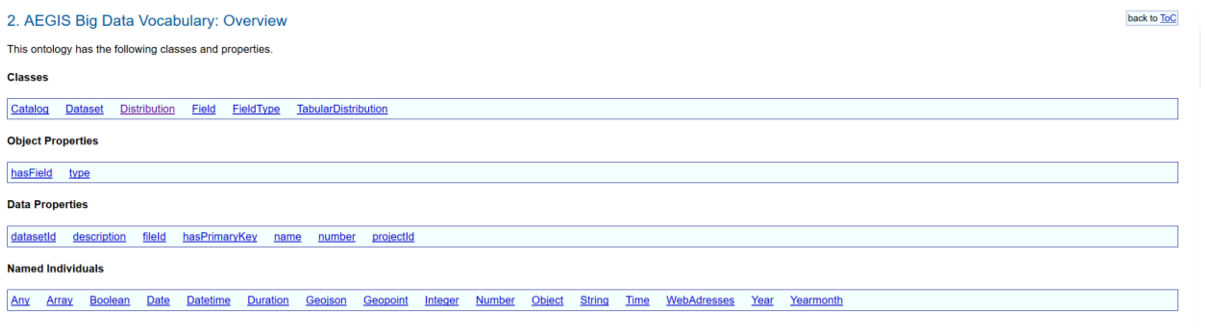
Figure 31: SPARQL User Interface of Virtuoso

Virtuoso only acts as a storage layer and is not supposed to be accessed directly by the users or any other component, with an exception to the SPARQL interface, which can be used for executing complex queries against the metadata of the AEGIS platform. Therefore, the AEGIS ontologies are publicly available: http://aegis.fokus.fraunhofer.de/. Figure 32 shows an example for the metadata stored in the triplestore. In Figure 33 an extract of the AEGIS ontology documentation can be seen.



```
<http://www.aegis-bigdata.eu/md/dataset/austria>
        a                   dcat:Dataset ;
        dct:description     "Weather data for austria" ;
        dct:publisher       [ foaf:homepage  "http://www.fokus.fraunhofer.de" ;
                              foaf:name      "Fraunhofer FOKUS"
                            ] ;
        dct:title           "austria" ;
        hops:datasetId      "3754629" ;
        dcat:contactPoint   [ vcard:hasEmail  "fabian.kirstein@fokus.fraunhofer.de" ;
                              vcard:hasName   "Fabian Kirstein"
                            ] ;
        dcat:distribution   <http://www.aegis-bigdata.eu/md/dataset/austria/distribution/Weather-Data-for-Graz-19-05-2018> , <http://www.aegis-bigdata.eu/md/dataset/austria/distribution/Weather-Dat
        dcat:keyword        "weather" , "austria" , "temperature" , "humidity" ;
        dcat:theme          <http://publications.europa.eu/resource/authority/data-theme/ENVI> .

<http://www.aegis-bigdata.eu/md/dataset/austria/distribution/Weather-Data-for-Graz-19-05-2018>
        dct:description      "Weater Data for Graz" ;
        dct:format           "CSV" ;
        dct:license          "CC-BY" ;
        dct:title            "Weather Data for Graz 19-05-2018" ;
        aegis:hasField       [ aegis:description  "The Humidity of the entry" ;
                               aegis:name         "Humidity" ;
                               aegis:number       "6" ;
                               aegis:type         "number"
                             ] ;
        aegis:hasField       [ aegis:description  "The Wind Direction of the entry" ;
                               aegis:name         "Wind Direction" ;
                               aegis:number       "11" ;
                               aegis:type         "number"
                             ] ;
        aegis:hasField       [ aegis:description  "The Avg. Temperature of the entry" ;
                               aegis:name         "Avg. Temperature" ;
                               aegis:number       "4" ;
                               aegis:type         "number"
                             ] ;
        aegis:hasField       [ aegis:description  "The Min. Temperature of the entry" ;
                               aegis:name         "Min. Temperature" ;
                               aegis:number       "7" ;
                               aegis:type         "number"
                             ] ;
        aegis:hasField       [ aegis:description  "The location of the entry (human-readable)" ;
                               aegis:name         "Location" ;
                               aegis:number       "1" ;
                               aegis:type         "string"
```

Figure 32: AEGIS Linked Data Example



2. AEGIS Big Data Vocabulary: Overview

back to ToC

This ontology has the following classes and properties.

Classes

Catalog Dataset Distribution Field FieldType TabularDistribution

Object Properties

hasField type

Data Properties

datasetId description fileId hasPrimaryKey name number projectId

Named Individuals

Any Array Boolean Date Datetime Duration Geojson Geopoint Integer Number Object String Time WebAdresses Year Yearmonth

Figure 33: Extract of the AEGIS Ontology documentation

Metadata Service

The Triplestore only offers (complex) Linked Data interfaces and no rich management functionalities. Therefore, an additional service is required, providing additional functionalities for the management of the metadata. This includes particularly the straight-forward creation of metadatasets. A first prototype is available here:

https://platform.aegis-bigdata.eu/hopsworks-api/aegis-metadata/

It offers the following core functionalities regarding the management of Linked Data-based metadata:

URI Management
The Metadata Service automatically provides consistent and comprehensive URIs for all entities upon creation time. Those will follow best-practices, especially promoted by the Semantic Interoperability Community (SEMIC) of the European Commission.

Access Control
Datasets may be not public and therefore the metadata needs to be protected too. RDF does not natively support access control. The Metadata Service enables the creation of hidden entities, which are not available via the SPARQL endpoint.

Multilingual Capacities
RDF natively supports the provision of multilingual literals. This feature is harnessed appropriately by adding support for it to the RESTful API of the Metadata Service and the integration of an automatic machine translation service.

Data Store Synchronisation
By definition the metadata refers to actual data in the AEGIS Data Store. Since data and metadata are managed by distinct service, a reliable synchronisation and connection is required. The Metadata Service is hooked into the AEGIS Data Store and enables a synchronisation of the metadata with the actual data.

Recommendation Service
The Metadata Service offers a recommendation service for querying similar or related data for a given dataset or file.

Thesauri and Vocabulary Management
The service integrates the LinDA Vocabulary and Metadata Repository for simplifying the metadata provision through auto-completion.

The Metadata Service interacts with the Virtuoso triplestore and offers a REST-API for creating, deleting and updating metadata. It suggests suitable and similar datasets based on an input dataset. Therefore, several characteristics of the dataset are matched against the stored metadata, e.g. keywords or the semantic tabular information. For future releases this feature will be extended and improved. The metadata service is developed in Java, based on the Vert.x framework.

Translation Middleware

The central task of the translation service is to translate the headings and descriptions of data sets of the metadata into all 24 EU languages. The description of a dataset, for example, is usually only available in one language and requires at least 24 translation requests. This service comprises a middleware for collecting these translation requests, which are bundled and sent to eTranslation - the translation service of the EU Commission. Because of the technical interface of eTranslation, it is necessary to bundle requests and re-assign them to their original texts. By the high amount of data is also important that the requests are processed over a specific period of time, because eTranslation does not translate many gigabytes of data immediately. The translation service also provides configuration parameters for this task. All requests and finished translations are always stored in a database management system (PostgreSQL) for reasons of failure safety. Communication with the external service handled via an encrypted HTTPS method (via JSON) so that no one can access the translations.

Integration

The Metadata Services requires tight integration into the AEGIS platform, since the metadata is present throughout the entire data value chain, from providing until visualizing data. For creating the metadata, the Metadata Service was integrated into the AEGIS frontend via the Data Annotator.

2.8. Query Builder

Query Builder provides the capability to interactively define and execute queries on data available in the AEGIS system. It is primarily addressed to the AEGIS users with limited technical background, but potentially useful for all, as it simplifies and accelerates the process of retrieving data and creating views on them. As explained also in Section 2.3, Query Builder also offers some simple data cleansing functionalities.

As already stated in previous revisions of this document, the name of the component may be misleading, since the word query can be interpreted as the part of the data value chain responsible for retrieving the appropriate data. However, when trying to build the dataset on which an analysis or a visualisation will be applied, the workflow to extract the meaningful parts of the data may include certain processing tasks that cannot be known a priori, in terms of filtering and cleansing. Therefore, and in order to leverage the computational power of the

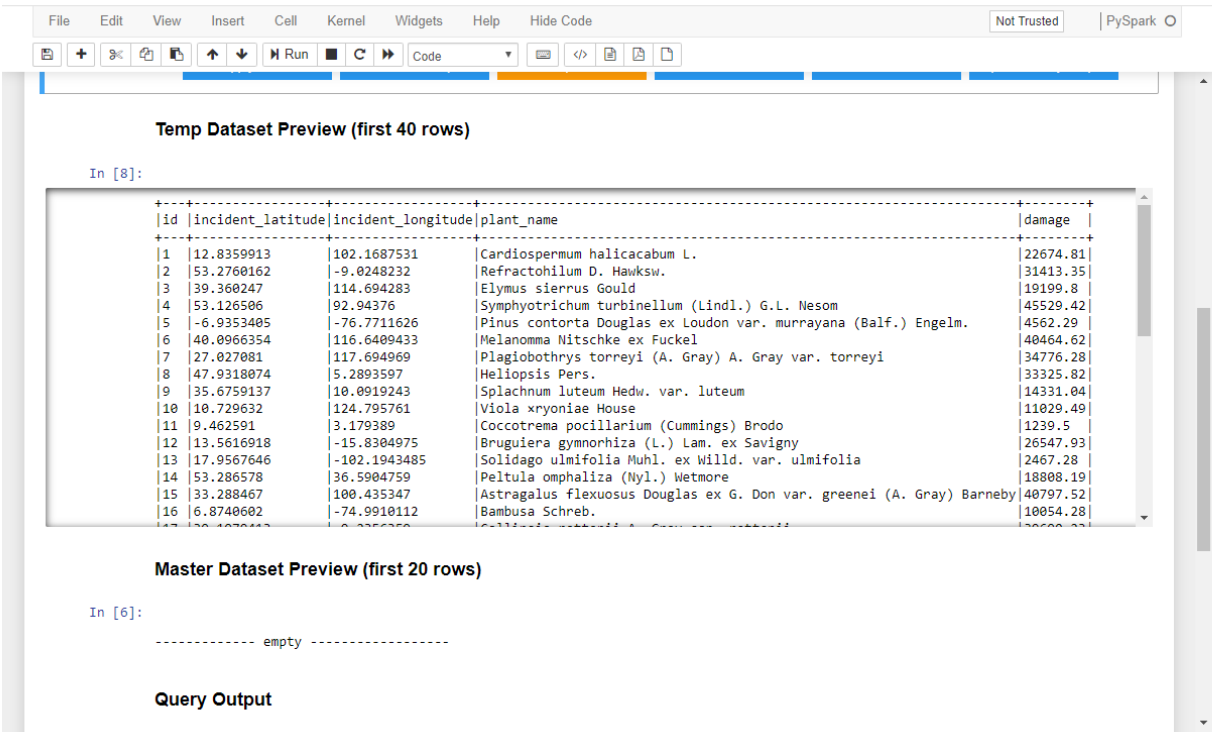
AEGIS system, Query Builder includes various such functionalities, e.g. null value replacement, filtering based on value, statistics through aggregation functions etc.

The final version of the tool is implemented in a Jupyter Notebook, as also explained in D4.2. The tool, potentially in slightly different flavours (to allow for more customization of the data manipulation processes) is directly accessible inside every newly created project through the AEGIS Jupyter Tools. For the final release, the following updates have been performed:

- User interface improvements
- Better integration with the AEGIS platform
- Embedded metadata browser was removed

The user can initially select a dataset and then see a list of included files in order to choose which one to load. AEGIS shared datasets are also supported. The new version can also handle opening a file directly from the Datasets page of AEGIS platform, using the context menu on CSV files.

Once the selected file is opened, it is loaded as a Spark Dataframe, called temp dataset (TempDF), and it is available for further data manipulation. At all times, the user can have up to two different datasets active in Query Builder: the temporary (TempDF) and the master (MasterDF). The temporary is the one currently being used and changed, whereas the master is used as a "storage point" for intermediate results while the user is processing data and as the final result once all data manipulation is over and the user is satisfied with the outcome.



Temp Dataset Preview (first 40 rows)

In [8]:

```
+---+-----------------+------------------+---------------------------------------------------------------------+--------+
|id |incident_latitude|incident_longitude|plant_name                                                           |damage  |
+---+-----------------+------------------+---------------------------------------------------------------------+--------+
|1  |12.8359913       |102.1687531       |Cardiospermum halicacabum L.                                         |22674.81|
|2  |53.2760162       |-9.0248232        |Refractohilum D. Hawksw.                                             |31413.35|
|3  |39.360247        |114.694283        |Elymus sierrus Gould                                                 |19199.8 |
|4  |53.126506        |92.94376          |Symphyotrichum turbinellum (Lindl.) G.L. Nesom                       |45529.42|
|5  |-6.9353405       |-76.7711626       |Pinus contorta Douglas ex Loudon var. murrayana (Balf.) Engelm.      |4562.29 |
|6  |40.0966354       |116.6409433       |Melanomma Nitschke ex Fuckel                                         |40464.62|
|7  |27.027081        |117.694969        |Plagiobothrys torreyi (A. Gray) A. Gray var. torreyi                 |34776.28|
|8  |47.9318074       |5.2893597         |Heliopsis Pers.                                                      |33325.82|
|9  |35.6759137       |10.0919243        |Splachnum luteum Hedw. var. luteum                                   |14331.04|
|10 |10.729632        |124.795761        |Viola xryoniae House                                                 |11029.49|
|11 |9.462591         |3.179389          |Coccotrema pocillarium (Cummings) Brodo                              |1239.5  |
|12 |13.5616918       |-15.8304975       |Bruguiera gymnorhiza (L.) Lam. ex Savigny                            |26547.93|
|13 |17.9567646       |-102.1943485      |Solidago ulmifolia Muhl. ex Willd. var. ulmifolia                    |2467.28 |
|14 |53.286578        |36.5904759        |Peltula omphaliza (Nyl.) Wetmore                                     |18808.19|
|15 |33.288467        |100.435347        |Astragalus flexuosus Douglas ex G. Don var. greenei (A. Gray) Barneby|40797.52|
|16 |6.8740602        |-74.9910112       |Bambusa Schreb.                                                      |10054.28|
```

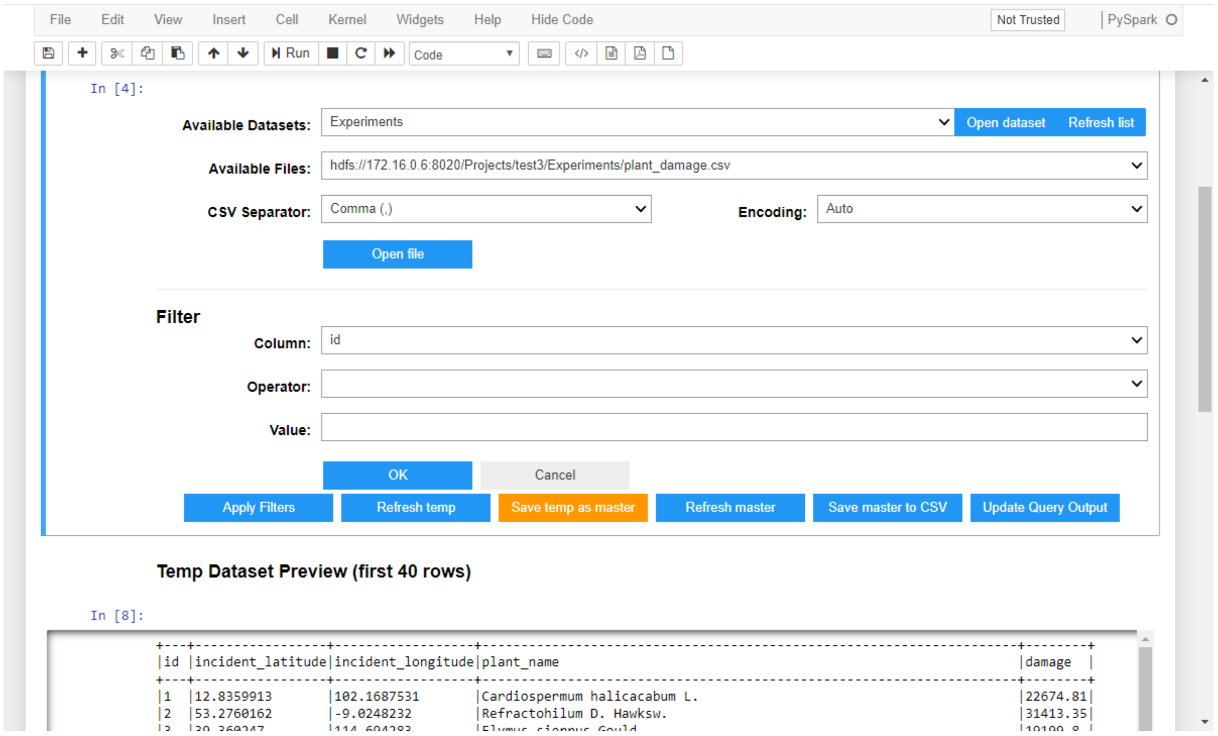
Master Dataset Preview (first 20 rows)

In [6]:

```
------------ empty -----------------
```

Query Output

Figure 34: Query Builder Temp dataset preview



Available Datasets: Experiments

Available Files: hdfs://172.16.0.6:8020/Projects/test3/Experiments/plant_damage.csv

CSV Separator: Comma (,)

Encoding: Auto

Filter

Column: id

Operator:

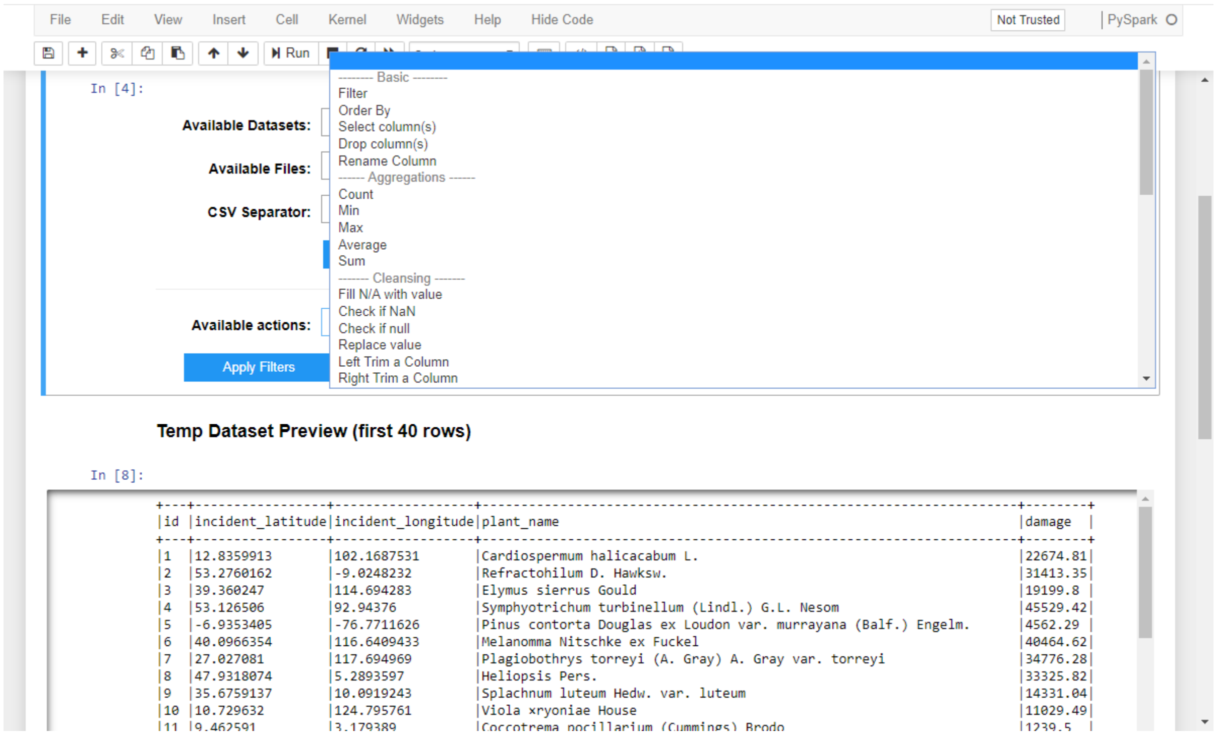
Value:

Temp Dataset Preview (first 40 rows)

```
+---+-----------------+------------------+------------------------------------------------------+--------+
|id |incident_latitude|incident_longitude|plant_name                                            |damage  |
+---+-----------------+------------------+------------------------------------------------------+--------+
|1  |12.8359913       |102.1687531       |Cardiospermum halicacabum L.                          |22674.81|
|2  |53.2760162       |-9.0248232        |Refractohilum D. Hawksw.                              |31413.35|
|3  |39.360247        |114.694283        |Elymus sierrus Gould                                  |19199.8 |
```

Figure 35: Query Builder data filters view

A number of filters and data processing methods are available for the user to select and apply on the temporary dataset, through the "Controls" panel. Indicatively, the user can fill in null values, filter out entries based on values, rename columns, replace values, select/drop columns etc.



Temp Dataset Preview (first 40 rows)

```
+---+-----------------+------------------+------------------------------------------------------+--------+
|id |incident_latitude|incident_longitude|plant_name                                            |damage  |
+---+-----------------+------------------+------------------------------------------------------+--------+
|1  |12.8359913       |102.1687531       |Cardiospermum halicacabum L.                          |22674.81|
|2  |53.2760162       |-9.0248232        |Refractohilum D. Hawksw.                              |31413.35|
|3  |39.360247        |114.694283        |Elymus sierrus Gould                                  |19199.8 |
|4  |53.126506        |92.94376          |Symphyotrichum turbinellum (Lindl.) G.L. Nesom        |45529.42|
|5  |-6.9353405       |-76.7711626       |Pinus contorta Douglas ex Loudon var. murrayana (Balf.) Engelm.|4562.29 |
|6  |40.0966354       |116.6409433       |Melanomma Nitschke ex Fuckel                          |40464.62|
|7  |27.027081        |117.694969        |Plagiobothrys torreyi (A. Gray) A. Gray var. torreyi  |34776.28|
|8  |47.9318074       |5.2893597         |Heliopsis Pers.                                       |33325.82|
|9  |35.6759137       |10.0919243        |Splachnum luteum Hedw. var. luteum                    |14331.04|
|10 |10.729632        |124.795761        |Viola xryoniae House                                  |11029.49|
|11 |9.462591         |3.179389          |Coccotrema pocillarium (Cummings) Brodo               |1239.5  |
```

Figure 36: Query Builder data filters view

As shown in the image, the final version has more than 30 operations available, organized in Basic operations, Aggregation operations, Joins, Time Conversions, Mathematic operations, Cleansing operations and others.

The user may also merge or append the temporary dataset with the master dataset. A list of selected data manipulation actions, either already applied or pending application, is always visible under the “Selected filters” panel. A preview of the data processing result is always available upon clicking the “Refresh temp” button.

When the result of a series of data processing tasks on the temporary dataset is satisfactory, it can be saved as the master dataset. The user may continue processing the same temporary dataset or open a new one or, when the query creation process is complete, can save the master dataset as a new csv file or export the query and continue to directly change the generated code. This code can be leveraged (a) by the advanced user as an easily acquired starting point to further elaborate on for more complex queries and (b) by the less technically skilled user as a means to understand the underlying code and facilitate learning. Finally, the result of the data manipulation, i.e. the master dataset, can be directly passed as input to more high-level AEGIS tools, like the Visualiser and the Algorithm Execution Container.

2.9. Visualiser

The Visualiser is the component enabling the advanced visualisation capabilities of the AEGIS platform. In accordance to the latest design and the specification of the component, as documented in deliverable D3.5, the purpose of the Visualiser remains two-fold: (1) to provide visualisations of the results generated by the Algorithm Execution Container and (2) to provide visualisations of the results generated by the queries composed and executed by the Query Builder.

For the realisation of the advanced visualisation capabilities of the AEGIS platform, the Jupyter[8] notebook development environment, incorporated in the AEGIS platform as part of the AEGIS integrated services, has been exploited. Jupyter supports multiple programming languages and is integrated with data processing frameworks such as Spark, which is utilised in the AEGIS platform. The Visualiser is implemented as a predefined Jupyter notebook following the microservices architecture. As such, a series of microservices where developed and orchestrated through Jupyter in order to implement the functionalities of the Visualiser, which includes the dataset selection, the dataset preview generation, the visualisation type selection, the visualisation configuration, the visualisation generation and the interactive dashboard. The Visualiser can be accessed through Jupyter that is integrated in the AEGIS Front-End as a service.

[8] http://jupyter.org/

In addition to Jupyter, two Python libraries were utilised, namely the Folium[9] and the python-highcharts[10] libraries. These libraries are two state-of-the-art open source charting libraries that facilitate the generation of a large variety of interactive charts and visualisations and are used within the Jupyter notebook.

To facilitate the advanced dataset or results visualisation process, an intuitive and easy-to-use user interface has been implemented guiding the user across all the execution workflow of the Visualiser component, as it was designed and documented in the deliverable D3.5. Visualiser offers a variety of visualisation formats which spans from simple static charts to interactive charts with multiple layers of information and several customisation options.

Based on the feedback received from the AEGIS stakeholders, the Visualiser focused on supporting advanced visualisation options for the Maps visualisation type. More specifically, the Visualiser is offering support for Heatmaps on top of Maps (Figure 37) and enhanced map visualisation with support for markers with custom labels and colours (Figure 38). Moreover, the Visualiser supports FastMarkerCluster on Maps and routes can be visualized as polylines with optional support of also adding markers on that routes. Furthermore, the Visualiser supports advanced configuration options on the visualisation parameters for the offered visualisation types.

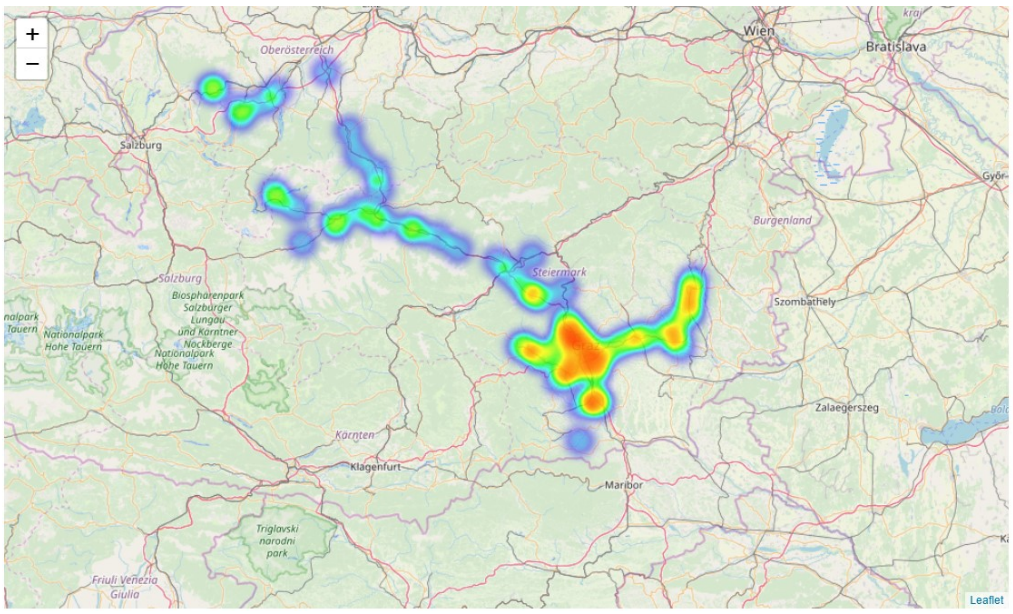


Figure 37: Visualiser Heatmaps on Maps

[9] http://folium.readthedocs.io/en/latest/

[10] https://github.com/kyper-data/python-highcharts

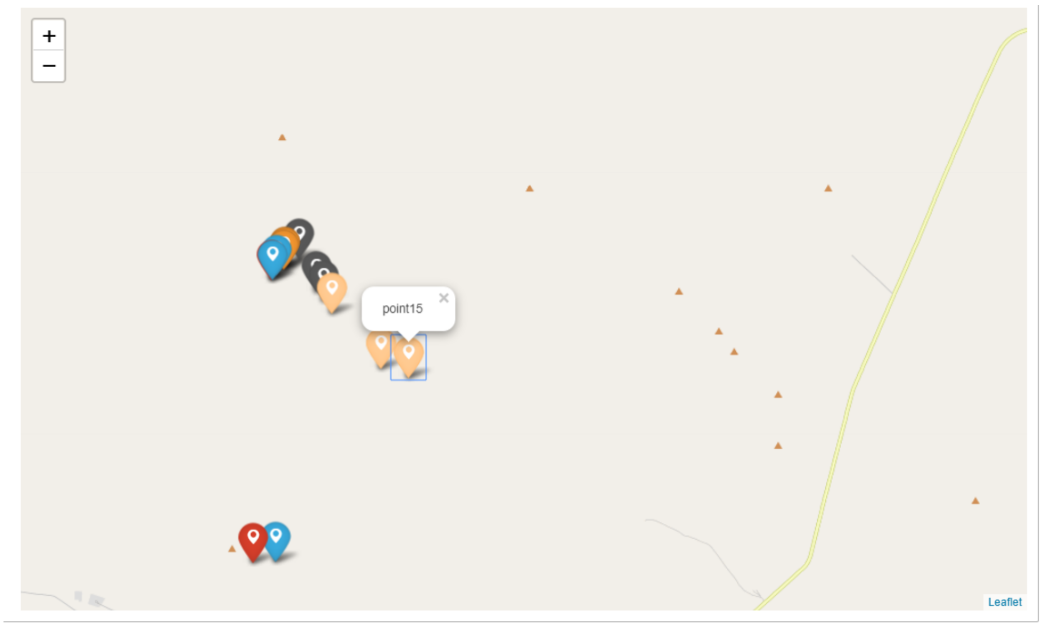


Figure 38: Visualiser Maps with markers with custom labels and colours

In detail, the Visualiser component supports the following visualisation types:

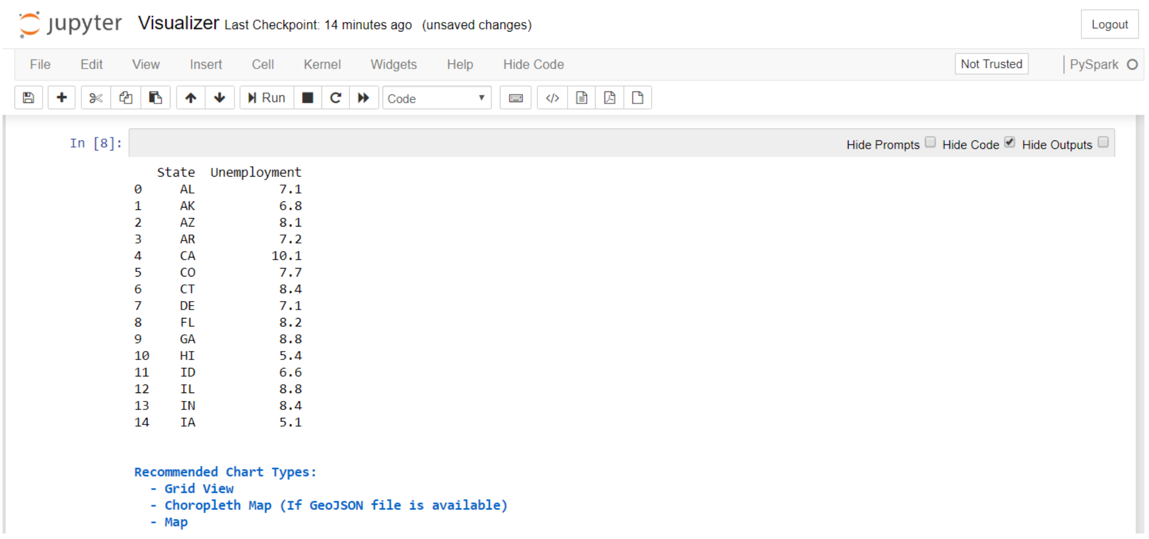
- Scatter plot
- Pie chart
- Bar chart
- Line chart
- Box plot
- Histogram
- Time series
- Heatmap
- Bubble chart
- Map (with support for HeatMaps on Maps, markers with custom labels and colours,FastMarkerCluster, Routes via polylines and Routes with Markers)
- Grid-View (Pretty Tabular Format)
- Area Chart
- Spline
- Choropleth Map
- Earthquake Risk Heatmap (Specific, only for Insurance Demonstrator)

The Visualiser is supporting exception handling that informs the user for any failure during the visualisation process with the prompt message. Finally, the Visualiser is supporting the export of the generated visualisation in the form of HTML file that can be saved in the AEGIS Data Store as a new asset of the user's project. Additionally, the Visualiser is also compatible with both Python 2.7 and 3.6 versions.

The Visualiser is also supporting the connection with the AEGIS Metadata Service. Upon loading a specific file from the Visualiser's file explorer, the Metadata Service is triggered by passing the full path of the file as parameter and the metadata associated with this file, if any, is returned. If metadata is available, the Visualiser performs a basic recommendation of chart types that can be used. This recommendation is based on the data types of the available variables of the dataset. An example of this feature is depicted in Figure 39. Another useful aspect of the metadata utilisation is that it can be used to pre-fill some fields during the parameterization of the chart, such as geospatial information in maps.

The execution workflow includes the following steps:

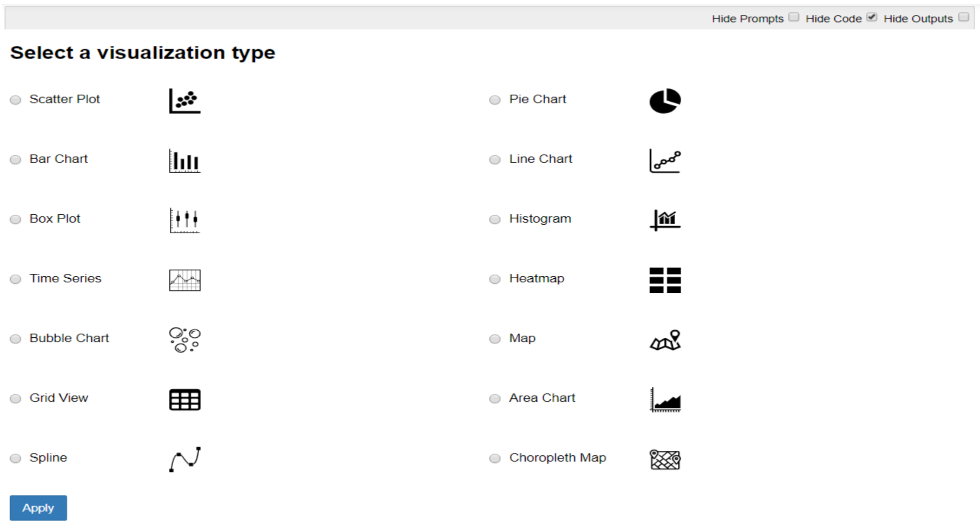
1. At first, when the Visualiser is loaded as an interactive notebook the user is presented with a list of options in order to define the dataset that will be utilised for the visualisation creation. The user is able to navigate through the list of available datasets within the project's folders and select the desired dataset. Upon selecting the desired dataset, a preview of the dataset in tabular format is presented to the user (Figure 39). Alternatively, the user can select a specific file from the dataset browser of the main AEGIS platform, right-click on it and then select to open it with the Visualiser. In this case, the drop-down menus will be prefilled with the path of the selected file.


jupyter
Visualizer Last Checkpoint: 14 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help Hide Code
Not Trusted
PySpark
Run
Code
In [8]:
Hide Prompts
Hide Code
Hide Outputs

```
   State  Unemployment
0     AL           7.1
1     AK           6.8
2     AZ           8.1
3     AR           7.2
4     CA          10.1
5     CO           7.7
6     CT           8.4
7     DE           7.1
8     FL           8.2
9     GA           8.8
10    HI           5.4
11    ID           6.6
12    IL           8.8
13    IN           8.4
14    IA           5.1


Recommended Chart Types:
  - Grid View
  - Choropleth Map (If GeoJSON file is available)
  - Map
```

Figure 39: Visualiser - dataset selection

2. In the next step, the user is presented with the list of available visualisation types (Figure 40). Once the desired visualisation type is selected, the user is presented with the list of available parameters for the specific visualisation type. The list of parameters includes a variety of options that spans from the variables that will be used in the visualisation and the titles that will be displayed in the visualisation axis to the selected visualisation's type specific parameters such as the aggregation function or class variable. An example of the visualisation parameters selection is displayed in Figure 41.

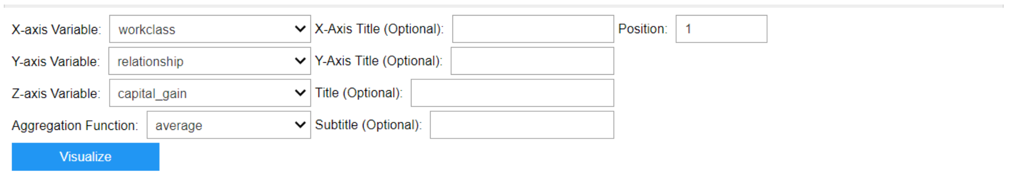


Hide Prompts ☐ Hide Code ☑ Hide Outputs ☐

Select a visualization type

○ Scatter Plot	○ Pie Chart
○ Bar Chart	○ Line Chart
○ Box Plot	○ Histogram
○ Time Series	○ Heatmap
○ Bubble Chart	○ Map
○ Grid View	○ Area Chart
○ Spline	○ Choropleth Map

Figure 40: Visualiser - visualisation type selection



X-axis Variable: workclass
X-Axis Title (Optional):
Position: 1
Y-axis Variable: relationship
Y-Axis Title (Optional):
Z-axis Variable: capital_gain
Title (Optional):
Aggregation Function: average
Subtitle (Optional):

Figure 41: Visualiser - visualisation parameters

3. Once the visualisation type has been selected and the corresponding parameters have been set, the user can trigger the visualisation creation. The following figures illustrate some examples of the visualisations offered by the Visualiser.

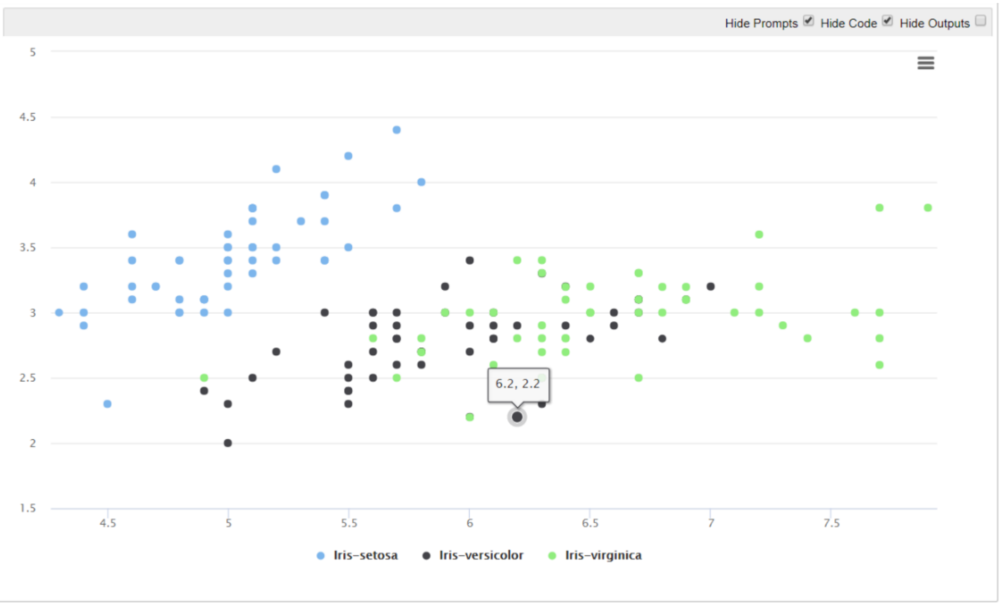


Figure 42: Visualiser – scatter plot

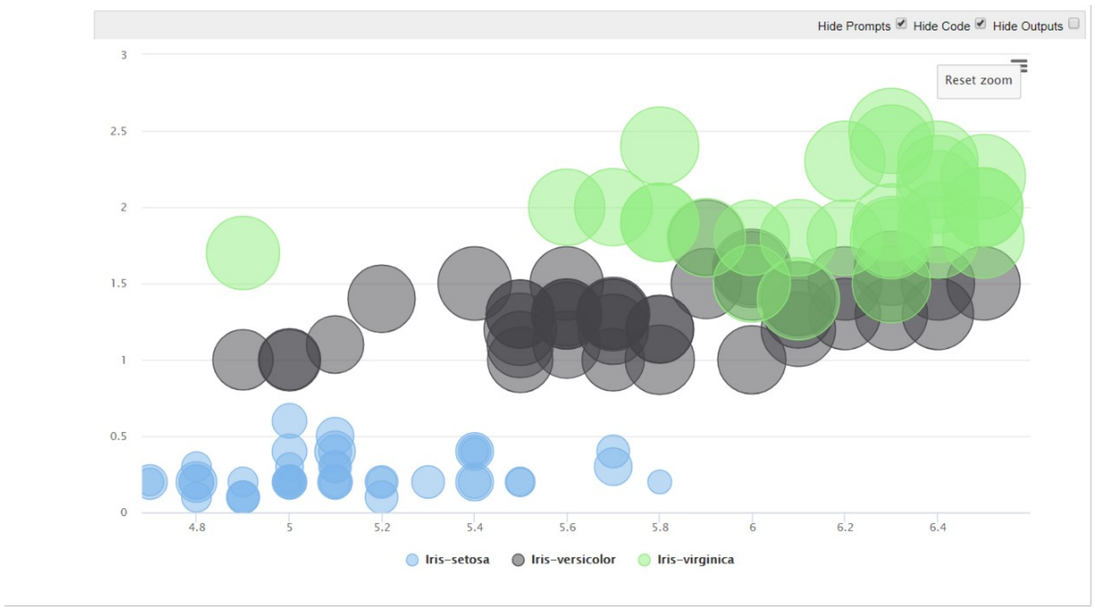


Figure 43: Visualiser - bubble chart

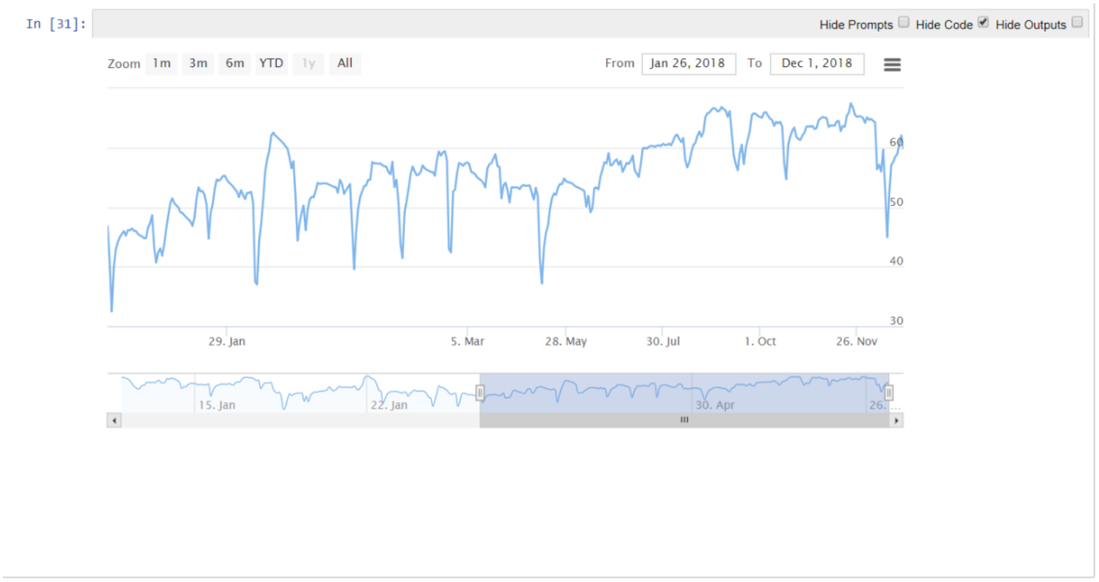


Figure 44: Visualiser - time series

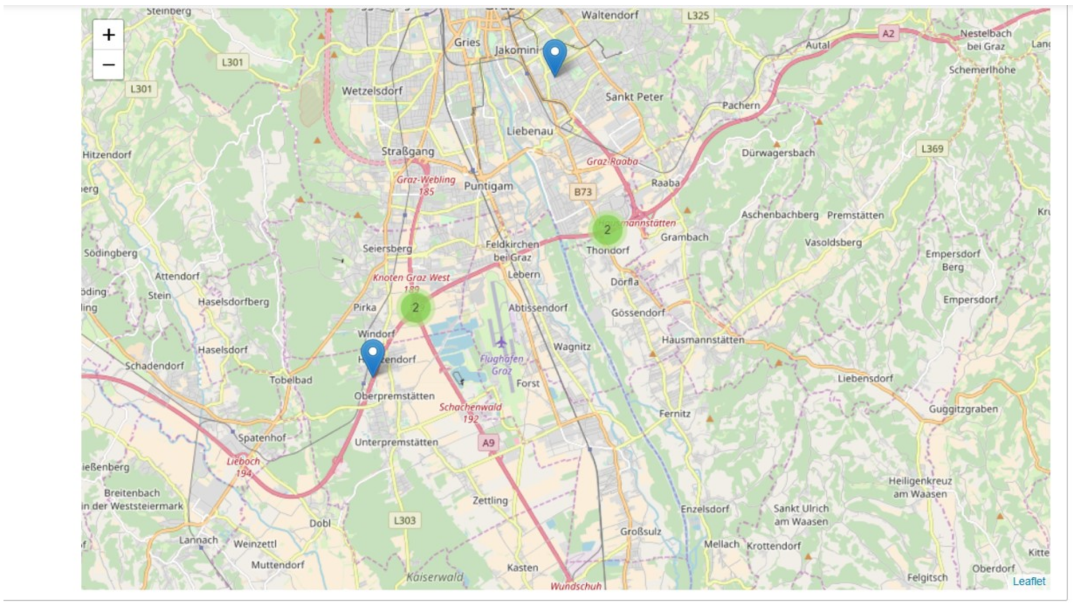


Figure 45: Visualiser – map

2.10. Algorithm Execution Container

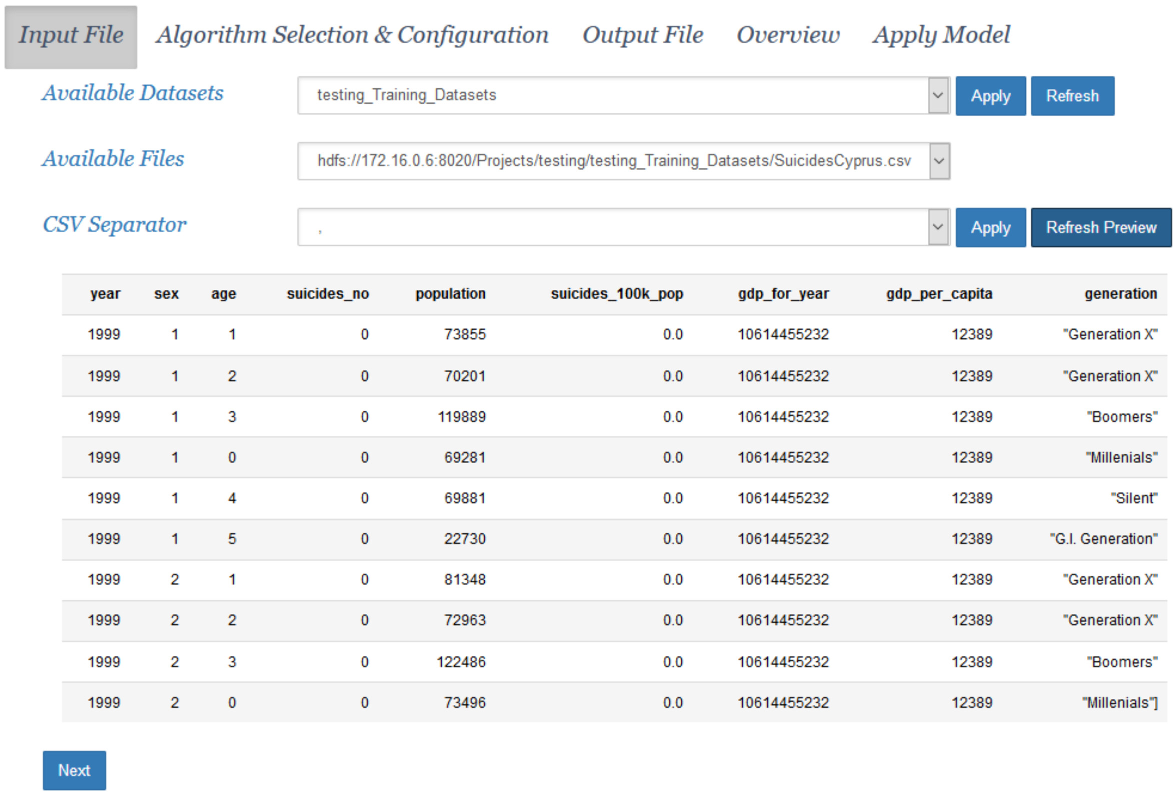
The final version of the AEGIS Algorithm Execution Container is based on the Jupyter notebook and includes UX and performance fixes over the version deployed and documented in D4.3.

The current Jupyter version of the container is now based on Python 3.6 (instead of Python 2.7 as in the previous versions) and JavaScript. Moreover, the module is now compliant with the https version of the AEGIS platform, as the internal calls used are now performed over the https protocol, ensuring the secure exchange of information to the AEGIS backend.

Upon launch of the corresponding Algorithm Execution Container (hereinafter container), the user has to initialised it through a dedicated button. The initialisation ensures that the Spark interpreter of the AEGIS platform is started and also creates the basic UI of the container, which is a simple five-tab window:

- An input file selection tab
- An algorithm selection and configuration tab
- An output folder selection tab
- An overview tab showing the current user selections and, when ready, the execution results
- A model application tab, to perform regression or classification with existing trained models on other datasets

Apart from the overview tab, the other three correspond to the basic steps towards applying an algorithm through the container. The first step is to select the file that contains the data to be used by the algorithm, through the interface shown in the next figure, which brings up also a preview of the selected file.



Available Datasets: testing_Training_Datasets

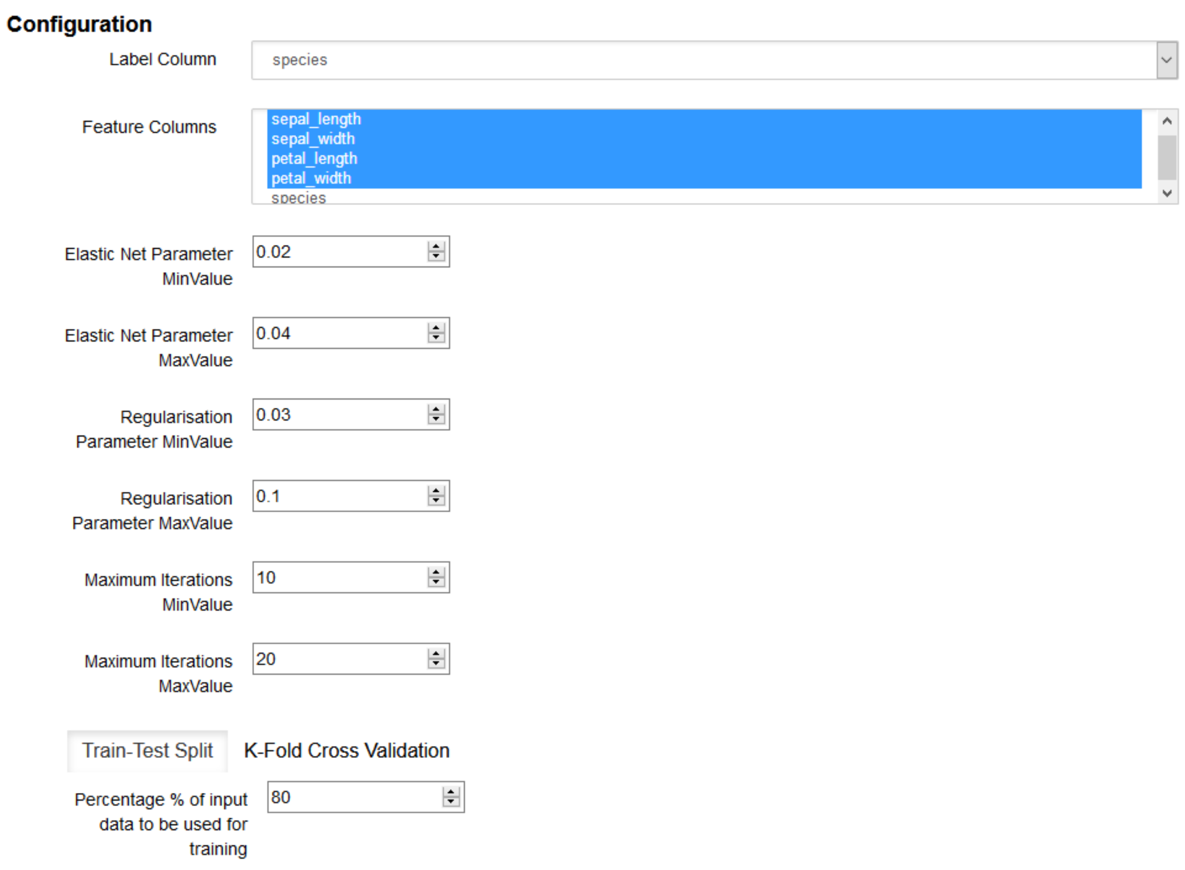
Available Files: hdfs://172.16.0.6:8020/Projects/testing/testing_Training_Datasets/SuicidesCyprus.csv

CSV Separator: ,

year	sex	age	suicides_no	population	suicides_100k_pop	gdp_for_year	gdp_per_capita	generation
1999	1	1	0	73855	0.0	10614455232	12389	"Generation X"
1999	1	2	0	70201	0.0	10614455232	12389	"Generation X"
1999	1	3	0	119889	0.0	10614455232	12389	"Boomers"
1999	1	0	0	69281	0.0	10614455232	12389	"Millenials"
1999	1	4	0	69881	0.0	10614455232	12389	"Silent"
1999	1	5	0	22730	0.0	10614455232	12389	"G.I. Generation"
1999	2	1	0	81348	0.0	10614455232	12389	"Generation X"
1999	2	2	0	72963	0.0	10614455232	12389	"Generation X"
1999	2	3	0	122486	0.0	10614455232	12389	"Boomers"
1999	2	0	0	73496	0.0	10614455232	12389	"Millenials"]

Figure 46: AEC Input File and Data Preview Tab

The next step is to select the algorithm to be applied. The provided algorithms are grouped under five categories (algorithm families). Once an algorithm is selected, a form from which to configure its parameters is shown to the user. A basic form validation is done for the case that a selected value for a parameter is not within the specified boundaries. In this point, the final version of the Algorithm Execution Container includes the option to set a grid of parameters, so that the algorithm may run within those spans, and it automatically selects the best model based on the results produced. A preview of the configuration of the grid parameters is shown in the next figure.



Configuration

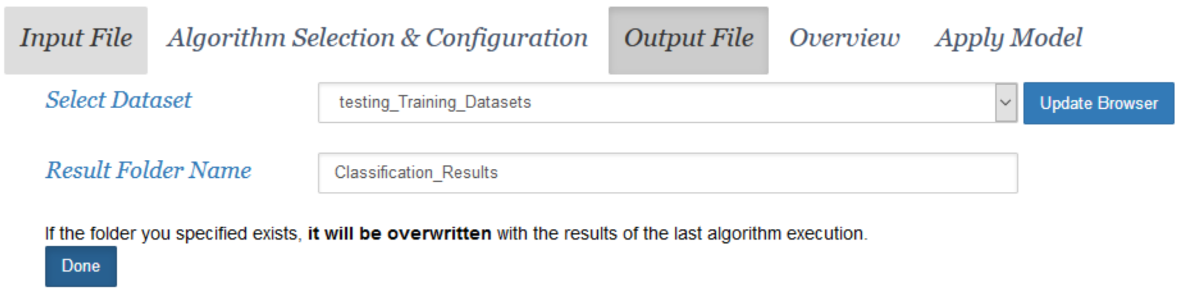
Label Column	species
Feature Columns	sepal_length, sepal_width, petal_length, petal_width (selected); species
Elastic Net Parameter MinValue	0.02
Elastic Net Parameter MaxValue	0.04
Regularisation Parameter MinValue	0.03
Regularisation Parameter MaxValue	0.1
Maximum Iterations MinValue	10
Maximum Iterations MaxValue	20

Train-Test Split K-Fold Cross Validation

Percentage % of input data to be used for training	80

Figure 47: AEC Algorithm Configuration Tab

The model that will be created when an algorithm is applied is saved in the user's datasets, in a folder specified in the last container tab, as shown below:

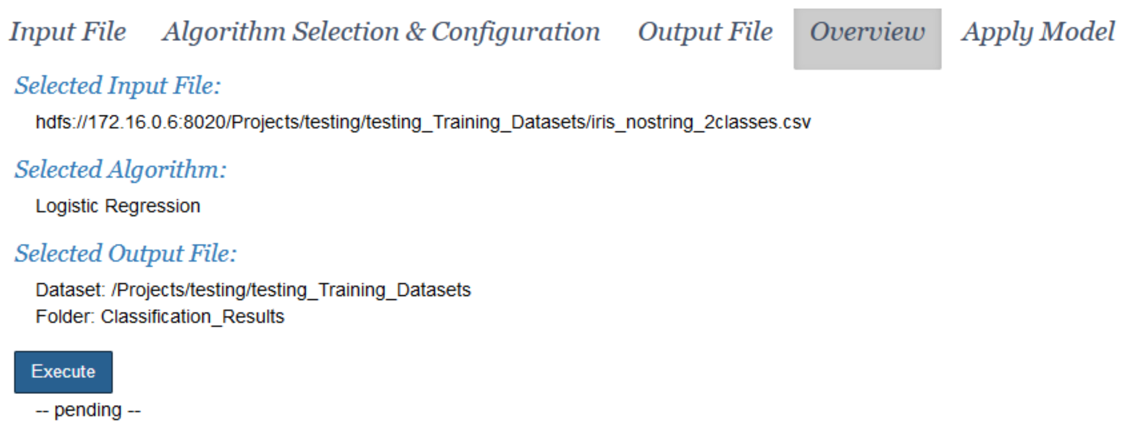


Select Dataset	testing_Training_Datasets
Result Folder Name	Classification_Results

If the folder you specified exists, **it will be overwritten** with the results of the last algorithm execution.

Figure 48: AEC Output File tab

Once this last step is concluded, by pressing the "Apply" button, the user is taken to the Overview Tab:



Selected Input File:
hdfs://172.16.0.6:8020/Projects/testing/testing_Training_Datasets/iris_nostring_2classes.csv

Selected Algorithm:
Logistic Regression

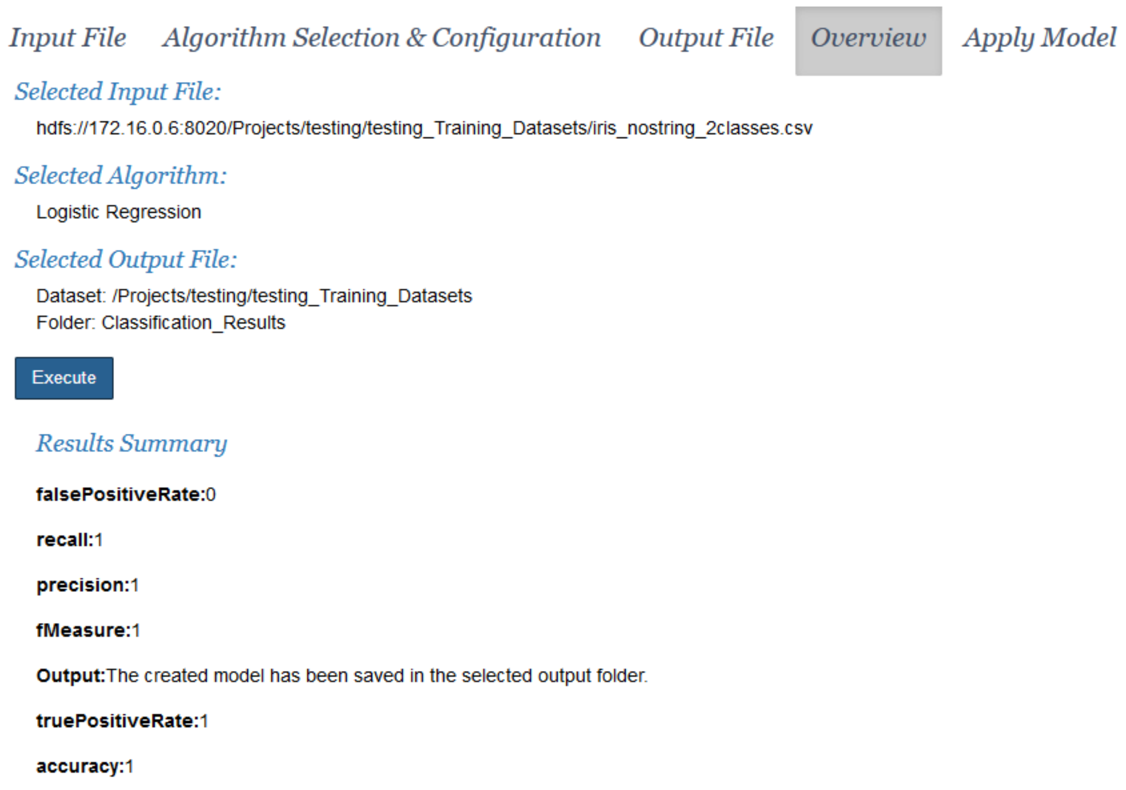
Selected Output File:
Dataset: /Projects/testing/testing_Training_Datasets
Folder: Classification_Results

-- pending --

Figure 49: AEC Overview Tab

There, by pressing Done, the algorithm is applied and when the execution is completed, some algorithm-dependent results are provided to the user in the same tab (Figure 50).

The final output of the analysis is stored back in the AEGIS Data Store, while the model that was used for the analysis, is also stored alongside with the analysis results.



Selected Input File:

hdfs://172.16.0.6:8020/Projects/testing/testing_Training_Datasets/iris_nostring_2classes.csv

Selected Algorithm:

Logistic Regression

Selected Output File:

Dataset: /Projects/testing/testing_Training_Datasets
Folder: Classification_Results

Results Summary

falsePositiveRate:0

recall:1

precision:1

fMeasure:1

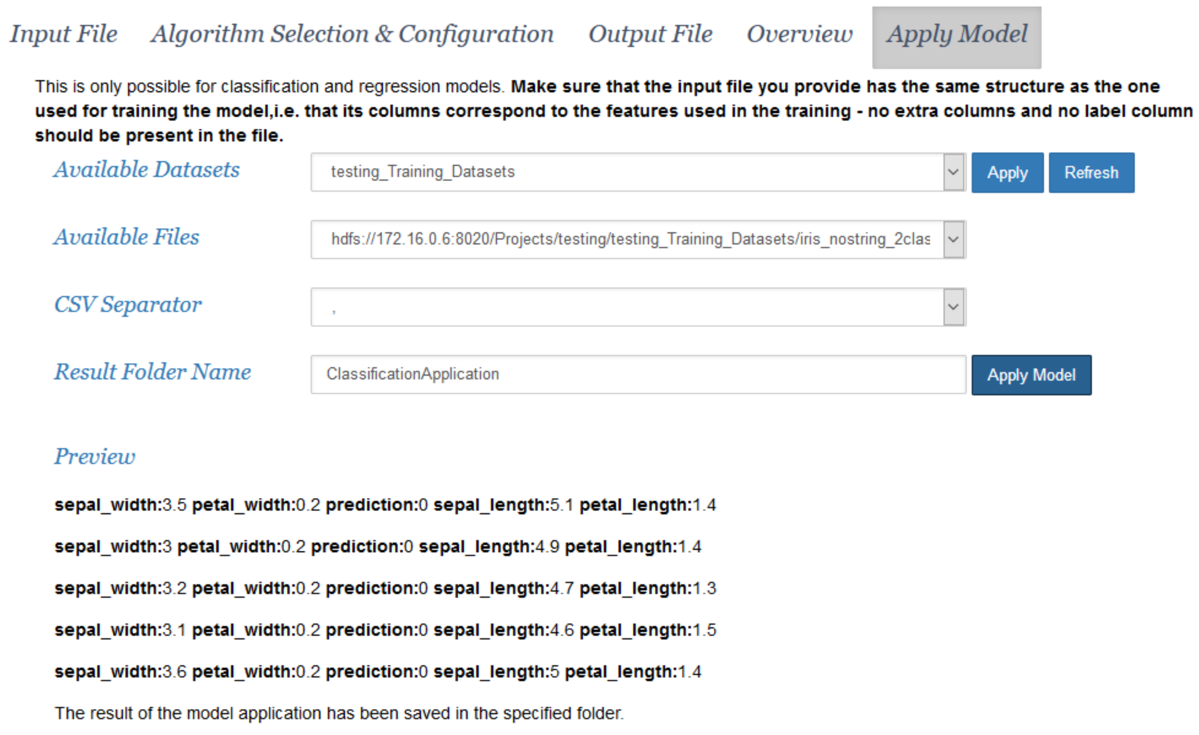
Output: The created model has been saved in the selected output folder.

truePositiveRate:1

accuracy:1

Figure 50: AEC Overview Tab with Results

In case the analysis that has been selected is of a classification or a regression, the model that has been created in the previous steps can be re-applied on new datasets, using the “Apply Model” tab, provided that the format of the input file is same as that of the one used to generate the model. This feature is shown in the next figure, where the user is able to select from existing files and re-apply the model.



This is only possible for classification and regression models. **Make sure that the input file you provide has the same structure as the one used for training the model,i.e. that its columns correspond to the features used in the training - no extra columns and no label column should be present in the file.**

Available Datasets	testing_Training_Datasets
Available Files	hdfs://172.16.0.6:8020/Projects/testing/testing_Training_Datasets/iris_nostring_2clas
CSV Separator	,
Result Folder Name	ClassificationApplication

Preview

sepal_width:3.5 **petal_width:**0.2 **prediction:**0 **sepal_length:**5.1 **petal_length:**1.4

sepal_width:3 **petal_width:**0.2 **prediction:**0 **sepal_length:**4.9 **petal_length:**1.4

sepal_width:3.2 **petal_width:**0.2 **prediction:**0 **sepal_length:**4.7 **petal_length:**1.3

sepal_width:3.1 **petal_width:**0.2 **prediction:**0 **sepal_length:**4.6 **petal_length:**1.5

sepal_width:3.6 **petal_width:**0.2 **prediction:**0 **sepal_length:**5 **petal_length:**1.4

The result of the model application has been saved in the specified folder.

Figure 51: Applying an newly created model to other datasets

As with the previous version, it needs to be mentioned, that in case an analyst is willing to run more complex and customised analyses, the module allows the direct edit of the underlying code.

2.11. AEGIS Front-end

Following the look and feel of the AEGIS institutional web site[11] (see Figure 52) an updated GUI/front-end for the AEGIS final integrated prototype has been developed, on top of Hopsworks, using as main technologies HTML and AngularJS[12].

[11] https://www.aegis-bigdata.eu/

[12] https://angularjs.org/

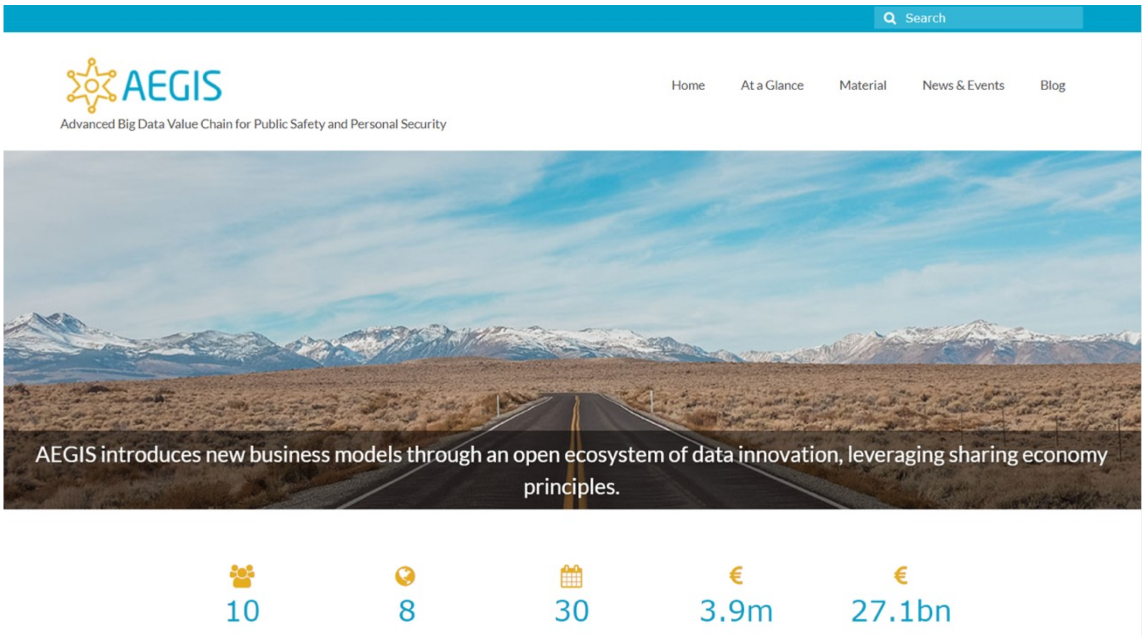

Search
AEGIS
Advanced Big Data Value Chain for Public Safety and Personal Security
Home
At a Glance
Material
News & Events
Blog
AEGIS introduces new business models through an open ecosystem of data innovation, leveraging sharing economy principles.
10
8
30
€ 3.9m
€ 27.1bn

Figure 52: AEGIS web site

The updates consist in many UI graphical improvements and adjustments, also due to the upgrade of Hopsworks versions.
A Landing page has been developed, including main links to Login and Features pages, as well as to objectives and project information (see Figure 53).

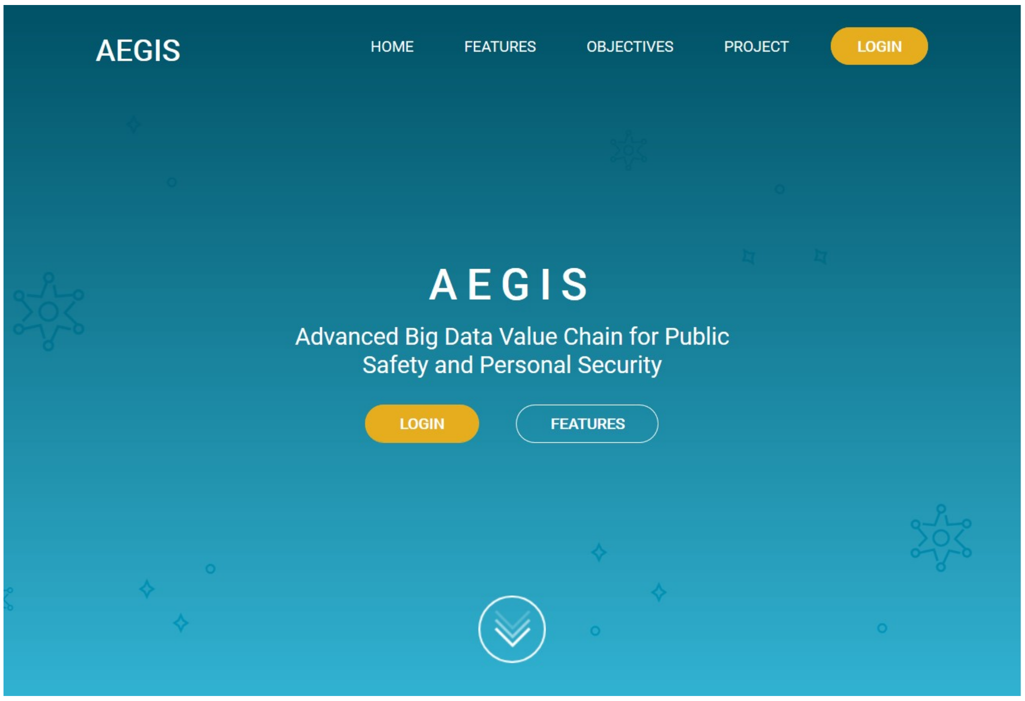

AEGIS
HOME
FEATURES
OBJECTIVES
PROJECT
LOGIN
AEGIS
Advanced Big Data Value Chain for Public Safety and Personal Security
LOGIN
FEATURES

Figure 53: Landing page

The home page has been redefined (see Figure 54) and now features one big space in the centre page, displaying the (current) user projects. The Projects menu on the right side has been removed accordingly. At the bottom of the page, the link to three main features are available:

- Getting started
- AEGIS Marketplace
- What's new

Once selected a project, the project page (see Figure 55: Project Datasets) with the available datasets is displayed. On the left column, the new menu features the following options:

- Datasets
- Settings
- Members
- Activity-stream
- Tools:
    - Extended Metadata
    - AEGIS Jupyter Tools
    - Jobs
    - Kafka
    - Model Serving

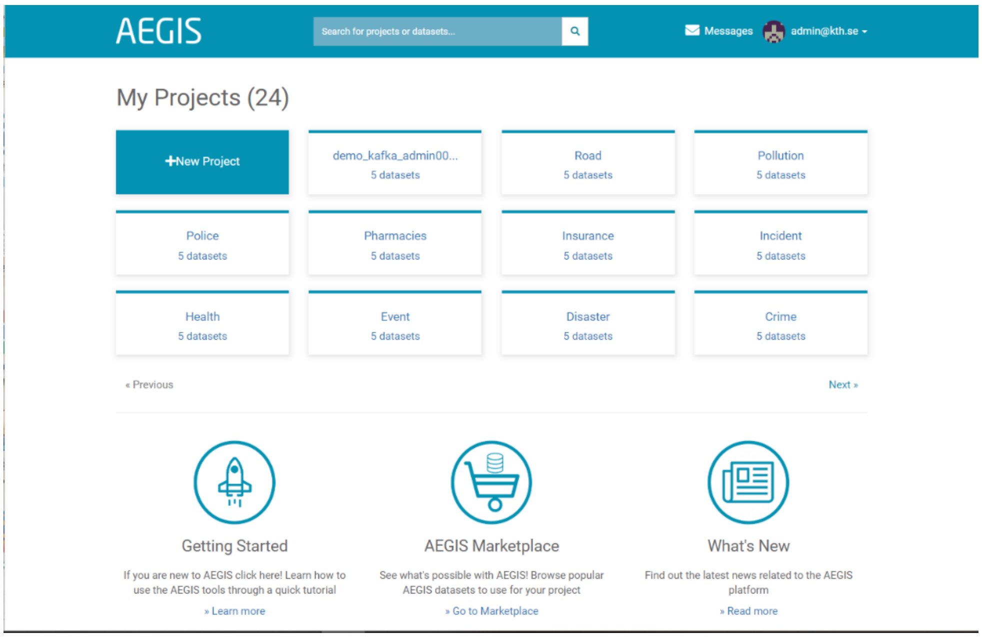


Figure 54: AEGIS home page

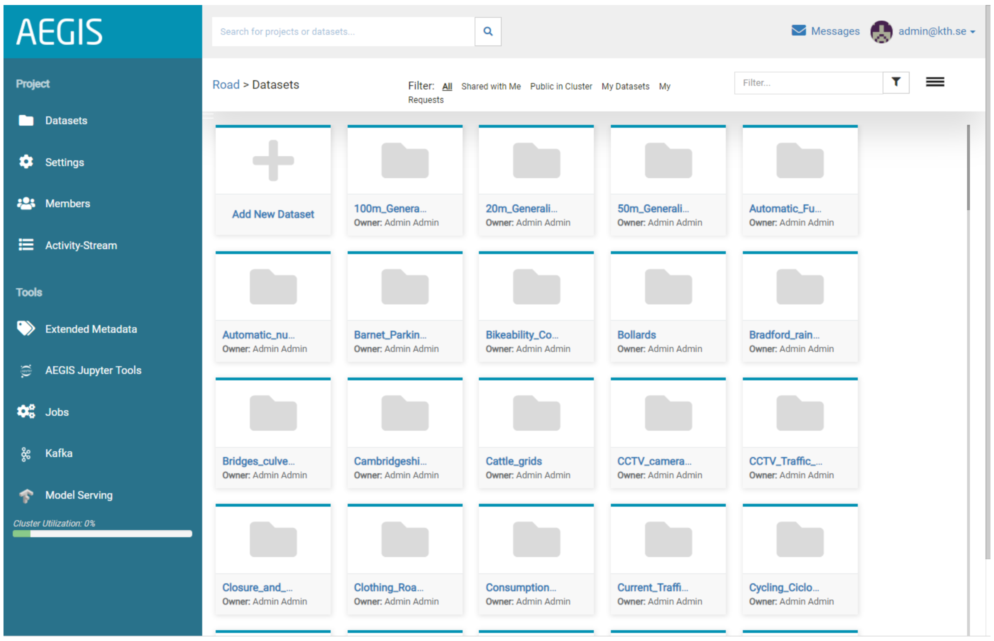


Figure 55: Project Datasets

The new AEGIS Jupyter Tools features the three project notebooks all in a single view, with links to the related documentation and source code.

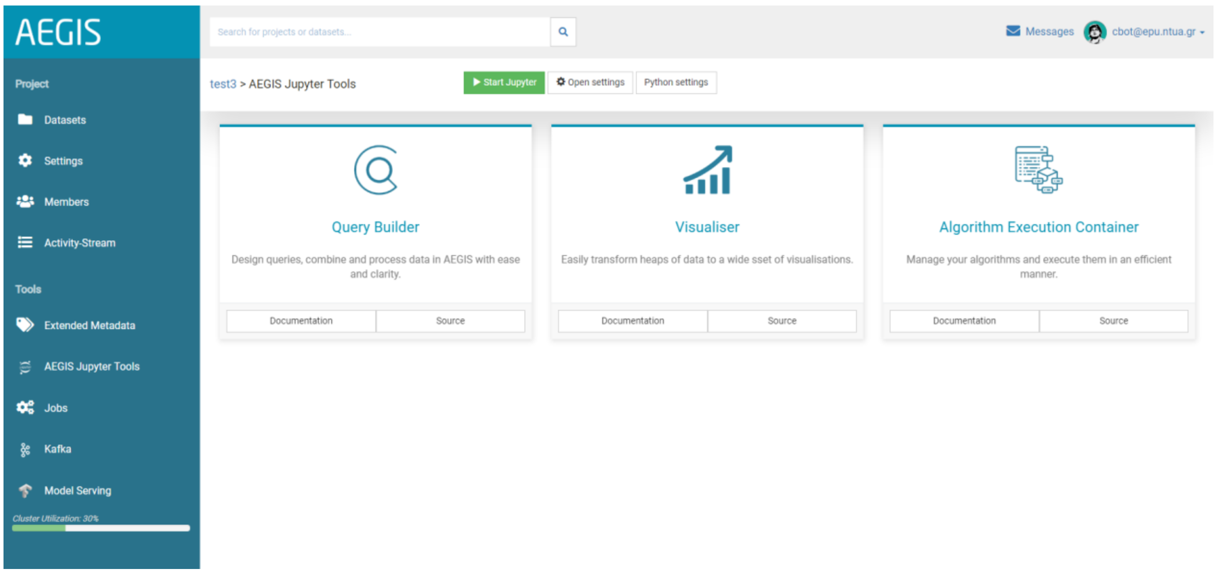
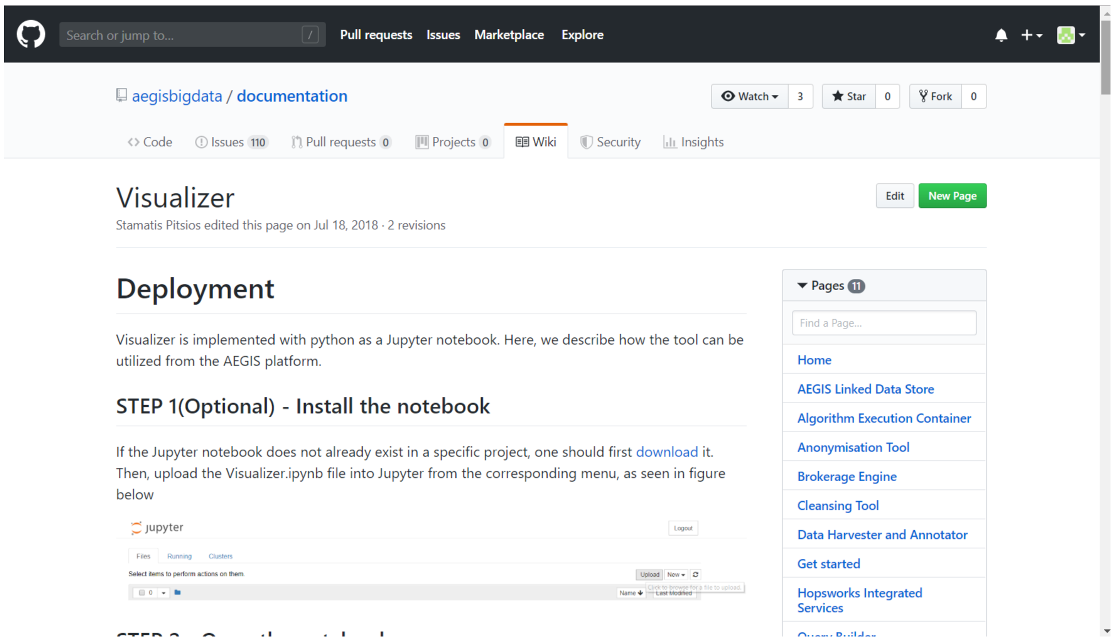

AEGIS
Search for projects or datasets...
Messages
cbot@epu.ntua.gr
Project
Datasets
Settings
Members
Activity-Stream
Tools
Extended Metadata
AEGIS Jupyter Tools
Jobs
Kafka
Model Serving
Cluster Utilization: 30%
test3 > AEGIS Jupyter Tools
Start Jupyter
Open settings
Python settings
Query Builder
Design queries, combine and process data in AEGIS with ease and clarity.
Documentation
Source
Visualiser
Easily transform heaps of data to a wide sset of visualisations.
Documentation
Source
Algorithm Execution Container
Manage your algorithms and execute them in an efficient manner.
Documentation
Source

Figure 56: AEGIS Jupyter Tools page


Search or jump to...
Pull requests Issues Marketplace Explore
aegisbigdata / documentation
Watch 3 Star 0 Fork 0
Code Issues 110 Pull requests 0 Projects 0 Wiki Security Insights
Edit New Page
Pages 11
Find a Page...
Home
AEGIS Linked Data Store
Algorithm Execution Container
Anonymisation Tool
Brokerage Engine
Cleansing Tool
Data Harvester and Annotator
Get started
Hopsworks Integrated Services

Visualizer

Stamatis Pitsios edited this page on Jul 18, 2018 · 2 revisions

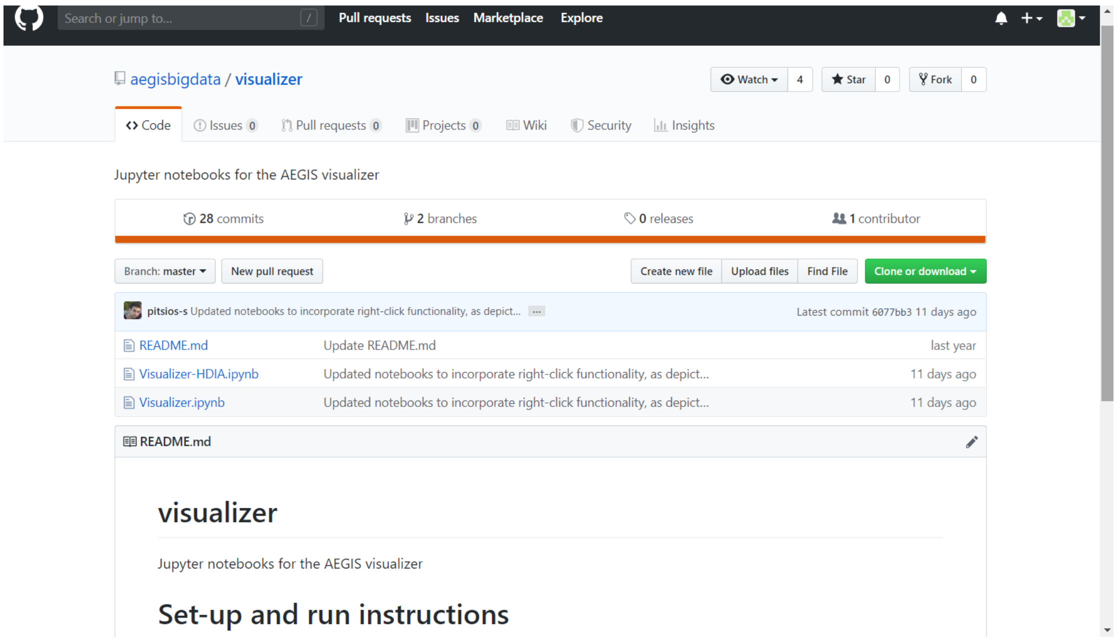
Deployment

Visualizer is implemented with python as a Jupyter notebook. Here, we describe how the tool can be utilized from the AEGIS platform.

STEP 1(Optional) - Install the notebook

If the Jupyter notebook does not already exist in a specific project, one should first download it. Then, upload the Visualizer.ipynb file into Jupyter from the corresponding menu, as seen in figure below

Figure 57: Visualizer documentation


Search or jump to...
Pull requests Issues Marketplace Explore
aegisbigdata / visualizer
Watch 4 Star 0 Fork 0
Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights
Branch: master
New pull request
Create new file Upload files Find File Clone or download

Jupyter notebooks for the AEGIS visualizer

28 commits 2 branches 0 releases 1 contributor

pitsios-s Updated notebooks to incorporate right-click functionality, as depict... Latest commit 6077bb3 11 days ago

README.md	Update README.md	last year
Visualizer-HDIA.ipynb	Updated notebooks to incorporate right-click functionality, as depict...	11 days ago
Visualizer.ipynb	Updated notebooks to incorporate right-click functionality, as depict...	11 days ago

README.md

visualizer

Jupyter notebooks for the AEGIS visualizer

Set-up and run instructions

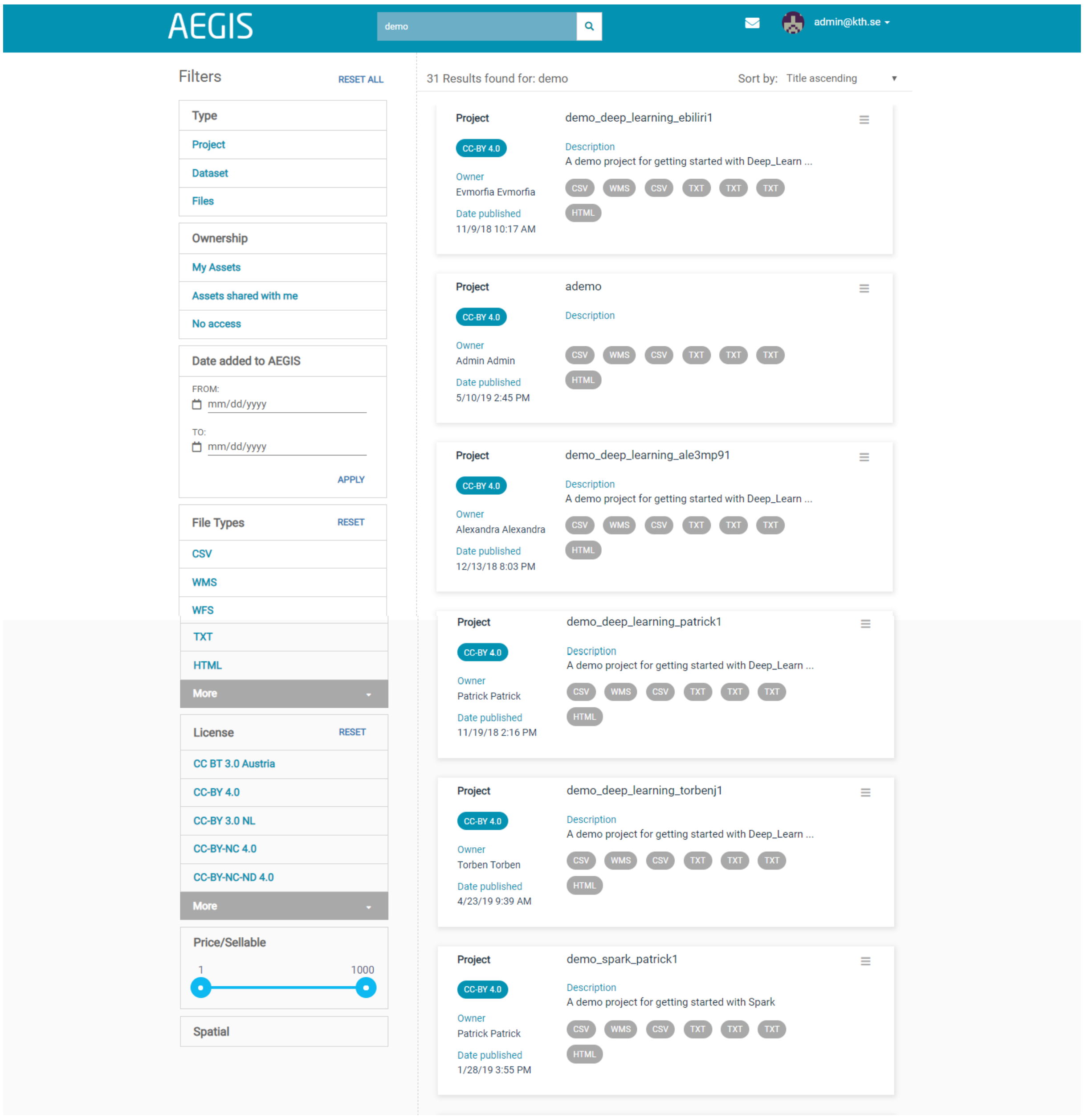
Figure 58: Visualizer source code

As far as the Datasets, the functionalities linked to a dataset have been enhanced and the user/dataset interaction logic has been modified as follows:

- Double-click on the dataset enters the datasets (browse files);
- left-click on the dataset shows details on the column on the right;

- right-click on the dataset opens the menu (browse files, remove dataset, permissions, add metadata, remove metadata, add or edit extended metadata, share with).

Moreover, a new page for the full search functionality has been implemented (see Figure 59).



31 Results found for: demo

Project — demo_deep_learning_ebiliri1
CC-BY 4.0
Description: A demo project for getting started with Deep_Learn ...
Owner: Evmorfia Evmorfia
CSV WMS CSV TXT TXT TXT HTML
Date published: 11/9/18 10:17 AM

Project — ademo
CC-BY 4.0
Description
Owner: Admin Admin
CSV WMS CSV TXT TXT TXT HTML
Date published: 5/10/19 2:45 PM

Project — demo_deep_learning_ale3mp91
CC-BY 4.0
Description: A demo project for getting started with Deep_Learn ...
Owner: Alexandra Alexandra
CSV WMS CSV TXT TXT TXT HTML
Date published: 12/13/18 8:03 PM

Project — demo_deep_learning_patrick1
CC-BY 4.0
Description: A demo project for getting started with Deep_Learn ...
Owner: Patrick Patrick
CSV WMS CSV TXT TXT TXT HTML
Date published: 11/19/18 2:16 PM

Project — demo_deep_learning_torbenj1
CC-BY 4.0
Description: A demo project for getting started with Deep_Learn ...
Owner: Torben Torben
CSV WMS CSV TXT TXT TXT HTML
Date published: 4/23/19 9:39 AM

Project — demo_spark_patrick1
CC-BY 4.0
Description: A demo project for getting started with Spark
Owner: Patrick Patrick
CSV WMS CSV TXT TXT TXT HTML
Date published: 1/28/19 3:55 PM

Figure 59: Full Search page

Major technical details about the Front-end technologies have been provided in AEGIS-*D3.5 Architecture and Revised Components Microservices and APIs Designs_v4.00_v1.0*.

2.12. Third Party AEGIS Services

AEGIS as a platform with specific open APIs is able to cooperate with various third party services that are interested either to consume some of AEGIS outputs and services, but also to provide AEGIS assets (such as datasets) to enrich its content and its offerings to stakeholders. The section below presents such a case, which was implemented during the project as a third-party service (e.g. not part of the AEGIS core platform) to provide data for the Insurance and the Smart Home demonstrators.

2.12.1. *Event Detection Module*

The Event Detection Module (EDM), residing as a demo at http://aegis-insurance.s5labs.eu, which has been developed during the AEGIS project, is an application that is able to monitor data pushed in the Twitter network, as well as RSS feeds, and provide notifications regarding events that are happening in specific locations, based on a trained model that evaluates whether the content discussed is an event of interest for an analyst that has fired up the application.

In more detail, the EDM can work in two different ways:

- As a system based on a trained model, which takes into consideration the location of tweets (using geofencing to identify tweets coming from a set of neighbouring regions), evaluating whether the tweets come from trustworthy ("followed") or not accounts, whether they refer to actual events and are not past ones, etc.
- As a simple monitoring service, that picks up specific keywords as they appear and displays them as events.

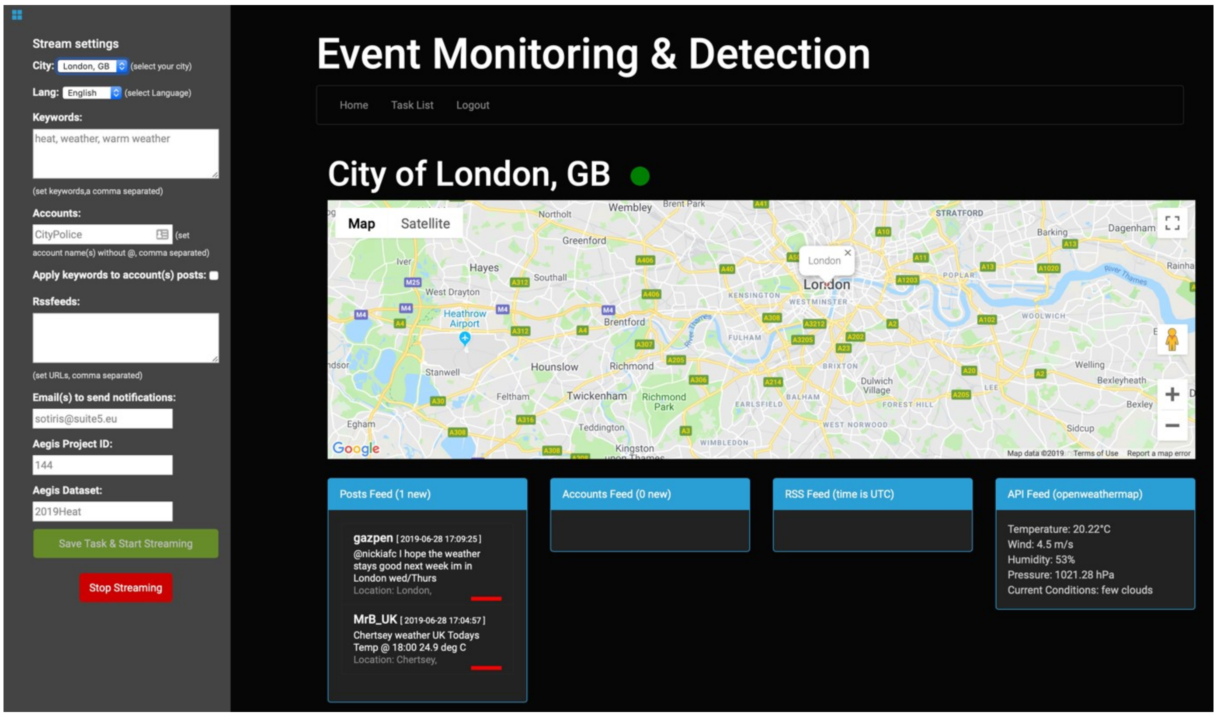
Apparently, the first mode of operation is of higher importance for analysts who would like to be punctually and in a trustworthy manner notified for specific events, and thus the focus was on this mode. For training the EDC model, a large set of tweets and relevant texts were collected and using a training set a model has been constructed using the AEGIS AEC and notebooks, which has been fed back to the EDM system in order for it to be able to identify events. Text inserted was multilingual, and so were the models developed, and as a result the EDM supports English, Greek, Italian and German.

As shown in the following figure which displays the dashboard of the system, a user is able to select specific keywords for events to monitor and specify selected accounts in Twitter, and/or selected RSS feeds, which could be governmental or news portals that push news, etc. witter accounts.

Moreover, the search can be either on a global scale, or in specific cities (for the cause of the demonstrator we experimented with London, Athens, Rome, Lazio, Genova and Gratz – e.g. places of the demonstrators). It needs to be noted that the locality is not only examined based on the "location" of the tweet according to its metadata (as it seems that only few users select the option to provide their location when tweeting), but is extracted out of the content of the message. Moreover, using the Google Geolocation API[13] the EDM can identify whether a location is in proximity of the selected area (city) of interest, so for example a tweet that

[13] https://developers.google.com/maps/documentation/geolocation/intro

includes "Soho" in its body will be displayed when one selects the London area, as EDM identifies Soho as an area within London.



Stream settings

City: London, GB (select your city)

Lang: English (select Language)

Keywords: heat, weather, warm weather (set keywords,a comma separated)

Accounts: CityPolice (set account name(s) without @, comma separated)

Apply keywords to account(s) posts:

Rssfeeds: (set URLs, comma separated)

Email(s) to send notifications: sotiris@suite5.eu

Aegis Project ID: 144

Aegis Dataset: 2019Heat

Posts Feed (1 new)

gazpen [2019-06-28 17:09:25]
@nickiafc I hope the weather stays good next week im in London wed/Thurs
Location: London,

MrB_UK [2019-06-28 17:04:57]
Chertsey weather UK Todays Temp @ 18:00 24.9 deg C
Location: Chertsey,

Accounts Feed (0 new)

RSS Feed (time is UTC)

API Feed (openweathermap)

Temperature: 20.22°C
Wind: 4.5 m/s
Humidity: 53%
Pressure: 1021.28 hPa
Current Conditions: few clouds

Figure 60: Dashboard of EDM

The main API that is used by the EDM is the Streaming API of Twitter[14], which allows the application to receive a constant feed from Twitter for the specific words under investigation, though due to certain limitation reasons set by Twitter, not all tweets are provided.

The outputs of the EDM, apart from being displayed on the frontend of the application, as shown in the figure above, are also pushed to AEGIS in case an analyst selects so. This is possible with the AEGIS Harvester, which is used to bundle the outputs of the EDM and upload them into a specific dataset under a project, so that analysts can then download this information or preform other set of analyses. Moreover, notifications for new events could also be send (apart from emailing) to the owner of this AEGIS project, using an Apache Kafka interface.

[14] https://developer.twitter.com/en/docs/tutorials/consuming-streaming-data.html

3. Conclusion

The document accompanies the fourth and final release of the AEGIS integrated prototype and provides information about the realization of the foreseen software prototype connected to a deployed version of platform. A full description of the platform functionalities developed is provided, as well as references to the software package of the core platform and its API, with corresponding supporting documentation on the deployment of each component and usage of the APIs.