Ref. Ares(2021)2145214 - 26/03/2021

Tailored IoT & BigData Sandboxes and Testbeds for Smart, Autonomous and Personalized Services in the European Finance and Insurance Services Ecosystem

D5.13 – Datasets for Algorithms Training & Evaluation - I

Revision Number 3.0
Task Reference T5.1
Lead Beneficiary UBI
Responsible Konstantinos Perakis
Partners AGRO ASSEN ATOS BOUN CP CTAG ENG FBK FTS GEN GFT INNOV ISPRINT JRC JSI PI PRIVE RB SIA UBI UPRC WEA
Deliverable Type Report (R)
Dissemination Level Public (PU)
Due Date 2021-02-28
Delivered Date 2021-03-26
Internal Reviewers FTS, GFT
Quality Assurance INNOV
Acceptance WP Leader Accepted and Coordinator Accepted
EC Project Officer Pierre-Paul Sondag
Programme HORIZON 2020 - ICT-11-2018

This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement no 856632

Contributing Partners

Partner Acronym	Role[1]	Author(s)[2]
UBI	Lead Beneficiary	Konstantinos Perakis, Dimitris Miltiadou
INSO	Contributor	Elena Femenia
INNOV	Contributor	John Soldatos
PRIVE	Contributor	Pablo Carballo
UPRC	Contributor	Dimitris Kotios
CP	Contributor	Marinos Xynarianos
JSI	Contributor	Maja Skrjanc
BOUN	Contributor	Can Ozturan
ENG	Contributor	Susanna Bonura
ATOS	Contributor	Juan Carrasco Alonso
ISPRINT	Contributor	Aristodemos Pnevmatikakis
WEA	Contributor	Carlos Albo Portero
AGRO	Contributor	Gregory Mygdakos
GFT	Internal Reviewer	Ernesto Troiano
FTS	Internal Reviewer	Juergen Neises
INNOV	Quality Assurance	Dimitrios Drakoulis

Revision History

Version	Date	Partner(s)	Description
0.1	2020-07-30	UBI	ToC Version

0.2	2020-08-10	UBI	Initial contributions on Section 2 and 3
0.3	2020-08-30	UBI	Contributions on Section 2 and 3
0.35	2020-09-15	UBI	Updated Contributions on Section 2 and 3
0.40	2020-09-17	INSO, INNOV, PRIVE, UPRC, CP, JSI, BOUN, ENG, ATOS, ISPRINT, WEA, AGRO	Contributions on Section 4
0.45	2020-09-30	UBI	Updated Contributions on Section 2 and 3
0.50	2020-10-15	INSO, INNOV, PRIVE, UPRC, CP, JSI, BOUN, ENG, ATOS, ISPRINT, WEA, AGRO	Updated contributions on Section 4
0.60	2020-10-25	INSO, INNOV, PRIVE, UPRC, CP, JSI, BOUN, ENG, ATOS, ISPRINT, WEA, AGRO	Updated contributions on Section 4

[1]Lead Beneficiary, Contributor, Internal Reviewer, Quality Assurance

[2]Can be left void

0.65	2020-11-05	INSO, INNOV, PRIVE, UPRC, CP, JSI, BOUN, ENG, ATOS, ISPRINT, WEA, AGRO	Updated contributions on Section 4
0.70	2020-11-13	UBI	Finalisation of sections 2, 3, 4 and 5
1.0	2020-11-15	UBI	Initial Version for Internal Review
1.5	2021-02-24	UBI	Updated Version for Internal Review
2.0	2021-03-05	UBI	Version for Quality Assurance
3.0	2021-03-26	UBI	Version for Submission

Executive Summary

The document at hand, entitled “D5.13 – Datasets for Algorithms Training & Evaluation - I” constitutes a

report of the preliminary efforts and the produced outcomes of Task T5.1 “Data Collection for Algorithms Training & Evaluation” of WP5. The purpose of this deliverable is threefold: a) to deliver the first version of the **INFINITECH Data Collection component**, b) to present the **core characteristics** and the **role of synthetic datasets within INFINITECH** and c) to document the **preliminary list of real and synthetic datasets** which will be exploited within INFINITECH.

Hence, the scope of the current report can be described in the following axes:

- To present the **INFINITECH Data Collection component** that is designed and implemented within the context of INFINITECH. In particular, the **high-level architecture of the component** is documented by presenting the component’s three main modules, namely the **Data Retrieval**, the **Data Mapper** and the **Data Cleaner**. Furthermore, the deliverable presents for each module the **detailed design specifications**, the **addressed use cases** and the respective **sequence diagrams**, as well as the **implementation details** of the first version of the component with the help of UML diagrams.
- To perform a comprehensive **analysis of the key characteristics of the synthetic datasets** and the definition of their role within INFINITECH. In this context, the deliverable presents the **main categories of synthetic datasets** and the **synthetic data generation process** along with the list of tools, libraries and frameworks that are utilised. Additionally, the **use cases and motivation for the usage of synthetic datasets** within the context of INFINITECH is documented along with the **initial list of synthetic datasets** that will be exploited by the INFINITECH pilots.
- To document the **details of the datasets that will be collected and utilised by the INFINITECH pilots**. Hence, the **preliminary list of the datasets**, real and synthetic, that will be leveraged, is documented by presenting the details of the **information included** on these datasets, their **format** and the **pseudonymization or anonymisation requirements**, where needed.

The outcomes of this deliverable will set the basis for the implementation activities of T5.1 and the rest of the tasks of WP5. However, the definition of the design specifications of the INFINITECH Data Collection component, the synthetic datasets generation and the list of datasets which will be exploited by the INFINITECH pilots, constitute a living process that will last until M27 as per the INFINITECH Description of Action. Hence, the deliverable at hand is a living document that will be constantly updated to incorporate the optimisations and enhancements that will derive from the analysis of the feedback, that will be collected from the pilots of the project and the stakeholders of the platform. The upcoming version of the deliverable, Deliverable D5.14, will document all the evolvements and updates and will be delivered on M27.

Table of Contents

List of Figures

List of Tables

Abbreviations/Acronyms

Abbreviation	Definition
AI	Artificial Intelligence
AML	Anti-Money Laundering
API	Application Programmable Interface
BDA	Big Data Application
CSV	Comma Separated Values
CTF	Combating Terrorist Financing
EO	Earth Observation
ES	Expected Shortfall
FTP	File Transfer Protocol
HDFS	Hadoop Distributed File System
HPC	High-performance computing
HTTP	Hypertext Transfer Protocol
ICT	Information and Communications Technology
IoT	Internet of Things
JSON	JavaScript Object Notation
JWT	JSON Web Token
LOCF	Last Observation Carried Forward

ML Machine Learning
NOCB Next Observation Carried Backward ORC
Optimized Row Columnar
PDF Portable Document Format
PNG Portable Network Graphics
PoC Proof of Concept
PSD2 Payment Service Directive 2
RA Reference Architecture
RWD Real-World Data
SME Small and Medium-Sized Enterprises UI User
Interface
UML Unified Modelling Language
VaR Value-at-Risk
XML Extensible Mark-up Language

1 Introduction

The scope of deliverable D5.13 “Datasets for Algorithms Training & Evaluation – I” is to document the preliminary efforts undertaken within the context of T5.1 “Data Collection for Algorithms Training & Evaluation” of WP5. In this context, the deliverable D5.13 constitutes the first iteration of the work performed under T5.1 and is prepared in accordance with the INFINITECH Description of Action on M17 of the project.

Datasets are the fuel of any Big Data enabled platform in which a vast amount of data are collected, harmonised, annotated, cleaned and stored in order to be exploited using a variety of data analysis approaches towards the aim of knowledge extraction and useful insights gain. One of the most demanding and challenging areas of the entire data lifecycle is the data collection which is an umbrella of processes and operations, spanning from the retrieval or reception of data originating from heterogeneous data source to their pre-processing, correlation, cleaning and eventually storage in an effective storage solution.

Despite the fact that data are produced in a tremendous amount of volume, there are several data privacy restrictions and regulations imposed, hence their collection and processing is in many cases limited or even prohibited. To overcome the problem of data unavailability or usage restrictions, synthetic datasets are generated and utilised. Synthetic datasets are generated utilising multiple techniques and methodologies depending on the nature of the real data that they are replicating, as well as the domain for which they will be utilised and exploited. As with all the aspects of any software solution there is no single norm or

approach that is suitable for all use cases.

Datasets constitute one of the main ingredients of the INFINITECH platform and within the context of the INFINITECH project, both real and synthetic datasets will be utilised by the project's pilots. However, the prerequisite for their exploitation is to ensure that they are effectively collected and eventually ingested into the INFINITECH platform to enable their exploration, analysis and processing through the execution of scenarios for the financial and insurance sectors. From these scenarios, the financial institutions will enhance their current core operational services and processes, while at the same time, they will be empowered to deliver new services, products and offerings to their clients. To this end, within the context of Task 5.1, the consortium has analysed the key requirements related to data collection in big data enabled platforms in order to formulate the design specifications of a data collection component which will be integrated and implemented as part of the INFINITECH RA that is tailored to the needs of the finance and insurance sectors.

The designed solution will be utilised towards the design and implementation of data collection pipelines that address the needs of data providers of the INFINITECH project, as well as the stakeholders of the financial and insurance sectors. Furthermore, the consortium has analysed the synthetic data generation methods and how they can be used in order to overcome the data privacy restrictions and data unavailability for the implementation of innovative financial and insurance services.

The INFINITECH pilots are implementing a variety of finance and insurance scenarios with different approaches and levels of complexity, depending on their scope. To this end, the required list of datasets for each pilot use case is carefully compiled in order to take into consideration and effectively address the peculiarities/specificities of each scenario. These datasets will be collected during the data collection process of each pilot. It should be noted that the details of the data collection processes of the pilots are documented in deliverable D7.1.

1.1 Objective of the Deliverable

The purpose of the deliverable is to report the outcomes of the work performed within the context of Task 5.1 at this phase of the project (M17). During this first period, the work that has been performed was mainly focused on the design and the implementation of the initial version of the INFINITECH Data Collection component, the analysis of the key characteristics of the synthetic datasets as well as the definition of their role in INFINITECH and finally the formulation of the preliminary list of required datasets of the INFINITECH pilots.

The first objective of the deliverable is to present the first version of the INFINITECH Data Collection component. The INFINITECH Data Collection component is designed and implemented aiming to address the need for a holistic mechanism that will empower the data providers to configure and execute data collection pipelines tailored to their needs**.** It is designed following a modular architecture composed by three distinct modules, namely the Data Retrieval, the Data Mapper and the Data Cleaner. In this context, the design specifications of each module are documented in detail by describing their scope and their concrete list of functionalities. Furthermore, the detailed use cases that each module addresses and that document the involvement of each stakeholder and components on each one of them, are also presented. Moreover, the supported workflows by each component are presented in the form of sequence diagrams. Finally, the implementation details of each module are documented, by presenting a high-level overview of the source code structuring via UML diagrams and the list of implemented functions by each module.

The second objective of the deliverable is to present the role of the Synthetic Datasets and their usage within the context of INFINITECH. To this end, a thorough analysis is executed in order to define their scope, document their advantages and limitations, as well as their classification into three major categories. Additionally, the approaches that are followed for the synthetic data generation process, are presented along with the list of tools and frameworks that are used during this process. Finally, the usage of the synthetic datasets within the context of INFINITECH project is documented by presenting the relevant use cases in the finance and insurance sectors, as well as the preliminary list of synthetic datasets that will be

utilised in the project.

The third and final objective of the deliverable is to document the preliminary list of datasets that will be exploited within the context of each pilot of INFINITECH. In this context, the deliverable presents the initial list of datasets that each pilot will leverage, documenting the details of the information included along with data format and the pseudo-anonymisation and anonymisation activities that will be followed, where needed.

Deliverable D5.13 constitutes the first iteration of the deliverable, and as per the INFINITECH Description of Action, Task T5.1 lasts until M27. Therefore, the second version of the deliverable, namely Deliverable D5.14, that will be delivered on M27 will constitute the final report of the work performed within the scope of Task 5.1. Deliverable D5.14 will include all the necessary updates and enhancements, where needed, taking into consideration the advancements of the project, as well as the feedback that will be received by the stakeholders of INFINITECH.

1.2 Insights from other Tasks and Deliverables

Deliverable D5.13 is released in the scope of WP5 "Data Analytics Enablers for Financial and Insurance Services" and documents the preliminary outcomes of the work performed within the context of T5.1 "Data Collection for Algorithms Training & Evaluation". The task is directly related to the outcomes of WP2 "Vision and Specifications for Autonomous, Intelligent and Personalized Services" in which the overall requirements of the INFINITECH platform are defined. Task 5.1 received as input the outcomes of Task 2.1, in which the collected user stories of pilots of the project and the extracted user requirements were formulated and reported within deliverable D2.1. In addition to this, the outcomes of T2.3, in which one can find the fundamental building blocks of the INFINITECH platform and their specifications in terms of technologies, as well as the elicited technical requirements that are linked to these building blocks, as reported in deliverable D2.5, were provided as input to T5.1. Furthermore, the outcomes of T2.7, in which the INFINITECH Reference Architecture (INFINITECH RA) was formulated, were also provided as input to T5.1 and had driven the design and implementation aspects of the Data Collection component that constitutes a core part of the INFINITECH platform. Finally, the work reported in this deliverable is tightly interconnected with the work performed in the rest of the tasks of WP5, namely T5.2, T5.3, T5.4 and T5.5, as it covers the required data collection aspects for the datasets that will be used on all the aforementioned tasks.

1.3 Structure

This document is structured as follows:

- Section **Error! Reference source not found.** introduces the document, describing the context of the outcomes of the work performed within the task and highlights its relation to the rest of tasks of the project and deliverables of the project.
- Section 2 documents the design specifications, the use case addressed along with relevant sequence diagrams and the implementation details of the INFINITECH Data Collection component. • Section 3 presents the key characteristics, advantages and limitations of the synthetic datasets, the details of the synthetic data generation process and how they will be leveraged in INFINITECH. • Section 4 presents the details of the datasets that will be collected from the INFINITECH pilots in order to execute their designed use cases. Finally,
- Section 5 concludes the document.

2 INFINITECH Data Collection

2.1 Design Specifications

Data Ingestion typically involves all the processes and operations that are performed for the gathering or retrieval of data from different data sources or data providers, the correlation and annotation of the included data entities with several ontologies and semantics and finally the cleaning of the data with multiple data cleaning operations before they are stored in the underlying storage solution of the system or the data warehouse. In the big data era, where a tremendous amount of diverse information is generated by a plethora of data sources, systems, devices and platforms, data ingestion becomes a crucial part of any designed solution. However, at the same time data ingestion becomes rather challenging not only because of the volume of data that grows exponentially, but also due to the nature of data sources that generate

data in a large variety of formats and at different velocities.

Towards this end, the INFINITECH Data Collection was designed aiming to provide an abstract and holistic mechanism for the data providers that will address the various connectivity and communication challenges with the variety of data sources that are exploited in the finance and insurance sector, as well as the peculiarities/specificities of the various data providers of the specific sectors. Hence, the scope of the Data Ingestion mechanism is threefold:

a) to enable the acquisition and retrieval of heterogeneous data from diverse data sources and data providers,
b) to facilitate the mapping of the entities included in the data to the corresponding entities of an underlying data model towards the data annotation and
c) to enable the data cleaning operations that will address the data quality issues of the acquired data.

The INFINITECH Data Collection is composed of three main modules:

a) the Data Retrieval that undertakes the responsibility to retrieve or receive the new datasets from a data source or data provider,
b) the Data Mapper that is responsible from the generation of the mapping between the entities of retrieved or received dataset and the ones of an underlying data model entities based on the data provider's input and
c) the Data Cleaner that undertakes the responsibility to perform the data cleaning operations on the retrieved or received dataset, again based on the data provider's input.

Figure 1 depicts the high-level architecture of the INFINITECH Data Collection. As illustrated, the INFINITECH Data Collection is capable of retrieving or fetching new datasets from a variety of data sources and to receive new datasets via its exposed RESTful APIs. The newly acquired dataset is fed in the process by the Data Retrieval in order to be provided as input to the Data Mapper. The Data Mapper performs the data mapping operations and generates the mapping between the data entities of the newly acquired dataset and the entities of an underlying data model, that is provided by the data provider, based on the input of the data provider. In the final step, the newly acquired dataset is provided as input to the Data Cleaner in which the data cleaning operations are performed based on the input of the data provider in order to provide the "cleaned data".

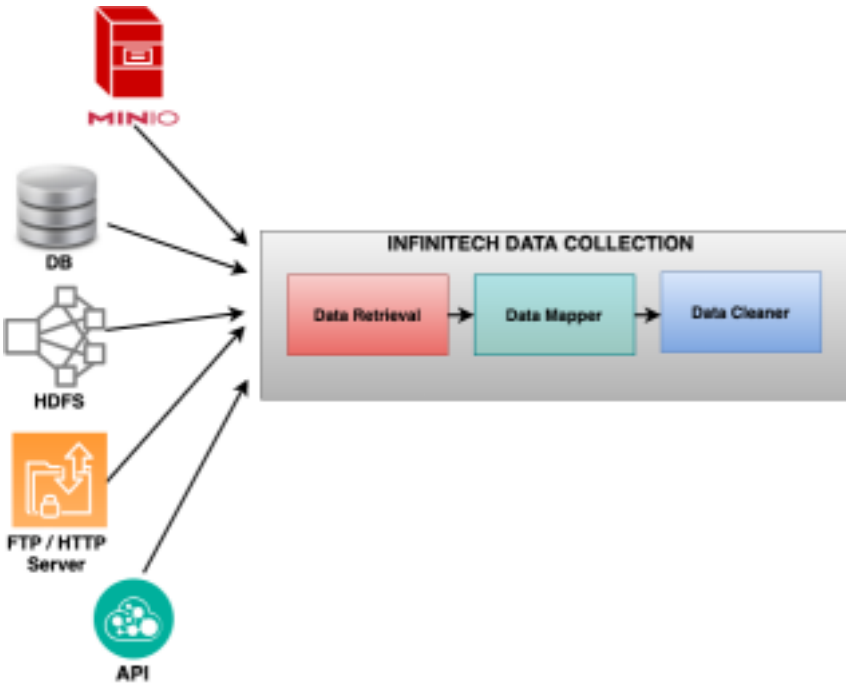

Figure 1: High-level architecture of the INFINITECH Data Collection

In the following subsections, the functionalities of the three main modules are presented in detail.

2.1.1 Data Retrieval

The scope of the Data Retrieval is to facilitate the data acquisition from any relational database, HDFS deployments, FTP or HTTP servers, MinIO storage servers as well as from any API of the data source. The prerequisite is that the appropriate information is provided by the data provider to the process prior to its invocation. Additionally, the Data Retrieval enables the reception of new datasets that are pushed from the data provider to its exposed RESTful APIs. Hence, the Data Retrieval supports all the aforementioned data source types which are considered the most commonly used data source types in every Big Data ecosystem. Nevertheless, the modular architecture of the described solution facilitates the expansion of the list of supported data source types in an effortless and effective manner upon the project's needs.

The process is able to retrieve new information either periodically or on-demand and can be automated as a background process using the concept of data source profiles that are configured by the data provider based on their needs.

Hence, the main functionalities of the Data Retrieval module are as follows:

a) The retrieval of new dataset from a data source by establishing the required connection to the data source's API and pulling the provided information based on the provided configuration b) The receival (pushing) of new dataset from a data source and local storage of the received data through well-defined RESTful APIs

c) The retrieval and fetching of files from an FTP or HTTP server by establishing the required connection and downloading the files locally

d) The data retrieval of new data from Relational Databases by establishing the appropriate connection and executing the appropriate query based on the provided configuration

e) The retrieval and fetching of files from an HDFS deployment or MinIO storage server by establishing the required connection and fetching the files locally

f) The creation of a data source profile that contains all the relevant information of properly connecting and retrieving new datasets from a specific data source which enables the automation of the data retrieval process

g) The periodic (in preconfigured time intervals) or immediate and on-demand retrieval of new datasets based on the provided configuration in the data source profile.

2.1.2 Data Mapper

The scope of the Data Mapper is to enable the mapping of the data entities included on a new dataset and the data model that is provided by the data provider. In this sense, the data provider is able to create the mappings for each entity of the new dataset to a specific data entity of the data model. To achieve this, at first the Data Mapper offers the means to integrate a data model during its initial configuration. Then, during processing, the data entities of the provided dataset are extracted and displayed to the data provider via its user friendly and easy-to-use user interface. Through this user interface the data provider is able to select the corresponding entities of the integrated data model that will be mapped to the entities of the dataset. The generated mappings are stored for later reuse in a JSON format.

The whole process can be performed as a background process also, if the data provider creates the corresponding data mapping profile for a specific dataset beforehand, which will be used in an automated way for the execution of the process, when a new dataset for the specific profile is received.

Hence, the main functionalities of the Data Mapper module, are as follows:

a) The integration of data model that is provided by the data provider during the module configuration
b) The extraction of the data entities of the input dataset which are represented to the data provider c) The display of the entities of the integrated data model
d) The creation of the mapping between the data entities of the input dataset and the entities of the integrated data model based on the data provider selection
e) The generation and storage of the produced mapping in JSON format
f) The creation of a data mapping profile that contains all the relevant information of generating the mapping in the form of mapping rules between the entities of the input dataset and the entities of the integrated data model
g) The automated or on-demand execution via API of the mapping process in the case of existence of data mapping profile.

It should be noted at this point that the development of a data model is out of scope of T5.1 and WP5 and that it is the responsibility of the data provider that exploits the Data Mapper module to provide it. However, the Data Mapper provides the means to fetch, interpret and integrate the provided data model in the mapping process via the module's internal configuration.

2.1.3 Data Cleaner

The scope of the Data Cleaner is to provide the data cleaning operations that will ensure that the provided input datasets, that are originating from a variety of heterogeneous data sources, are clean and complete to the extent possible. The specific functionalities enable the detection and correction (or removal) of inaccurate, corrupted or incomplete values in the data entities of the datasets towards the increase of the data quality of the specific data, as well as its value during data processing. To this end, the Data Cleaning operations are performed on top of the input dataset in order to produce the "cleaned" results.

Under the hood, the data cleaning process is a four-step process that includes:

a) the validation of the values of the data entities against a set of constraints,
b) the correction of the errors identified based on a set of data correction operations, c) the data completion of the values for the required/mandatory data entities with missing values with a set of data completion operations and
d) the maintenance of complete history records containing the history of errors identified and the data cleaning operations that were performed to address them.

The cleaning process is based on the set of data cleansing rules that are defined by the data provider on a data entity level via the Data Cleaner's user friendly and easy-to-use user interface and include the data validation rules, as well as the data correction and data completion actions performed for these rules.

In the same manner as with the Data Mapper, the Data Cleaner has been designed in a way that enables the execution of the process as a background process with the only prerequisite being the creation of a data cleaning profile for a specific dataset beforehand that will be leveraged in order to automated the process execution upon the receival of a new dataset for which the corresponding data cleaning profile can be applied.

To this end, the main functionalities of the Data Cleaner module are as follows:

a) The extraction of the data entities of the input dataset which are represented to the data provider b) The definition of data cleaning rules for each extracted data entity that include the validation of the values of the specific data entity against a set of constraints, the desired data correction operation or the missing value handling operation
c) The execution of the data cleaning operations on the input dataset based on the set of data cleaning rules defined by the data provider towards the generation of the "cleaned" dataset
d) The creation of a data cleaning profile that contains all the data cleaning rules of the specific dataset
e) The automated or on-demand execution via API of the cleaning process in the case of existence of data cleaning profile.

2.2 Use Cases and Sequence Diagrams

As explained in Section 2.1, the INFINITECH Data Collection is composed of three main modules, namely the Data Retrieval, the Data Mapper and the Data Cleaner. In this section, the detailed documentation of all the use cases encapsulated on each module are presented describing in detail all the information of each use case. Furthermore, for each use case the corresponding sequence diagram that depicts the interactions of the involved stakeholders and the involved components are also presented.

2.2.1 Data Retrieval

2.2.1.1 API to pull information

The specific functionality undertakes the responsibility of retrieving (pulling) information from a specific data source provider either upon the triggering of the process (on-demand pull) or in a predefined time interval (scheduled pull). To initiate the process, the mechanism of the Data Retriever requires the definition of the configuration (profile) of the specific data source. The configuration contains all the connections details required in order to properly access and retrieve the required information in the form of JSON file from a data source's APIs. The acquired information is locally stored in order to be used in the next steps of the process.

Table 1: Definition of the Data Source Profile (API)

Stakeholders	Data Source Provider
Pre	1. The existence of an accessible API capable of providing the requested information upon the successful connection and request 2. The existence of the configuration details of the respective API endpoint
Post	1. The data source profile is available

	1. Data Source Name

	2. Data Source Owner 3. Data Source Type = API 4. API URL path 5. Authentication Details (Username/Password, JWT, etc) 6. Pull time interval (None or number of seconds, minutes, days)
	1. The data source provider initiates a request to the Data Retrieval's API endpoint to register the new data source profile 1. An acknowledgement is returned and the data source profile is stored in the list of known data sources
	1. The new data source profile is available in the list of known sources for the data retrieval process
	1. The new data source profile is not available in the list of known data sources

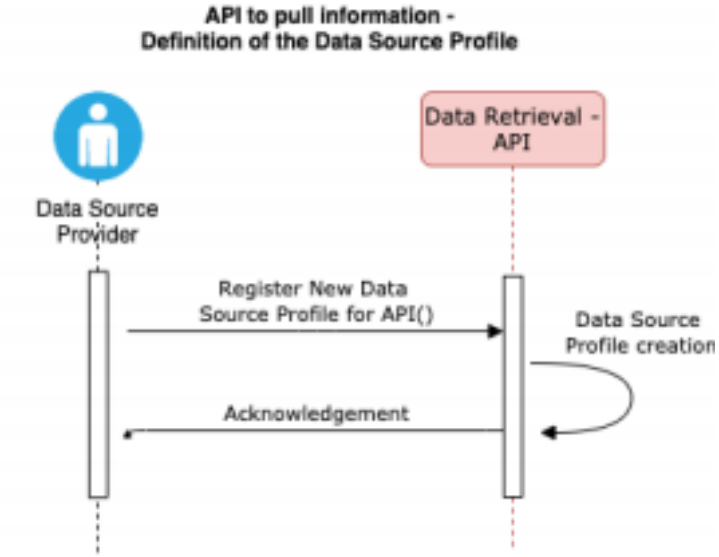

Figure 2: API to pull information – Definition of the data source profile (API)

Upon the successful creation of the data source profile in the previous step, the data retrieval via an API of the data provider can be triggered on-demand. In this case, the Data Retrieval component will initiate a new pull request to the data provider's API, utilising the information included in the data source profile and the new dataset will be retrieved and stored locally for further processing.

Table 2: API Data Retrieval (on-demand)

Stakeholders	Data Source Provider
Pre	1. The existence of the data source profile

Post	1. New information has been retrieved and stored locally for further processing
	1. Data Source Profile ID

	1. The data source provider initiates a request to the Data Retrieval's API endpoint to retrieve new information based on a specific data source profile 2. The Data Retrieval initiates the request to the respective API and retrieves the new information 3. The new information is stored locally in a JSON format
	1. The new information is available locally as a JSON format file
	1. The request is rejected and no information is retrieved from the API of the data source profile

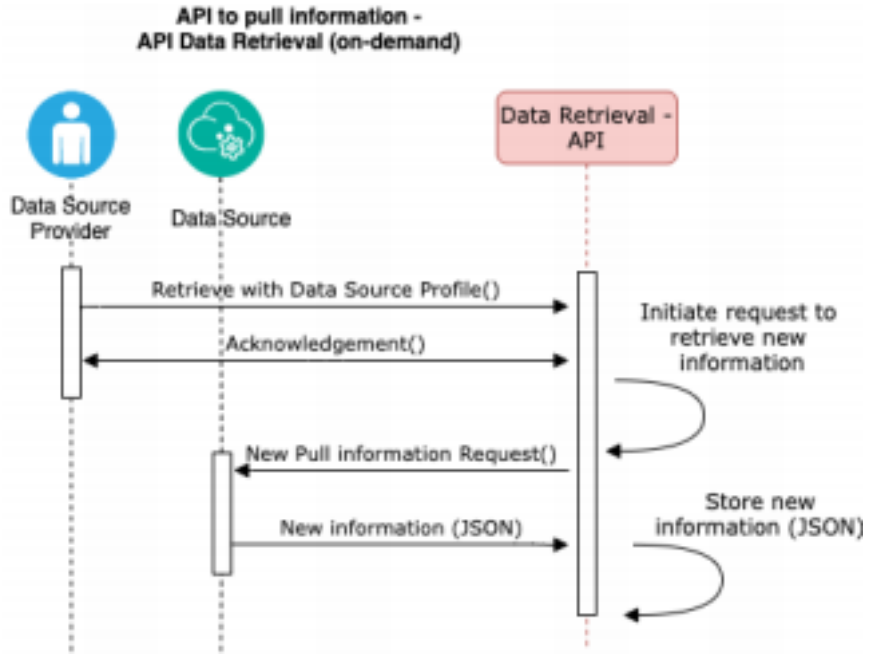

Figure 3: API Data Retrieval (on-demand)

An alternative way is to preconfigure the process via the respective information in the data source profile to be executed in predefined time intervals (scheduled pull). In this case, the Data Retrieval component will initiate a new pull request to the data provider's API based on the schedule defined in the data source profile and the new dataset will be retrieved and stored locally for further processing.

Table 3: API Data Retrieval (scheduled)

Stakeholders	N/A
Pre	1. The existence of the data source profile 2. The Pull time interval is set in the respective data source profile

Post	1. New information is retrieved and stored locally for further processing based on the configured pull time interval
	1. Data Source Profile ID

	1. The Data Retrieval initiates the request to the respective API and retrieves the new information based on a specific data source profile and the configured pull time interval 2. The new information is stored locally in a JSON format
	1. The new information is available locally as a JSON format file
	1. The request is rejected and no information is retrieved from the API of the data source profile

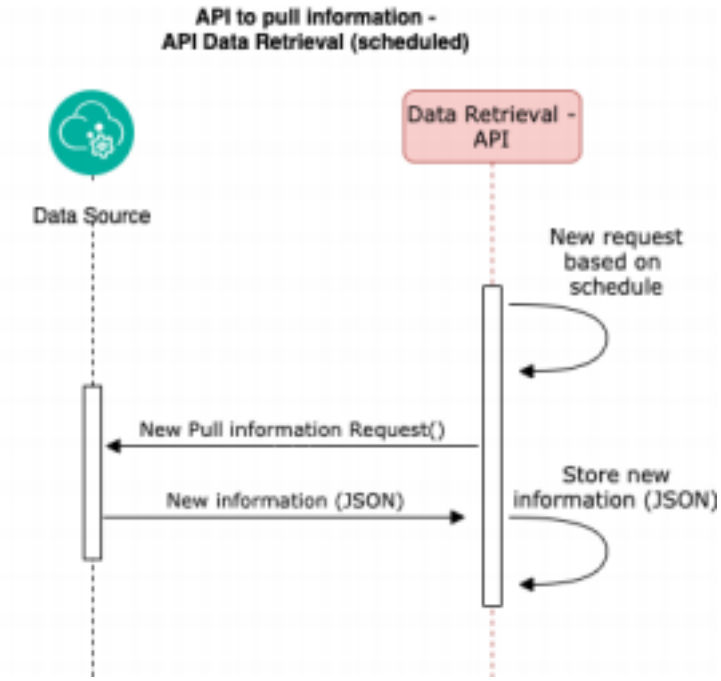

Figure 4: API Data Retrieval (scheduled)

2.2.1.2 API to push information

An alternative option for the data provider to ingest new datasets is the utilisation of the RESTful API that is provided by the Data Retrieval. In detail, the data provider initiates a request to the Data Retrieval APIs to push new information providing the required information and the new dataset in the form of an attachment file or in the request body in the JSON format. The Data Retrieval processes the request and stores the received file locally for further processing.

Table 4: API to push information

Stakeholders	Data Source Provider

Pre	N/A
Post	1. New information is pushed to the Data Retrieval and stored locally for further processing
	1. Data Type (CSV, JSON, XML) 2. Data Source Name

	3. Data Source Owner 4. Authorisation Token (JWT) 5. Data Source File (included as an attachment or in the request body)
	1. The data source provider initiates a request to the Data Retrieval's API endpoint to push new information 2. The Data Retrieval's retrieves the new information stores it locally in the provided file format for further processing
	1. The new information is available locally in the provided file format
	1. The request is rejected and no information is stored locally

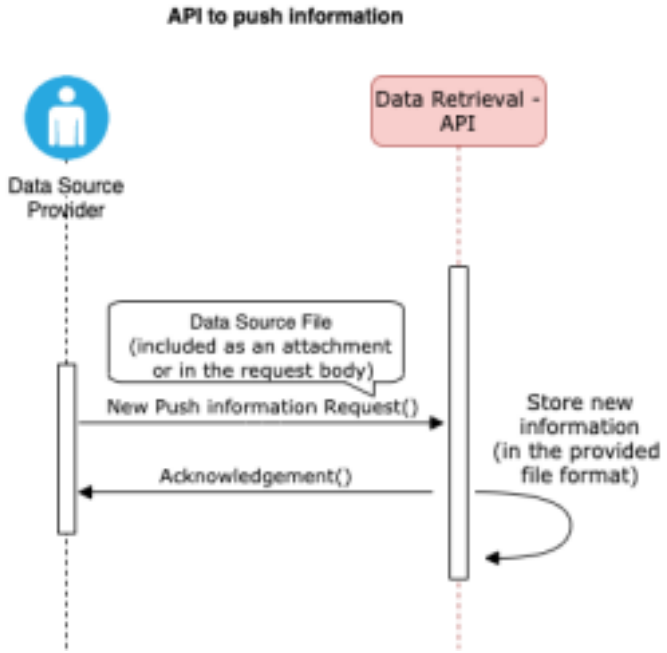

Figure 5: API to push information

2.2.1.3 Fetch files from an FTP / HTTP server

The specific functionality undertakes the responsibility of retrieving (pulling) files from a specific data source provider's FTP or HTTP server, either upon the triggering of the process (on-demand pull) or in a predefined time interval (scheduled pull). To initiate the process, the mechanism of the Data Retriever requires the definition of the configuration (profile) of the FTP or HTTP server. The configuration contains all the connection details required in order to properly access and retrieve the required file from the data source

provider's FTP or HTTP server. The acquired information is locally stored in order to be used in the next steps of the process.

Table 5: Definition of the Data Source Profile (FTP or HTTP)

Stakeholders	Data Source Provider
Pre	1. The existence of an accessible FTP or HTTP server capable of providing the requested information upon the successful connection and request 2. The existence of the configuration details of the respective FTP/HTTP server

Post	1. The data source profile is available
	1. Data Source Name 2. Data Source Owner 3. Data Source Type = FTP or HTTP 4. File path 5. Connection URL 6. Connection Port 7. Connection Username 8. Connection Password 9. Pull time interval (None or number of seconds, minutes, days)
	1. The data source provider initiates a request to the Data Retrieval's API endpoint to register the new data source profile 2. An acknowledgement is returned and the data source profile is stored in the list of known data sources
	1. The new data source profile is available in the list of known sources for the data retrieval process
	1. The new data source profile is not available in the list of known data sources

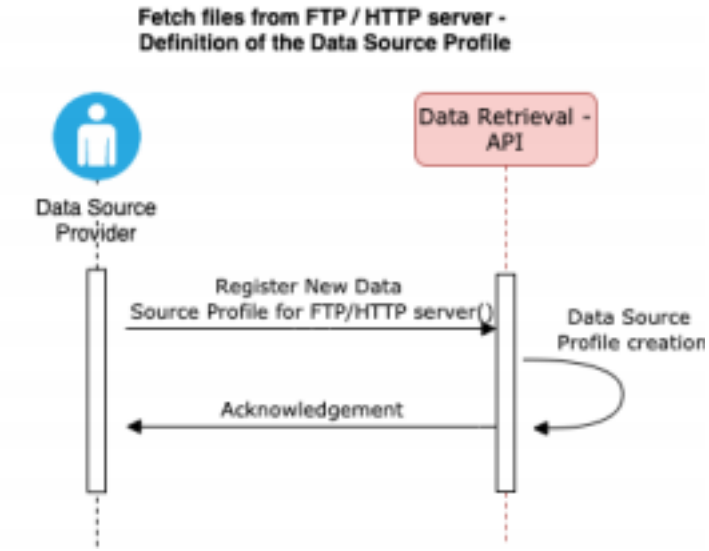

Figure 6: Definition of the Data Source Profile (FTP or HTTP)

Upon the successful creation of the data source profile in the previous step, the data retrieval process can be triggered via the API of Data Retrieval from the data provider on-demand. In this case, the Data Retrieval component will initiate a new request to the respective FTP or HTTP, utilising the information included in the data source profile and the new file will be retrieved and stored locally for further processing.

Table 6: FTP or HTTP Data Retrieval (on-demand)

Stakeholders	Data Source Provider

Pre	1. The existence of the data source profile
Post	1. New information is retrieved and stored locally for further processing
	1. Data Source Profile ID
	1. The data source provider initiates a request to the Data Retrieval's API endpoint to retrieve new information from the FTP or HTTP server based on a specific data source profile 2. The Data Retrieval initiates the request to the respective server and retrieves the new information 3. The new information is stored locally in the provided file format
	1. The new information is available locally in the provided file format
	1. The request is rejected and no information is retrieved from the server defined in the data source profile

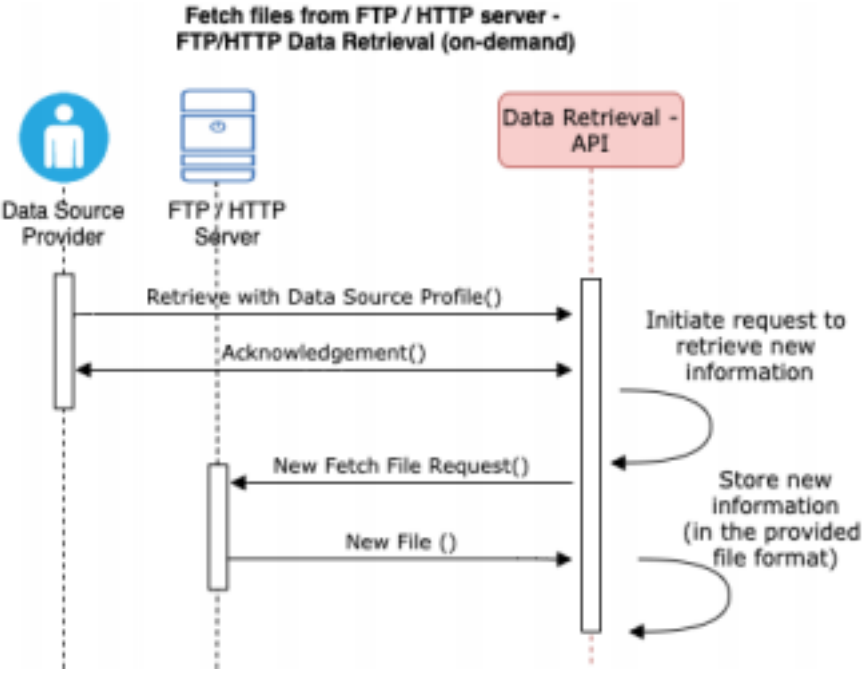

Figure 7: FTP or HTTP Data Retrieval (on-demand)

An alternative option provided is to preschedule the execution of the process via the respective information in the data source profile in order to be executed in predefined time intervals (scheduled pull). In this case, the Data Retrieval component will initiate a new request to the respective FTP or HTTP based on the schedule defined in the data source profile and the new file will be retrieved and stored locally for further processing.

Table 7: FTP or HTTP Data Retrieval (scheduled)

Stakeholders	N/A

Pre	1. The existence of the data source profile 2. The Pull time interval is set in the respective data source profile
Post	1. New information isretrieved and stored locally for further processing based on the configured pull time interval
	1. Data Source Profile ID
	1. The Data Retrieval initiates the request to the respective server and retrieves the new information based on a specific data source profile and the configured pull time interval 2. The new information is stored locally in the provided file format
	1. The new information is available locally in the provided file format
	1. The request is rejected and no information is retrieved from the server defined in the data source profile

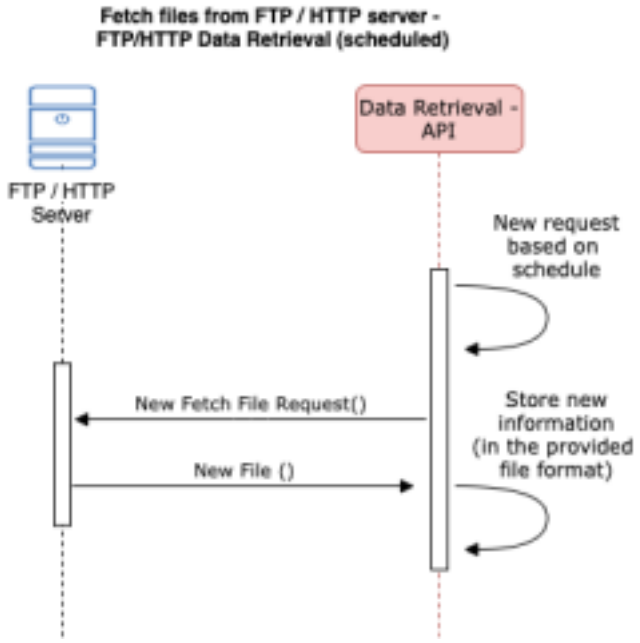

Figure 8: FTP or HTTP Data Retrieval (scheduled)

2.2.1.4 Relational Databases

This functionality undertakes the responsibility of retrieving new data from Relational Databases by establishing the appropriate connection and executing the appropriate query based on the provided configuration. The operation can be executed either upon the triggering of the process (on-demand) or in a predefined time interval (scheduled). To initiate the process, the mechanism of the Data Retriever requires the definition of the configuration (profile) of the specific relational database, containing all the connection details required in order to properly access and retrieve the required information from the respective database. The acquired information is locally stored in order to be used in the next steps of the process.

Table 8: Definition of the Data Source Profile (DB)

Stakeholders	Data Source Provider

Pre	1. The existence of an accessible SQL Database capable of providing the requested information upon the successful connection and request 2. The existence of the configuration details of the respective DB deployment
Post	1. The data source profile is available
	1. Data Source Name 2. Data Source Owner 3. Data Source Type = DB 4. DB name 5. Executed DB Query 6. Connection URL 7. Connection Port 8. Connection Username 9. Connection Password

	10. Pull time interval (None or number of seconds, minutes, days)
	1. The data source provider initiates a request to the Data Retrieval's API endpoint to register the new data source profile 2. An acknowledgement is returned and the data source profile is stored in the list of known data sources
	1. The new data source profile is available in the list of known sources for the data retrieval process
	1. The new data source profile is not available in the list of known data sources

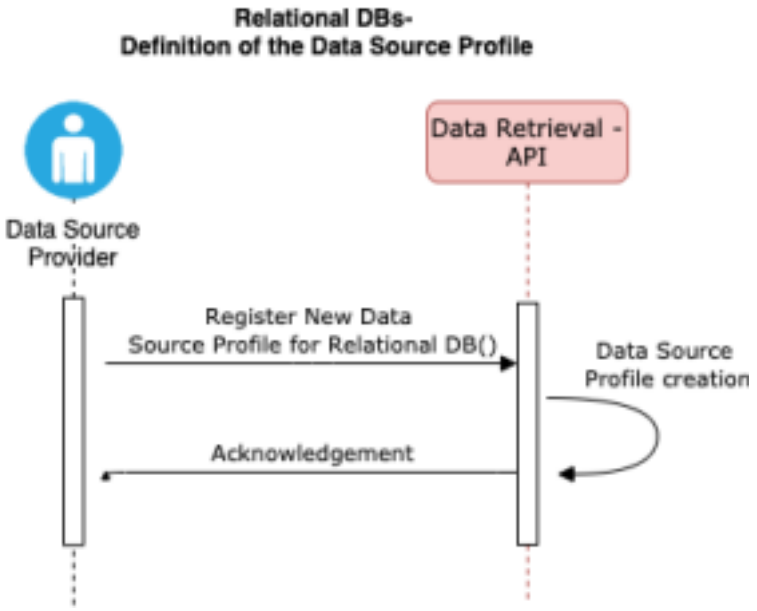

Figure 9: Definition of the Data Source Profile (DB)

Upon the successful creation of the data source profile in the previous step, the data retrieval process can be triggered via the API of Data Retrieval from the data provider on-demand. In this case, the Data Retrieval

component will initiate a new connection to the respective relational database utilising the information included in the data source profile, execute the defined query and retrieve the results and store them locally for further processing.

Table 9: DB Data Retrieval (on-demand)

Stakeholders	Data Source Provider
Pre	1. The existence of the data source profile
Post	1. New information is retrieved and stored locally for further processing
	1. Data Source Profile ID

	1. The data source provider initiates a request to the Data Retrieval's API endpoint to retrieve new information from the defined DB based on a specific data source profile 2. The Data Retrieval initiates the request to the respective DB and retrieves the new information 3. The new information is stored locally in the provided file format
	1. The new information is available locally in the provided file format
	2. The request is rejected and no information is retrieved from the server defined in the data source profile

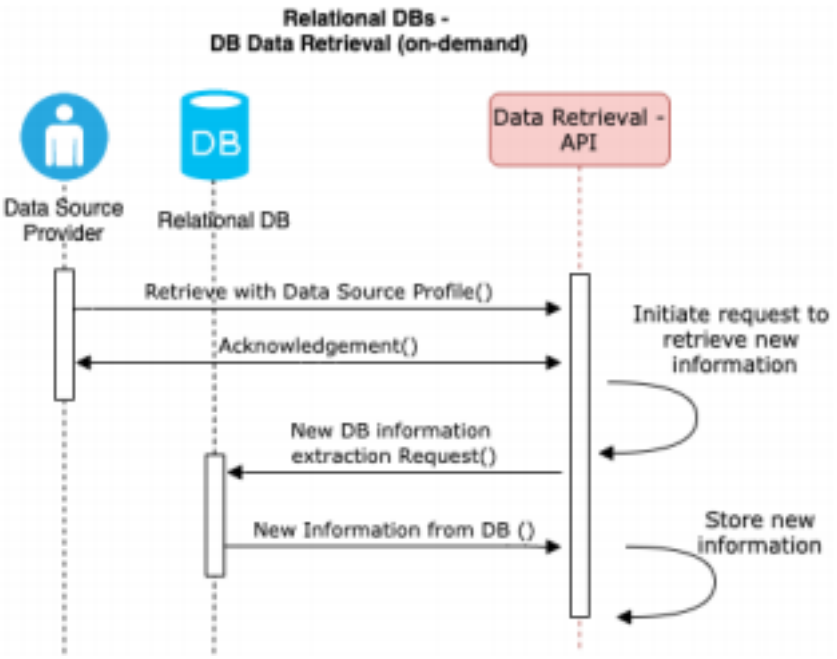

Figure 10: DB Data Retrieval (on-demand)

In the same manner as with the rest of the options, the execution of the process can be prescheduled by setting the respective information in the data source profile,so as to be executed in predefined time intervals

(scheduled pull). In this case, the Data Retrieval component will initiate a new connection to the respective relational database based on the schedule defined in the data source profile, execute the predefined query, retrieve the results and store them locally for further processing.

Table 10: DB Data Retrieval (scheduled)

Stakeholders	N/A
Pre	1. The existence of the data source profile 2. The Pull time interval is set in the respective data source profile
Post	1. New information isretrieved and stored locally for further processing

	based on the configured pull time interval
	1. Data Source Profile ID
	1. The Data Retrieval initiates the request to the respective DB and retrieves the new information based on a specific data source profile and the configured pull time interval 2. The new information is stored locally in the provided file format
	1. The new information is available locally in the provided file format
	1. The request is rejected and no information is retrieved from the server defined in the data source profile

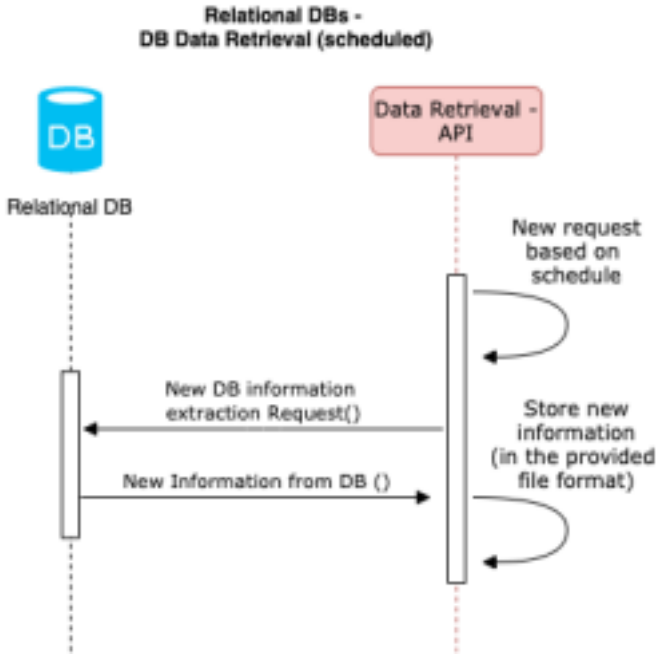

Figure 11: DB Data Retrieval (scheduled)

2.2.1.5 Retrieve files from HDFS

The specific functionality undertakes the responsibility of retrieving (pulling) files from a specific data source provider's HDFS deployment, either upon the triggering of the process (on-demand pull) or in a predefined time interval (scheduled pull). To initiate the process, the mechanism of the Data Retriever requires the

definition of the configuration (profile) of the HDFS. The configuration contains all the connection details required in order to properly access and retrieve the required file from the data source provider's HDFS deployment. The acquired information is locally stored in order to be used in the next steps of the process.

Table 11: Definition of the Data Source Profile (HDFS):

Stakeholders	Data Source Provider

Pre	1. The existence of an accessible HDFS deployment capable of providing the requested information upon the successful connection and request 2. The existence of the configuration details of the respective HDFS deployment
Post	1. The data source profile is available
	1. Data Source Name 2. Data Source Owner 3. Data Source Type = HDFS 4. HDFS File Path 5. Connection URL 6. Connection Port 7. Authentication details 8. Pull time interval (None or number of seconds, minutes, days)
	1. The data source provider initiates a request to the Data Retrieval's API endpoint to register the new data source profile 2. An acknowledgement is returned and the data source profile is stored in the list of known data sources
	1. The new data source profile is available in the list of known sources for the data retrieval process
	1. The new data source profile is not available in the list of known data sources

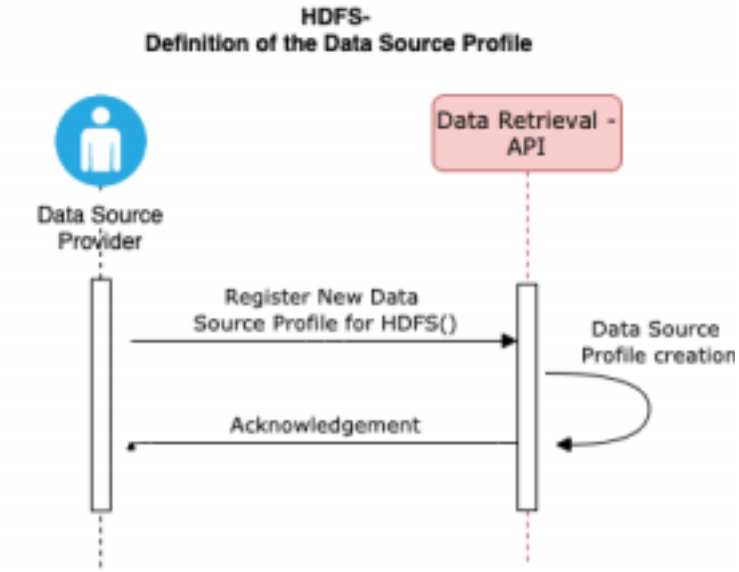

Figure 12: Definition of the Data Source Profile (HDFS)

Upon the successful creation of the data source profile in the previous step, the data retrieval process can be triggered via the API of Data Retrieval from the data provider on-demand. In this case, the Data Retrieval component will initiate a new request to the respective HDFS deployment, utilising the information

included in the data source profile and the new file will be retrieved and stored locally for further processing.

Table 12: HDFS Data Retrieval (on-demand)

Stakeholders	Data Source Provider
Pre	1. The existence of the data source profile
Post	1. New information is retrieved and stored locally for further processing
	1. Data Source Profile ID
	4. The data source provider initiates a request to the Data Retrieval's API endpoint to retrieve new information from the defined HDFS deployment based on a specific data source profile 5. The Data Retrieval initiates the request to the respective HDFS deployment and retrieves the new information 6. The new information is stored locally in the provided file format
	2. The new information is available locally in the provided file format
	3. The request is rejected and no information is retrieved from the server defined in the data source profile

Figure 13: HDFS Data Retrieval (on-demand)

On the other hand, the execution of the process can be executed in predefined time intervals (scheduled pull). In this case, the Data Retrieval component will initiate a new connection to the HDFS deployment based on the schedule defined in the data source profile, retrieve the new file and store it locally for

further processing.

Table 13: HDFS Data Retrieval (scheduled)

Stakeholders	N/A
Pre	1. The existence of the data source profile 2. The Pull time interval is set in the respective data source profile
Post	1. New information isretrieved and stored locally for further processing based on the configured pull time interval
	1. Data Source Profile ID
	1. The Data Retrieval initiates the request to the respective HFDS deployment and retrieves the new information based on a specific data source profile and the configured pull time interval 2. The new information is stored locally in the provided file format
	1. The new information is available locally in the provided file format
	1. The request is rejected and no information is retrieved from the server defined in the data source profile

Figure 14: HDFS Data Retrieval (scheduled)

2.2.1.6 Retrieve files from MinIO

The specific functionality undertakes the responsibility of retrieving (pulling) files from a specific data source provider's MinIO storage server. The process can be triggered at any time (on-demand pull) or automatically in a predefined time interval (scheduled pull). As with the rest of the cases, Data Retriever requires the definition of the configuration (profile) of the MinIO storage server that should contain all the required information (connection details), in order to be able to establish a connection to the storage server and

retrieve the required file. The retrieved file is locally stored and is fed to the Data Mapper or Data Cleaner for further processing.

Table 14: Definition of the Data Source Profile (MinIO):

Stakeholders	Data Source Provider
Pre	1. The existence of an accessible MinIO storage server capable of providing the requested information upon the successful connection and request 2. The existence of the configuration details of the respective MinIO storage server
Post	1. The data source profile is available
	1. Data Source Name 2. Data Source Owner 3. Data Source Type = MinIO 4. Bucket Name 5. File Path 6. Endpoint 7. Access Key 8. Access Secret 9. Pull time interval (None or number of seconds, minutes, days)

	1. The data source provider initiates a request to the Data Retrieval's API endpoint to register the new data source profile 1. An acknowledgement is returned and the data source profile is stored in the list of known data sources
	1. The new data source profile is available in the list of known sources for the data retrieval process
	1. The new data source profile is not available in the list of known data sources

Figure 15: Definition of the Data Source Profile (MinIO)

Once the data source profile has been successfully registered in accordance with the previous step, the data retrieval process can be initiated by utilising the API of Data Retrieval and the on-demand retrieval of the requested file is triggered. Under the hood, the Data Retrieval component will initiate and establish a connection to the specified MinIO storage server, based on the information included in the data source profile and pull locally the requested file.

Table 15: MinIO Data Retrieval (on-demand)

Stakeholders	Data Source Provider
Pre	1. The existence of the data source profile
Post	1. New information is retrieved and stored locally for further processing
	1. Data Source Profile ID
	1. The data source provider initiates a request to the Data Retrieval's API endpoint to retrieve new information from the specific MinIO storage server based on a specific data source profile

	2. The Data Retrieval initiates the request to the respective MinIO storage server and retrieves the new information 3. The new information is stored locally in the provided file format
	1. The new information is available locally in the provided file format
	1. The request is rejected and no information is retrieved from the server defined in the data source profile

Figure 16: MinIO Data Retrieval (on-demand)

While the process can be triggered and executed at any time, there is also the option to execute the process automatically in predefined time intervals (scheduled pull) provided that the respective parameter is set in the data source profile. In this case, the Data Retrieval component initiates a new connection to the MinIO storage based on the Pull time interval that is set in the data source profile, retrieves the new file and stores it locally for further processing.

Table 16: MinIO Data Retrieval (scheduled)

Stakeholders	N/A
Pre	1. The existence of the data source profile 2. The Pull time interval is set in the respective data source profile
Post	1. New information isretrieved and stored locally for further processing based on the configured pull time interval
	1. Data Source Profile ID
	1. The Data Retrieval initiates the request to the respective MinIO storage server and retrieves the new information based on a specific data source profile and the configured pull time interval 2. The new information is stored locally in the provided file format

	1. The new information is available locally in the provided file format
	1. The request is rejected and no information is retrieved from the server defined in the data source profile

Figure 17: MinIO Data Retrieval (scheduled)

2.2.2 Data Mapper

2.2.2.1 Data Mapping (on demand without profile)

For the Data Mapping process, the data provider is able to perform an on-demand mapping process for a specific dataset without the use of data mapping profile. This process can be executed only through the user interface of the Data Mapper, as the data provider will be instructed to provide their input by selecting the proper entities of the underlying data model that will be mapped to the entities that are extracted by their dataset. During this process, the data provider is given the option to save the provided information as a data mapping profile that can be used to automated the process in the future executions.

Table 17: Data Mapping (on demand without profile)

Stakeholders	Data Source Provider
Pre	1. A new dataset is available for the data mapping process 2. The existence of a data model that is integrated during the process start-up
Post	1. The data mapping of the data entities of the datasets with the underlying integrated data model is available
	N/A
	1. The data source provider selects the dataset that will be utilised as input in the data mapping process.

	2. The data entities of the dataset are extracted and presented to the data source provider 3. For each data entity, the data source provider selects the corresponding entity from the underlying integrated data model that the data entity will be mapped 4. The data source provider is presented with the option to save the provided information as a data mapping profile for the specific dataset 5. The data mapping between the dataset and the underlying integrated data model l is produced and stored locally in JSON format
	1. The data mapping is available in JSON format 2. In case the data mapping profile option is selected, the existence of the data mapping profile for later reuse
	1. The data mapping is not available. 2. In case the data mapping profile option is selected, the data mapping profile is missing

Figure 18: Data Mapping (on demand without profile)

2.2.2.2 Data Mapping (on demand with profile)

For the Data Mapping process, the data provider is able to perform an on-demand mapping process for a specific dataset, utilising an existing data mapping profile. This process can be executed through the user interface of the Data Mapper and upon selection of the data mapping profile. The profile is then applied to the specific dataset and the results are available for verification by the data source provider. Upon verification, the data mapping is produced and stored for later use.

Table 18: Data Mapping (on demand with profile)

Stakeholders	Data Source Provider
Pre	1. A new dataset is available for the data mapping process 2. The existence of the underlying integrated data model 3. The existence of the data mapping profile for the specific dataset
Post	1. The data mapping of the data entities of the datasets with the underlying integrated data model is available
	1. Data Mapping Profile ID
	1. The data source provider selects the dataset that will be utilised as input in the data mapping process. 2. The data entities of the dataset are extracted and presented to the data source provider 3. The data source provider selects the data mapping profile for the specific dataset in order to be applied 4. The data mapping profile is applied to the specific dataset is available for verification by the data source provider 5. Upon verification, the data mapping between the dataset and the underlying integrated data model is produced and stored locally in JSON format
	1. The data mapping is available in JSON format
	1. The data mapping is not available.

Figure 19: Data Mapping (on demand with profile)

2.2.2.3 Data Mapping Profile Registration via API

An alternative way of utilising the Data Mapper is with the use of the provided RESTful APIs. However, this option requires the definition of the data mapping profile of the specific dataset. The data mapping profile contains all the required information that will enable the automation of the data mapping process.

Table 19: Data Mapping Profile Registration via API

Stakeholders	Data Source Provider
Pre	1. The existence of the configuration details for the data mapping of the datasets
Post	1. The data mapping profile is available
	1. Data Source Name 2. Data Source Owner 3. Data Mapping rules
	1. The data source provider initiates a request to the Data Mapper's API endpoint to register the new data mapping profile 2. An acknowledgement is returned and the data mapping profile is stored in the list of data mapping profiles
	1. The new data mapping profile is available in the list of data mapping profiles for the data mapping process
	1. The new data mapping profile is not available in the list of known data mapping profiles

Figure 20: Data Mapping Profile Registration via API

2.2.2.4 Data Mapping (via API with profile)

The data mapping process can be triggered via the respective RESTful APIs, with the only precondition being the existence of a data mapping profile. The data source provider can initiate a request to the Data Mapper's API endpoint to perform the data mapping on the selected dataset based on a specific data mapping profile. Alternatively, the Data Retrieval could be the one to trigger the Data Mapper as part of the automated end
to-end data collection process.

Table 20: Data Mapping (via API with profile)

Stakeholders	Data Source Provider or Data Retrieval
Pre	1. A new dataset is available for the data mapping process 2. The existence of the common information mode of the INFINITECH 3. The existence of the data mapping profile for the specific dataset
Post	1. The data mapping of the data entities of the datasets with the common information mode of the INFINITECH is available
	1. Dataset Name 2. Local file path of the dataset 3. Data Mapping Profile ID

	1. The data source provider or Data Retrieval initiates a request to the Data Mapper's API endpoint to perform the data mapping on the selected dataset based on a specific data mapping profile 2. The Data Mapper performs the data mapping operations on the selected dataset and produces the data mapping. 3. The new data mapping information is stored locally in a JSON format
	1. The data mapping is available in JSON format
	1. The data mapping is not available.

Figure 21: Data Mapping (via API with profile)

2.2.3 Data Cleaner

2.2.3.1 Data Cleaning (on demand without profile)

In the same manner as for the data mapping, for the Data Cleaning process, the data provider is able to perform an on-demand cleaning process for a specific dataset without the use of data cleaning profile. This process can be executed only through the user interface of the Data Cleaner, since the input from the data provider is required in order to define the cleaning rules that include the validation rules, as well as the data cleaning and data completion actions that will be performed in case the validation identifies an error. During this process, the data provider is given the option to save the provided information as a data cleaning profile that can be used to automated the process in future executions.

Table 21: Data Cleaning (on demand without profile)

Stakeholders	Data Source Provider
Pre	1. A new dataset is available for the data cleaning process
Post	1. The produced "cleaned" dataset is available

	N/A
	1. The data source provider selects the dataset that will be utilised as input in the data cleaning process. 2. The data entities of the dataset are extracted and presented to the data source provider 3. For each data entity, the data source provider selects the data validation rule, as well as the data cleaning and data completion action that will be performed in case of a data validation error. 4. The data source provider is presented with the option to save the provided information as a data cleaning profile for the specific dataset 5. The data cleaning operation is performed and the “cleaned” dataset is available.
	1. The “cleaned” dataset is available 2. In case the data cleaning profile option is selected, the existence of the data cleaning profile for later reuse
	1. The data cleaning is not performed. 2. In case the data cleaning profile option is selected, the data cleaning profile is missing

Figure 22: Data Cleaning (on demand without profile)

2.2.3.2 Data Cleaning (on demand with profile)

In the case where a data cleaning profile exists, the Data Cleaning process can be executed by the data provider on demand through the user interface of the Data Cleaner. In detail, upon the selection of the

input dataset, the entities of the dataset are extracted and the data provider can select a data cleaning profile that can be used in the process. The Data Cleaning process is executed by applying the cleaning rules set in this profile and the "cleaned" dataset is available for further processing.

Table 22: Data Cleaning (on demand with profile)

Stakeholders	Data Source Provider
Pre	1. A new dataset is available for the data cleaning process 2. The existence of the data cleaning profile for the specific dataset
Post	1. The produced "cleaned" dataset is available
	1. Data Cleaning Profile ID
	1. The data source provider selects the dataset that will be utilised as input in the data cleaning process. 2. The data source provider selects the data cleaning profile for the specific dataset in order to be applied 3. The data cleaning profile is applied to the specific dataset is available for verification by the data source provider 4. Upon verification, the data cleaning operation is performed and the "cleaned" dataset is available.
	1. The "cleaned" dataset is available

	1. The data cleaning is not performed.

Figure 23: Data Cleaning (on demand with profile)

2.2.3.3 Data Cleaning Profile Registration via API

The execution of the Data Cleaning process can be performed also by utilising the RESTful APIs of the Data Cleaner. In this case, the first step is the registration of a data cleaning profile for a specific dataset via the respective API. The data cleaning profile contains all the required information that will enable the automation of the data cleaning process.

Table 23: Data Cleaning Profile Registration via API

Stakeholders	Data Source Provider
Pre	1. The existence of the configuration details for the data cleaning of the datasets
Post	1. The data cleaning profile is available
	1. Data Source Name 2. Data Source Owner 3. Data Cleaning rules
	1. The data source provider initiates a request to the Data Cleaner's API endpoint to register the new data cleaning profile 2. An acknowledgement is returned and the data cleaning profile is stored in the list of data cleaning profiles
	1. The new data cleaning profile is available in the list of data cleaning profiles for the data cleaning process

	1. The new data cleaning profile is not available in the list of known data cleaning profiles

Figure 24: Data Cleaning Profile Registration via API

2.2.3.4 Data Cleaning (via API with profile)

The data cleaning process can be triggered via the respective RESTful APIs of the Data Cleaner, provided that a data cleaning profile has been registered beforehand. In this case, the data source provider can initiate a request to the Data Cleaner's API endpoint to perform the data cleaning on the selected dataset based on a specific data cleaning profile. Furthermore, in terms of automation of the end-to-end data collection process, the Data Mapper could trigger the Data Cleaner in order to perform the next step in the process.

Table 24: Data Cleaning (via API with profile)

Stakeholders	Data Source Provider or Data Mapper
Pre	1. A new dataset is available for the data cleaning process 2. The existence of the data cleaning profile for the specific dataset
Post	1. The produced "cleaned" dataset is available
	1. Dataset Name 2. Local file path of the dataset 3. Data Cleaning Profile ID
	1. The data source provider or Data Mapper initiates a request to the Data Cleaner API endpoint to perform the data cleaning on the selected dataset based on a specific data cleaning profile 2. The Data Cleaner performs the data cleaning operations on the selected dataset and produces the "cleaned" dataset.

	1. The "cleaned" dataset is available
	1. The data cleaning is not performed.

Figure 25: Data Cleaning (via API with profile)

2.2.4 Data Collection Usage Examples

As documented in Section 2.1, the INFINITECH Data Collection has a modular architecture composed of three distinct modules, namely the Data Retrieval, the Data Mapper and the Data Cleaner, which are designed with a variety of functionalities. One of the core aspects of the design specifications of all three modules, is their high level of modularity and that they are highly configurable in terms of both operation and integration. As a result, they enable the design and execution of many different data collection pipelines that are tailored to the needs of each data provider. In this sense, a data collection pipeline can be designed by the data provider in a way that the data provider can utilise the user interfaces of all three modules (or any number of them) to execute a data collection process, where in this case manual steps, executed by the data provider, will be included in the process. On the other hand, a data collection pipeline can be designed by the data provider in order to operate in a fully automated manner, in which case the whole process is executed via the use of APIs without any manual intervention.

In the following paragraphs, two main examples of the utilisation of the INFINITECH Data Collection are presented. In the first example, the data provider utilises the user interfaces of all three modules in order to retrieve a new dataset from the API of a specific data source.

Hence, in the first step, the data provider registers the new data source profile for the respective API in the Data Retrieval via its user interface. Upon the creation of the new data source profile, the data source provider invokes via the user interface the retrieval of the new dataset based on this specific data source profile. Under the hood, the Data Retrieval initiates a new request to the respective API and fetches the new dataset in the form of a JSON file.

Once the dataset retrieval is completed, the data provider is moved to the next step that includes the data mapping process. The data provider exploits the user interface of the Data Mapper in order to select the newly fetched dataset. Upon the selection of the dataset, the data entities of the dataset are extracted and presented to the data source provider. The data source provider is able to select the entity from the underlying integrated data model that correspond to each specific dataset entity and create the respective data mapping.

Upon the successfully data mapping completion, the data source provider is moved to the third and final step that includes the data cleaning process. In the same manner, the data entities of the dataset are extracted and presented to the data source provider and the data source provider is able to set the data cleaning rules

for each entity of the dataset. Upon the completion of the data cleaning process, the produced “cleaned” dataset is available for later usage. The described example execution is depicted in Figure 26.

Figure 26: Data Collection Usage Example – Manual

On the other hand, the same process can be fully automated by exploiting the provided RESTful APIs of the INFINITECH Data Collection.

During the first step, the data source provider registers the corresponding data source profile, data mapping profile and data cleaning profile for the specific dataset that will be retrieved from the API of the data source. Once all profiles are successfully registered, the data source provider initiates the retrieval of the new dataset based on the defined data source profile via the corresponding Data Retrieval's API. The Data Retrieval initiates the request to the corresponding data source API and retrieves the new dataset.

Upon the successful retrieval of the new dataset, the data source provider is informed and they initiate the data mapping process with the respective data mapping profile via the Data Mapper's API. Upon the successful data mapping generation, the data provider is informed and they initiate the final step of the data cleaning process by invoking the respective API of the Data Cleaner with the corresponding data cleaning profile.

Once the data cleaning process is complete, the data provider is informed and the "cleaned" data are ready for usage. The described example execution is depicted in Figure 27.

Figure 27: Data Collection Usage Example – Automated via APIs

It should be noted that the presented end-to-end executions are only indicative use cases of the modularity and configurability of the INFINITECH Data Collection. As highlighted in the previous sections, the INFINITECH Data Collection can be configured in multiple ways depending on the needs of the data source provider, addressing multiple use cases where the three modules are properly configured and combined.

2.3 Implementation of the INFINITECH Data Collection

As described in section 2.1, the INFINITECH Data Collection is composed by three distinct modules namely the Data Retrieval, the Data Mapper and the Data Cleaner, which are integrated in order to formulate the configurable solution that enables the data source providers to perform data ingestion processes tailored to their needs. The implementation of each of the three modules is driven by the design specifications that are also documented in section 2.1 of the current deliverable. In the following subsections, the implementation details of each module are presented, providing a high-level overview of how the source code is organised with the help of UML diagrams along with the implementation details of the respective functions of each module.

2.3.1 Data Retrieval

Data Retrieval undertakes the responsibility of the data acquisition process, supporting the direct retrieval of new datasets from a variety of data sources such as FTP / HTTP servers, relational databases, HDFS deployments and MinIO storage servers, as well as through the APIs of a data source, and the receival of new dataset via its own exposed API. The process is facilitated with the use of data source profiles and can be executed in an on-demand (immediately) or scheduled manner. The class diagram of the Data Retrieval is depicted in Figure 28.

Figure 28: Data Retrieval UML diagram

As displayed in the UML diagram in Figure 28, the Data Retrieval is composed by the *DataRetrievalService* that constitutes the main service of the module and that is responsible for handling all the interactions and the communication of the Data Retrieval module with the other two modules of the INFINITECH Data Collection, as well as for the orchestration of the internal services of the module.

In detail, the *DataRetrievalService* interacts with the *DataSourceProfileService*, the *PullDataService*, the *PushDataService*, the *FileHandlerService*, the *DBHandlerService*, the *HDFSHandlerService* and the *MinIOHandlerService,* that constitute the internal services of the Data Retrieval module. These internal services are not exposed to the rest of the modules and the interaction with these services is explicitly performed through their internal interfaces that are exposed only to the *DataRetrievalService*.

The *DataRetrievalService* implements the following main functions:

- ***registerNewProfile (dataSourceProfile: Object)*:** The specific function is performing the necessary actions in order to register a new data source profile. The function receives the data source profile and interacts with the *DataSourceProfileService* in order to store the new data source profile.
- ***retrieveFromAPI (dataSourceProfileID: String):*** The specific function is performing the necessary

actions to execute a request to pull information from the data source's API, utilising the details included in the data source profile. The request can be triggered on demand or based on a scheduled execution. The function interacts with the *PullDataService* for the retrieval and storage of the new dataset.

- ***receive* (dataSourceInfo: Object, dataset: Object):** The specific function is performing the necessary actions to receive new datasets via the exposed external API of the Data Retrieval. The functions receives the required information and the new dataset and interacts with the *PushDataService* in order to store it locally for further processing.
- ***fetchFilesFromServer (dataSourceProfileID: String):*** The specific function is performing the required actions to connect and retrieve files from an FTP or HTTP server utilising the relevant information included in the selected data source profile. It interacts with the *FileHandlerService* in order to retrieve and store the new file, while the process can be triggered on demand or on a scheduled manner.
- ***pullDataFromDB (dataSourceProfileID: String):*** The specific function is performing the required actions to connect and execute a query in the relational database whose relevant connection details are included in the specific data source profile. The function retrieves the new information by interacting with the *DBHandlerService* in order to perform the query and to store the results as a new dataset. The process can be triggered on demand or on a scheduled manner.
- ***fetchFilesFromHDFS (dataSourceProfileID: String):*** The specific function is performing the required actions to connect and retrieve files from an HDFS deployment utilising the relevant information included in the selected data source profile. The new files are retrieved by interacting with the *HDFSHandlerService* and the process can be triggered on demand or on a scheduled manner.
- ***fetchFilesFromMinIO (dataSourceProfileID: String):*** The specific function is performing the required actions to connect and retrieve files from a MinIO storage server, initiating a connection and fetching the respective files, utilising the relevant information included in the selected data source profile. The process is either executed on demand or on a scheduled manner by interacting with the *MinIOHandlerService*.

The *DataRetrievalService* is providing the external REST interface, namely the *DataRetrievalController*, that is exposed by the INFINITECH Data Collection and that undertakes the execution of the data acquisition process for the data sources. The interface has the following main functions:

- ***registerDataSourceProfile (dataSourceName: String, dataSourceOwner: String, dataSourceType: Enum, apiURL: String, pullInterval: integer)*:** This function handles the requests for a new data source profile registration as received by a data source provider, and provides the information required to the *registerNewProfile* function.

• ***pullData (dataSourceProfileID: String):*** This function handles the requests for pulling a new dataset from a data source, as defined in the data source profile and utilises the *retrieveFromAPI* function. • ***pushData (datatype: Enum, dataSourceName: String, dataSourceOwner: String, authToken: String, body: String):*** This function handles the requests where data are pushed to the Data Retrieval from the data source provider. It compiles the data source information and passes the information along with the received dataset to the *receive* function.

- ***pullFromServer (dataSourceProfileID: String):*** This function handles the requests where new datasets are fetched in the form of files from an FTP or HTTP server providing the information required to the *fetchFilesFromServer* function.

• ***pullFromDB (dataSourceProfileID: String):*** This function handles the requests where new datasets are retrieved by connecting to a relational database utilising the *pullDataFromDB* function. • ***pullFromHDFS (dataSourceProfileID: String):*** This function handles the requests where new datasets are fetched in the form of files from a HDFS deployment as defined in the data source profile utilising the *fetchFilesFromHDFS* function.

- ***pullFromMinIO (dataSourceProfileID: String):*** This function handles the requests where new datasets are fetched in the form of files from a MinIO storage server, in accordance with the information

included in the data source profile utilising the *fetchFilesFromMinIO* function.

The *DataSourceProfileService* is an internal service that undertakes the complete lifecycle management of the data source profile. Hence, it undertakes the generation, storage and retrieval of the data source profiles of the process, supporting the operations performed by the *DataRetrievalService.* In this sense, the *DataSourceProfileService* implements the following internal interfaces:

- ***createNewProfle (dataSourceProfile: Object):*** This function is responsible for the creation and local storage of a new data source profile as provided by the data source provider.
- ***getProfile (dataSourceProfileID: String):*** This function undertakes the retrieval of a specific data source profile based on the provided profile identifier.
- ***getAllProfiles (dataSourceOwner: String):*** This function undertakes the retrieval of all the data source profiles of a data source provider, based on the provided owner identifier.

The *PullDataService* is an additional internal service that provides all the functionalities related to the retrieval of new datasets from the API of a data source based on the information defined in the data source profile. Hence, the *PullDataService* implements the following internal interfaces:

- ***pullDataAPIcall (dataSourceProfileID: String):*** This function is responsible for initiating a request to the API specified in the data source profile. The retrieved datasets are stored locally for further processing.
- ***schedulePullDataAPI (dataSourceProfileID: String):*** This function undertakes the scheduling of an API pull request based on the time interval value set on the data source profile. In this case, a timer is created and when the timer expires the *pullDataAPIcall* function is triggered.

The *PushDataService* is the internal service that undertakes the handling of requests where a data source provider is pushing new information to the Data Retrieval via its exposed API. Hence, the *PushDataService* implements the following internal interface:

- ***receiveDataAPICcall (dataSourceInfo: Object, dataset: Object):*** This function is responsible for handling the requests where new datasets are pushed to the API by the data source provider. It receives the new datasets and stores them locally for further processing.

The *FileHandlerService* is the internal service responsible for the retrieval of new files from an FTP or HTTP server based on the provided connection details and file path. To this end, the *FileHandlerService* implements the following internal interfaces:

- ***fetchFiles (dataSourceProfileID: String):*** This function is responsible for connecting and retrieving a file from the server. The retrieved file is stored locally for further processing.

- ***scheduleFetchFiles (dataSourceProfileID: String):*** This function undertakes the scheduling of connection and retrieval of files from a server based on the time interval value set on the data source profile. Initially, a timer is set and upon the expiration of the timer the *fetchFiles* is triggered**.**

The *DBHandlerService* is the internal service that undertakes the connection and query execution in the configured relational database utilising the details included in the data source profile. Hence, the *DBHandlerService* implements the following internal interfaces:

- ***queryDB (dataSourceProfileID: String):*** This function is responsible for connecting and executing the predefined query to the selected relational database. The retrieved results are stored locally as a new dataset for further processing.
- ***scheduleQueryDB (dataSourceProfileID: String):*** This function undertakes the scheduling of a connection and a predefined query execution based on the time interval set on the data source profile. In this case, a timer is created and when the timer expires the *queryDB* is triggered**.**

The *HDFSHandlerService* is the internal service responsible for the retrieval of new files from an HDFS

deployment utilising the connection details and file path that is set in the data source profile. To this end, the *HDFSHandlerService* implements the following internal interfaces:

- ***fetchFilesHDFS (dataSourceProfileID: String):*** This function is responsible for connecting and retrieving a new file from the HDFS deployment. The retrieved file is stored locally for further processing.
- ***scheduleFetchFilesHDFS (dataSourceProfileID: String):*** This function undertakes the scheduling of the connection and retrieval of files from the HDFS deployment. In detail, a timer is set based on the time interval value on the data source profile and upon the expiration of the timer the *fetchFilesHDFS* is triggered**.**

Finally, the *MinIOHandlerService* is the internal service that undertakes the handling of requests for the retrieval of new files from a MinIO storage server. The service utilises the connection details (access key and access secret), the bucket name and the file path included in the respective data source profile and implements the following internal interfaces:

- ***fetchFilesMinIO (dataSourceProfileID: String):*** This function undertakes the connection and retrieval of a new file from the MinIO storage server. The retrieved file is stored locally for further processing.
- ***scheduleFetchFilesMinIO (dataSourceProfileID: String):*** This function is responsible for the scheduled execution of the connection and retrieval of a file from the from the MinIO storage server. Based on the information set on the data source profile, a timer is triggered and once the timer expires the *fetchFilesMinIO* is triggered.

2.3.2 Data Mapper

Data Mapper is responsible of the generation of the mapping of the data entities of a specific dataset with the respective entities of the underlying integrated data model based on the data source provider's input. In this sense, the Data Mapper facilitates the annotation of the data entities of a selected dataset with the relevant entities that will enable effective and efficient exploitation of the dataset within the context of the INFINITECH platform. In the same manner as with the Data Retrieval module, the process exploits the concept profiles, namely the data mapping profiles, and can be executed in an on-demand (immediately) or scheduled manner. Figure 29 depicts the class diagram of the Data Mapper.

Figure 29: Data Mapper UML diagram

As illustrated in the UML diagram (Figure 29), the *DataMapperService* is the main service of the module, acting as the interface of the module with the rest of the modules of the INFINITECH Data Collection and providing the level of abstraction on top of the internal services of the module that are not exposed outside the module. In this sense, the *DataMapperService* interacts with the *MappingProfileService* and the *MappingGenerationService* via their internal interfaces which are exposed only to the *DataMapperService.*

The *DataMapperService* implements the following main functions:

- ***registerMappingProfile (dataMappingProfile: Object)*:** The specific function is performing the necessary actions in order to register a new data mapping profile. The function receives as input the new data mapping profile and invokes the *MappingProfileService* in order to process and store the new profile.
- ***generateMapping (dataMappingSelection: Object, filePath: String):*** The specific function is performing the necessary actions in order to generate the mapping of the data entities to the respective entities of the underlying integrated data model. The function receives the input from the data source provider along with the path of the local file, and invokes the *MappingGenerationService* in order to execute the mapping process.
- ***generateMapping (dataMappingProfileID: String, filePath: String):*** The specific function is generating the mapping of the data entities to the respective entities of the underlying integrated data model utilising an existing data mapping profile. The function receives as input the identifier of the data mapping profile along with the path of the local file and invokes the *MappingGenerationService.*

The *DataMapperController* is the single external interface of the *DataMapperService* that is exposed by the INFINITECH Data Collection. The interface enables the execution of the data mapping process for a selected dataset. The interface has the following main functions:

- ***registerMappingProfile (dataSourceName: String, dataSourceOwner: String, mappingRules: Object)*:** This function handles the requests for the registration of a new data mapping profile as

received by a data source provider and invokes the *registerMappingProfile* function. The process can be invoked either through the respective API call or via the user interface of the Data Mapper. • ***executeMapping (newMapping: [], filePath: String):*** This function handles the requests for the execution of the data mapping process based on the data source provider selection and invokes the *generateMapping* function. The process can be invoked via the user interface of the Data Mapper. • ***executeMappingProfile (dataMappingProfileID: String, filePath: String):*** This function handles the requests for the execution of the data mapping process based on the data mapping profile identifier and invokes the *generateMapping* function. The process can be invoked either through the respective API call or via the user interface of the Data Mapper.

The *MappingProfileService* is the internal service that is responsible for all operations related to the creation, storage and retrieval of the data mapping profiles. Hence, its scope is to support the data mapping process performed by the *DataMapperService.* To this end, the *MappingProfileService* implements the following internal interfaces:

- ***createMappingProfle (dataMappingProfile: Object):*** This function undertakes the creation and local storage of a new data mapping profile as provided by the data source provider.
- ***getProfile (dataMappingProfileID: String):*** This function facilitates the retrieval of a specific data mapping profile based on the provided profile identifier.
- ***getAllProfiles (dataSourceOwner: String):*** This function enables the retrieval of all the data mapping profiles of a data source provider based on the provided owner identifier.

The *MappingGenerationService* is the internal service that undertakes the responsibility of generating the mapping of the data entities to the respective entities of the underlying integrated data model. The mapping process is either performed utilising the data source provider's input or by utilising an existing data mapping profile. In this sense, the *MappingGenerationService* implements the following internal interfaces:

- ***mapWithInput (newMapping: [], filePath: String):*** This function undertakes the execution of the mapping process by utilising the data source provider's input and the selected dataset. The produced mapping is stored locally for further exploitation.
- ***mapWithProfile (dataMappingProfileID: String, filePath: String):*** This function undertakes the execution of the mapping process by utilising the existing data mapping profile and the selected dataset. The produced mapping is stored locally for further exploitation.

2.3.3 Data Cleaner

Data Cleaner undertakes the responsibility of the data cleaning process that safeguards the accuracy and completeness of the selected dataset to the extent possible, maximizing its value and usability before being stored and further processed. The Data Cleaner implements the data cleaning process, that is documented in Section 2.1.3, and is composed of four (4) steps: a) the data validation, b) the data correction, c) the missing values imputation and d) the complete logging of the performed data cleaning operations. The class diagram of the Data Retrieval is depicted in Figure 30.

Figure 30: Data Cleaner UML diagram

As displayed in Figure 30, the DataCleaner module is composed by the main service, namely the *DataCleanerService*, and a set of the internal services, each one undertaking a specific step of the data cleaning process, namely the *ConfigService, the ValidatorService*, the *CleanserService*, the *CompleterService* and the *LoggerService.*

The *DataCleanerService* orchestrates the execution of the data cleaning process by exploiting the internal services via the internally exposed interfaces that each service offers explicitly to the main service. The *DataCleanerService* executes the data cleaning process based on the set of data cleansing rules that are defined by the data source provider. In these rules, the configuration that will customise the operations of the internal services of *the ValidatorService*, the *CleanserService* and the *CompleterService* is defined. The *ConfigService* and the *LoggerService* are supplementary services that facilitate the execution of the data cleaning process. Hence, the *DataCleanerService* implements the following main functions:

- ***add_config (dataCleaningProfile: Object)*:** The specific function undertakes the registration of a new data cleaning profile as defined by the data source provider. The function receives as input the new data cleaning profile and invokes the *ConfigService* in order to process and store the new profile.
- ***clean_data (dataCleaningRules: Object, filePath: String):*** The specific function is performing the necessary actions in order to execute the data cleaning process based on the data cleansing rules set by the data source provider. The function receives the data cleansing rules along with the path of the local file and invokes the *ValidatorService*, the *CleanserService* and the *CompleterService* services respectively based on these rules.
- ***clean_data (dataCleaningProfileID: String, filePath: String):*** The specific function is orchestrating the execution of the data cleaning process utilising an existing data cleaning profile. The function receives as input the identifier of the data cleaning profile along with the path of the local file and

based on the rules defined in the profile it invokes *ValidatorService*, the *CleanserService* and the *CompleterService* services.

The *DataCleanerService* provides the *CleanController* interface that constitutes the single external exposed interface that the INFINITECH Data Collection offers from the specific service. Through this interface, the data cleaning process is initiated and executed. The interface has the following main functions:

- ***registerCleaningProfile (dataSourceName: String, dataSourceOwner: String, cleaningRules: Object)*:** This function handles the incoming requests for the registration of a new cleaning mapping profile as initiated by a data source provider and invokes the *add_config* function. It should be noted that the process is either invoked directly via the offered API or via the user interface that is offered by the Data Cleaner.
- ***clean (cleaningRules: Object, filePath: String):*** This function handles the incoming requests for the execution of the data cleaning process originating by the user interface of the Data Cleaner. The

function receives the input of the data source provider selection and invokes the *clean_data* function.
- ***clean (dataCleaningProfileID: String, filePath: String):*** This function handles the incoming requests for the execution of the data cleaning process and invokes the *clean_data* function for the specified data cleaning profile. The process can be invoked either through the respective API call or via the user interface of the Data Cleaner.

The *ConfigService* is the internal service that undertakes the profile management operations that involve the registration, storage and retrieval of the data cleaning profiles. To this end, the *ConfigService,* in accordance with the rest of profile management services of the INFINITECH Data Collection, implements the following internal interfaces:

- ***create_config (dataCleaningProfile: Object):*** This function handles the registration and storage of a new data cleaning profile as defined by the data source provider.
- ***get_config (dataCleaningProfileID: String):*** This function handles the retrieval of a specific data cleaning profile based on the provided profile identifier.
- ***get_all_configs (dataSourceOwner: String):*** This function facilitates the retrieval of all the data cleaning profiles of a data source provider based on the provided owner identifier.

The *ValidatorService* is the internal service that is responsible for the data validation checks that are performed on the selected dataset. In this sense, the service offers an extended list of data validation checks which are performed against the data entities of the selected dataset based on the preferences of the data source provider. In particular, within the data cleaning rules, the data validation rules are defined and are translated to a set of constraints that the specific data entity should conform with. The list of data validation rules includes the following:

- Conformance to a data type (i.e. Boolean, Integer, String, etc.)
- Conformance to a list of acceptable values (i.e. “Yes” or “No”)
- Conformance to a value range (i.e. the minimum and maximum acceptable values) • Conformance to a value representation format (i.e. all dates are following the YYYY-MM-DD format) • Conformance to a value uniformity (i.e. all time-stamps are in UTC format)
- Conformance to uniqueness (i.e. duplicate values are not acceptable)
- Conformance to non-empty value (i.e. all mandatory fields should have values) • Conformance to cross-field validity (i.e. the sum of fields with percentage values must be equal to 100)
- Conformance to cross-field dependency (i.e. in case the field is set to a value then the other field should be set to a value)

In this sense, the *ValidatorService* implements the following internal interface:

- ***validate_data (validationRules: Object, data: DataFrame):*** This function undertakes the execution of

the data validation process. The function receives as input the data validation rules and the dataset as a Data Frame and performs the data validation checks. The output of the function is the conformance errors identified which are used for the data completion and data imputation step. The function interacts with the *LoggerService* in order to generate the respective records.

The *CleanserService* is the internal service that is responsible for the data correction operations that are performed on the selected dataset. The service provides a list of corrective operations which are performed against the conformance errors identified by the *ValidatorService*, based preferences of the data source provider, as defined in the data correction rules. The *CleanserService* offers, among others, the following data correction operations:

• Inconsistent value rejection and removal of a value from a record of a dataset. •
Inconsistent value rejection and removal of a complete record of a dataset.

- Inconsistent value replacement with a statistical value as the minimum or maximum value observed, the mean or median value or the most frequent value.
- Inconsistent value replacement with a specific value

In this sense, the *CleanserService* implements the following internal interface:

- ***cleanse_data (cleaning_rules: Object, validation_errors: [], data: DataFrame):*** This function performs the execution of the data correction process. The function receives as input the data correction rules, the conformance errors identified by the *ValidatorService* and data as a Data Frame. It performs the corrective actions and provides the updated, "corrected", dataset as the output. The function interacts with the *LoggerService* in order to generate the respective records.

The *CompleterService* is the internal service that undertakes the missing value handling operations that are performed against the selected dataset. The service provides a set of data imputation operations that are performed against the errors identified by the *ValidatorService,* which are related to non-empty value conformance as defined in the data completion rules by the data source provider. To this end, the *CompleterService* offers the filling of the missing values based on the following data imputation operations:

- Using statistical methods such as the minimum or maximum value observed, the mean or median value or the most frequent value
- Using the Linear Regression algorithm
- Using the k-Nearest Neighbours algorithm
- Using the Moving Average method
- Using the Last Observation Carried Forward (LOCF) and Next Observation Carried Backward (NOCB) methods
- Direct data imputation with a predefined value

Hence, the *CompleterService* implements the following internal interface:

- ***check_data_completeness (dataCompletionRules: [], conformanceErrors: [], data: DataFrame):*** This function performs the execution of the data imputation process. The function receives as input the data correction rules, the conformance errors identified by the *ValidatorService* and the data as a Data Frame. The output of the function is the updated, "completed", dataset. The function interacts with the *LoggerService* in order to generate the respective records.

The *LoggerService* is the complementary internal service that is utilised during the whole process execution in order to create and maintain the complete history records that contain the errors that were identified during the whole data cleaning operation process, as well as the corrective and missing value handling operations that were performed against these errors. Thus, the *LoggerService* offers the following internal interfaces:

- ***create_logs(dataSourceName: String, dataSourceOwner: String, validation_errors: Data Frame):*** This function is responsible for creating a new log file based on errors identified during the validation step, as long as the actions taken during cleaning and missing value handling steps.

• ***get_all_log_files:*** This function is responsible for returning the names of all the available log files. • ***log_contents(log_file: String):*** This function returns the contents of a specific log file that its name is provided as parameter.

3 INFINITECH Synthetic Datasets

In the age of Big Data, a large variety of data sources produce data at an exponential level. Public and private organisations are struggling to effectively collect all these enormous amounts of data that contain valuable detailed information that researchers and decision makers need in order to perform analyses, formulate predictions, evaluate their strategies and ultimately solve both simple or complex problems. Despite the increased availability of datasets and the acknowledged request for unrestricted availability of datasets imposed by the researchers and decision makers, the preservation of the privacy aspects of the individuals whose sensitive and private information is often included in the datasets, is equally important [1].

Thus, several data privacy-preserving techniques, such as pseudonymization, anonymisation, pseudo anonymisation, masking techniques, are employed on datasets in order to cope with the privacy concerns. However, besides the pure need for availability of datasets there is also the need for reliable, relevant and adequate data that meet certain characteristics, which are critical for the aspired analysis. The data privacy preserving techniques are posing several restrictions on these required characteristics, hence the quality of the utilised datasets is in most cases highly compromised. Nevertheless, the generation of Synthetic Datasets, in which private and sensitive data in the original dataset are replaced with synthetic data, is a viable alternative to the data privacy preserving techniques.

3.1 The characteristics of Synthetic Datasets

Synthetic datasets are datasets that contain artificially generated data instead of real data which are usually generated with the help of algorithms and a variety of data modelling techniques. In recent years, synthetic data generation has gained focus and significant effort has been invested in research for this topic, not only for its effective usage in a privacy preserving manner, but also for its effectiveness to support validation of new algorithms and applications which require data that are either not available or not accessible due to privacy concerns [2].

One of the major advantages of synthetic data is that they can be made publicly available with minimum risk of data disclosure and maximum utility [1], since the data included is randomly generated with constraints to hide sensitive private information and retain certain statistical information or relationships between attributes in the original data [2]. Hence, synthetic datasets can eliminate the barriers in numerous cases when privacy requirements limit data availability, the way data can be used and shared between multiple third parties in a collaborative nature thus effectively support research and innovation, as well as decision making.

Another major advantage of synthetic datasets is that they are generated in a such way that they address

specific needs or conditions and with specific characteristics that are not available in existing (real) data. Synthetic datasets are often generated by exploiting several algorithms in a manner that enables more flexibility on the data manipulation aspect towards the effective testing of a broader range of conditions and use cases. In detail, synthetic datasets generation enables [3]:

a) the control over the data distributions used for the testing and validation of an algorithm's performance,
b) the fair performance comparison between different algorithms and
c) the creation of data records with the finest level of granularity in each attribute.

Hence, synthetic data can be valuable to a wide range of activities, including effective testing of new software systems and applications, training and validation of machine learning models, as well as all the cases where data are needed but they are not available, are too expensive to be generated as real data, or do not exist at all.

In general, synthetic datasets can be classified into the following three main categories:

a) Fully Synthetic Data: The fully synthetic data does not contain any real data. For this reason, re identification of any individual is almost impossible, while at the same time it is ensured that all variables of the datasets are fully available. In this case, during the fully synthetic data generation, the procedure that is followed utilizes multiple imputation in order to replace the values of certain attributes for all the data points in the dataset [1]. The process includes multiple steps in which the density function of attributes in real data is identified and the parameters of these functions are estimated, before generating a privacy protected series of randomly selected values from these estimated density functions [2].

b) Partially Synthetic Data: The partially synthetic data contain real data and only the sensitive data is replaced with synthetic data. Hence, only data that raise risk of personal or sensitive information disclosure is replaced. Their generation is largely dependent on the utilised imputation model and the risk of disclosure is higher than in fully synthetic data, as real data is still included in the dataset. To generate the synthetic values for the selected attributes, multiple imputation and model-based techniques are also used [2].

c) Hybrid Synthetic Data: The hybrid synthetic data are generated utilising a limited volume of real data or synthetic data that were generated by domain experts. In the course of hybrid synthetic data generation, the distribution of the real data is analysed and the nearest record in the synthetic data is chosen, while ensuring both the relationship and the integrity between other attributes of the dataset. Hybrid synthetic data holds the advantages of both fully and partially synthetic data, providing adequate privacy preservation with high utility compared to fully synthetic and partially synthetic data, but at the cost of more memory and processing time [2].

In order to benefit from the usage of the synthetic datasets, it is important to possess a thorough knowledge of the domain for which data will be generated, as well as to follow a set of principles during the generation process, and finally to choose the correct methodology for the generation process, depending on the needs that these datasets will cover. During the synthetic data generation, a set of principles and restrictions are implied. Synthetic datasets are domain-dependent and for this reason it is crucial to utilise a real dataset during the synthetic data generation process, in order to ensure the properties of the dataset are satisfied. For this reason, a good understanding of the domain for which data are generated is also required [2]. Furthermore, both the domain and the data type that are generated are significantly affecting the complexity of the required synthetic data generation process.

Synthetic datasets have also restrictions as it is the case of real data. It should be acknowledged that during the synthetic data generation process, specific attributes of the data are only replicated, hence in principal general trends are simulated. While fully synthetic data has strong resistance to disclosure risk, these data lack truthfulness [2]. In addition to this, most of the times fully synthetic data are extremely useful for

research purposes, however their usage in commercial products is usually limited. On the other hand, the support of partially synthetic data generation is very limited from the available techniques as most of the techniques that are available are mostly suitable for fully synthetic data generation. Finally, the complexity of the process is strongly dependent on the type of data that should be replicated, as for example static data is usually less challenging than streaming data for which the distribution of data is not usually known beforehand.

The complexity of the synthetic data generation process is highly dependent on the domain for which data should be synthesized, as well as the usage purpose of the generated synthetic datasets. As such, usually it goes beyond the simplistic solution of drawing numbers from a distribution that is created either based on the observation of the existing distribution from real data or from a comprehensive understanding of how the distribution would be like in the dataset in the case where real data do not exist. Usually, data modelling techniques are applied with the aim of replicating the required data. The existence or not of real data practically drives the decisions related to the data modelling technique. In the first case, the real data are exploited in order to generate fully synthetic or partially synthetic data by extracting the characteristics of the data and modelling them using usually probability density functions in order to perform multiple imputation. In the latter case where real do not exist or are not available due to privacy restrictions, the domain expert should define rules, constraints and relationships to effectively model the data, thus it is

imperative to have extensive knowledge of the domain for which data will be generated [2]. In both cases, machine learning and deep learning models are utilised in the process in order to perform the multiple imputation, treating the values of the selected attributes as missing values which are generated using models such as Decision Trees, Random Forest, Support Vector Machine and Generative Adversarial Networks, and more [1].

The synthetic data generation process is facilitated by a large variety of libraries, frameworks and tools that the domain experts can utilise based on their needs and preferences. The most dominant libraries and frameworks are based on the Python and R programming languages, while there are also libraries and frameworks offered in other programming languages such as Java.

Scikit-learn[3]is the python well-established ML library that offers an extensive list of ML algorithms which are exploited in a large variety of synthetic data generation processes, such as the regression problem generation, classification problem generation, clustering problem generation, anisotropic cluster generation, concentric ring cluster data generation and moon-shaped cluster data generation. On the other hand, when it comes for categorical data generation, pydbgen[4]is a lightweight python library that is capable of generating a large database with multiple tables, filled with meaningful yet random data which can be ingested into a database, loaded as a dataframe in Pandas[5] or saved as an Excel file for further processing. Synthpop[6]is also a popular R library utilised for producing synthetic versions of microdata containing confidential information enabling their sharing and exploratory analysis. DataGenerator[7]is a Java based library that is capable of producing large volumes of data, framing data production as a modelling problem, with a user providing a model of dependencies among variables and the library traversing the model to produce relevant datasets. Another example is several online open-source or commercial platforms that enable the generation of realistic synthetic (usually structured) datasets which are mostly utilised for testing purposes such as Mockaroo[8], GenerateData[9] and OnlineDataGenerator[10]. In addition to these libraries and online tools, a plethora of candidate libraries and frameworks are available for usage depending on the scope of the synthetic data generation process and the preferences of the domain experts.

3.2 The role of Synthetic Datasets in INFINITECH

The financial and insurance sectors generate data in massive and increasing volume with high velocity and in a large variety of data types and formats. Despite the availability of these data, multiple restrictions and barriers are imposed to their usage and sharing due to the nature of the personal and sensitive information that is included on these datasets. As both sectors are highly regulated with strict legislations and processes,

these collected data are mostly stored in silos within the organisations and severe limitations are applied when it comes to processing and sharing them even within the same organisation's different departments and especially outside the organisation's boundaries. As a consequence, the various research and business development activities are facing the lack of data that will enable them to perform the required and effective analysis, prediction generation and strategy evaluation.

Towards this end, the synthetic data generation approach is providing an appealing solution to overcome these restrictions and the problem of data unavailability in the finance and insurance sectors, as it ensures the required privacy preservation and at the same provides the means to the researchers and decision

[3]Scikit-learn, https://scikit-learn.org/stable/

[4] pydbgen, https://pydbgen.readthedocs.io/en/latest/

[5] Pandas Python, https://pandas.pydata.org/

[6]Synthpop, https://www.synthpop.org.uk/

[7] DataGenerator, https://finraos.github.io/DataGenerator/

[8] Mockaroo, https://www.mockaroo.com/

[9] GenerateData, http://generatedata.com/

[10] Online Data Generator, https://www.onlinedatagenerator.com/

makers to perform the research and business development activities. In this sense, within the context of INFINITECH the synthetic data generation will be leveraged in order to overcome the aforementioned difficulties where needed and to facilitate the design and implementation of innovative financial and insurance services. In particular, the use cases and motivation for synthetic data generation, which are applicable in the finance and insurance sectors and are taken under consideration in INFINITECH, can be described in the following axes [4]:

- **Internal data use restrictions**: The imposed privacy requirements that limit data availability and prevent data sharing internally within a finance or insurance organization can be mitigated with the use of synthetic datasets.
- **Lack of historical data**: In some cases, the lack or limited availability of historical data which are needed in order to effectively perform the required analysis is posing limitations. Hence, with the use of synthetic data, these barriers can be removed.
- **Class imbalance**: The datasets which are utilised in several use cases, such as fraud detection, are imbalanced hence traditional machine learning techniques and anomaly detection techniques will often fail. Thus, synthetic data with more realistic attribute values generated with the appropriate data imputation techniques are characterized as more suitable for the aspired analysis.
- **Training advanced ML models**: Training of advanced machine learning models require vast amount of data, which should also probably be transferred into the appropriate infrastructure that is capable of performing such computationally intensive operations and might be outside of the boundaries of the organisation. Hence, the usage of synthetic datasets resolves both the need for large volume of data and the imposed privacy requirements.
- **Data Sharing**: The main restriction of the data originating from the finance and insurance sectors is that data sharing is severely restricted. Thus, data sharing between the organisations or within the research community which will enable the design, experimentation and implementation of innovative financial and insurance services, is very limited. However, the nature of synthetic data offers the potential to meet the imposed data sharing restrictions and enable the research and innovation.

Within the context of INFINITECH, several pilots will exploit the benefits offered by the usage of synthetic datasets towards the validation of the INFINITECH offerings from both a technical/technological and a business/economic perspective. In this context, synthetic datasets will be prepared by the pilot's domain

experts in order to be exploited during the execution phase of each pilot. The format of the synthetic datasets will vary from semi-structured (*CSV, JSON, GeoJSON*) to unstructured formats (*txt, NetCDF*) depending on the needs of each pilot. In the same manner, the volume of each synthetic data that will be exploited will vary from 10 MB per executed use case to 20 GB, depending on the executed scenario.

The main categories of the synthetic data that will be utilized within the context of the INFINITECH Pilots are the following:

a) Customer Profiles
b) Customer Accounts
c) Customer Data
d) Financial Transaction Data
e) Connected Car Data
f) Traffic Data
g) Activity Tracking Data
h) Crop Biophysical Parameters and Loss Data
i) Seasonal Crop Yield Prediction Data
j) SME Company Profile Data

These synthetic data will be available in the respective testbed of each pilot that will exploit them. The data collection process that will be utilised in order to ingest the respective synthetic data along with any real data

that will exploited into the testbed of each pilot is presented in Section 4. The list of synthetic datasets of each pilot along with in-depth details is described in Appendix B**.**

4 INFINITECH Pilot Use Cases

The scope of the current section is to document the details of the datasets, both real and synthetic, that will be collected by the INFINITECH pilots in order to be harmonized, anonymised and ingested into the underlying storage. As mentioned already in previous sections, the INFINITECH pilots aspire to implement and execute a list of diverse scenarios from the finance and insurance sectors with different approaches based on their scope. To this end, different approaches will be followed in order to address the requirements and the peculiarities of each scenario. The details of the design specifications and implementation of each pilot for the data collection process along with the aspired scenarios that will be executed, are documented in deliverable D7.1.

In the following paragraphs, the focus is on presenting the preliminary list of the datasets that will be leveraged by the pilots, accompanied by the details for the information included, their data format and the anonymisation needs, where applicable. These datasets will be collected by the data collection process of each pilot as documented in D7.1.

It should be noted that the description of the data collection processes of Pilot #3, Pilot #7 and Pilot #15 is not included in the current version of the deliverable for various reasons. In particular, Pilot #3 and Pilot #7 at the moment of writing the deliverable, are currently in redesign state due to partner swap in the consortium of the project as part of the latest amendment process. Additionally, Pilot #15 was recently introduced in the project. The details of these pilots will be included in the upcoming version of the deliverable.

4.1 Pilot #1 - Invoices Processing Platform for a more Sustainable Banking Industry

The scope of pilot #1 is to design, integrate and deliver a data-intensive system that is capable of extracting valuable information from notary invoices with a threefold purpose: a) to establish the sustainability index of each notary based on the number of physical copies that are issued, b) to provide financial institutions with the information (properly indexed) about the documents that are finally generated by notarial services required by the bank and c) to promote notarial services from those with the higher sustainability score.

One core part of the specific pilot is related to the dataset that will be collected in order to be fed into the pre-processing, processing and post-processing processes of the specific pilot that include the digitalisation of the collected data, as well as their effective storage and indexing. The following table presents the details of the dataset that will be exploited in these processes.

Table 25: Pilot #1 List of datasets

Dataset Name	Dataset description	Data format	Anonymization
On-premise Bankia system	Real data from notary service invoices will exported in PDF, Image (PNG) or Text formats. The dataset will be composed with data from 32,300 real invoices issued by 3,000 different notaries and the volume of the dataset it is expected to be around 2 TB	PDF/ PNG/ text	No anonymisation

4.2 Pilot #2 - Real-time risk assessment in Investment Banking

The scope of Pilot #2 is to design and implement a real-time risk assessment and monitoring procedure that aims to facilitate the generation of risk information for asset management with two standard metrics, namely the VaR (Value-at-Risk) and the ES (Expected Shortfall). To this end, the pilot will perform the measurement of market risks of assets portfolios, while also evaluating what-if scenarios for pre-trade analysis.

With regards to the datasets that will be exploited, the pilot will mainly leverage Foreign Exchange (FOREX) market data in order to calculate the Value-at-Risk (VaR) for various portfolio compositions. In detail, two main types of input datasets will be used, the first one includestrades associated with a portfolio composition to be analysed ("TradeData"), while the second one includes price data ("TickData") regarding the FOREX market. The pilot will also leverage alternative data, such as derived analysis data and news data, that will be used in the performed analysis. All data types have quite simple structures and are presented in the table below:

Table 26: Pilot #2 List of datasets

Dataset Name	Dataset description	Data format	Anonymization
TradeData	Trades associated with a portfolio composition. The TradeData will comprise the following three columns: ▪ SymbolID, i.e. the name of the instrument in FOREX trading e.g., GBPUSD for the exchange of GBP to USD ($) ▪ Timestamp (in UNIX format) denotes when the trading took place. ▪ Quantity, i.e. the amount traded which will be a negative number in case of selling and a positive number in case of buying.	CSV / Real time Data stream	No anonymisation
TickData	Price data regarding the FOREX market. The price data will comprise the following four columns: ▪ Line Number, ▪ SymbolID i.e. the name of the instrument in FOREX trading e.g., GBPUSD for the exchange of GBP to USD ($) ▪ Timestamp in Unix format ▪ Closing Price i.e. the market price for the instrument	CSV / Real time Data stream	No anonymisation
Derived analysis data	Risk measures and correlation matrices retrieved from online trading platforms.	CSV	No anonymisation
News articles and Twitter data	Open source sample data for Market Sentiment Analysis	Text	No anonymisation

To facilitate the process, both TradeData and TickData datasets will be provided in two flavours, the Real time market feeds associated with the real-time operation of the market ,and the historical datasets that will be used for various calculations. The market data are real data, retrieved from a financial data service provider while the trade data will be the trading signals generated by JRC's algorithmic trading strategies, and no real trades will be executed. The data will be accessible to their end-users (i.e. traders) and no anonymization processes will take place.

For testing and experimentation purposes existing datasets in CSV formats will be used. Hence, the format of the data will be CSV and real-time data stream containing text and numeric data related to public market

data and synthetic portfolio composition data.

To support experimentation, validation and deployment in production, the data collection processes will leverage data from the following data sources:

- Forex APIs, which will be mainly used for testing and experimentation purposes. They will provide access to both real-time and historical values of FX data.
- Data from the JRC Trading Platforms, which will be used to provide real-time trading values with high ingestion rates.
- Data from the JRC Trading Datawarehouse, which will provide access to historical information (e.g., close values) for the forex assets that will be entailed in the pilot.

4.3 Pilot #4 Personalized Portfolio Management ("Why Private Banking cannot be for everyone?")

The scope of Pilot #4 to develop and integrate within the SaaS based Privé Managers Wealth Management Platform an Optimization algorithm (further on called Privé Optimizer "AIGO"), as well as to improve and expand its capabilities as an artificial intelligence engine to aid investment propositions for retail clients. Hence, pilot #4 aims to exploit the capabilities of AI-Based Portfolio construction for Wealth Management processes in order facilitate the advisors and/or end-customers utilising the Privé Managers Wealth Management Platform to effectively consume the offered risk-profiling and investment proposal capabilities, starting from their personal risk-awareness.

The innovative AIGO genetic algorithm is capable of proposing investments which are in turn evaluated based on a set of fitness factors which are composed by easy-to-use, personalized set of criteria. Based on the fitness factors, "health" score will be generated for each portfolio that will drive the definition of the "fittest" investments. To achieve this, a series of datasets will be leveraged which are presented in the following table:

Table 27: Pilot #4 List of datasets

Dataset Name	Dataset description	Data format	Anonymization
Customer Transactions Data	Customer Transactions Data fetched directly from the Bank or an Asset Manager. It consists of customer securities and cash transactions through their deposit accounts.	CSV	Confidential data
Financial Market Price Data	Financial Market Price Data fetched from several Market Data Providers. It consists of price data for Stocks, Bonds, Mutual Funds and or other assets like certificates/warrants.	Text	Open, partially license agreements with data providers needed
Financial Market Asset Master Data	Financial Market Asset Master Data fetched from several Market Data Providers. It consists of asset related characteristics (e.g. expiration date, minimum investment amount, asset class breakdowns).	Text	Open, partially license agreements with data providers needed

Customer Risk Profile Data	Customer Risk Profile Data fetched directly from the Bank or an Asset Manager. It consists of customer Risk Profile Data through their account data and profiling, based on B2B customers parameters.	CSV	Confidential data
Mutual Fund, ETF and Structured Products Breakdown	Mutual Fund, ETF and Structured Products Breakdown fetched from several Market Data Providers. It consists of asset breakdowns based on bank data or market data providers breakdown.	CSV	Open/Confidential data, partially license agreements with data providers needed

Customer Economic Outlook	Customer Economic Outlook fetched directly from the Bank or an Asset Manager based on questionnaires and Customer Profiles.	CSV	Confidential data

With regards to the anonymisation of the aforementioned confidential datasets, no anonymisation is foreseen as the data will be privately kept within the premises of Privé. Additionally, the input that will be taken from these datasets in order to be fed to the optimisation process will not include any sensitive or private data.

4.4 Pilot #5b - Business Financial Management (BFM) tools delivering a Smart Business Advise

The scope of Pilot #5b is to provide the means to the SMEs which are clients of Bank of Cyprus (BoC) to effectively and efficiently manage their financial health, in multiple areas such as cash flow management, continuous spending/cost analysis, budgeting, revenue review and VAT provisioning by employing a set of AI-powered Business Financial Management tools and harnessing available data to generate personalized business insights and recommendations.

Hence, the multiple diverse datasets will be collected and processed in the course of development of the specific pilot. In detail, most of the datasets that will be utilized, will be extracted from BoC databases and are related to financial transactions, account profiles and relevant data of BoC's SME customers. Furthermore, BoC's SME customers transaction data with other banks which are retrievable from Open Banking under PSD2 will also be collected, as well as open-source macroeconomic data from sources like Eurostat or data.gov.cy.

The following table depicts the list of datasets that will be used, providing for each dataset a short description, the respective data format and the anonymization requirements.

Table 28: Pilot #5b list of datasets

Dataset Name	Dataset description	Data format	Anonymization
Transaction Data from the Bank	SME Customers Transactions Dataset from BoC	CSV	Already anonymized

Transaction Data from Open Banking (PSD2)	SME Customers Transactions Dataset from financial institutions other than BoC. BATCH input (e.g. every 6 hours or every night)	CSV	Already anonymized
Accounts Data from Bank	Account data regarding SME customers of BoC. E.g. Balances, Available amount, account type	CSV	Already anonymized
Customer Data from Bank	Customer Demographics	CSV	Already anonymized
Other Data (Market)	Macroeconomic SME related data from public/private resources (e.g. https://www.data.gov.cy/ & https://ec.europa.eu/eurostat/data/database)	CSV / JSON	Open source data
Other Data from SME	Other Data that is provided by the SME ERP/Accounting system (e.g. number of customers, suppliers, stock). Non- transactions related data	CSV/ JSON	Investigating GRAD anonymization solution

Transaction Data from SME	Data that is provided by the SME ERP/Accounting system and relates directly or indirectly to account debit/ credit transactions. For instance, invoice data; non PSD2 data (i.e. payment account related) e.g. saving accounts in financial institutions other than BOC	CSV/ JSON	Investigating GRAD anonymization solution

As described in the table above, most of the datasets will already be anonymised before being imported to the INFINITECH platform. Data originating from BoC activities and databases will be pseudonymized before being extracted to the platform using the tokenization approach, where sensitive data are being replaced with non-sensitive equivalents, which can only be reversed in the tokenization vault that resides on BoC premises.

4.5 Pilot #6 - Personalized Closed-Loop Investment Portfolio Management for Retail Customers

The scope of Pilot #6 is to design and deliver a personalized investment recommendations system tailored to the needs of the retail customers of NBG. In this process, a variety of diverse datasets from different data sources in large volumes will be leveraged towards the aim of providing investment recommendations to retail customer more targeted, automated, effective, as well as context-aware (i.e. tailored to state of the market).

The datasets will be generated from NBG Operational Databases in CSV/TXT format and will be in a way anonymized, as all critical information will be converted to specific IDs that the reference to the actual information will be privately kept within the bank premises. The list of the datasets that will be utilised is as follows:

Table 29: Pilot #6 List of datasets

Dataset Name	Dataset description	Data format	Anonymization
Deposit Account Transactions	Customers' transactions performed through their deposit accounts for the last two (2) years, for each deposit product account that the customer possesses. The Deposits Account transactions data include category amount, date, time and channel that were performed.	CSV / Text	Already anonymized
Cards Transactions	Customers' transactions performed through their cards for the last two (2) years, for each card product that the customer is a card holder. The Cards Transactions data include date, time, card type, merchant category, amount in euros and foreign currency, instalment or cash purchase, transaction type and channel (POS, e-commerce, ATM).	CSV / Text	Already anonymized
Instruments Historical Prices	Historical prices of investment instruments that NBG offers to the customers for the last two (2) years, including for each instrument the details of date, closing price, closing price currency, total transactions volume, market code, effective and ending date.	CSV / Text	Already anonymized

Investment Related Transactions	Customers' transactions related to investment products for the last two (2) years, containing date, time, account used, instrument, settled amount, settled currency, maturity date, quantity, instrument price, investment amount gross and net, financial market, investment transaction type.	CSV / Text	Already anonymized
Instruments Characteristics	Detailed characteristics for all instruments that will be considered in the pilot, including instrument type, instrument asset class, currency, ISIN, issue date, maturity date, instrument pieces (e.g. shares).	CSV / Text	Already anonymized
CRM Data	Customer related data like demographics, product ownership and responses to MIFID questionnaires. The data that will be available for the pilot include customer type, birth date, gender, marital status, child number, profession category, origin country, type of employment, customer risk category (bank's calculation engine).	CSV / Text	Already anonymized

4.6 Pilot #8 - Platform for Anti Money Laundering Supervision

(PAMLS)

The scope of Pilot #8 is to design and implement a platform that performs anti-money laundering Supervision (PAMLS) with the aim of enhancing the efficiency and effectiveness of the existing supervisory activities in the area of anti- money laundering and combating terrorist financing (AML/ CTF). Pilot #8 will be leveraging the characteristics of Big Data which are owned by the Bank of Slovenia (BOS) and other competent authorities (FIU). To this end, the PAMLS platform will offer, among others, a risk assessment tool, a screening tool, a search engine and a distributed channel.

Within the context of the specific pilot, multiple real datasets originating from various heterogeneous data sources will be exploited. The relevant information of the aforementioned datasets is summarized in the following table:

Table 30: Pilot #8 list of datasets

Dataset Name	Dataset description	Data format	Anonymization
TARGET2 transactions	Transactions executed by the Slovenian payment institutions within TARGET2 (Trans-European Automated Real-time Gross Settlement Express Transfer System): ▪ high value (above 50.000 EUR), urgent transactions in EUR; ▪ transactions processed through BOS payment systems (responsible BOS Payment Settlement and Systems department - PPS). TARGET2 transactions include following data relevant for PAMLS: *transaction date, transaction value, payer data (transaction account no., name and address), payer bank data and BIC code, payee data (transaction account no., name and address), payee bank data and BIC code, intermediary bank data.*	XLSX (txt or numeric)	Anonymised

SEPA transactions	Transactions executed by the Slovenian payment institutions within SEPA (Single Euro Payments Area): ▪ domestic and international transactions within SEPA area in EUR under 50.000EUR value; ▪ transactions processed through payment systems by third party provider. SEPA transactions include following data relevant for PAMLS: *transaction date, transaction value, payer data (transaction account no., name and address), payer bank data and BIC code, payee data (transaction account no., name and address), payee bank data and BIC code, intermediary bank data.*	XLSX (txt or numeric	Anonymised

Slovene Financial Intelligence Unit (FIU) transactions (public data)	Transactions above 15.000 EUR, related to high risk countries and reported to the FIU.	XLSX (txt or numeric	Open Data
FI identification data	Identification information about Financial Institution (FI) that they send by report to the BOS (reports are confidential). The data are statistical data on the FI inherent risk and control environment (number of clients, number of Suspicious transactions reports (STR) etc.)	XLSX (txt or numeric	Not anonymised
ePRS data	Slovenian Business Register (public data on legal entities).	XLSX (txt or numeric	Open Data
eRTR data	Slovenian Transactions Accounts Register (public data on legal entities).	XLSX (txt or numeric	Open Data

It should be noted that due to the high confidentiality requirements, the transactions analysed within PAMLS datasets will not be transferred outside BOS premises and will not be shared on the INFINITECH platform.

Since the data included in both the TARGET2 and the SEPA transactions contain both personal and confidential data, it will be ensured that the data privacy and data confidentiality aspects are properly addressed. In this context, within the TARGET2 and SEPA transactions datasets, the payer data (transaction account number, name and address) and payee data (transaction account number, name and address) will be anonymised prior to being delivered to PAMLS in order to eliminate the disclosure of personal or confidential information. With regards to the FIU transactions, the ePRS and eRTR datasets, there is no need for anonymisation as they are publicly available datasets provided by the FIU. In the same manner, the FI identification data owned by BOS do not require any anonymisation to be applied as they are compiled by statistical data with no personal or confidential data.

4.7 Pilot #9 - Analyzing Blockchain Transaction Graphs for Fraudulent Activities

The scope of Pilot #9 is to design and develop a parallel and scalable system that will facilitate the formulation of a massive Bitcoin and Ethereum blockchain transaction graph with distributed dynamic data structures,

exploiting the capabilities of an HPC cluster. In this transaction graph, an analysis will be performed utilising a variety of graph and machine learning algorithms in order to investigate and identify blockchain account transactions that can be traced to fraudulent activities or accounts.

Within the context of Pilot#9, massive actual public anonymous chain data which are available from Ethereum and Bitcoin chains will be collected and utilised. These raw data will be obtained from actual blockchain nodes or gateways or a service like Google BigQuery. Due to the nature of the blockchain technology, as well as the peculiarities of the data stored within a blockchain network, the collected

blockchain datasets differ from the usual datasets. In particular, the Bitcoin raw blocks are parsed in order to be stored as files that contain the respective transactions, while the Ethereum raw blocks are firstly joined and then parsed in order to be stored as files also. The collected transactions are stored in bzip2 compressed files. The following table presents the details of the collected datasets.

Table 31: Pilot #9 list of datasets

Dataset Name	Dataset description	Data format	Anonymization
Ethereum Mainnet Blockchain Dataset	Blockchain data collected from Ethereum Mainnet containing: • Blocks from 0 to 9499999 • Time coverage from 30.07.2015 to 17.02.2020 • 633 762 485 transactions • 69 223 762 addresses • 24 646 152 of 31 Major ERC20 Token Transfer Transactions • Symbols of 31 Major ERC20 Tokens: USDT PAX EURS GUSD TRYB BAT CEO LNK HT HEDG MKR CRO VEN INO INB SNX MOF ZRX SXP OKB XIN SAI HOT DAI HPT BUSD XAUT USDC SUSD HOG QCAD • 18 GB zipped data or 70 GB unzipped data	bzip2	No anonymisation
Bitcoin Blockchain Dataset	Bitcoin blockchain data containing: • Blocks from 0 to 623467 • Time coverage from 03.01.2009 to 29.03.2020 • 516.305.390 transactions • 91 GB zipped data or 310 GB unzipped data	bzip2	No anonymisation

4.8 Pilot #10 - Real-time cybersecurity analytics on Financial Transactions' BigData

The main objective of the Pilot #10 is to significantly improve the detection rate of malicious events (i.e. frauds attempts) and enable the identification of security-related anomalies while they are occurring with the real-time analysis of the financial transactions of a home and mobile banking system.

Within the context of the pilot, the datasets that will be used are related to several types of transactions (SEPA bank transfer, foreign bank transfers, internal transfers of funds, PCTU, SMWCA, STFTS). In detail, synthetic datasets that are consistent with the real data present will be created in the data operations

environment. During the synthetic data creation, models will be built from real world data and domain knowledge. The data will be synthetized using a set of generators which will be based on a lightweight agent based modelling framework while using inferred distributions. Dimensional data will be generated either from classical parametric random generators, or from inferred non-parametric distributions. Every synthetic

dataset will be represented as a scenario, in which dimensional data will be generated either from classical parametric random generators, or from inferred non-parametric distributions. The datasets are then produced as interactions between agents, which use the dimensional data as input to determine their behaviour. The starting point for the synthetic dataset creation will be randomly generated personal data with no correlation with real people. The datasets will be in CSV or ORC format with temporal coverage for the whole year 2019 and an estimated size of 3,5 GB and will be treated as a confidential dataset. The following table documents the details of the collected datasets.

Table 32: Pilot #10 list of datasets

Dataset Name	Dataset description	Data format	Anonymization
Bank Transfer SEPA	Single Euro Payments Area (SEPA) transactions that cover predominantly normal bank transfers.	CSV or ORC	No anonymisation (fully synthetic datasets)
Foreign Bank Transfers	International transfer of funds between banking accounts held in financial institutions located in different countries	CSV or ORC	No anonymisation (fully synthetic datasets)
Internal transfers of funds	Transactions involving transfer of funds from the Bank current account of one payer to that of another Bank account's holder	CSV or ORC	No anonymisation (fully synthetic datasets)
PCTU	Top-ups phone credit	CSV or ORC	No anonymisation (fully synthetic datasets)
SMWCA	Sending Money to people Who do not have a Current Account (SMWCA) transactions as a means of payment for transferring money to those not holding a bank account	CSV or ORC	No anonymisation (fully synthetic datasets)
STFTS	Secure telematic fund transmission system (STFTS) where neither the sender nor the receiver of the money being transferred need to hold a banking account	CSV or ORC	No anonymisation (fully synthetic datasets)

4.9 Pilot #11 - Personalized insurance products based on IoT connected vehicles

The main objective of Pilot #11 is to improve the risk profiles in car insurance by using the information collected from connected vehicles and applying Artificial Intelligence technologies. The overall concept of Pilot #11 is summarised in **Error! Reference source not found.**. Within the context of this specific pilot, a "Pay as you drive" service capable of adapting the insurance costs to the actual driver's way of driving will be developed. A "Fraud detection" service aiming to help insurance companies will be also developed.

In order to prevent data protection issues, an Anonymization Tool that is offered by GRAD will be used. Within Pilot #11, the Anonymization Tool will be exploited in order to effectively address all the data privacy concerns related to the location privacy. Therefore, the specialised anonymisation algorithm offered by the tool will be applied to the geolocated data with the aim of ensuring that user's exact spatial coordinates will not be revealed as this could lead to a possible re-identification of the data subjects.

The open standard FIWARE NGSI serves as the basis for the implementation of the data gathering process within Pilot #11 and specifically an instance of its NGSIv2 interface. In this context, all data stored in the Connected Car Platform, that manages the data collection process of Pilot #11, will be generated according to the **FIWARE Vehicle Data model** [5], the **FIWARE Alert Data model** [6] and the **FIWARE Weather Observed Data model** [7] and in NGSI format, so that they can be retrieved using the same standard and the whole process will be aligned with the published ETSI NGSI-LD specification [8].

Table 33: Pilot #11 List of datasets

Dataset Name	Dataset description	Data format	Anonymization
Connected Vehicles	Data collected from connected vehicles (provided Automotive Technology Centre of Galicia – CTAG). CANBus + NMEA (location) from real connected vehicles.	CSV	GRAD Anonymisation Tool
Simulated Vehicles	Simulated Urban mobility data (mainly vehicles CAN Signals) generated through SUMO tool. One scenario reporting 30K vehicles (around 8 Gb of data)	JSON	No anonymisation (synthetic datasets)
Roads datasets	Roads data extracted from OpenStreetMap (2 Gb for considered scenarios).	JSON	N/A
Weather information	Collected from AEMET weather stations. 2k registries (around 1 Mb of data).	JSON	N/A
Traffic Events	Traffic events published by DGT. 1k registries (around 0,5 Mb of data).	JSON	N/A
Historical datasets (CANBus data + NMEA)	Historical datasets (CANBus data + NMEA) from real connected vehicles	CSV	Anonymised
Historical datasets (traffic events)	Historical datasets from traffic events	JSON	N/A
Motor Insurance Data	Data concerning motor insurance including data from the policies (duration, covers), data from vehicles (licence No, VIN etc.) and data from drivers (age, experience etc.)	CSV	Anonymised

4.10 Pilot #12 - Real World Data for Novel Health-Insurance

products

The scope of Pilot #12 is to assist health insurance companies in improving the risk insurance profiles using the information collected by activity trackers and questionnaires, and by applying IoT & ML technologies on the collected information. In this context, the pilot aims to deliver two distinct services, one performing risk assessment and another one for fraudulent behaviour detection.

To this extend, Real-World Data (RWD) spanning from physiological, psychological to social and environmental aspects, as well as synthetic data (simulated lifestyle) will be collected in the context of the

pilot. These data are either measured using sensors on devices, or are reported by the participants via questionnaires. Both the measurements and the questionnaire posting and answering are managed by the Healthentia platform[11] of Innovation Sprint. Besides the RWD, synthetic data will be provided, utilising a simulation tool that Innovation Sprint is developing. The data types will be similar to the ones collected by actual Pilot #12 participants. The produced synthetic data will be stored in Healthentia platform, so they will be accessed by the Pilot #12 INFINITECH infrastructure in the same as manner as with the actual data.

In detail, in the initial, proof of concept (PoC), phase of Pilot #12 the data that will be collected are documented in the following table.

Table 34: Pilot #12 List of datasets

Dataset Name	Dataset description	Data format	Anonymization
Physiological	The data physiological data collected are related to steps, distance travelled, time spent in different activity intensity categories, energy burned, floors climbed, exercise sessions. In the case where the participant has a Fitbit device, then resting heart rate, time spent in different heart rate zones, time to bed and wake up, time spent in different sleep categories is also measured. In the case where no Fitbit device is available, questionnaires are employed to collect time of sleep and wake up. In both cases, liquid and food intake is quantified via questionnaires, as well as body weight and various symptoms (blood pressure, body temperature, cough, diarrhea, fatigue, headache and pain).	JSON	GRAD Anonymisation Tool
Psychological	Current mood is collected via a questionnaire	JSON	GRAD Anonymisation Tool
Social	A weekly questionnaire is employed to collect the impression of the participants on how they cope with instrumental activities of daily life.	JSON	GRAD Anonymisation Tool

As the data collected by the Healthentia platform contain personal and / or sensitive data, the Anonymisation Tool offered by GRAD will be used in order to anonymise the data in a way the participants' privacy is preserved, hence removing any data privacy concerns. In particular, different anonymization algorithms (such as generalization, randomization, etc.) will be applied to avoid the existence of data

combinations that could lead to a possible re-identification of the data subjects. Additionally, a set of privacy and utility metrics that allow to measure the risk that remains after anonymizing the data and the impact of the anonymization process on the quality of the data, will be calculated by the Anonymisation Tool and the results of the applied anonymisation algorithms will be assessed before the final decision is reached. The synthetic data will be bypassing the Anonymization Tool, as they do not need anonymisation.

4.11 Pilot #13 - Alternative/automated insurance risk selection - product recommendation for SME

The main objective of Pilot #13 is to implement a data analysis platform applying machine learning and artificial intelligence technologies to better predict the insurance needs of SMEs. In this context, the platform

[11] Healthentia Platform, https://innovationsprint.eu/healthentia/

will generate a risk map of the SMEs based on their daily activities and will predict how the risk will vary on time. Therefore, the pilot will design and implement a service that effectively monitors the current risks of SMEs, as well as their risk variance in the future, in order to improve the control of the accident rate, the renewal of insurance policies and offer personalised insurance cover.

To this end, real data that will be obtained from a variety of open data sources such as the web, social media profiles, official registers, opinion platforms, business directories, e-commerce platforms and more, will be leveraged. The dataset will be composed of data originating from 50000 EU SMEs and will have both structured and unstructured data in image (PNG) format and text format, while the estimated data volume is expected to be around 1 TB of data. The following table summarize the main data sources that will be exploited in the context of Pilot #13:

Table 35: Pilot #13 List of datasets

Dataset Name	Dataset description	Data Format	Anonymisation
SMEWIF	SMEs website information and functionalities. Description of the text contained in the website of the companies, services and structure of the company: • Name, Brand Name • Business Activity • Phone Number, Brand Address, Website Phone Number • Description from company • Website • Security Protocol • Play Store, APP Store • Location from website • Nº Opinions, Average Rating • E-Mail • Business Hours • Categories • Claimed	AWS S3 / Dynamo DB (text format)	Not required (Open data)
ROPS	Review and opinions platforms. Reputation information and opinions of clients about	AWS S3 / Dynamo DB	Not required (Open data)

	products and services.	(text format)	
EUBD	European SMEs Business Directories. Official and legal information about the companies, social object, activities, other companies where they have equity: • Legal Name • Registered Address • Address • Court • Last Announcements in Commercial Register • Incorporated • Company Type • Age of company • Number of employees • SHARE CAPITAL • SIC/NACE CODE • VAT NUMBER	AWS S3 / Dynamo DB (text format)	Not required (Open data)

GIO	SMEs geolocation information and characteristics images and geographical information.	AWS S3 / Dynamo DB (text format	Not required (Open data)
SMSIP	Social media SMEs information and presence. Evaluation and track of the channels where the SMEs have online presence.	AWS S3 / Dynamo DB (text format	Not required (Open data)

In addition to the real data, synthetic data will be leveraged towards the aim of increasing the prediction accuracy and the usability of the designed machine learning models. The synthetic data creation will aim to produce hybrid synthetic data by building models from real world data and domain knowledge and leveraging a simulator that will be driven by these models to generate synthetic data. Within the context of the pilot, either open data or synthetic data will be utilised. Hence, the need for any anonymisation process is not foreseen.

4.12 Pilot #14 - Big Data and IoT for the Agricultural Insurance Industry

The objective of Pilot #14 is to deliver a commercial service module which will enable insurance companies to exploit the untapped market potential of Agricultural Insurance (AgI), taking advantage of innovations in Earth Observation (EO), weather intelligence & ICT technology. Towards this end, EO will be leveraged to implement a crucial new supplementary information source which will be utilised by insurance companies in their products design and risk assessment processes related to nature disasters. On the other hand, weather intelligence related to data assimilation, numerical weather prediction and ensemble seasonal forecasting will be leveraged with the aim of effectively verifying the occurrence of catastrophic weather events, as well as predicting predict future perils, providing crucial information to the agricultural insurance companies for possible threats of their portfolios.

In this context, Pilot #14 will mainly exploit the data produced by the satellite and the weather intelligence

engine which will be complemented by anonymised in-situ data of the insured parcels that are currently used both as input and calibration data for the existing insurance services. In detail, the following data will be utilised in Pilot #14:

Table 36: Pilot #14 List of datasets

Dataset Category	Dataset Name	Dataset description	File Format
Earth Observation data	Sentinel-2	Sentinel-2 products follow a specific naming code as described below: *S2A_MSIL2A_YYYYMMDDTHHMMSS_Nxxyy_ROOO_Txxxxx_<Product Discriminator>* where: • S2A or S2B = Spacecraft • MSI = Instrument • L2A = Processing level • YYYYMMDDTHHMMSS = Sensing start time and sensing stop time of the first line of granule in date UTC time format • Nxxyy = Processing Baseline number (e.g. N0204) • ROOO = Relative Orbit number (R001 - R143) • Txxxxx = Tile Number field	netcdf4, tif
	Sentinel-1	The top-level Sentinel-1 product folder naming convention is composed of upper-case alphanumeric characters separated by an underscore: *MMM_BB_TTTR_LFPP_YYYYMMDDTHHMMSS_YYYYMMDDTHHMMSS_O OOOOO_DDDDDD_CCCC.EEEE* where:	netcdf4, tif

		• The Mission Identifier (MMM) denotes the satellite and will be either S1A for the Sentinel-1A instrument or S1B for the Sentinel 1B instrument. • The Mode/Beam (BB) identifies the S1-S6 beams for SM products and IW, EW and WV for products from the respective modes. • Product Type (TTT) can be RAW, SLC, GRD or OCN. • Resolution Class (R) can be F (Full resolution), H (High resolution) or M (Medium resolution) • The Processing Level (L) can be 0, 1 or 2. • The Product Class can be Standard (S) or Annotation (A). Annotation products are only used internally by the PDGS and are not distributed. • Polarisation mode (PP) can be of dual, single or partial-dual type. • The product start and stop date and times are shown as fourteen digits representing the date and time, separated by the character 'T'. • The absolute orbit number at product start time (OOOOOO) will be in the range 000001-999999. • The mission data-take identifier (DDDDDD) will be in the range 000001-FFFFFF. The product unique identifier (CCCC) is a hexadecimal string generated by computing CRC-16 on the manifest file. The CRC-16 algorithm used to compute the unique identifier is CRC-CCITT (0xFFFF).	

	Landsat-8	Following is a typical naming of a Landsat-8 product: *LXSS_LLLL_PPPRRR_YYYYMMDD_yyyymmdd_CX_TX_prod_band* where: • L: Landsat • X:Sensor (“O” = OLI; “T” = TIRS; “C” = OLI/TIRS) • SS: Satellite (“08” = Landsat 8) • LLLL: Processing correction level (“L1TP” = Precision Terrain; “L1GT” = Systematic Terrain; “L1GS”= Systematic) • PPP: Path • RRR: Row • YYYY: Year of acquisition, MM Month of acquisition, DD Day of acquisition • yyyy: Year of processing, mm Month of processing, dd Day of processing • CX: Collection number (“01”, “02”, etc.) • TX: Collection category (“RT” = Real-Time; “T1” = Tier 1; “T2” = Tier 2) • prod: Product, such as “toa” or “sr” • band: such as “band<1-11>,” “qa,” or spectral index.	tif, tfw, xml, hdf, hdr, nc, or img
	PROBA-V Dry Matter Productivity (DMP)	The DMP collection 300 m v1 product follows the naming standard as follows: *c_gls_<product>[-<RTx>]_<YYYYMMDDhhmm>_<AREA>_<SENSOR>_V<Major.Minor.Run>* where: • <product> is the DMP300. • <RTx> is an optional parameter. It is used whenever a real-time product is provided: • RT0: Near Real Time product; • RT1 or RT2: Consolidated Real Time product (in convergence period), where the number equals the number of times the RT0 product was successively updated; • RT5: Final consolidated Real Time product.	tif, tfw, xml, hdf, hdr, nc, or img

		• <YYYYMMDDhhmm> gives the temporal location of the file. YYYY, MM, DD, hh, and mm denote the year, the month, the day, the hour, and the minutes, respectively. • <AREA> gives the spatial coverage of the file. For example, if the <AREA> is GLOBE, then the full globe product is available. • <SENSOR> gives the name of the sensor family used to retrieve the product, so VGT referencing SPOT-VEGETATION, and PROBAV for PROBA-V. • <Major.Minor.Run> gives the version number of the product. “Major” increases when the algorithm is updated. “Minor” increases when bugs are fixed or when processing lines are updated (metadata, colour quicklook, etc.). “Run” increases whenever a new processing run (with the same major and minor version) is performed without a change in the algorithm (e.g. due to a change in input data). This version refers to Major = 1.	

Weather and Climate Data	Numerical weather predictions	A typical weather product follows the naming standard as follows: *<Weather_Parameter>_<AOI>_<Year>-<Month>-<Day>-<Hour>:00:00* where: • <Weather_Parameter> is the corresponding weather parameter (Temperature, RH etc.) • <AOI> is the Area of Interest • <Year> is the corresponding year • <Month> is the corresponding month • <Day> is the corresponding day of the month • <Hour> is the corresponding hour	tif, tfw, xml, hdf, hdr, nc, or img
Insured Parcels	Parcel Data	In order to be ingested by the system, all the required information related to the insured parcels are collected in a Shapefile format file reporting the following information for each parcel: • Parcel ID number • Boundaries (i.e. lat/lon coordinates of the corners of the field) • Soil texture (i.e. % sand, % clay, % silt) • Crop type • Size (in ha) • Country/county/area • Insured from (the type of the damage, the parcel is insured of) • Insured Value • Contract Duration • Total insured area (in case not the entire parcel is insured) • Total Insured value • Expected yield (per parcel or per ha)	Shapefile format file

5 Conclusions

The purpose of this deliverable entitled “D5.13 – Datasets for Algorithms Training & Evaluation - I” was to report the preliminary outcomes of the work performed within the context of T5.1 “Data Collection for Algorithms Training & Evaluation” of WP5. To this end, the deliverable documented the initial design specifications, the addressed use cases along with the respective sequence diagrams and the implementation details of the INFINITECH Data Collection component. Furthermore, it documented the characteristics and role of the synthetic datasets, as well as how they will be exploited in INFINITECH. Finally, it presented the details of the datasets that are collected within the context of the pilots of INFINITECH.

At first, the deliverable introduced the INFINITECH Data Collection by describing its scope, the motivation for the development of the component and the challenges that it addresses. Following the definition of the component’s scope, the high-level architecture of the component, that is composed of three main modules, namely the Data Retrieval, the Data Mapper and the Data Cleaner, was presented. For each sub-component, the design specifications were documented in detail by presenting their main functionalities. Following the design specifications, the use cases that each sub-component addresses were presented along with the relevant sequence diagrams that depict the interactions between the sub-components and the respective stakeholders. Moreover, the implementation details of each sub-component were presented by documenting the source code structuring via UML diagrams and the details of the incorporated functions.

Following the INFINITECH Data Collection documentation, an in-depth analysis of the synthetic datasets was presented. Within this analysis, the advantages and limitations of the synthetic datasets were presented along with their categorisation into three main categories. The analysis provided an overview of the approaches that are followed during the synthetic data generation process along with the list of tools and

frameworks that are utilised within this process. Moreover, the motivation and use cases for which the synthetic datasets will be leveraged within the context of INFINITECH project were presented, accompanied by the preliminary list of synthetic datasets which will be used by the INFINITECH pilots.

Finally, the deliverable presented the list of datasets that will be exploited by the INFINITECH pilots. The composed list of datasets of each pilot is tailored to their needs and it is driven by the details of the financial and insurance scenarios that will be executed within the context of each pilot. To this end, the preliminary list of datasets that will be utilised were presented, focusing on the description of the information included, the format of each dataset and the needs for anonymisation in order to cope with the data privacy requirements.

It should be noted at this point that the current deliverable reports the initial design specifications and implementation details of the INFINITECH Data Collections component, as well as the initial list of datasets of the INFINITECH pilots, accompanied by the preliminary list of synthetic datasets that will be exploited by the pilots. The outcomes of this deliverable will drive the implementation activities that will be performed within the context of T5.1, as well as of the rest of the tasks of WP5. Nevertheless, as the project evolves and new requirements may arise, as the result of the feedback that will collected by the pilots of the project and the stakeholders of the platform, optimisations and enhancements might be introduced and they will be documented in the upcoming version of the deliverable which will be delivered on M27 in deliverable D5.14.

Table 37 - Conclusions (TASK Objectives with Deliverable achievements)

Objectives	**Comment**
Enable the exploitation of the variety of finance and insurance sector data sources.	The proposed solution successfully enables the collection of information from the variety of data sources that are exploited in the finance and insurance sector. It facilitates the effortless and efficient data collection, data mapping and data cleaning towards the ingestion of the information in the underlying storage in order to be further processed and exploited by the INFINITECH Machine Learning services.

Provide the mechanism that addresses the various connectivity and communication challenges of the complete data collection process	The INFINITECH Data Collection delivers the required abstract and holistic mechanism for the data providers of INFINITECH to effectively and efficiently collect, map into a data model and clean the required information from a large variety of data sources. Furthermore, through its microservice-based architecture it provides the means for further expansion of the supported data sources upon needs.
Design and deliver the required solution that enables the connection and retrieval of the information for different data sources	The detailed design specifications, as well as the first release, of the INFINITECH Data Collection, are delivered with the current deliverable. The designed functionalities successfully the needs of the INFINITECH data provider to connect and retrieve information from a data source. The implementation of the INFINITECH Data Collection is an ongoing process that will last until M27.

Table 38: Conclusions – (map TASK KPI with Deliverable achievements)

KPI	**Comment**

Data Collection mechanism available for enabling the collection and ingestion of data in an INFINITECH testbed	*Target Value = 1* The specific KPI is successfully achieved with the presented solution, namely the INFINITECH Data Collection, whose first version was released in the current deliverable, accompanied by the details detailed design specifications which were also documented.
Coverage of different data source types	*Target Value = 6* Per the design specifications documented in the current deliverable, the INFINITECH Data Collection is capable of: • Retrieving new information from an API • Receiving new information from its exposed API • Fetching files from an FTP or HTTP server • Retrieve new information from a Relational Database • Retrieve new information from a HDFS deployment • Retrieve new information from a MinIO storage server
Number of services exposed to the clients of the INFINITECH Data Collection.	*Target Value >= 3* The INFINITECH Data Collection exposes three main services: a) the Data Retrieval service, b) the Data Mapper service, c) the Data Cleaner service

Appendix A: Literature

[1] A. Z. R. a. B. S. Dandekar, “Comparative evaluation of synthetic data generation methods,” in *ACM Conference (Deep Learning Security Workshop)*, 2017.

[2] H. Surendra and H. Mohan, “A review of synthetic data generation methods for privacy preserving data publishing,” *International Journal of Scientific & Technology Research,* vol. 6, no. 3, pp. 95-101, 2017.

[3] V. Ayala-Rivera, P. McDonagh, T. Cerqueus and L. Murphy, “Synthetic Data Generation using Benerator Tool,” University College Dublin, 2013.

[4] S. Assefa, D. Dervovic, M. Mahfouz, T. Balch, P. Reddy and M. Veloso, “Generating synthetic data in finance: opportunities, challenges and pitfalls,” JPMorgan Chase & Co, 2020.

[5] “FIWARE Vehicle Data model,” [Online]. Available: https://fiware datamodels.readthedocs.io/en/latest/Transportation/Vehicle/Vehicle/doc/spec/index.html.

[6] “FIWARE Alert Data Model,” [Online]. Available: https://fiware datamodels.readthedocs.io/en/latest/Alert/doc/spec/index.html.

[7] “FIWARE Weather Observed Data model,” [Online]. Available: https://fiware datamodels.readthedocs.io/en/latest/Weather/WeatherObserved/doc/spec/index.html.

[8] “ETSI NGSI-LD specification,” [Online]. Available: https://www.etsi.org/deliver/etsi_gs/CIM/001_099/009/01.01.01_60/gs_CIM009v010101p.pdf.

[9] “FIWARE,” [Online]. Available: https://www.fiware.org/developers/.

[10] “NGSI specifications,” [Online]. Available: http://fiware.github.io/specifications/ngsiv2/stable/.

Appendix B: INFINITECH List of synthetic datasets

Pilot #	Dataset Name	Short Description	Dataset Provider	Type	Dataset Format	Dataset Volume	Synthetic Data Category	Synthetic Data Methodology / Strategy	Library, Framework, Tool utilised
P10	PI Financial transactions data								
P11	ATOS Connected Car dataset (FIWARE NGSI Vehicle data models)								
P11	ATOS Traffic Dataset (FIWARE NGSI								

Road &
RoadSegment
data models)

Synthetic real time dataset related to financial
transactions of the following: STFTF , PCTU, Bank Transfer SEPA , Foreign Bank Transfers, Internal Transfer of funds, SMWCA

Real time dataset related to connected vehicles (location, status, speed, acceleration, fuel consumption, etc.) driving within selected
scenario

Real time datasets related to traffic (road occupation, traffic status, etc.) referred to the cities included in out SUMO-Based simulation tool (Cologne, Luxemburg,
Monaco)

Text /
Numeric

FIWARE NGSI
Vehicle extended data
model

FIWARE NGSI
Road and RoadSeg ment
extended data
models

CSV 3.5GB per Year

JSON ~10MB per
simulatio
n

JSON ~10MB per
simulatio
n

Fully
Synthetic

Fully
Synthetic

Partially Synthetic

Build models from real world data and domain knowledge. The data will be synthetized using a set of generators which will be based on a lightweight agent-based modelling framework while using inferred distribution. Datasets extracted from Real time simulation. Historical datasets would be also generated

Datasets extracted from Real time simulation. Historical datasets would be also generated
Python
Libraries

Atos SUMO 2 NGSI

Atos SUMO 2 NGSI

P12 iSprin t
Healthentia Simulated
Activity tracking data (steps, sleep etc.) and reported data regarding nutrition, health and quality of life self assessments
Numeric al Data
JSON Depends on the
scenario
executio
n
Fully
Synthetic
Build models from real world data and domain knowledge. Use a simulator driven by the models to generate synthetic data
Python Libraries

P13 WEA SMEs synthetic raw data
SMEs raw data to complete the model and algorithm
Text JSON 120 Kb per
target/co
mpany
Hybrid Synthetic
Build models from real world data and domain knowledge. Use a simulator driven by the
Python Libraries

P14 AGRO Crop Biophysical
Parameters
Crop Biophysical Parameters that will be used from crop health and crop productivity assessment. Typical examples of these parameters are Biomass, Chlorophyll content, Leaf Area Index and Yield.
Raster GeoJSON / NETCDF
Depends on the
size of
the Pilot area

Partially Synthetic
models to generate synthetic data
Datasets are produced using ESA SNAP ANN
AgroApps OCTAPUSH/ Weather Intelligence Engine

P14 AGRO Crop Loss Crop loss assessment after an insured peril
Raster GeoJSON / NETCDF
Depends on the
size of
the Pilot area
Partially Synthetic
Datasets are produced using AGROAPPS Ensemble EO change detection
methodology and machine learning methodology for translating damages to actual crop losses
AgroApps OCTAPUSH/ Weather Intelligence Engine

P14 AGRO Seasonal Crop Yield Prediction
Crop Yield Climatology and Seasonal Crop Yield Prediction
Raster GeoJSON / NETCDF
Depends on the
size of
the Pilot area
Partially Synthetic
Datasets will be produced by using the GECROS crop simulation model
AgroApps OCTAPUSH/ Weather Intelligence Engine