Active Mobility Innovations for Green and safe city sOlutionS

Grant Agreement n°101104268

D1.3 – Big Data Platform

This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No. 101104268 (AMIGOS). Views and opinions expressed are those of the author(s) only and do not necessarily reflect those of the European Union or CINEA. Neither the European Union nor the granting authority can be held responsible for them.

This project is part of the CIVITAS initiative.

Start date of project	01/06/2023
Duration of project	48 months
Deliverable n° & name	D1.3 – Big Data Platform
Version	1.0
Work Package n°	1
Due date of the deliverable	M12 – 31/05/2024
Participant responsible	Tree Technology (TREE)
Main authors	Alejandro Gámez, Cristian Robledo, Krzysztof Trawinski, Ramón Casado, Tatiana Silva (TREE)
Website	www.amigos-project.eu

Nature of the deliverable		
R	Document, report (excluding the periodic and final reports)	
DEM	Demonstrator, pilot, prototype, plan designs	
DEC	Websites, patents filing, press & media actions, videos, etc.	
DATA	Data sets, microdata, etc.	
DMP	Data management plan	
ETHICS	Deliverables related to ethics issues	
SECURITY	Deliverables related to security issues	
OTHER	**Software, technical diagram, algorithms, models, etc.**	**X**

Dissemination level		
PU	Public, fully open, e.g. web (Deliverables flagged as public will be automatically published in CORDIS project's page)	X
SEN	Sensitive, limited under the conditions of the Grant Agreement	
CI	Classified, EU RESTRICTED, CONFIDENTIAL or SECRET under the Commission Decision No2015/444	

Quality Procedure			
Date	**Version**	**Reviewers**	**Comments**
01/04/2024	0.1	Alejandro Gámez (TREE)	Table of Contents
29/04/2024	0.2	Alejandro Gámez, Ramón Casado, Cristian Robledo, Krzysztof Trawinski, Tatiana Silva (TREE)	First complete version of the deliverable
24/05/2024	0.3	Alejandro Gamez, Ramón Casado, Tatiana Silva (TREE)	Address comments from ESTACA colleagues (internal reviewers)
27/05/2024	0.4	Tatiana Silva (TREE)	Final version to be shared with the coordinator and WP1 leader
03/06/2024	0.5	Avihai Degani (IPG)	Inputs about RT AI based platform

07/06/2024	0.6	Tatiana Silva (TREE)	Integrated version with IPG inputs, some amendments requested to IPG
11/06/2024	0.7	Avihai Degani (IPG)	Additional inputs according to the requests from TREE (figures)
13/06/2024	1.0	Tatiana Silva (TREE)	Integration with the new inputs, final version

Contenido

List of Figures

List of Tables

1. Project abstract

To reach carbon neutrality, cities must adopt new, more adapted energy models for urban mobility, relying on zero-emission and active mobility modes. The uptake of sustainable mobility solutions relies on their inclusivity, affordability and safety, as well as their consistency with users' needs. Through co-creation activities and innovative digital tools, the AMIGOS project will identify present and future mobility challenges for 5 cities (living labs) and 10 urban areas (safety improvement areas). The digital tools include a Mobility Observation Box and an application for the collection of new mobility data, which will feed a big data platform for their analysis and digital twins to visualize mobility scenarios. They will allow urban stakeholders to identify mobility challenges and will serve as a basis for the co-development of adapted mobility solutions: towards reducing traffic, increasing public and active mobility modes, improving safety and co-habitation between different mobilities for the 5 cities, and towards increased safety for the 10 urban areas.

Therefore, key stakeholders such as public authorities and vulnerable users will be included in the definition of technological and policy solutions mobility solutions which will be implemented in the cities. Their environmental, safety, economic and social impacts will be assessed, in addition to their medium- and long-term impact and their replicability, in view of their implementation in 5 twin cities.

2. Executive summary

D1.3 addresses the challenge of developing a platform for AMIGOS data storage and processing. Most of the data will be mainly provided by the cities (car traffic, bikes traffic, pedestrians, noise and air pollution data, Mobility Observation Box data from each city…) in different points in time. This data will feed the services to be developed in WP3 (for example, air and noise pollution improvements or optimizing the existing transport network). This data will be crucial to the development and utilization of the three digital urban twins. It will present interesting data for city authorities, local transport authorities and researchers.

The platform was designed after asking to the AMIGOS cities about the data they will be sending to the Platform and understanding their requirements, as well as the requirements of the partners who may consume the data to develop a concrete service.

The data is a combination of accumulated data: Big Data and the real-time (RT) streaming data. The Big Data Platform is provided by TREE and the real-time AI based platform is provided by IPG.

The real time data is the data representing current values provided by relevant providers e.g. IoT. This data, after being analysed, defines events to be handled by the city as soon as possible e.g. air quality related events. Another use case related to real time data is the parking availability and complements the predicted probability of finding an available parking spot at a specific area.

The Big Data platform will provide with insights, like predictions and pattern recognition to be used together with the Real-Time Data to define events that will help to come up with recommendations to address the specific per city challenges.

3. Abbreviations

Table 1 List of abbreviations.

Abbreviation	Definition
ACID	Atomicity, Consistency, Isolation, and Durability
API	Application Programming Interface
AWS	Amazon Web Services
ETL	Extract, Transform and Load
ML	Machine Learning
MOB	Mobility Observation Box
TREE	Tree Technology S.A.
IPG	Gallery IP Telephony

4. Introduction

Smart cities utilize multiple technologies to improve the performance of transportation services leading to higher levels of comfort of their citizens. This involves reducing costs and resource consumption in addition to more effectively and actively engaging with their citizens. One of the recent technologies that has a huge potential to enhance smart city services is big data analytics. As digitization has become an integral part of everyday life, data collection has resulted in the accumulation of huge amounts of data that can be used in various beneficial application domains. Effective analysis and utilization of big data is a key factor for success in many business and service domains, including the smart city domain. To this aim, Big data analysis and machine learning algorithms can play a fundamental role to bring improvements in city policies and urban issues.

This document presents the work done in "T1.5 Big Data Platform" within Work Package 1 "Preliminary analysis and stakeholder engagement", encompassing the Big Data Platform and the RT AI platform.

The purpose of the document is to explain the Big Data Platform and the real-time AI based platform, their architecture and how they were developed.

The next steps planned for these two platforms are included, as they will need to evolve during the project.

5. Overview of the Big Data Platform

The AMIGOS Big Data platform is a comprehensive Big Data solution designed to facilitate the management and analysis of large-scale city data. Being cloud-based, this platform focuses on collecting both structured and unstructured information from multiple cities, enabling a holistic and collaborative approach to data management, and the use of the cloud brings scalability, accessibility and security to the platform. The AMIGOS platform aims to improve city-to-city collaboration and data-driven decision making, thus contributing to more effective urban planning and improved citizen services.

5.1. General Vision

The Big Data platform of the AMIGOS project is a system designed to collect, store and process both **structured and unstructured information** from different cities and different data sources.

Structured information refers to data that is organised and has a defined format, which makes it easy to store, search and analyse. This data is usually defined in tables and assigned data types.

Unstructured information, on the other hand, is information that does not follow a specific model or format, making it difficult to process using traditional methods. Examples of unstructured data are free text, images, videos and data collected from urban sensors. This type of information is becoming increasingly prevalent and its importance in urban and behavioural analysis is fundamental.

The AMIGOS platform is designed as a cloud-based solution. This means that all data collected from different cities is stored on servers that are in data centres accessible via the internet, rather than being confined to a single physical location.

A cloud-based platform offers multiple key benefits such as extraordinary scalability, allowing storage and processing capacity to adjust as data volumes increase without requiring additional infrastructure. It facilitates accessibility from any location, which is essential for collaboration between different cities and departments, especially in projects involving multiple government entities, like AMIGOS. In addition, the pay-as-you-go model of cloud services allows the consortium to avoid large expenditures on hardware and maintenance, providing a cost-efficient solution. Finally, cloud platforms offer robust security measures and disaster recovery policies, ensuring the protection and rapid restoration of data, if necessary.

In conclusion, the Big Data platform of the AMIGOS project seeks to leverage Big Data technology to process both structured and unstructured information efficiently and securely thanks to its implementation in the cloud. These benefits not only enable more effective and collaborative management between cities but also open up new possibilities in the analysis of large volumes of data.

5.2. Architectural overview

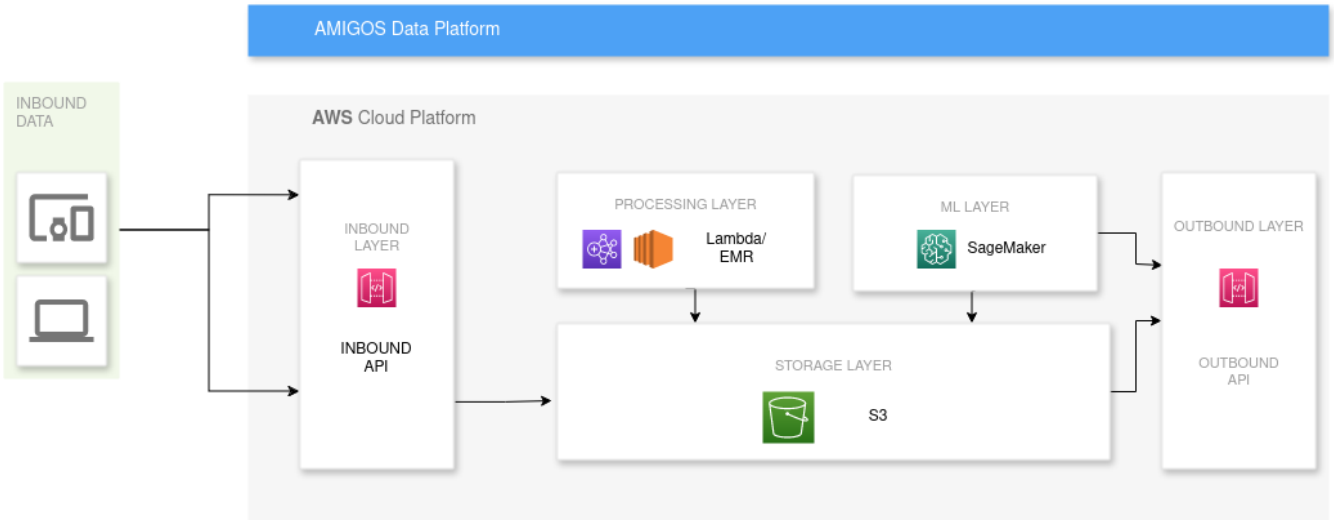

Figure 1 Big Data Platform architecture.

The Big Data platform architecture (Figure 1) of the AMIGOS project focuses on a distributed cloud model that integrates scalable services and resources to efficiently handle large volumes of data generated by cities. This architecture consists of three main components:

- **Inbound layer**: This layer oversees ingesting information from different data sources and different cities. This layer is based on the use of an API (Application Programming Interface) to allow the loading of information and to allow following a previous agreed structure before uploading the information to the platform.
- **Storage Layer and Processing Layer**: Centralised on cloud servers, this layer uses distributed storage and parallel processing technologies to handle and store large data sets. It includes the Data Lake itself and the processes that structure the information for further analysis.
- **Outbound Layer**: Provides information consumption APIs that allow authorised external users (partners from the consortium, in this case) to access structured data and the KPIs and results of analytical processes.

Together, these components work in sync to ensure that the platform not only collects and stores data efficiently, but also provides valuable insights that can be acted upon in a timely manner by users and cities (see Figure 2).

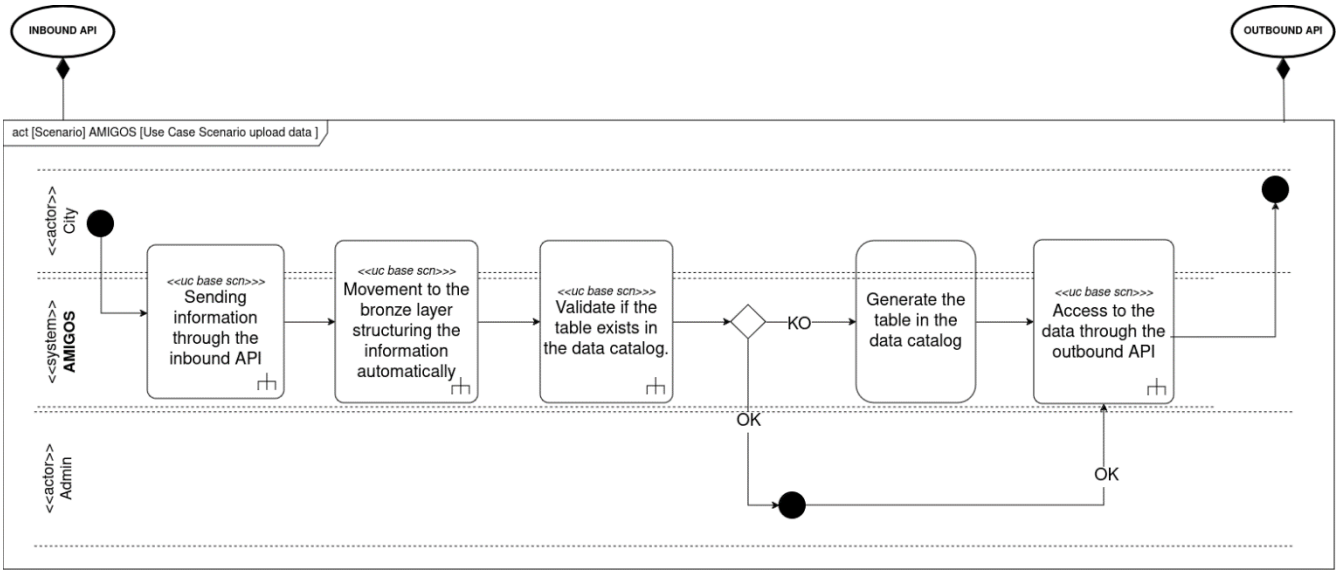

Figure 2 Interaction between components.

Start: The process begins with a call to the Inbound API, where an entity, possibly a city, uses the Inbound API to upload data.

Raw to bronze: Once the data is received, the information is moved to the bronze layer, which we will describe in depth in the next point. This step involves a transformation where the information is automatically structured.

Validation: The next step is a validation where the system checks whether the table structure sent already exists in the data catalogue. This is crucial to avoid duplication and inconsistencies in the Bronze layer.

- Generation or Access: Depending on the result of the validation, the workflow is bifurcated:
- If the table does not exist (KO branch), the system proceeds to "Generate the table in the data catalogue". This involves creating a new entry in the catalogue for the new data.

- If the table does exist (OK branch), the system allows the ingestion of the information in the bronze layer into the already existing table. This means that the data is now available and can be accessed by other systems or processes.

End process: Finally, the process concludes with the data already loaded and accessible through the Outbound API, allowing other services or entities to use the updated information.

6. Data Lake architecture

The Data Lake is a warehouse architecture designed to store large amounts of data (Figure 3). This architecture facilitates the massive storage of unstructured, structured and semi-structured data, and is ideal for projects that need to store large volumes of data from different sources for further analysis and processing.

As described below, the Data Lake of the Big Data platform of the AMIGOS project has been structured in different layers and sub-layers to facilitate both access management and the life cycle of the information and the structure of the data itself (Figure 3).

The structure of the first level layers raw, bronze, silver and gold will be described in the following subsections, but the sub layers by domains, cities and dates will allow the segmentation of accesses and permissions to the data, both by cities and by domains if necessary. The dates will allow the easy management of the life cycles of the information.

6.1. Raw Layer

The raw layer is the first line of ingestion in a Data Lake architecture, where data is stored in its raw and original form without any prior processing. It is critical to maintain the integrity of the data during storage to ensure that it is authentic and reliable for future analysis.

6.2. Bronze layer

The bronze layer is responsible for the first stage of raw data processing. Here, the data undergoes a light transformation, that may include cleaning of obvious errors, normalisation of formats and structuring the data in tabular format, in particular delta lake format. This delta format is the one that will be used from now on in the bronze, silver and gold layers. Delta Lake stands out for its ability to guarantee data integrity with Atomicity, Consistency, Isolation, and Durability (ACID) operations, evolutionary schemas, versioning and efficient processing on platforms such as Apache Spark.

6.3. Silver layer

In the silver layer, the already transformed data undergoes further cleaning and preparation for use. This may involve removing duplicates, correcting errors, imputing missing values and improving data quality. Often, it is here that data is structured to fit into defined schemas, facilitating analysis and insights.

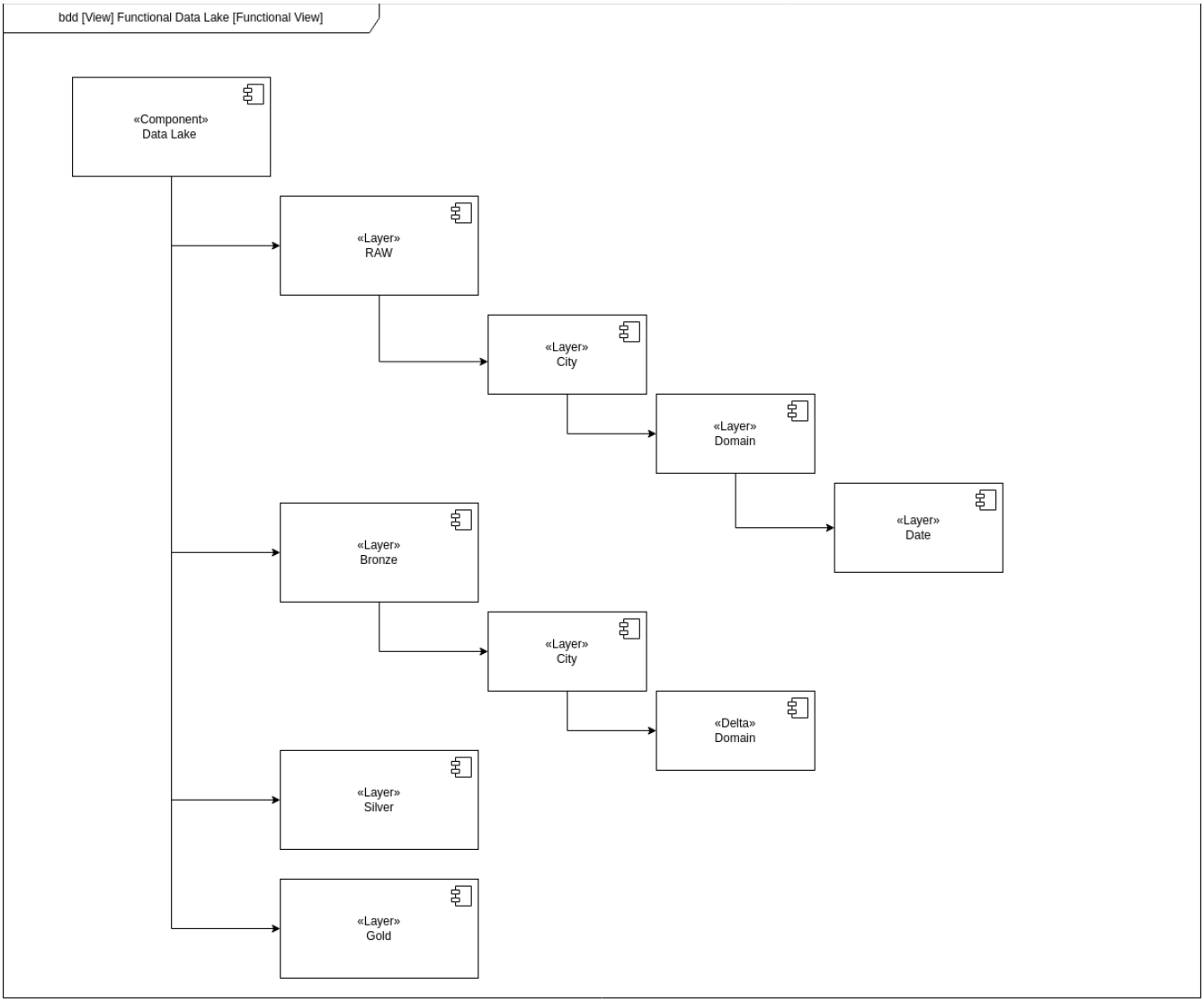

Figure 3 Functional view of the Data Lake

6.4. Gold layer

The gold layer represents the most refined level of Data Lake, where data is fully prepared, enriched and optimised for high-level analysis. The data in this layer is used for insight discovery, advanced reporting and to feed business intelligence applications. This layer ensures that data is ready to be consumed by end users and decision applications.

6.5. Data Movement process

The data movement process refers to the orchestration and automation of the flow of data through the different layers of the Data Lake. This is where the business rules and logic necessary to move data efficiently and securely are established, ensuring that it is in the right place, in the right format and at the right time. This includes scheduling ETL (extract, transform and load) jobs, monitoring data flows and implementing data governance policies.

7. Data Ingestion Layer

An API, short for "Application Programming Interface," is a set of rules and definitions that facilitates communication between different software systems or components. Through predefined methods and data structures, an API allows developers to interact with the system exposing the API, whether to retrieve data, send commands, or perform specific operations. Among APIs, REST APIs are a specific type designed following the principles of REST architecture (Representational State Transfer). A REST API uses standard internet protocols, such as HTTP, and typical methods like GET, POST, PUT, and DELETE to manage interactions with the web service. This makes REST APIs highly efficient, scalable, and easy to integrate into web applications.

7.1. Ingestion architecture

The Inbound API is a single endpoint for data reception. Its resources are organised according to specific data domains (Figure 4).

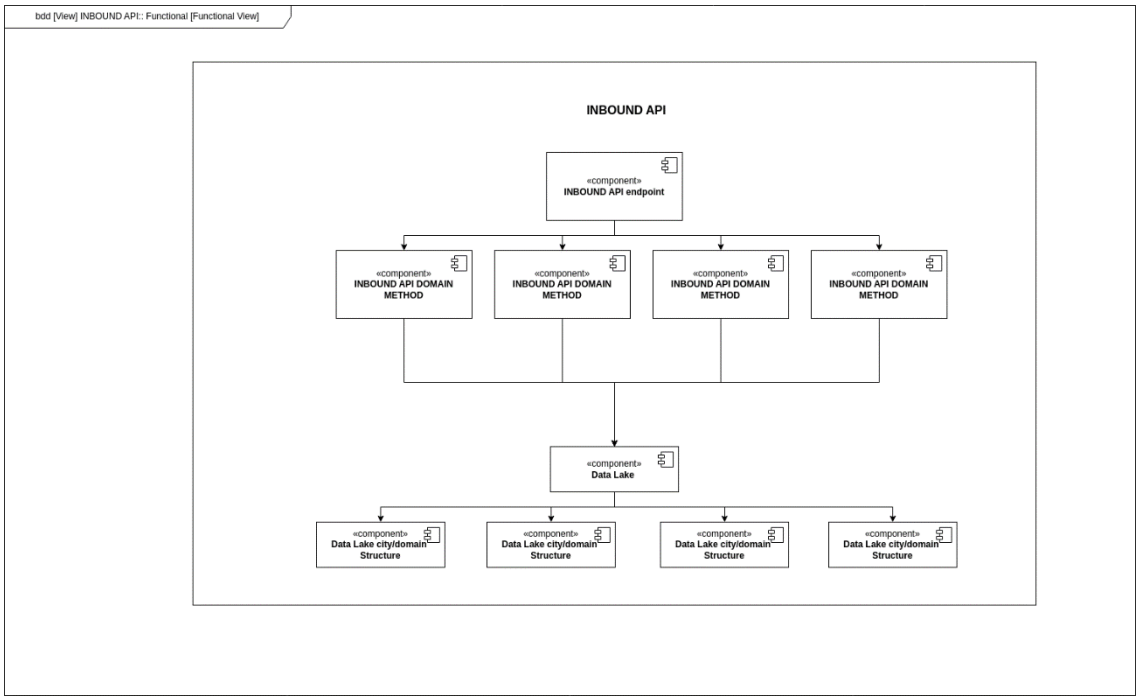

Figure 4 Ingestion Architecture.

7.2. APIs by data domains

The goal of the data ingestion layer is to act as the main conduit through which data flows into the Data Lake ecosystem, ensuring the efficient and secure capture of data from multiple sources, preparing them for subsequent processing and analysis.

Focused on a REST API, the proposed architecture offers a single endpoint for data reception, organizing resources according to specific data domains. Scalability, security and high availability are the cornerstones of this architecture, supported by Amazon Web Services (AWS), such as API Gateway and AWS Lambda.

The authentication and format of the data are validated with each API request, ensuring that only appropriate data is admitted. Once validated, the data are sent to the "raw" layer of the Data Lake, which maintains the data in its original form, ensuring integrity and traceability in accordance with relevant regulations.

The inbound API is structured into different domains, with specific validations to ensure that both the header and the necessary fields are correct.

When data is ingested into the raw layer, it is structured as follows: raw/{city}/{domain}/{year}/{month}/{day}/{filename}.json. This directory structure aids in the organization and retrieval of data by specific city, domain, and date. For example: raw/LasRozas/AirQuality/2024/04/22/2024-04-22-13-58-56-677216.json

Subsequently, the data ingestion triggers an event that launches a Lambda function. This function is tasked with transferring the data to the bronze layer, where it is stored in delta tables incrementally, adhering to the directory format bronze/{city}/{domain}. This process ensures a smooth and efficient data integration, enabling optimal updates and querying of the data.

The inbound API is secured with a security token that must be sent with each API call; this token is generated by Tree Technology (TREE). Upon receiving a request, the API Gateway validates the access token against a list of authorized tokens. This includes verifying the token's integrity and ensuring it has not expired. All access to the API is logged. This includes the access time and the API resource that was requested. These records are essential for auditing API usage.

7.2.1. Domains

The API is organized into several domains, each representing a specific data area reflecting different aspects of urban life. Each domain corresponds to a unique resource within the API, allowing for the ingestion and management of specific data from each area. Below are the available domains:

- **Air Quality.** This domain handles data related to air quality in the city. The required fields for this domain are: *["date", "name", "geolocation", "pm2.5", "pm10", "no", "no2", "nox", "so2", "trs", "o3", "co2", "temperature", "humidity"]*. For an example of submission, see Annex A.
- **Noise.** The noise domain collects data on ambient noise levels. The required fields for this domain are: *["date", "name", "geolocation", "LAeq", "LA10", "LA90", "LAmax"]* For an example of submission, see Annex A.
- **Traffic**. Traffic data are collected under this domain. The required fields for this domain are: *["state_class", "timestamp", "timestamp_utc", "street_class", "geom", "crs"]* For an example of submission, see Annex A.
- **Mob Flow.** This domain focuses on analyzing movement patterns of people within the city. The required fields for this domain are: *["Id", "Appears_at", "Last_at", "Enter", "Leave", "Typ", "Stunde", "Tag", "Monat", "Wochentag", "Typname", "Polygonpfad"]*. For an example of submission, see Annex A.
- **Mob Interactions** The mob interactions domain studies how people interact with each other and with city infrastructures in their daily mobility. The required fields for this domain are: *["ID_1", "ID_2", "min_actual_distance", "larger_speed", "distance_remaining", "duration", "score_1", "score_2", "time_to_distance_remaining", "frame", "video_second", "video_minute", "speed_1", "speed_2", "x_position_1", "y_position_1", "x_position_2", "y_position_2", "acc_diff_1", "acc_diff_2", "max_acc_1", "max_dec_1", "max_acc_2", "max_dec_2", "x_position_at_min_1", "y_position_at_min_1", "x_position_at_min_2", "y_position_at_min_2", "angle_at_min_distance_remaining", "First_Polygon_1", "Final_Polygon_1", "First_Polygon_2", "Final_Polygon_2", "Ex_Polygon_1", "Ex_Polygon_2", "max_accel_1", "min_accel_1", "max_accel_2", "min_accel_2", "max_accel_frame_1", "min_accel_frame_1", "max_accel_frame_2", "min_accel_frame_2", "confl_x_1", "confl_y_1", "confl_x_2", "confl_y_2", "old_min_dist", "interact_lateralc", "interact_accelc", "interact_decelc", "interact_lateral", "interact_accel", "interact_decel", "aggregate_lateralc", "aggregate_accelc", "aggregate_decelc", "aggregate_lateral", "aggregate_accel", "aggregate_decel", "Type_1", "Type_2", "video_0_date_time"]*. For an example of submission, see Annex A.

Each of these domains is designed to facilitate specific capture and structured management of related data, ensuring that information is accessible and usable for analysis and decision-making needs. The description of data fields is provided in Annex B.

For more details on this structure, our Swagger documentation can be consulted in the following website: http://amigos-swagger-bucket.s3-website-eu-west-1.amazonaws.com/#/ .

7.3. CSV Ingestion Tool

This tool is designed to simplify data incorporation for municipalities, consisting of a Python script and a configuration file named "config.ini." The purpose of this tool is to automate the transfer of information, taking a locally stored CSV file as the origin. The script processes this file and makes requests to the API in accordance with the directives set out in the configuration file. In this way, data is transmitted efficiently and integrated

directly into our platform. We have provided this tool to our partner cities to facilitate their data integration efforts.

8. Data Consumption APIs Layer

The Data Consumption layer constitutes a pivotal component within data processing architectures, particularly in contexts such as data warehousing and big data analytics. Contrary to the Data Ingestion layer, the Data Consumption layer is positioned at the endpoint of the data processing pipeline. It encompasses processes and technologies responsible for consuming processed or transformed data and delivering it to its intended destination.

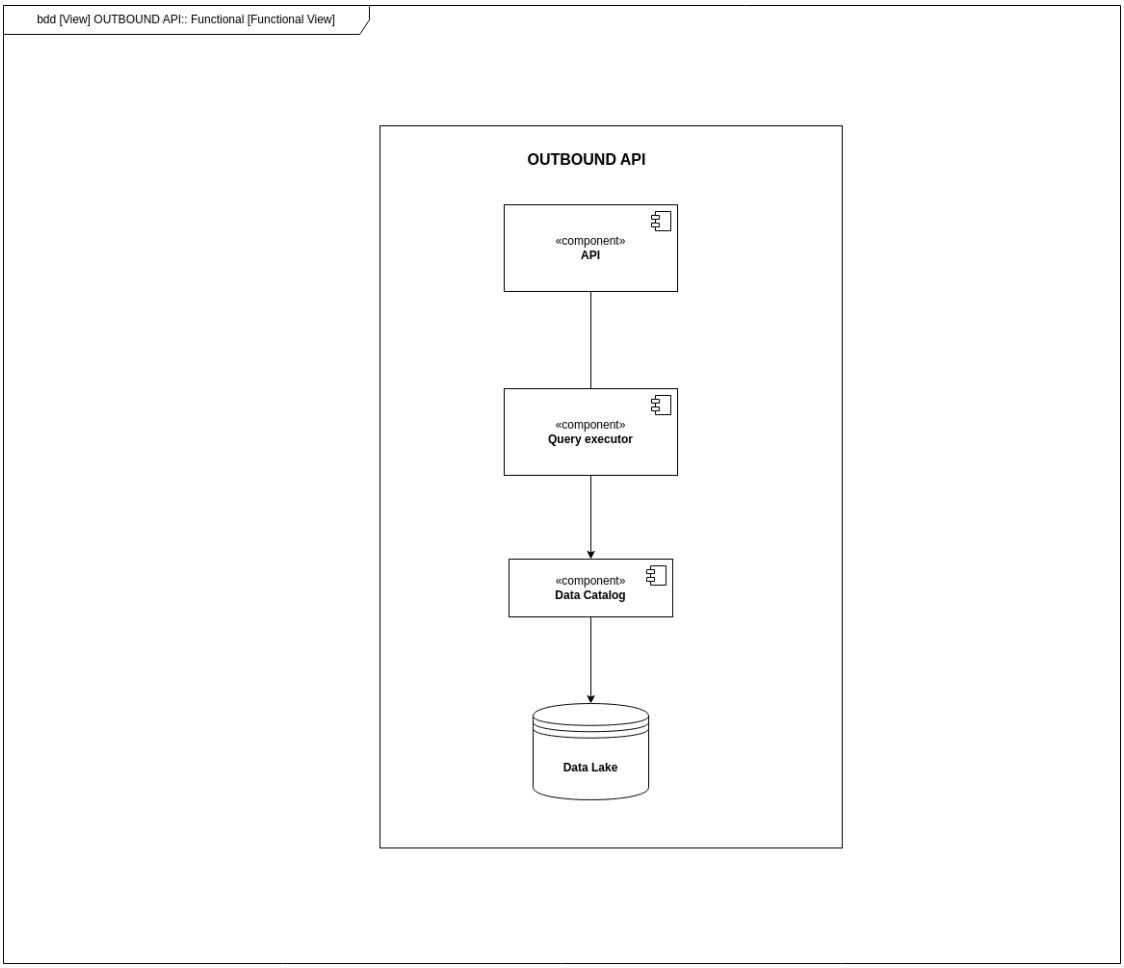

Figure 5 Outbound API structure.

8.1. Functionality and access

The architecture or functional view of the Data Consumption layer is shown in Figure 5. It is form by four different components:

- **The API component** serves as the interface through which processed or transformed data is delivered to external partners. It acts as a gateway for accessing data from the Data Lake, so it provides a single

endpoint through which partners can request specific data or retrieve data tailored to their needs. Additionally, it ensures security, authentication, and authorization mechanisms to control access to sensitive data and maintain data integrity. We use the AWS API Gateway service to implement this outbound API.

- **The Query Executor** is a component responsible for executing queries against the data stored in the Data Lake. It is the main component of the Outbound layer, since it allows not only to query the requested data based on the query parameters that partners can specify through the API, but also to construct the necessary query to access the data. In addition, the executor optimizes query execution to ensure efficient utilization of resources and timely delivery of results. It leverages distributed computing techniques such as Apache Spark to parallelize query execution and improve performance, especially when dealing with large datasets. This component is supported by AWS ElasticMap Reduce (EMR). EMR is a web service that facilitates processing of large amounts of data in a quick and cost-efficient manner. Amazon EMR uses Hadoop, an open-source framework, to distribute data and processing across resizable clusters. Amazon EMR is used in many applications, including log analysis, web indexing, data warehousing, machine learning, and bioinformatics. is a web service that facilitates processing of large amounts of data in a quick and cost-efficient manner. Amazon EMR uses Hadoop, an open-source framework, to distribute data and processing across resizable clusters. Amazon EMR is used in many applications, including log analysis, web indexing, data warehousing, machine learning, and bioinformatics.
- **The Data Catalog** serves as a centralized repository or metadata store that provides comprehensive information about the data available in the Data Lake. It acts as a catalog or index, organizing and describing the structure, format, lineage, and other metadata attributes of the data assets stored. It also facilitates data governance, data lineage tracking, and data quality management by maintaining a record of data assets, their ownership, and lineage information. This data catalog service is provided by AWS Glue.
- Finally, **the Data Lake** component, which has been already described, is the warehouse system designed to store the data ingested by the Data Ingestion layer.

This integrated framework provides partners with a seamless and standardized interface for accessing and retrieving data. They only need to specify the query parameters or data requirements through the API, which orchestrates the execution of the query via the executor and retrieves the requested data stored in the data lake.

In the context of querying data, parameters refer to the criteria or conditions specified by users to filter, retrieve, or manipulate specific subsets of data. Parameters typically correspond to the column names and values within the dataset and are used to customize the query results based on specific requirements. This enables partners to precisely retrieve the data they require, facilitating efficient data consumption and analysis.

9. Analytics and ML Layer

The three-layered structure from Bronze to Gold layers provides a simple and understandable data model. Thus, it empowers easy use of business intelligence tools on top of the gold layer to obtain actionable insights and guide decision-making. It also facilitates implementation of Machine Learning (ML) algorithms capable of solving complex and non-linear problems.

ML is a field of artificial intelligence that focuses on the study, development and use of computer algorithms that can learn from data and make predictions or decisions without using explicit instructions. Huge expansions of ML techniques in last years, led to its use in everyday life. Among many fields, ML was applied to natural language processing, computer vision, speech recognition, self-driving car, urban mobility, and medicine.

Analytics and ML layer plays a crucial role in the context of big data, where the abrupt volume of data makes it difficult for humans to understand and work with it. Analytics and ML algorithms can help overcome these challenges by obtaining data from data lake and automatically detecting patterns, which allows obtaining valuable predictions and insights.

In AMIGOS, Analytics and ML layer will be used in the tasks defined in WP3. For example, ML algorithms will be implemented to optimize public transport based on the supply and demand, as well as decrease traffic congestions leading to air and noise pollution improvements, as described in Subtasks 3.1.2 and 3.3.1., respectively. In the following subsection tools, technologies and processes are explained.

9.1. Tools and Technologies

There are several tools and technologies that are commonly used for the analytics and ML layer. Amazon EMR is one of the solutions used. It provides built-in machine learning tools that leverage the Hadoop framework to create a variety of scalable ML algorithms like TensorFlow, Apache Spark MLlib, and Apache MXNet. EMR makes it easy to develop, visualize, and debug machine learning applications.

Another option is Amazon SageMaker, which is a fully managed cloud-based ML service provided by Amazon Web Services (AWS). It aims to simplify the process of building, training, and deploying ML models by handling much of the underlying infrastructure. SageMaker supports a wide range of ML frameworks and algorithms, including popular frameworks such as TensorFlow or PyTorch. Besides, it allows implementing ML models, but also offers an option for users with no coding experience via visual interface.

Morever, Kubeflow also can be used. It is a cloud-native platform designed for ML operations, encompassing pipelines, training, and deployment of ML models. Kubeflow simplifies the process of deploying machine learning workflows on Kubernetes by providing a platform for building, deploying, and managing multi-step ML workflows based on Docker containers.

The development of the analytics and ML models is done commonly in Python, Pyspark and jupyter notebook, independently of the underlying technology. It allows use of the ML libraries such as Scikit-learn, Apache Spark MLlib, XGBoost, and TensorFlow. Using the R language with RStudio is also possible, however this option is much less frequent.

9.2. ML Processes, Model Training and Deployment

The ML workflow includes several steps to follow when it comes to generating a model:

1. **Data preparation**. Once the data is obtained, it needs to be carefully pre-processed before training the model. This step includes exploratory data analysis, missing values handling, feature generation and extraction, outlier removal, data normalization, and data partitioning.
2. **Model training and evaluation**. In this step, the model generation technique with its initial parameters is selected, the model is trained and evaluated based on defined performance metrics and validation.
3. **Model optimisation and evaluation**. This step is usually executed in an iterative way. The model parameters are optimized using hyperparameter tunning method to obtain the best performance. In some cases, a regularization technique is applied to avoid overfitting of the model.

4. **Model deployment**. The last step is to integrate the final model into the deployment environment. Scalability, costs, security, and integration capabilities are factors to be considered.

10. Security and Compliance

10.1. Security measures

Encryption at Rest

The Data Lake leverages the encryption at rest provided by S3 by default to ensure the protection and confidentiality of stored data. This automatic encryption, managed by the service, uses advanced algorithms to encrypt data before it is saved to disk, with the encryption keys managed and protected by S3's own infrastructure. This ensures that without the appropriate credentials, data is inaccessible, even if unauthorised physical access to the storage media occurs.

API Security

The security of application programming interfaces (APIs) is essential to protect against unauthorised access. Unique API keys per user are employed to ensure that only clients with the appropriate credentials can access or modify the resources exposed by the APIs. This level of authentication helps to track and limit usage, providing an additional layer of security.

10.2. Data compliance

The platform stores information for auditing and enables control of data access and manipulation activities within the Data Lake, facilitating transparency and supporting ongoing compliance.

The Big Data platform features granular permission segmentation based on access roles to ensure that data relevant to a particular city is only accessible to authorised users in that city. This not only strengthens data security and privacy, but also allows each city to manage its own data effectively, without exposing sensitive information to unauthorised external entities.

11. Overview of the Real Time AI based Platform

11.1. The holistic topology of the IPG Solution

The Real-Time AI based platform and related solutions provided by IPG works alongside TREE's Big Data platform and both complement each other, sharing data and insights to be used for the services development as well as the Digital Twins Component.

One example of the use of the RT AI based Platform is the real time location of buses and how they are deviated from the planned schedule. This processed data will increase the usage of public and private shared vehicles, by enabling a smooth ride from A to B. Commuters are reluctant to use mass transport if schedule plan is not met.

To be able to address the cities' challenges as were detailed in the first months of AMIGOS project, a special solution topology and specific data, in terms of content and availability is required. The holistic topology is depicted by the Figure 6.

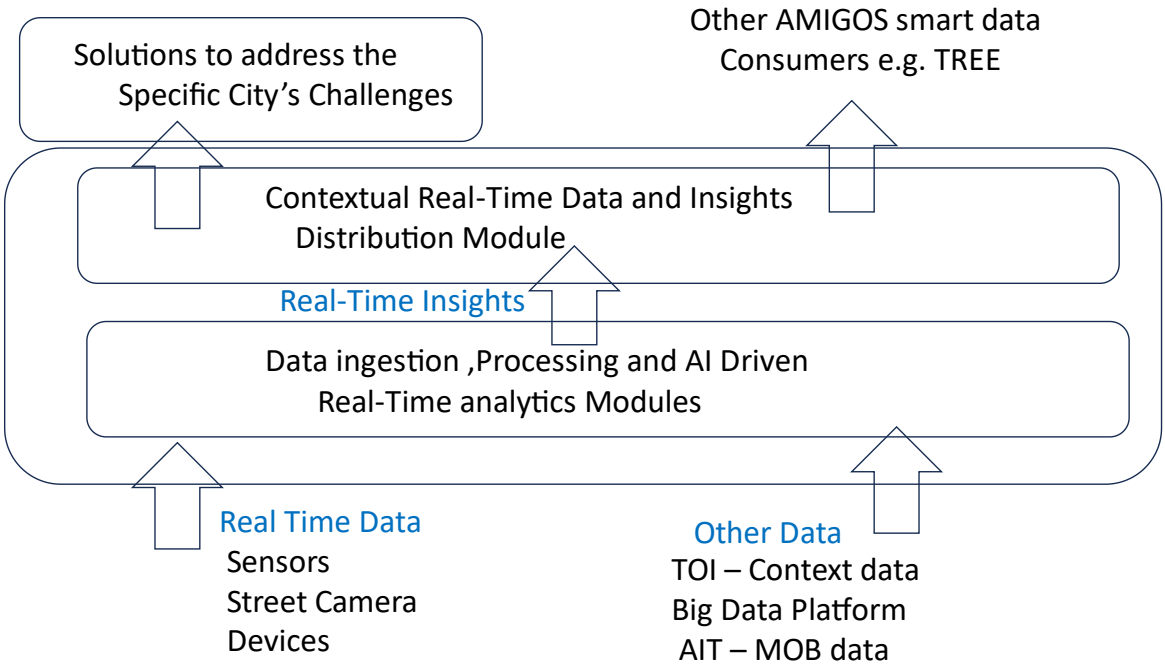

Figure 6 Holistic topology of the RT AI Platform.

The system's topology encompasses everything from collecting relevant data from various sources to providing services that address the specific challenges of each AMIGOS city.

The data to be processed is real-time and complements the Big Data. Examples of such type of data are data coming from sensors when a threshold has been met, traffic data originated by street cameras, actual location of buses, availability of parking spots. It is the actual situation data, and it is important as standalone event, such as data from air-quality sensors. Additionally, it involves a stream of data that needs to be analysed accordingly, unlocking insights from the stream.

The AI and Real Time data insights, complement the insights provided by the Big Data via its platform. Besides, the solution provides with data in different modes to other AMIGOS partners via outbound dedicated API (POST) to be used by them to fulfil their tasks (develop a service).

11.2. Handling The Data

Data is the "fuel" for the services to be developed. Having the right data is crucial. Data that does not accurately reflect the current situation in real time, with the lowest possible latency, will most likely lose its relevance and value. That's why it is crucial to not only locate the right data, but also to have real-time access with low latency. The data aspect of the project was managed with the context and the services necessary to address the identified challenges.

This task was executed in parallel with the development of the Data Collection Module. It was carried out by data scientists using a "bottom down" approach, starting with the definition of the needed data and continuing through to locating and preparing it for further processing. The data handling flow is depicted in Figure 7.

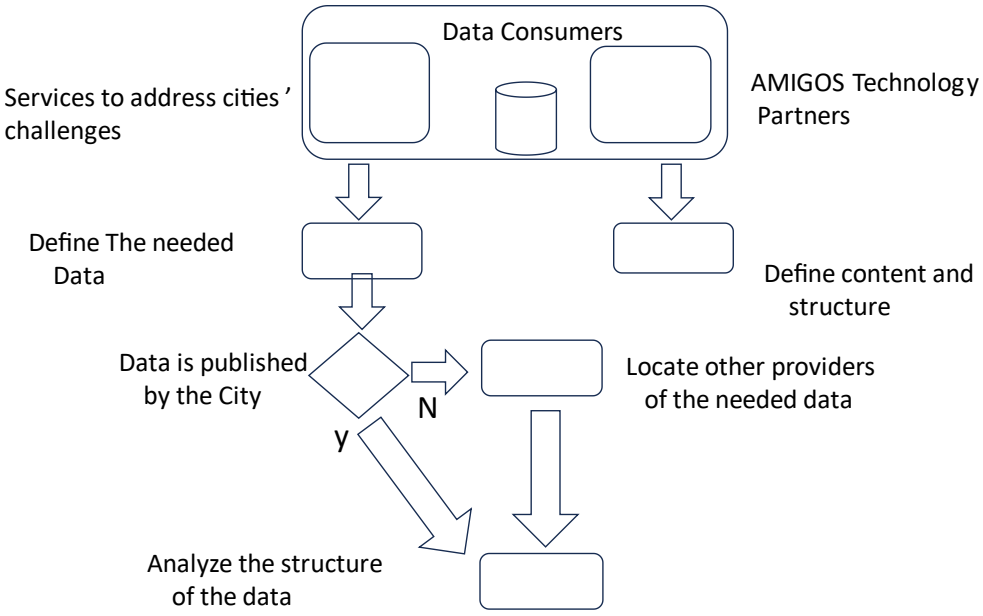

Figure 7 Data handling: bottom down approach.

Data handling is an ongoing process that involves managing data by city, challenge, location, boundaries (SIA, LL) and appropriate data sources. This process supports the implementation of services to be developed by IPG and the integration with the Big Data Platform, potentially enhancing Digital Twins by introducing real time data.

As illustrated in Figure 7, the process starts with identifying the needed data and locating it. If the city has published the data, we proceed to analyse and prepare it for further processing. Otherwise, we enter a new phase to find alternative data sources. At this stage, we have handled the collection of air quality data for the city of Hamburg using the Data Collection Module, which is described in the hereafter section (Section 11.3).

11.3. Data Collection Module

The Data Collection Module has been developed as a versatile tool capable of supporting the collection of any data from any source and through any interface. Its main advantage is its ability to access various data sources from a single point via REST_API, a protocol supported by all data providers. The logic of the module is illustrated in Figure 8 below.

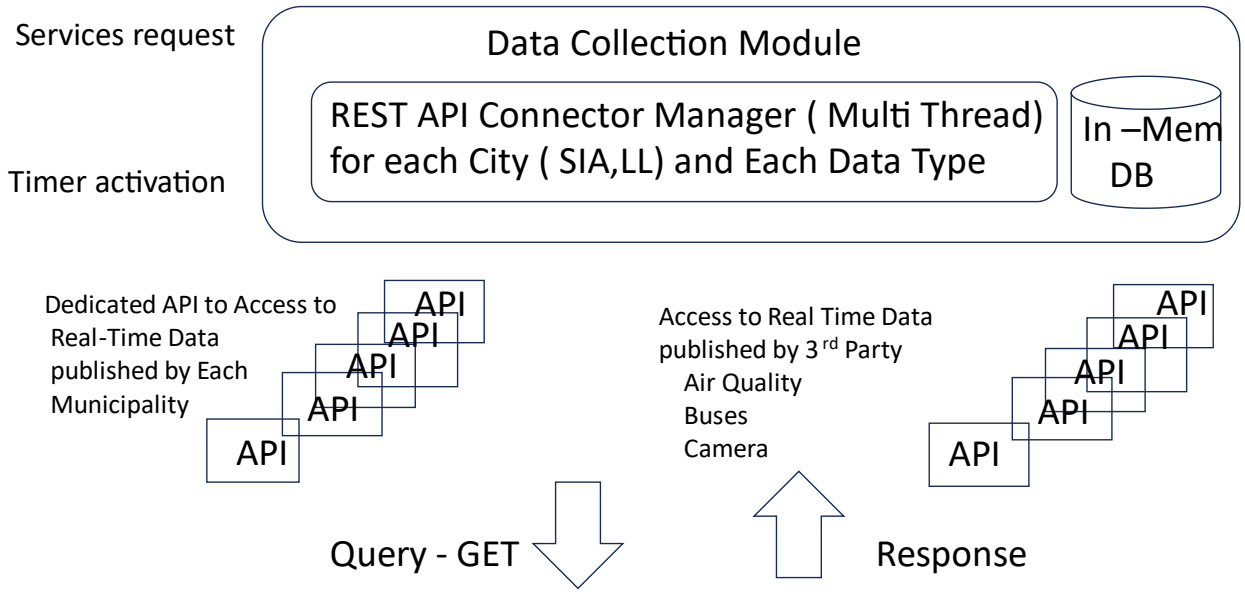

Figure 8 Real-time data collection.

As depicted in Figure 8, the Data Collection Module is activated either by a higher-level module in the topology or by timer for periodic sampling of specific data, such as air quality values in particular locations. This mechanism creates a data stream that enables real-time AI algorithms to make predictions at a later stage.

Once activated, the module sends an HTTP GET request to the appropriate API, which acts as a gateway to the data, and collects the data to be used by upper layers or published to external consumers. Real-time data is used by the upper layers, while accumulated data is pushed to the Big Data Platform using an HTTP POST.

The module selects an API connector, sends a GET request to the correct target, defines the geographical boundaries, and ensures the collected data has the appropriate location context, targeting SIA and LL. It is designed to handle various cities' proprietary API structures and different domains such as air quality, mobility, and public safety.

11.4. Data Publication and Distribution Module

After data is collected through a specific request or upon timer expiration, it undergoes a process of normalization before being published and distributed, as depicted in Figure 9.

The Data Collection Module feeds data to this module, which uses normalization algorithms to prepare the data so that consumers receive it according to their requirements.

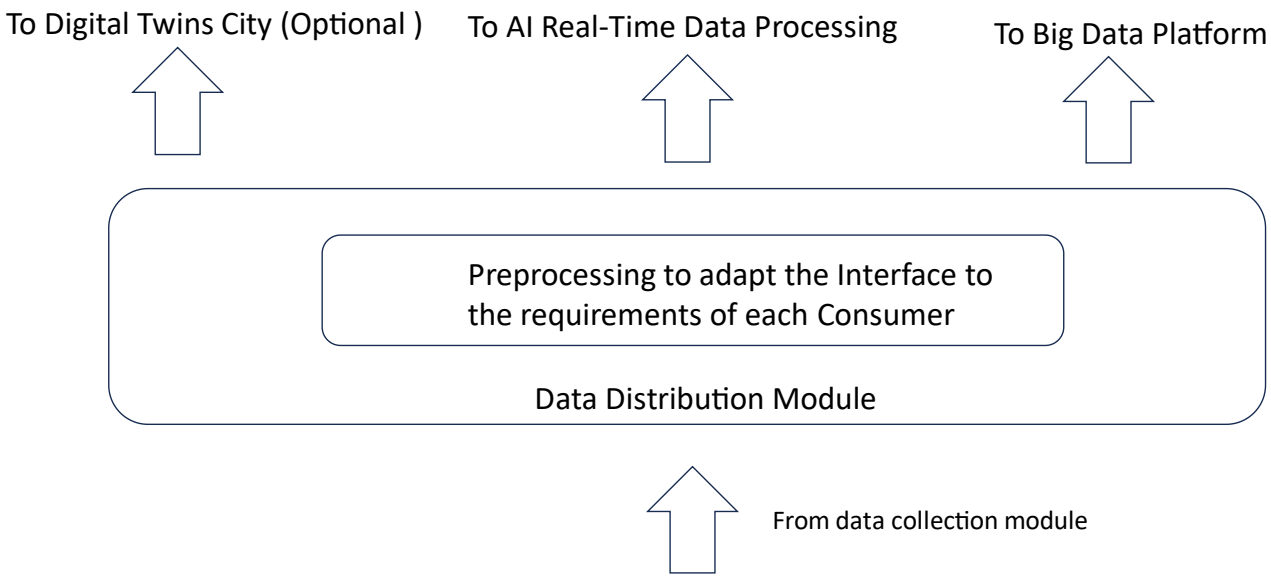

Figure 9 Data publication and distribution.

11.5. Technology – General

The solution was developed using the most advanced software design and implementation methodologies such as microservices and Docker. As its name implies, it deals with time- sensitive data and services. This requires writing the code efficiently and using in-memory database. We used RedisDB for this purpose, which enables us to run the system both as on-premises and in the cloud. The solution is cloud vendor agnostic, but since the Big Data Platform runs on AWS, we might consider using this cloud vendor as well.

12. Conclusion

This deliverable provides an overview of the work carried out in T1.5 Big Data Platform, including the RT AI-based platform. An explanation of the concept of Big Data Platform is included as well as the architecture and explanation of the different components developed. Besides, the difference between the Big Data Platform and the RT AI-based platform is explained.

The Big Data Platform was already tested with synthetic data, data sent from IPG related to Hamburg city and data from MOB, but from June 2024 cities will start feeding the Platform with their real data, which will be processed withing WP3 “Implementation and testing of solutions in LLs and SIAs”.

The RT AI based Platform was tested with the data from Hamburg city (provided by the city and from other sources available), starting from air quality and data gathered by thermal street cameras. The Data Module to collect data was implement for this purpose. The design was made considering the need of taking data from any source, supporting any interface (in particular, supporting cities that do not have IT resources to adapt the data to the API structured by TREE for the Big Data Platform). The functionality to “push” data to the TREE’s Big Data Platform was implemented, based on the technical requirements agreed.

In the coming months, it is planned to expand the support from the Hamburg city to other cities, so it complies with the fast response time required.

The Platforms will evolve, together with the development of the project, as new requirements, features and needs will be defined.

13. References

Cesario, Eugenio. "Big data analytics and smart cities: applications, challenges, and opportunities." *Frontiers in big data* 6 (2023): 1149402.

Apache MXNet: https://mxnet.apache.org/

Apache Spark MLlib: https://spark.apache.org/mllib/

Apache Spark: https://spark.apache.org/

API Gateway: https://aws.amazon.com/api-gateway/

Delta Lake: https://delta.io/

EMR: https://aws.amazon.com/emr/

Glue: https://aws.amazon.com/glue/

Jupyter notebook: https://jupyter.org/

Kubeflow: https://www.kubeflow.org/

Lambda: https://aws.amazon.com/pm/lambda/

Pyspark: https://spark.apache.org/docs/latest/api/python/

Pytorch: https://pytorch.org/

S3: https://aws.amazon.com/s3/

SageMaker: https://aws.amazon.com/sagemaker/

Tensorflow: https://www.tensorflow.org/

XGBoost: https://xgboost.readthedocs.io/

14. Annexes

14.1. Annex A

Air Quality:


```
{
  "date": "2024-04-12T17:00:00Z",
  "name": "Estación Medioambiental Centro",
  "geolocation": "19.432608,-99.133209",
  "pm2.5": 12.3,
  "pm10": 55.2,
  "no": 10.0,
  "no2": 20.0,
  "nox": 30.0,
  "so2": 60.5,
  "trs": 0.1,
  "o3": 25.0,
  "co2": 400.5,
  "temperature": 22.5,
  "humidity": 50000
}
```


Noise:


```
{
    "date": "2024-04-10T15:45:00Z",
    "name": "Zona Urbana Río",
    "geolocation": "32.5332",
    "LAeq": 68.4,
    "LA10": 72.1,
    "LA90": 65.0,
    "LAmax": 80.5
}
```


Traffic:


```
{
    "state_class": "inactive",
    "timestamp": "2024-04-12T17:00:00Z",
    "timestamp_utc": "2024-04-12T17:00:00Z",
    "street_class": "commercial",
    "geom": "POINT(34.0522 -118.2437)",
    "crs": "EP45:4326"
}
```

Mob Flow:


```
{
  "Id": 12345,
  "Appears_at": 1593574800,
  "Last_at": 1593578400,
  "Enter": "2024-04-10T09:00:00Z",
  "Leave": "2024-04-10T17:00:00Z",
  "Typ": 1,
  "Stunde": 9,
  "Tag": 10,
  "Monat": 4,
  "Wochentag": 3,
  "Typname": "Example",
  "Polygonpfad": 67890
}
```


Mob interactions:

```
{
    "ID_1": 101,
    "ID_2": 202,
    "min_actual_distance": 15.0,
    "larger_speed": 120.5,
    "distance_remaining": 250.75,
    "duration": 3600,
    "score_1": 85.6,
    "score_2": 90.2,
    "time_to_distance_remaining": 300,
    "frame": 1500,
    "video_second": 30,
    "video_minute": 5,
    "speed_1": 88.1,
    "speed_2": 92.3,
    "x_position_1": 50.2,
    "y_position_1": 75.4,
    "x_position_2": 52.2,
    "y_position_2": 78.4,
    "acc_diff_1": 0.1,
    "acc_diff_2": -0.2,
    "max_acc_1": 1.5,
    "max_dec_1": 1.2,
    "max_acc_2": 1.4,
    "max_dec_2": 1.3,
    "x_position_at_min_1": 45.7,
    "y_position_at_min_1": 73.2,
    "x_position_at_min_2": 47.1,
    "y_position_at_min_2": 74.5,
    "angle_at_min_distance_remaining": 25.5,
    "First_Polygon_1": 1024,
    "Final_Polygon_1": 2048,
    "First_Polygon_2": 3072,
    "Final_Polygon_2": 4096,
    "Ex_Polygon_1": 512,
    "Ex_Polygon_2": 256,
    "max_accel_1": 1.8,
    "min_accel_1": 0.5,
    "max_accel_2": 2.0,
    "min_accel_2": 0.6,
    "max_accel_frame_1": 150,
    "min_accel_frame_1": 75,
    "max_accel_frame_2": 145,
    "min_accel_frame_2": 65,
    "confl_x_1": 102.4,
    "confl_y_1": 204.8,
    "confl_x_2": 307.2,
    "confl_y_2": 409.6,
    "old_min_dist": 30.2,
    "interact_lateralc": 0.03,
    "interact_accelc": 0.04,
    "interact_decelc": 0.05,
    "interact_lateral": 0.06,
    "interact_accel": 0.07,
    "interact_decel": 0.08,
    "aggregate_lateralc": 0.09,
    "aggregate_accelc": 0.10,
    "aggregate_decelc": 0.11,
    "aggregate_lateral": 0.12,
    "aggregate_accel": 0.13,
    "aggregate_decel": 0.14,
    "Type_1": 1,
    "Type_2": 2,
    "video_0_date_time": "20240410_093000"
}
```

14.2. Annex B

Air Quality:

date	Date of the data collection
name	Name of the place of measurement
geolocation	Geolocation in decimal degrees format
pm2.5	The measure of particulate matter with size of 2.5 microns or less
pm10	The measure of particulate matter with size of 10 microns or less
no	The measure of nitrogen oxide
no2	The measure of nitrogen dioxide
nox	The measure of nitrogen oxides
so2	The measure of sulfur dioxide
trs	The measure of total reduced sulfur
o3	The measure of ozone
co2	The measure of carbon dioxide
temperature	The measure of temperature in °C
humidity	The measure of humidity in %

Noise:

Column name	Description
date	Date of the data collection
name	Name of the place of measurement
geolocation	Geolocation in decimal degrees format (from 1 to 4)
LAeq	The measure of equivalent continuous sound level
LA10	The measure of noise level exceeded for 10% of the measurement period
LA90	The measure of noise level exceeded for 90% of the measurement period
LAmax	The measure of maximum sound level

Traffic:

state_class	The condition class of the traffic situation
timestamp	Date of the data collection
timestamp_utc	Date of the data collection in UTC
street_class	The class of the street (unknown, main, highway, service)
geom	The coordinates of the street defined by points
crs	The code of coordinate reference system

Mob flow:

Id	Id of the polygon
Appears_at	
Last_at	
Enter	Timestamp of entering the polygon
Leave	Timestamp of leaving the polygon
Typ	Type of polygon
Stunde	Hour
Tag	Day
Monat	Month
Wohentag	Name of the week
Typname	Type of the vehicle
Polygonpfad	

Mob interactions:

Column name	Description
ID_1	
ID_2	
min_actual_distance	
larger_speed	
distance_remaining	
duration	
score_1	
score_2	
time_to_distance_remaining	
frame	
video_second	
video_minute	
speed_1	
speed_2	
x_position_1	
y_position_1	
x_position_2	
y_position_2	
acc_diff_1	
acc_diff_2	
max_acc_1	
max_dec_1	
max_acc_2	
max_dec_2	
x_position_at_min_1	

y_position_at_min_1	
x_position_at_min_2	
y_position_at_min_2	
angle_at_min_distance_remaining	
First_Polygon_1	
Final_Polygon_1	
First_Polygon_2	
Final_Polygon_2	
Ex_Polygon_1	
Ex_Polygon_2	
max_accel_1	
min_accel_1	
max_accel_2	
min_accel_2	
max_accel_frame_1	
min_accel_frame_1	
max_accel_frame_2	
min_accel_frame_2	
confl_x_1	
confl_y_1	
confl_x_2	
confl_y_2	
old_min_dist	
interact_lateralc	
interact_accelc	
interact_decelc	
interact_lateral	
interact_accel	
interact_decel	
aggregate_lateralc	
aggregate_accelc	
aggregate_decelc	
aggregate_lateral	
aggregate_accel	
aggregate_decel	
Type_1	
Type_2	
video_0_date_time	