D2.3

Decision Support System & APIs v1

Deliverable number: (D2.3)
Author(s): Dimitra Politaki, Antonis Mygiakis, Ioanna Fegardiotou
Author’(s’) affiliation (Partner short name): INLE

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 861598. LEAD is a project under the CIVITAS Initiative. Read more - civitas.eu

Deliverable No.	**2.3**		
Work Package No.	**2**	Work Package Title	Digital Twin Model and Simulation Environment
Task No.	**2.3**	Task Title	Decision Support System & APIs (v1)
Date of preparation of this version:	**26/03/2021**		
Authors:	**Dimitra Politaki (INLE)** **Ibad Kureshi (INLE)** **Ioanna Fergadiotou (INLE)** **Antonis Mygiakis (INLE)** **Abdelhadi Belfadel (IRTX)** **Sebastian Hörl (IRTX)** **Rodrigo Tapia (TUDELF)** **Ioanna Kourounioti (TUDELF)** **Angel Batalla (LMT)** **Bråthen Svein (MOLDE)** **Nagy Vivien (BKK)**		
Status (F: final; D: draft; RD: revised draft):	**F**		
File Name:	**LEAD-Decision_Support_System_&_APIs_D2.3_v21-INLE_final.doc**		
Version:	**1.2**		
Task start date and duration	**01/02/2021 – 31/08/2022**		

This document is issued within the frame and for the purpose of the LEAD project. This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 861598.

Revision History

Version No.	Date	Details
1.0	26/03/21	Initial Deliverable Structure and Chapter Summaries
1.1	22/06/21	First draft setup
1.2	28/06/21	4. Interface Design Principles and Frameworks
1.3	29/06/21	LL6 Oslo
1.4	30/06/21	3.2 Model Library
1.5	30/06/21	LL2 – The Hague
1.6	30/06/21	LL5 – Budapest
1.7	30/06/21	LL3 – Lyon
1.8	14/07/21	LL1 – Madrid
1.9	23/07/21	First version for initiating the review process
2.0	19/08/21	Restructuring, added more technical details
2.1	24/08/21	Update after reviews

Reviewers List

Name	Company	Date	Signature
Abdelhadi Belfadel	ISX	02/08/2021	
Horváth Adrián	SZE	09/08/2021	

Executive Summary

This document describes the LEAD Decision Support System (DSS) and Application Programming Interfaces (APIs), which is the technological core of the LEAD project, reflecting the system's design decisions by month fifteen (M15) of the project. The scope of this document is to detail the design specification of the DSS and the LEAD dashboards and interfaces towards producing a scalable, robust, flexible, and reusable platform for urban logistics operations.

By following a design procedure that engaged urban logistics experts and the Living Lab stakeholders, discussions and virtual meetings and taking into account related deliverables and the requirements originating from the nature of the problems LEAD aims to address, the Decision Support System architecture has been designed and will be developed/implemented as part of the LEAD platform that will be available as a complete framework to facilitate Living Labs execution while enabling modularity, scalability and high performance.

LEAD develops a scalable and robust digital twin platform for urban logistics that enables collecting, integrating, and processing streams of data, including contextual data. The platform is based on an open architecture ensuring both scalability and interoperability as well as on open software standards, adhering to a privacy-by-design approach to ensure that privacy and ethical issues are respected. The platform includes ready-to-go integration connectors for the seamless ingestion of multiple, voluminous, and heterogeneous data-in-motion and data-at-rest sources. Moreover, the platform with the efficient management of diverse data sources and the provision of the connectors (with a continuous system availability) will be able to support analytics and efficient complex data driven simulations that can reproduce operational conditions.

With reference to deliverable D2.1 "Technical Requirements – Solution Architecture", this report focuses on the LEAD platform Decision Support System, the platform's application programming interfaces with internal and external components and the platform's dashboards. The design of the LEAD DSS is primarily driven by the Living Labs business and interoperability requirements, considering the data diversity, volume and availability of large and complex data collections as presented in Section 3, and the Living Labs use case scenarios as described in WP3 (D3.1).

To provide a high-quality communication framework for LEAD project, the connectors and the platform should be developed for and be deployed in a distributed environment. As such, the platform's components can take advantage of the clustered deployment, scalability, integrate performance tests and metrics and allow for the identification of issues and related corrective actions in a timely manner, well before the Living Labs utilise the platform. Hence, this document presents a detailed technical specification analysis for the Decision Support System, the LEAD APIs, and the dashboards of the platform.

1 Contents

List of Figures

List of Tables

Acronyms

Acronyms	Explanation
ABM	Agent Based Model
API	Application Programming Interface
BEV	Battery Electric Vehicles
CNG	Compressed Natural Gas
CSV	Comma Separated Values
DDDAS	Data-Driven Dynamic Application System
DSS	Decision Support System
DT	Digital Twin
EV	Electric Vehicles
FLP	Future Location Prediction
GA	Grant Agreement
GDPR	General Data Protection Regulation
GTFS	General Transit Feed Specification
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LL	Living Lab
MASS-GT	Multi-Agent Simulation System for Goods Transport
MATSim	Multi-Agent Transport Simulation
ML	Model Library
NDA	Non-disclosure agreement
PI	Personal Information
TP	Trajectory Prediction
SLA	Service-Level Agreement

UCC	Urban Consolidation Centre
UI	User Interface
W3C	World Wide Web Consortium
WCAG	Web Content Accessibility Guidelines

1. Introduction

LEAD focuses on the operational level of last mile logistics, where stakeholders are trying to tackle optimisation problems in increasingly complex supply chains, while also accounting for environmental performance. For designers of such supply chains, a digital twin can help simulate the impact of changes prior to implementation and better align performance with business requirements. The benefits of implementing a digital twin include a common understanding, definition, and measurement of relevant KPIs. A digital twin makes it also possible to compare different perspectives - such as strategic goals, risk, operational or process key figures[1].

From a technical point of view, the LEAD Digital Twinning platform will support the execution of experiments in local/small scale urban logistics networks (Living Labs), employing multiple simulation models, resulting in proof-of-concept implementations of optimised operations captured in predefined KPIs. These data-driven experiments need to be designed holistically to optimize value across the full range of extended last mile processes. A digital twin of a Living Lab provides the potential to build a context-related model for business processes. The model visualizes how business processes, and in particular their changes, can affect value creation. It also enables the process simulation results to be aligned with the desired operational and strategic results therefore leading to the creation of suitable solutions for last mile value chains and the mapping of data to the process steps at all levels. A digital twin is not a one-time implementation; it requires an agile approach and follows a growth model. The initial design is inspired by the state-of-the-art of Digital Twins solutions and driven by the Living Labs specifications. The platform will be updated through iterations with the LLs stakeholders setting the foundations of a scalable future DT framework.

The objective of the LEAD Decision Support System and APIs task (T2.3) is to design and develop the technological core of the LEAD project. First, the technical requirements are collected, and an architecture is proposed. In the subsequent tasks, the Digital Twin model library is set up and the graphical user interface (dashboards) for interactive what-if analysis as well as the APIs connecting the different system components are developed.

1.1 Mapping LEAD Outputs

This deliverable is based on the commitments made on LEAD's Grant Agreement. In the following, a mapping between these commitments, as described in the Deliverable and the Tasks, and the project's respective outputs and work performed is presented. A list of the deliverable's components and task description together with the corresponding chapter of this report that discusses the approach that was followed and work that has been carried out can be seen on Table 1.

[1] *A Requirements Driven Digital Twin Framework: Specification and Opportunities, June 2020IEEE Access PP(99):1-1, DOI:10.1109/ACCESS.2020.3000437*

Table 1: Adherence to LEAD's Deliverable & Tasks Descriptions

LEAD GA Component	Document & Justification chapters
DELIVERABLE	
D2.3 Decision Support System and APIs *This task will involve three parallel activities that are tightly integrated to establish the top user-focused layer of the LEAD architecture.*	The deliverable covers the core of work required for D2.3. Chapter 2 covers the interfaces of the LEAD platform while chapter 3 focuses on the DSS. Chapter 4 presents the principles followed by the platforms' user interfaces and some initial designs. An outline of the future work to be carried out on the respective components (D2.4) until M27 is finally provided in chapter 5.
TASKS	
ST 2.3.1: LEAD API's *As LEAD's development incorporates open-source big data solutions for communication and data handling, this task will establish the protocols and connections required.* *(i) The LEAD platform to connect to external systems.* *(ii) Inter-module communication for each of the modelling and simulation components.* *(iii) The user interfaces*	**Chapter 2** This chapter covers aspects of API design and data exchange with the LEAD platform. Chapter 2.1 describes the data pipeline and how each LL has been integrated into the platform Inter-module communications are defined in detail in chapter 2.2. Finally, in chapter 2.3 the interfaces provided for the dashboards and the graphical components of the platform are discussed.
ST2.3.2: Cognitive Decision Support Systems *Using the principles of Bayesian Inference, this sub-task will create a cognitive engine that will take as input either outcomes of the prescriptive what-if scenarios or the various predictive models to make decisions.* *As there is considerable uncertainty when modelling the real world, there are never perfect outcomes. The cognitive engine of the DSS will chose the best of the sub-optimal outcomes based on the programmed parameters.*	**Chapter 3** This chapter outlines the functional and non-functional requirements of the DSS subsystem of the LEAD platform. The bulk of this work will be carried out in Period 2 of the project and reported in D2.4 (M27), Section 0 provides an outlook of Cognitive Decision Support System and outlines the work that will be carried out.
ST2.3.3: LEAD Interfaces and Dashboards	**Chapter 4**

From a user and operator perspective there are two interfaces – the visual design interface where the simulation scenarios can be configured and executed from, and the LEAD Dashboard. *The later includes configurable widgets to allow the end user to configure the interfaces as they want displaying real-time data and simulation data based on their preference and requirements.* *Once again open-source frameworks will be used to create the underlying interfaces and the API's will be used to ensure modularity.*	This chapter outlines the work carried out concerning the interfaces of the LEAD Digital Twin platform. Initially, basic design principles (4.1) and the user requirements as received by the project partners (4.2) are discussed. Then an initial implementation of the scenario creation user interface is presented along with a user journey (4.3). Finally, an outlook of the work to be completed is provided to the reader.

1.2 Deliverable Overview and Report Structure

This report presents to the reader an in-depth description of the Decision Support System of the LEAD platform as well as the interfaces the platform uses to communicate with the world, between the main components of the platform and with the platform's dashboards.

Chapter 2 Error! Reference source not found.concerns the interfaces that the LEAD platform offers regarding:

- the connections with the outside world; that could be external sources such as sensors, city monitoring systems, geospatial data, orders data and more
- the methods used for the communication among the platform's modules
- the integration of the user interfaces and dashboards

The Cognitive Decision Support System is discussed in **chapter 3**. Initially, the methodology used to define the technical specifications and the main actors are presented. Then, the functional and non-functional requirements are discussed as a main factor for the definition of the platform's architecture and main functionalities.

In **chapter 4** an initial approach on the graphical user interfaces and dashboards is suggested. It mainly concerns the scenario creation and the monitoring solutions available.

Finally, the conclusion (**chapter 5**) concerns a general overview of the work carried out as well as next steps and discussion points for the enhancement of the platform according to the business objectives and the partners' feedback.

2. LEAD API's

2.1. External Data sources

This section describes the data sources that have been identified in the LLs and will be ingested by the LEAD Platform. In this deliverable we only examine the nature of the existing data sources and external interfaces and their characteristics. D2.5 describes how these data sources are connected to the LEAD platform and how the data is ingested.

Data is ingested from external data sources in a real-time, periodical, or one-off manner into the LEAD platform. This data is used during the execution of the models. Most of these data sources will vary between LLs, both in terms of content and format thus making the interoperability of the models between LLs a great challenge.

In the following sections, an overview of the data sources is presented, as provided and currently used by the LLs. The data sources are described in terms of data structures and data volumetrics information. From the platform's perspective, such input is of great value to align schemas that could work together while also promoting the usage of more interactive data exchange methods that would eventually lead to the creation of the Digital Twin platform.

2.1.1. LL1 – Madrid

Madrid LL intends to help urban freight deliveries within the city centre by improving the deliveries' efficiency. This goal will be achieved through a logistic micro hub using light electric commercial vehicles. At the same time, a Digital Twin will help identifying behavioural pattern and anticipate events while further exploration of the potential of data management will be performed (i.e., evaluating flows and congestion), Finally, the deliveries' efficiency is going to be further enhanced by applying a route optimization engine in different scenarios while also combining different vehicle typologies.

The overall aim is to explore alternative and sustainable business models fostering public-private cooperation mechanisms with courier and logistic companies, with the goal of contributing to the development of urban policies.

2.1.1.1. Data definitions

In this Section, the data definitions for the Madrid Living Lab are listed as provided by the project's partners. These data definitions refer to basic entities of the modelling and will be updated based on the needs of the models and the evolution of the LL.

Table 2 LL1 data definitions: Vehicle

VEHICLE
Description:

The vehicle data type refers to the delivery vehicles of Madrid's living lab. The entity includes properties defining the vehicle's type and usage. The vehicles data are occasionally updated and are currently included in the simulations as CSV files.	
Field	**Description**
ID	Unique vehicle identifier
Name	Name of the vehicle
StartTimeWorkday	Departure time from the hub
EndTimeWorkday	Arrival time to the hub
DepartureLocation	Coordinates of the route starting location
ReturnBase	TRUE if the route ends where it started, FALSE otherwise
ArrivalLocation	Coordinates of the route ending location
WorkBreakStart	Time the vehicle can start a break
WorkBreakEnd	Time the vehicle has to end the break
WorkBreakDuration	Allowed break duration
CapacityUnits	Units that the vehicle can load (parcels, pallets, envelopes etc.)
InitialUnits	Units in the vehicle left before a planning run (parcels, pallets, envelopes etc.)
MaxDailyKM	Maximum kilometres a vehicle can travel in a day

Table 3 LL1 data definitions: Service

SERVICE
Description: The service data type refers to a delivery that needs to be fulfilled. It includes data such as the service location, units to be delivered and the duration of the service. The data are continuously generated and currently included in the simulations as CSV files.

Field	Description
ID	Unique service identifier
Name	Receiver name (optional)
Duration	Time to serve; measured from driver leaving the vehicle to driver returning to vehicle
ServiceAddress	Address of the service (optional)
ServiceHouseNumber	Street number of the service (optional)
ServiceCity	City of the service (optional)
PostalCode	Zip code of the service (optional)
ServiceLocation	Coordinates of the delivery point - MANDATORY
ServiceWindow1Start	Initial time to deliver the service
ServiceWindow1End	Final time to deliver the service
LoadUnits	Units to pick up (parcels, pallets, envelopes etc.)
UnloadUnits	Units to drop off (parcels, pallets, envelopes etc.)
WayBillCustomer	Unique identifier assigned to the service

Table 4 LL1 data definitions: Settings

SETTINGS	
Description: The settings data entity refers to simulation settings that need to be set for the execution of a model. It includes several factors and flags that alter the configuration of a simulation. They are currently provided as an XML file.	
Field	**Description**

Route weight	Time or Distance
MinHoursBreak	Number of working hours before drivers have to take a mandatory break
AllocateMode	Number of Services, Load Distribution or Work
AllVehicles	Uses all available vehicles or the minimum possible (Y/N)

2.1.1.2. Volumetrics

Table 5 LL1 volumetrics

Data	Type	Volume	Rate
Vehicle	Historical	1MB	Every optimization run
Service	Historical	1MB	1000 orders/day
Settings	Geographic	10kB	Static

2.1.2. LL2 – The Hague

The Hague LL aims to experiment with new forms of parcel distribution services such as crowd shipping. The municipality also wants to understand the impact of new logistics service platforms and their integration.

As such, we first implement the MATSim activity-based model for the passenger transport for the city of Hague. Secondly, we use the parcel module implementation of MASS-GT (parcel generation and parcel tour formation). The final aim is to combine these models and use them to investigate the impact of the crowd shipping implementation. Additionally, we will explore the effect of combining networks of different providers for the parcel delivery market.

2.1.2.1. Data Definitions

In this section, the data definitions for the Living Lab of Hague are listed as provided by the project's partners. The entities are designed to accommodate the use cases of the LL but basic principles are shared between LLs. The data definitions are still evolving together with the development of the simulation models.

Table 6 LL2 data definitions: Zone Data

Zone Data
Description:

Data defining a specific zone in the municipality. The distinction could be independent of the administrative zones and follow the logistics' needs. The zone data are static, and they are currently provided to the simulations as input files.	
Field	**Description**
ID	Area code
Name	Zone name
Centroid	Location of the zone centroid
Municipality	Municipality name
Coordinates	X, Y-coordinates of zone
UCC	Location of the UCC
Perimeter	Perimeter of the zone
Areas	Areas within the zone
Surface	Surface of the zone

Table 7 LL2 data definitions: Depot

Depot	
Description: The depot data type describes locations where the parcels are stored. The depot data locations are static, and they are currently provided to the simulations as input files.	
Field	**Description**
ID	ID of the depot
Name	Depot name
Company	Company managing the depot

Location	Location of the depot
Latitude	Coordinates – latitude
Longitude	Coordinates - Longitude

Table 8 LL2 data definitions: Parcel

Parcel	
Description: Aggregated data on the parcel demand in a city with information on the market share for each courier company. Among others, they are used to define market shares of the local-to-local, local-to national and national-to-local. The data are continuously collected and provided to the models in an aggregated format over a certain period.	
Field	**Description**
ID	Parcel ID
Type	Type of the parcel
Segment	Segment of the parcel
VehicleType	Type of the vehicle carrying the parcel
Company	Courier company transporting the parcel
Depot	Depot ID
OriginZone	Zone ID
DestinationZone	Zone ID

Table 9 LL2 data definitions: Activity

Activity

Description: The activity data type refers to the trips made in the area from all the members of each household. The entity concerns continuously collected data that are provided to the models as input files.	
Field	**Description**
ID	Activity ID
Type	Activity type
Mode	Mode type
HouseholdID	Household ID
MemberID	Member ID
TripID	Trip ID
Origin	Origin of the trip
Destination	Destination of the trip

Table 10 LL2 data definitions: Vehicle

Vehicle	
Description: Information related with the parcel delivery vehicles. Since the LL2 concerns crowd shipping applications the vehicle data are dynamic and flexibility concerning the vehicle data is required. Currently, the data are provided to the models as input files.	
Field	**Description**
ID	Vehicle ID
Type	Vehicle type
Capacity	Carrying capacity

Table 11 LL2 data definitions: Parcel Delivery Time

Parcel Delivery Time	
Description: Cumulative probability distribution of the parcel delivery time in hours. The final definition of this data type is not yet completed.	
Field	**Description**
ID	

2.1.2.2. Volumetrics

Table 12 LL2 volumetrics

Data	Type	Volume	Rate
Activity data - Albatross	Historical	TBC	Static
Zone data	Open source	TBC	static
Parcel demand	Open source	TBC	static

2.1.3. LL3 – Lyon

In Lyon, data is used to set up large-scale agent-based transport simulations of the city. This process is based mostly on open and publicly available datasets as well as data that will be gathered by logistics operators.

At the time of writing of this document it is not defined which datasets can be provided by the operators, so a general description is provided. These data will help to validate assumptions and inform on a macroscopic level the development and calibration of the agent-based models. The third pillar is the use of cameras on the Confluence peninsula to detect the number of utility vehicles entering and exiting the study area. Information on the acquisition, storage and processing of this data is provided in the following.

2.1.3.1. Data Definitions

1.2.1.1.1 Publicly available data

Table 13 LL3 data definitions: Insee Datasets

Insee Datasets

Description: Data openly provided by Insee (National Statistical Institute of France) are used for the agent-based simulations	
Source	**Description**
https://www.insee.fr/fr/statistiques/3625223 https://www.insee.fr/fr/statistiques/3627376	Disaggregate and aggregated population census data (RP)
https://www.insee.fr/fr/statistiques/3566008 https://www.insee.fr/fr/statistiques/3565982	Origin-Destination data for work and education commutes (RP-MOBPRO, RP-MOBSCO)
https://www.insee.fr/fr/statistiques/3560118	National income tax registry (FiLoSoFi)
https://www.insee.fr/fr/statistiques/3568638	Service and facility registry (BPE)
https://www.insee.fr/fr/information/2017499	Hierarchical index of spatial zoning systems in France
https://www.insee.fr/fr/information/2017499	Spatial statistical zoning system (IRIS)

Table 14 LL3 data definitions: Other Open Datasets

Other Open Datasets	
Description: Further openly available data used for the agent-based simulations	
Source	**Description**
https://www.data.gouv.fr/fr/datasets/base-sirene-des-entreprises-et-de-leurs-etablissements-siren-siret/	Enterprise census of France (SIRENE)
https://geoservices.ign.fr/documentation/diffusion/telechargement-donnees-libres.html#bd-topo	The National address database (BD-TOPO) from the National Geographic Institute (IGN)

https://www.statistiques.developpement-durable.gouv.fr/enquete-nationale-transports-et-deplacements-entd-2008	The national household travel survey from the Ministry of Ecological Transition
https://download.geofabrik.de/europe/france/rhone-alpes.html	Geofabrik cut outs of network data from OpenStreetMap for spatial simulations

Table 15 LL3 data definitions: Public Transport Data

Public Transport Data	
Description: GTFS schedules are used to represent the public transport in the simulations. The datasets are provided publicly.	
Source	**Description**
https://ressources.data.sncf.com/explore/dataset/horaires-des-train-voyages-tgvinouiouigo/information/	SNCF-TGV
https://ressources.data.sncf.com/explore/dataset/sncf-intercites-gtfs/information/	SNCF - Intercités
https://ressources.data.sncf.com/explore/dataset/sncf-ter-gtfs/information/	SNCF - TER
https://transport.data.gouv.fr/datasets/horaires-theoriques-du-reseau-transports-en-commun-lyonnais-1/	TCL Lyon

1.2.1.1.2 Proprietary data for agent-based simulation

For the research project, the regional household travel survey for the Auvergne – Rhône-Alpes region was provided to IRTX by CEREMA[2]. It contains information on the movements of a sample of persons from the region, along with their sociodemographic profile.

1.2.1.1.3 Proprietary data from logistics operators

Currently, discussions are being held with logistics operators that may be interested in contributing to the project. However, no specific agreement has been made to date. In case that data are going to be

[2] *https://www.cerema.fr/en*

provided, a non-disclosure agreement will be signed between IRTX, the logistics operator and potentially SPL Lyon. Currently, the list of requested data sets includes:

1. Detailed information on the packages distributed by the operator (typology, origin, destination, distribution centre)
2. Number of packages: per day at the Confluence district
3. Number of packages: weekly / monthly / seasonal variations (e.g., Christmas)
4. Number of packages: by distribution centre,
5. Origin of packages before being delivered to Confluence
6. Typology of the vehicles used
7. Size of the packages (statistical distributions)
8. Total volume of B2B activities of the operator in the region / the department
9. OD Matrix (origin -> destination sector) for B2B as input for extrapolating to the whole region
10. Exchange volumes between different economic sectors:
    - 10.1. Number / volume of shipments
    - 10.2. Number / volume send and/or received by sector

For a higher-level analysis, additional information is requested, such as:

1. Business numbers
2. Types of distributed freight
3. General scheme of logistic organisation
4. Number of daily tours
5. Percentage of sub-contracting
6. Average size / weight of shipments
7. Distance travelled motorized and non-motorized per day
8. Number, types, capacities of vehicles
9. Number of people mobilised
10. Distance covered per tour
11. Energy consumption per day / per tour
12. Emissions produced per day / per tour
13. GPS information on one or multiple delivery activities
14. Cost of organisation and cost of perturbations
15. Perception of the service of citizens

1.2.1.1.4 Video data collected on road network

There is the intention of deploying video cameras on the road network with the purpose of segregating logistics traffic flows. The collected data are expected to include:

- Videos of urban traffic (vehicles) entering and exiting the district
- Number plate recognition (to be confirmed)

Videos should be collected from the rear of the traffic. As such, images will not include details on drivers or passengers' identity.

The deployment of cameras, data collection, transfer, storage, and treatment processes will be detailed in a specific authorisation demand which will be duly assessed by French authorities against GDPR regulations.

2.1.3.2. Volumetrics

The open dataset for agent-based simulation is retrieved once for the course of the project as it is updated on an annual basis. In the operating LEAD platform, this will be the frequency of consumption. The proprietary data are only available on request; therefore, they are not going to be updated automatically. The household surveys are performed on average every five years. The frequency of transmission from the logistics operators will be defined in coordination with them. For the time being, no real-time or periodical data exchange is envisioned.

Concerning the video data an experimental process will be elaborated to detail the schedule and volumes of data to be collected onsite. Video capture will be limited and subject to authorizations delivered by national authorities in accordance with GDPR regulations.

2.1.3.3. Special Security Requirements

There are no security requirements for the open datasets. On the other hand, the proprietary household survey data cannot be publicly available, hence they need to be stored in a secure environment. Similarly, before the acquisition of the operators' data, an NDA needs to be signed with each operator. Afterwards the data need to be stored in a secured environment with controlled access to ensure that only parties included in the NDA have access.

For the video data a dedicated storage container will be implemented to consolidate the batch of data between collection points and processing servers. Data will be then processed once anonymized. All these processes to be deployed within an IRT SystemX secured environment are subject to authorizations delivered by national authorities in accordance with GDPR regulations.

2.1.4. LL4 – Budapest

Budapest Living Lab focuses on the inner-city area, enclosed by the Grand Boulevard, where the density of retailers/ shops, loading areas and population are high. The delivery time of these shops overlaps with the peak hours, so it generates congestion in the narrow streets of the city and increases noise and air pollution.

The Digital Twin will define the best location of a mini hub based on the value case scenarios as presented in D3.1, which would be a solution to these problems. The value case scenarios contain a combination of facilities, various types of vehicles and different durations of operational time windows. In Budapest the BILK is functioning as a UCC, so in the framework of the project the Budapest Living Lab tries to find the best place for the mini hub. The scenarios will investigate the advantages of using a UCC and a mini hub for serving a certain area. The local stores will be served with electric vans; the line haul from UCC to mini hub will be implemented with CNG and hydrogen fuel cell trucks. The operational time windows for serving local stores will be between 7h, 12h and 24h. The variable time window will provide some changes in the volume of the traffic because the delivery processes will be able to avoid the peak-hours and they will be more evenly distributed throughout the day.

2.1.4.1. Data Definitions

In this section, the data definitions for the Living Lab of Budapest are listed as provided by the project's partners. Budapest's use cases are reflected to the inclusion of a mini-hub data entity that is used to investigate the specific scenarios of the LL. The rest of the data entities follow similar patterns as generally observed from other LLs.

Table 16 LL4 data definitions: Store

Store	
Description: Data related to the stores that send the parcels. The data are occasionally updated and provided to the model execution as files.	
Field	**Description**
ID	Store ID
Name	Store Name
Address	Store address information; street and number
Zipcode	Store zip code
Latitude	Geo-information of store's location – latitude
Longitude	Geo-information of store's location – longitude

Table 17 LL4 data definitions: Mini-hub

MiniHub	
Description: Data related to the mini hub. The data are occasionally updated and provided to the model execution as files.	
Field	**Description**
ID	Mini Hub ID
Name	Mini Hub Name
Address	Mini Hub address information; street and number
Zipcode	Mini Hub zip code

Latitude	Geo-information of mini hub's location – latitude
Longitude	Geo-information of mini hub's location – longitude
TimeWindow	Time window information
LoadingTime	Time that takes for a parcel to be loaded from the mini hub

Table 18 LL4 data definitions: Fleet

Fleet	
Description: The entity concerns data related to the fleet of the vehicles that are used for the parcel deliveries. The data are occasionally updated and currently provided to the simulations as files.	
Field	**Description**
ID	Vehicle ID
Type	Vehicle Type
Manufacturer	The vehicle manufacturer
Fuel	Fuel Type
Quantity	How many similar vehicles are in the fleet for the region of Budapest
WorkingTime	Working time per day in hours

Table 19 LL4 data definitions: Origin-Destination

Origin-Destination	
Description: The OD-matrix of the private transportation of the parcels.	
Field	**Description**

ID	OD ID
Origin	Origin
Destination	Destination

Table 20 LL4 data definitions: Orders

Orders	
Description: The orders are a data entity that concerns the orders that customers are placing to the stores and need deliveries. The data are continuously collected and currently provided to the simulations as input files.	
Field	**Description**
ID	Order ID
StoreID	Store ID
DeliveryDate	Date the order has been / will be delivered
DeliveryTimeWindow	The time window for the delivery date
UnloadingTime	Required time to unload the order
RouteSequence	The route that the vehicle will follow to deliver the order
VehicleID	The ID of the vehicle to deliver the order
Quantity	Order quantity

2.1.4.2. Volumetrics

Table 21 LL4 volumetrics

Data	Type	Volume	Rate
Store Information	Historical	100kB	Static
OD-matrix	Historical	100kB	Static

Fleet Information	Historical	100kB	Static
Order Information	Historical	100kB	50-70 orders/day

2.1.5. LL5 – Oslo

LL5 Oslo plans to explore the micro hub concept with four scenarios that are based on home delivery concepts of primarily larger items from the furniture superstore IKEA at Slependen, 8 km west of Oslo CBD. Other stores in the same area may also be added (scenarios 3 and 4 as presented in D3.1).

The customer will approve ex-ante by either physically visiting an IKEA store and order the goods to be shipped to home or by placing an order online. Because of the strong competition in the market of ordinary packages, LL Oslo focuses on shipments or large items that provide service to smaller items as needed, e.g., as part of larger shipments.

The practical testing of the four scenarios is handled by the project's partner NIMBER. The location of the IKEA store and the micro-hub at Lysaker, 3 km west of Oslo CBD is in line with the current main markets for NIMBER. The micro-hub for the LEAD experiments is in the vicinity of the market for customers and close to the main arterial road E18. The location is well suited for experiments with variants of home deliveries based on both dedicated trips with e-vans and non-dedicated trips by using crowd-shipping.

IKEA facilities will be the departure point. The set of alternative scenarios will challenge the current B2C transport pattern. The current model with customers bringing the items home themselves will be the baseline scenario.

2.1.5.1. Data Definitions

In this section, the data definitions for the Living Lab of Oslo are listed as provided by the project's partners.

Table 22 LL5 data definitions: Store

Store	
Description: Information related to the store. IKEA will be the main store from where all deliveries start, but it is likely to add more stores in the future related to the scenarios 3 and 4.	
Field	**Description**
ID	Store ID
Status	Store is open (active) or closed
Name	Name of the store
Address	Street, number and zipcode; the address information of the store

Website	Store website
Preparation	If the store prepares the orders for delivery
PickUpPoints	Shows if the store can be a pickup point for the NIMBER bringers
DeliveryService	Delivery service to the customers as defined by the four scenarios as explained in D3.1

Table 23 LL5 data definitions: Trip

Trip	
Description: Information related to the trips for delivering the orders to the client.	
Field	**Description**
ID	Trip ID
ClientID	ID of the client (if applicable)
OrderID	May be more than one order per trip
VehicleType	Type of the vehicle (car, van)
FuelType	The type of fuel
CargoType	Cargo type based on NIMBER's volume/weight classes
Type	Dedicated / Non-dedicated
Distance	Distance from store (direct) or from hub (indirect) and to customer
DepartureTime	Time of day when departing from store
DeliveryTime	Time of day when delivering to customer
ReschedulingByCustomer	If customer has changed time of delivery
ReschedulingByStore	If store has changed time of delivery
ReschedulingByBringer	If bringer has changed time of delivery
Delay	Number of minutes the delivery is outside of the time window
CargoStoredTime	The time the cargo has been stored in the micro hub
CargoLoadingTime	The micro hub's loading time

CargoUnloadingTime	The micro hub's unloading time
Payable	Costs distribution for the customer

Table 24 LL5 data definitions: Order

Order	
Description: Information related to the orders received by a store. The data are continuously collected and provided to the model execution as input files.	
Field	**Description**
ID	Order ID
ClientID	ID of the client (if applicable)
Date	Order date
Delivery	Delivery date
TimeWindow	Delivery time window
DeliveryMethod	Delivery method according to the 4 delivery scenarios
NumberSKU	Number of different products / codes in the order
OrderValue	Cost of client order (if applicable)
Route	The route the vehicle will follow to deliver the order

2.1.5.2. Volumetrics

The table below presents the data volume per data definition. Data will be assigned to the variables listed above for each main group in the table (the volumes need to be worked out later):

Table 25 LL5 volumetrics

Data	Type	Volume	Rate
Store Information	Historical	TBD	Static

Order Information	Historical	TBD	TBD orders / day
Trip	Historical	TBD	TBD orders / day

2.1.6. LL6 – Porto

Porto LL aims at transforming SONAE's deliveries to be based on electric mobility and optimize its fleet using electrical vehicles (EVs). Framing the problem, the first goal is to position the EDV charging stations to SONAE's stores by considering parameters such as vehicle type, delivery lead time, distance travelled, traffic, frequency, and duration of EV stop. Secondly, we focus on routing optimization and rescheduling orders finding alternative shortest paths. We are supposed to use optimization algorithms such as location-allocation algorithms and machine learning algorithms such as distributed sub-trajectories clustering and Future Location Prediction (FLP) - Trajectory Prediction (TP).

2.1.6.1. Data Definitions

In this section, the data definitions for the Living Lab of Porto are listed as provided by the project's partners.

Table 26 LL6 data definitions: Store

Store	
Description: Information related to the stores receiving orders. The data are occasionally updated and provided to the model execution as files.	
Field	**Description**
ID	Store ID
Name	Name of the store
Status	Store is open (active) or closed
Address	The store's address
Zipcode	The store's zipcode
City	The city the store is located
Latitude	Latitude of the store's location

Longitude	Longitude of the store's location
Phone	Store's phone number
Website	Store's website
IsWarehouse	True if the store is a warehouse
IsPreparation	True if this store prepares the orders for delivery
IsPickUpPoint	True if the store can be a pickup point for the customer
DeliveryService	True is the store provides delivery services to the customer
RenewableEnergy	True if the store has been set with renewable energy sources
Route	The route the vehicle will follow to deliver the order

Table 27 LL6 data definitions: Charging Locations

Charging Locations	
Description: Data related to the locations of the BEV charging stations. It is unlikely for the data to be updated and currently they are provided to the simulations as files.	
Field	**Description**
ID	Charging station ID
Type	Charging station type
Address	Address of the charging station in the SONAE network
Latitude	Latitude of the charging station's location
Longitude	Longitude of the charging station's location
Availability	Charging station's time availability

Table 28 LL6 data definitions: Fleet

Fleet	
Description: Data related to the delivery vehicle's fleet. The data are occasionally updated and provided to the model execution as files.	
Field	**Description**
ID	Vehicle ID
Type	Vehicle type
Manufacturer	Vehicle manufacturer
Model	Vehicle manufacturer model
Year	Vehicle's construction year
Fuel	Fuel type of the vehicle (diesel, electric, etc)
Quantity	Fleet quantity of vehicles of the same type
RechargeDuration	Time needed for an electric vehicle to recharge
AverageDowntime	Average time the vehicle stays parked
ServiceFrequency	Times per year the vehicle needs to be serviced
ServiceDuration	Average duration of a service for the vehicle
LicensePlate	The vehicle's license plate number

Table 29 LL6 data definitions: Order

Order
Description:

Data related to the received orders by the stores. The orders are continuously collected and currently provided to the simulations as input files.	
Field	**Description**
ID	Order ID
ClientID	Client ID
VehicleID	Delivery vehicle ID
Date	Order date
DeliveryDate	Date that the order will be delivered
PreparationStore	ID of the store where the delivery is going to be prepared
Zipcode	Postal code of the delivery location
DeliveryMethod	Delivery method (e.g., home delivery, take away etc)
DiscountCard	Premium cards for delivery fee discounts (entegra zero etc)
NumSKU	Number of products / codes in the order
NumNegativeCoolSKU	Number of products in the order that need refrigerator
TotalCost	Total client cost of the order
RouteSequence	The route the vehicle will follow to deliver the order

2.1.6.2. Volumetrics

The table below presents the data volume per data type definition:

Table 30 LL6 volumetrics

Data	Type	Volume	Rate
Store Information	Historical	35kB	Static
Order Information	Historical	50MB	300 orders / day
Charger Locations	Historical	20kB	Static

Fleet Information	Historical	21.9KB	Static

2.2. Inter-module Communication

This section describes the interfaces linking the platform's main components: the DDDAS, the DSS and the Model Library. The overall architecture of the LEAD platform, as presented in detail in the D2.5 LEAD DDDAS System, is going to be briefly discussed to provide context for the interactions between the components. Subsequently, the interfaces of the three main components are analysed.

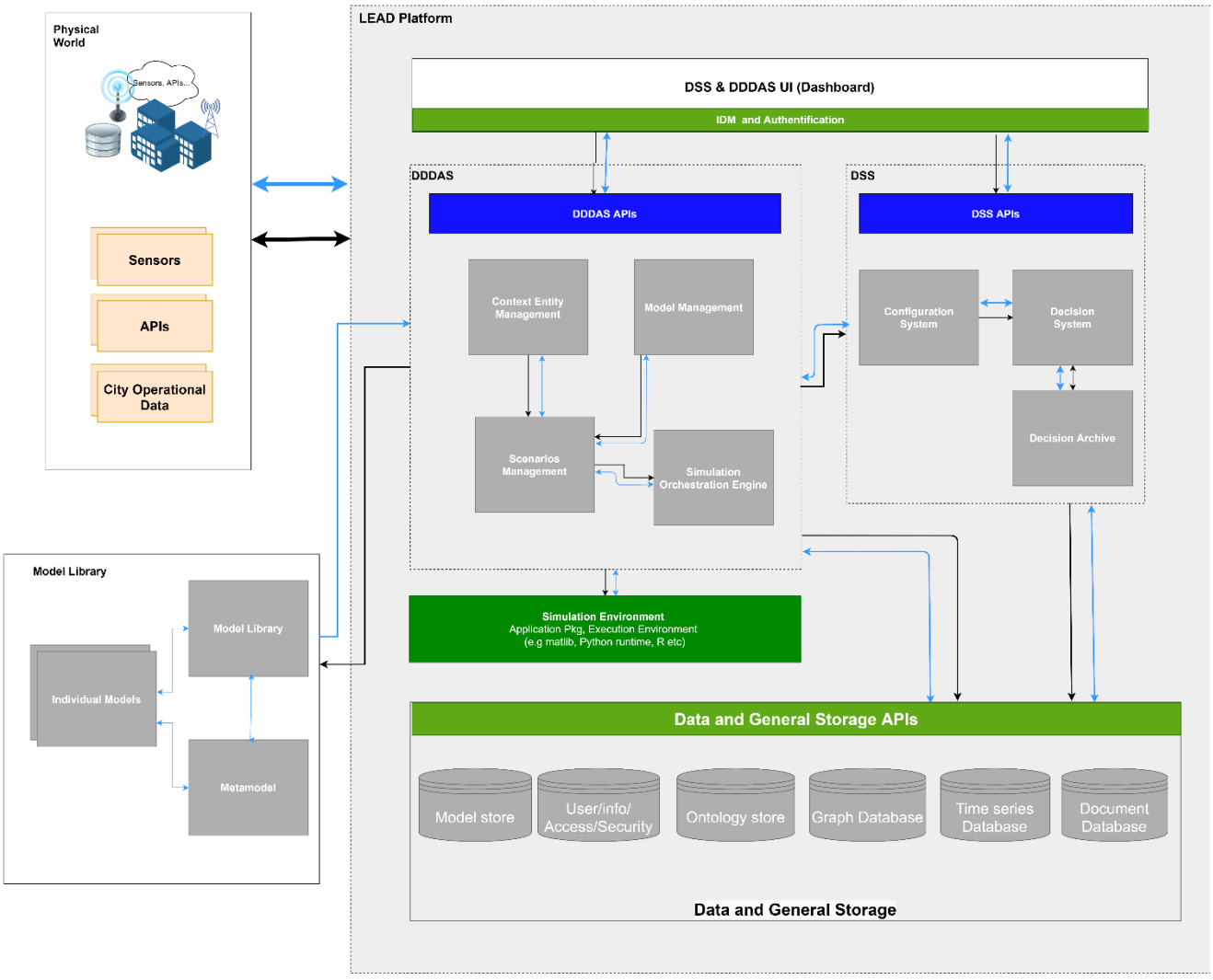

Figure 1 LEAD platform high-level architecture

2.2.1. LEAD Platform's Architecture

A general overview of the LEAD platform can be observed in Figure 1. As seen in the figure, the LEAD Platform is designed to ingest data from the Physical World while the Figure 1 Model Library provides model information and metadata to the platform and can be either part of the platform or an external

component. The DDDAS component manages the entities, the scenario creation, and the simulation execution. The DSS applies cognitive algorithms to the results of the simulations in order to select the best scenario based on the parameters and selected KPI targets. All the functionality is provided to the user through dedicated User Interfaces. Finally, all the components have access to data services as needed to accommodate the LL use cases and the execution of the models.

Based on these architecture and design principles, an initial implementation of the platform was deployed (Figure 2). An event streaming platform (Apache Kafka) ingests data (real-time, batch, imported, etc.) from the physical world. The platform is considered an industry standard for such applications and offers high levels of stability, availability, and scalability. Furthermore, it is also very modular and offers Kafka Connect that allows data ingested into Kafka topics to be easily stored in different data structures as needed by the models and living labs.

Data of a more static nature can also be ingested into the platform in different structures or in the filesystem based on the specification and implementation of the models. To that end, both relational and non-relational databases are offered, and the schemas of the tables or collections are defined in iterative discussions with the living labs.

Often, models are developed using files as data input. Therefore, normalized naming has been defined for the data paths so that both ingested data from external source and generated data from the execution of models can be uniquely identified through their metadata that are stored in the RDBMS.

Moreover, Openroute Service[3] has been chosen in coordination with the living labs as the preferred routing service and has been deployed as part of the LEAD platform. It is a cutting-edge crowd-sourced solution that offers several SDKs in different languages that match the preferences of the modellers.

Finally, monitoring of the platform has been considered as a core component of the platform and solutions have been put in place from the initial steps of the platform's deployment. To that end, Prometheus is used as the data collection service. It uses a flexible HTTP pull model and stores real-time metrics in a time series database. Grafana provides the potential for a standardized visualization of such metrics but also for (e.g.) data generated by the execution of the models.

[3] *https://openrouteservice.org/*

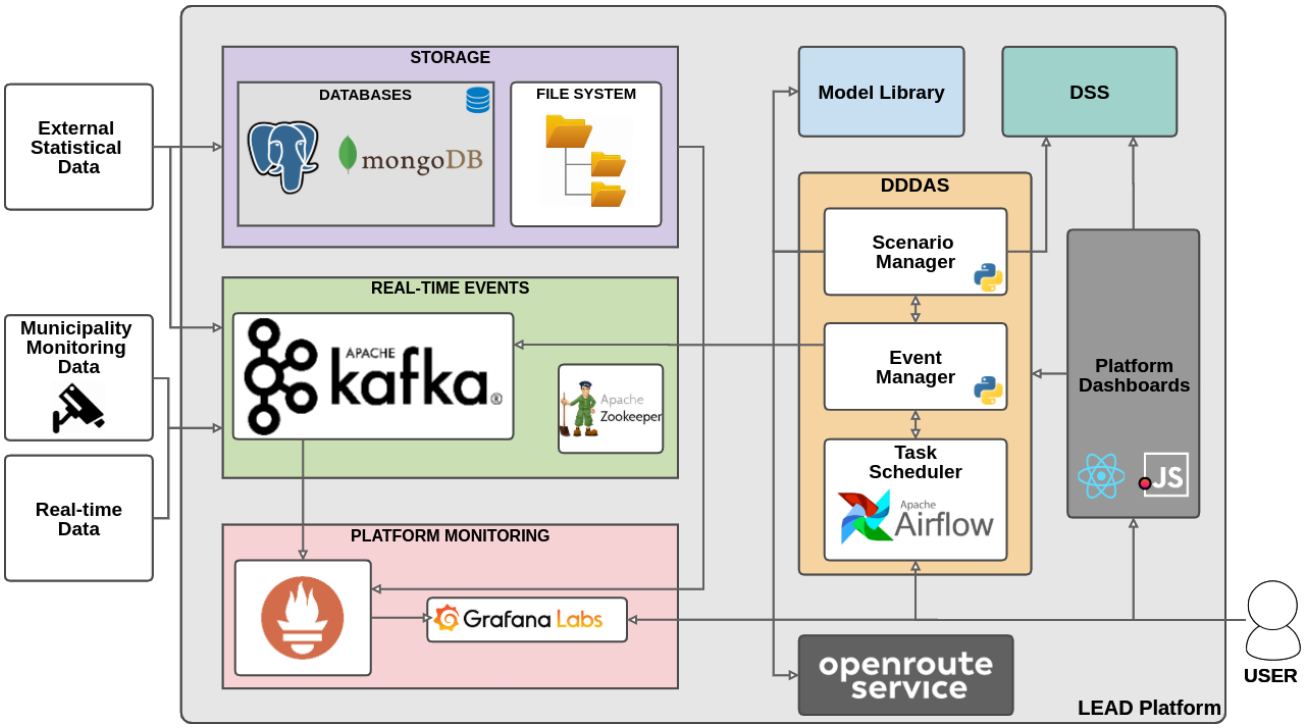

Figure 2 LEAD Architecture implementation

2.2.2. DDDAS

The dynamic data driven application system will operationalize the user created scenarios by linking the KPIs with the models, the data sources, and the simulation environment. At the first step, based on the selected KPIs and models, the DDDAS reads in and pre-processes data from the physical world through the corresponding interfaces or Kafka topics and retrieves any additional content from web servers (e.g., with tools such as wget, ftps etc) as requested by the model.

The DDDAS communicates with the Model Library to access metadata containing the information regarding the environment, inputs, and outputs of a model. Therefore, each model is expected to follow specific principles that will allow the platform to deliver it and to be parameterized so that multiple instances can be created for parallel execution.

After the data preparation step, the DDDAS uses a task scheduler to initialise the environment so that the requested model can be executed. Depending on the model's implementation and delivery, this could involve downloading the appropriate container images or retrieving the code from an accessible repository. The DDDAS uses containerization and virtualization to scale the system and the task scheduler handles the resources management and monitoring of the execution pipelines. At the end of the execution cycle, it is expected that the DDDAS will produce multiple model execution outcomes at the next time step based on the parameter sweeps of the various models in the scenario pipeline.

These model execution outcomes will be sent to the DSS which with select the best one using Bayesian methods. For the lifetime of the scenario, the DDDAS will continue to monitor the data sources at each timestep and, depending on established thresholds, re-initiate the simulation scenario pipeline. Depending on the LL the DDDAS will also monitor and provide metrics to the user about the

scenario. Situations where the processing time of the simulation exceeds the length of the time-step, will be raised with the user to ensure the DT can perform as close to real time as possible. Finally, given the flexibility provided by the software architecture and the data sources, further platforms for IoT and Digital Twins are investigated to be included in the platform, such as Eclipse Ditto or Predix.

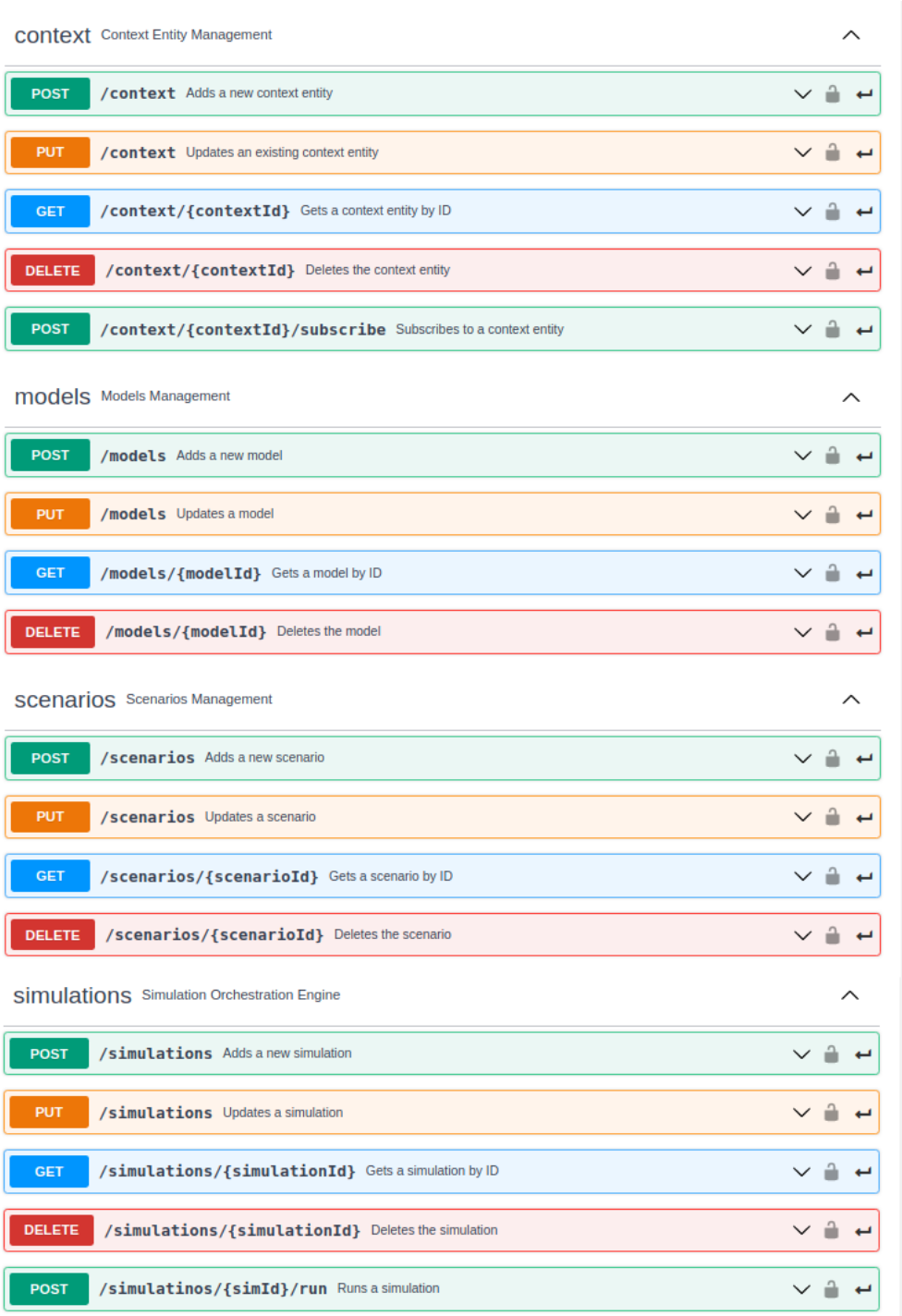

Figure 3 Initial version of the DDDAS API

An initial version of the DDDAS API is presented in Figure 3 with endpoints for CRUD actions on the entities, the models, the scenarios, and the simulations. The DDDAS handles the inclusion of sensors from the physical world as platform entities while the user can manage the models through an API that manages the Model Library. Through DDDAS, the user can manage scenarios by defining a sequence of models to be executed together with their parameters. Finally, the simulation is also defined through user inputs and model requirements and is handled through a dedicated endpoint. This API has been designed using the OpenAPI specification[4] and an interface from where users can view and test the API is also provided (for more details on the API design, see D2.1[5]).

2.2.3. Model Library

The Model Library component (ML) is a collection of all the models available for the deployment of the DTs. The ML's role is to provide all the necessary meta-information through its database so that the platform can access the models' source, their requirements, inputs, and outputs and as well their execution pipelines and possible model interactions. The ML is a dynamic component that should support existing model updates and the inclusion of new ones. The ML consists of two main elements: 1) the model library itself; 2) the metamodel.

The ML consists of a database that contains the models and the APIs that connect to the external models that are available for the DT platform. To facilitate version management in a decentralized way, each model would have its own GitHub repository from where the DT platform can clone the repository. In each repository, besides the model itself, a JSON file will be included with key information of each model, such as execution environment parameters, inputs, and outputs. The structure of the json will be the ML Metamodel.

The ML metamodel aims to guide towards the selection, use of existing models, and support the development of new models, by mapping the interconnections between the models. This mapping is made to assist the deployment of the models in a machine-readable way for the DT platform and the adoption of the system from other living labs.

As shown in Figure 2, the ML is connected to the DDDAS from which it receives two types of queries. The first one regards the query of a particular model, where the ML component returns the repository location for model cloning and the Json file of said model. The second query refers to the metamodel information, where the DDDAS can ask for specific models to obtain a KPI, interconnections between models and an overview of each model inputs and outputs.

For example, if the user wants the KPI "CO2 emissions", the platform queries the metamodel to find which models provide this KPI, what are the models' inputs and which of those could be provided by other models together with any data transformations needed. Once the models are selected, the ML is queried to obtain the repositories and the JSON files with the execution commands and the data validation.

[4] *https://swagger.io/specification/*
[5] *D2.1 – LEAD Solution Architecture*

2.2.4. DSS

The Decision Support System module employs Bayesian inference techniques[6] to decide which outcome of the simulation scenario is most likely achievable and addresses the KPIs of the defined scenario. It then recommends to the city operators the interventions required in the physical world to achieve the predicted outcome. At each time interval the DSS continues to build its knowledge base of input parameters, model outputs, predicted outcomes, and real-world readings. Finally, it forms labels for the user interface, making simulation results readable and comprehensive to the end-users.

To accommodate the functionality above, the DSS exposes an API that in currently in its initial design phase (Figure 4). Three main entities have been identified and a basic CRUD paradigm has been followed. First, the decision models are managed in a similar fashion as implemented for the DDDAS. The user can explore, create, and modify decision models through the respective interface. The configuration concerns the definition of the parameters that relate to a decision model. Finally, the best scenarios as defined by the decision models are stored and can also be managed from the users.

[6] *Richard Mcelreath, Statistical Rethinking: A Bayesian Course with Examples in R and STAN, CRC Press, 2020, ISBN: 036713991X, 9780367139919*

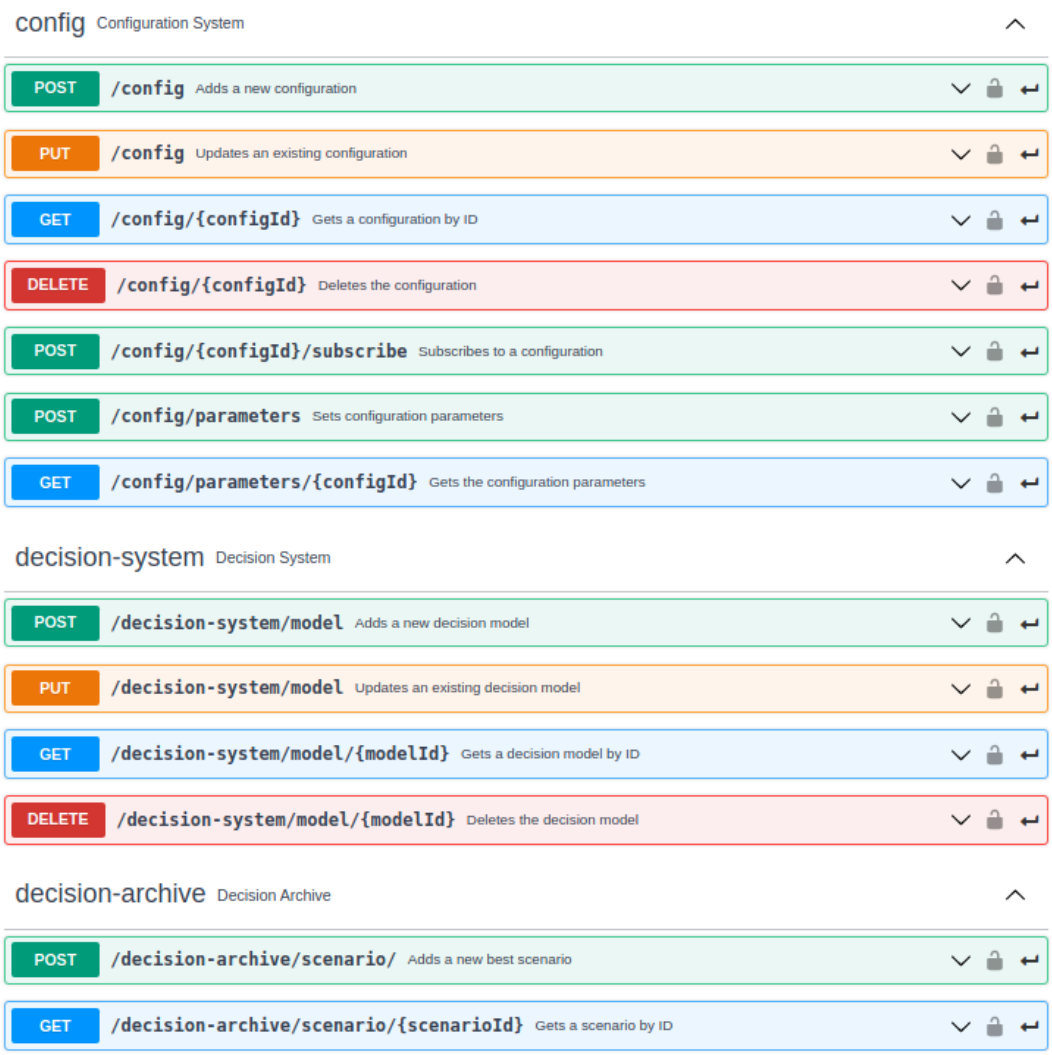

Figure 4 Initial version of the DSS API

The Cloud Computing side of the LEAD Platform will be based on existing open-source technology. Several cloud solutions exist in the market. The hybrid cloud stack provided by Cloudify[7], can integrate cloud and local environments. Other existing platforms are OpenStack[8], Cloud Foundry[9], and Apache CloudStack[10]. Classical approaches adopt the idea of deploying full virtual machines (KVM[11]). Based on the initial design phase, LEAD will follow a more modern approach by applying containerization techniques (Docker[12], Kubernetes[13]). With respect to the Simulation Orchestration Engine, solutions

[7] *https://cloudify.co/*
[8] *https://www.openstack.org/*
[9] *https://www.cloudfoundry.org/*
[10] *https://cloudstack.apache.org/*
[11] *https://www.linux-kvm.org/page/Main_Page*
[12] *https://www.docker.com/*
[13] *https://kubernetes.io/*

for IT automation to configure systems, deploy software and orchestrate IT tasks will be considered such as Ansible[14], SLRUM[15] and KVM[16].

2.3. Frontend Interfaces

The LEAD platform provides a variety of graphical interfaces to the end-user. The access to the dashboards is restricted only to authorized users and the registration is made through an invite / request system. A visitor to the LEAD platform can either request access to the platform through a contact form or be directly invited by the platform's administrators. Forgotten password and reset password functionality are also included. The API is also designed based on the OpenAPI specification and the endpoints are presented in Figure 5.

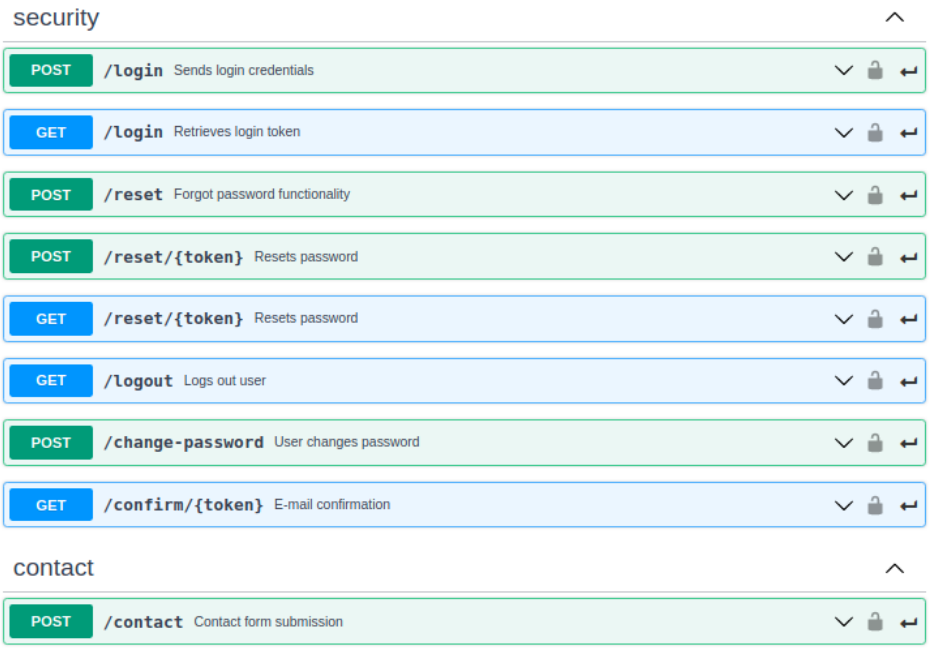

Figure 5 Initial version of the Security and Contact APIs

The main use case of the provided frontend is to allow an authenticated user to create and execute a scenario. A scenario consists of a model sequence, the parameters of the models and a set of KPI targets. To create the scenario, the user is presented with widgets containing dynamically created fields based on the selected models' metadata. The scenario creation graphical interface is still a work in progress that by design requires communication with the DDDAS, subscribes to Kafka topics and queries the databases to access various information related to the users, the models, the scenarios and more.

Subsequently, the dashboard sends the user defined scenario to the DDDAS simulation manager which transforms and sends the information to the task scheduler. The task scheduler also provides

[14] *https://www.ansible.com/*

[15] *https://slurm.schedmd.com/documentation.html*

[16] *https://www.linux-kvm.org/page/Main_Page*

a dashboard where the users can monitor several metrics relative to the running jobs. Status, logs, and graphs related to the steps and settings of currently existing execution pipelines are all presented to the user.

Finally, for further real-time performance metrics and model outputs Grafana dashboards[17] are provided to the users. These solutions collect data from special-purpose exporters and store them so that they can be queried and presented to the user either from the frontend or from any other visualization platform.

3. Cognitive Decision Support Systems

In this chapter, we capture the technical specifications for Cognitive DSS of the LEAD Platform. Firstly, the methodology used for the definition of the technical requirements is presented; the usage scenarios follow in section 0 and the main actors are presented next. Finally, the functional and non-functional requirements are explained in sections 3.4 and 3.5 respectively.

Both functional and non-functional requirements are crucial to guide the design of the architecture and the implementation of the platform during the project's development cycle, and ultimately as a re-usable asset from the project.

3.1. Methodology

An overview of the methodology that is used to define the technical requirements for Cognitive DSS is shown in Figure 6. Four main components have been identified:

1. the usage (business) scenarios,
2. the key actors,
3. the assets,
4. the functional and
5. non-functional requirements.

A key actor is a human that specifies a role played by a user that interacts with the LEAD platform. An asset is an entity that communicates with/or are used by the platform during the execution or realization of a use case. Usage (business) scenarios specify a series of actions or events between an actor and a system to achieve a goal. These goals are based on functional and non-functional requirements.

The requirements can be categorised based on their priority as displayed in Table 31. The category key is then going to be used throughout the tables that define the requirements. The categories describe the status of the requirements that have been discussed and agreed on by the stakeholders and will be formed through iterative discussions.

[17] https://grafana.com/grafana/dashboards

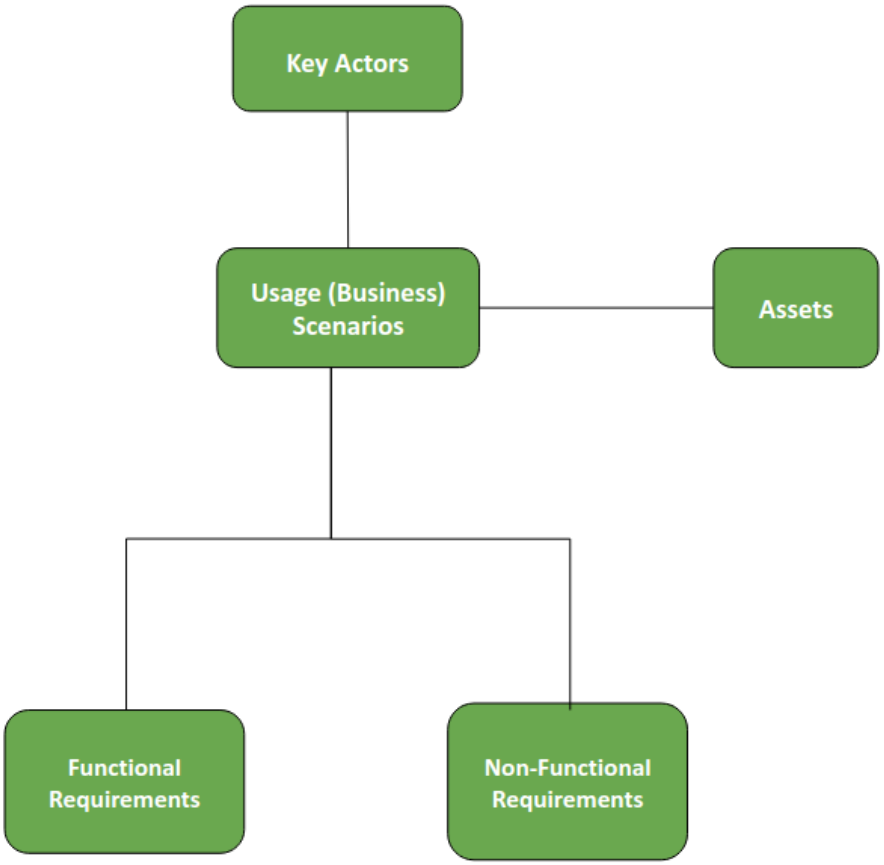

Figure 6 Methodology schema for the definition of the technical requirements

Table 31 Key requirement categories

Category	Key	Description
MUST	**M**	A requirement that must be satisfied for the platform to succeed
SHOULD	**S**	A high-priority item that should be included in the platform. Often a critical requirement which can be satisfied differently if strictly necessary
COULD	**C**	A requirement which is considered desirable but not necessary. It will be included if time and resources permit.
WON'T	**W**	The stakeholders have agreed that the requirement will not be implemented in the current project but may be considered for the future).

3.2. Usage Scenarios

A list of the usage scenarios, as defined in the previous section is shown in Table 32. Through these scenarios, we capture the overall user interactions with the Cognitive DSS component of the LEAD project.

Table 32 Cognitive DSS usage scenarios

Usage Scenario	Description
Designing, executing, and reviewing simulation scenarios	Refers to proposing and selecting the best strategies and scenarios, configuring model parameters, starting a simulation, displaying evaluation reports, and confirming a strategy.
Decision making for optimal outcomes in the DT (DSS)	Refers to analysing historic information, current physical world status, DDDAS/model outputs, and scenario KPIs to recommend optimal configurations for the next time-step.
Visualisation and data representation	Refers to visualising and analysing data using statistical, modelling, and AI-based analytics tools.

3.3. Actors & Assets

A definition of the key actors and assets of the platform have been presented in Section 3.1. It concerns any person or system respectively that has a role interacting with the LEAD platform. The actors and assets as identified in the current state of the project is listed in Table 33 and Table 34 respectively.

Table 33 DSS actors

Actor	ID	Description
Data Owner	**DO**	The person responsible for data curation including classification according to GDPR and Commercial Sensitivity
Modeller	**MO**	The person responsible for constructing or programming analytical models
Analyst	**AN**	The person responsible for designing and running DT scenarios
System Administrator	**SA**	The person responsible for operational availability of the platform
User Administrator	**UA**	The person responsible for user registration and onboarding

Table 34 DSS assets

Asset	ID	Description
WAN	**WA**	The wide area network the platform is connected to
LAN	**LA**	The local area network associated with the platform
Data pipeline	**DP**	Data transport, transformation and storage mechanisms
Packaged Application	**PA**	Application software used by the platform (e.g., MATSim)
Custom Software	**CS**	Scripts and applications of the platform
External System	**ES**	Systems external to the platform (e.g., weather, traffic APIs)

3.4. Functional Requirements

Functional requirements generally describe what a system or product must do, providing in detail what is being requested by the users. The functional requirements are presented in tables based on the component they concern along with the key actors, assets, and their priority category.

Table 35 Functional requirements: Dashboard

Dashboard				
Functional Requirement	**ID**	**Key Actors Involved**	**Assets Involved**	**Category**
Select scenario/strategy	A.1	MO, AN	ES, PA	M
Configure logistics profile	A.2	DO, MO, AN, SA	WA, LA, CS, PA	M
Start simulation	A.3	UA, SA	CS	M
Display evaluation report	A.4	DP, SA	WA, LA	M
Confirm strategy	A.5	MO, AN, SA, UA	-	S

Table 36 Functional requirements: DSS

Decision Support System				
Functional Requirement	**ID**	**Key Actors Involved**	**Assets Involved**	**Category**
Decide which outcome of the simulation scenario is most likely achievable and addresses the KPIs of the defined scenario	B.1	MO, AN, SA	WA, LA, DP, CS	M
Retrieve new KPIs for comparison	B.2	MO, AN	ES, CS	C
Generation of evaluation report	B.3	SA, US	PA, ES	S
Send notifications to the user	B.4	UA, SA	WA, LA, DP, PA, CS, ES	C

Table 37 Functional requirements: Data analysis & visualization

Data analysis & visualization			
Functional Requirement	**ID**	**Key Actors Involved**	**Category**
Ability to run supervised & semi-supervised analytics and models	C.1	MO, AN	M
Ability to run unsupervised / unattended analytics and models	C.2	MO, AN	M
Export reports in a range of formats (e.g., pdf, excel) from unattended models	C.3	AN	M
Run classical statistics and models based on many, varied sources (unstructured and unstructured)	C.4	MO, AN	M
Ability to perform a wide range of text analysis techniques on focus group data and social media	C.5	MO, AN	M

3.5. Non-Functional Requirements

Non-functional requirements concern aspects of usability, security, performance, reliability, supportability, testability, and maintainability of the DSS. The following tables provide a description of the non-functional requirements based on such aspects along with their priority category.

Table 38 Non-functional requirements: Usability

Usability		
ID	**Category**	**Description**
US.1	S	Custom information screens (e.g., for adding metadata) should be designed to make user tasks and workflows intuitive and efficient
US.2	M	Save analysis and visualization outputs to local drives
US.3	M	Record and retain metadata and volumetrics information associated with a dataset
US.4	M	It must be possible to access the platform remotely
US.5	M	The platform must be accessible to all project team members

Table 39 Non-functional requirements: Usability

Security		
ID	**Category**	**Description**
SE.1	M	Physical security to Tier 1 minimum
SE.2	M	Secure access control to the platform
SE.3	S	Unified user management for the platform with IP restrictions, and permission delegation options
SE.4	W	Single sign-on
SE.5	M	Access to platform must be supported in writing from institutional PIs (and all other PIs will be informed)
SE.6	M	Connection to LEAD intranet will use SSH
SE.7	S	Hold all data in an encrypted vault (AES128 min.)
SE.8	**S**	Offer an encrypted mail drop for the most sensitive information

Table 40 Non-functional requirements: Information Management

Information Management		
ID	**Category**	**Description**
IM.1	**M**	All analysis outputs should be checked back into the platform (even if saved locally)
IM.2	**S**	Outputs are automatically classified based on a matrix (see below) Status may be changed based on Data Governance meeting
IM.3	**C**	Support for distributed code repositories

Table 41 Non-functional requirements: Infrastructure

Infrastructure		
ID	**Category**	**Description**
IN.1	M	Platform must be protected against unauthorised intrusion and viruses
IN.2	M	Use gigabit or greater networking between platform hardware components
IN.3	S	Aggregate distributed physical resources into one, shared compute and data resource platform
IN.4	S	Support low latency, parallel processing
IN.5	C	Support long-running services
IN.6	S	The platform should support multiple Infrastructure models - bare metal, private cloud, public / hybrid cloud

Table 42 Non-functional requirements: Integration

Integration		
ID	**Category**	**Description**
IT.1	M	Platform is a single platform as far as user is concerned
IT.2	W	Cloud and non-cloud components need to be fully integrated

IT.3	M	An air gap is permissible between the cloud and non-cloud components during the project
IT.4	C	The platform should provide integrated application support with rich API layer

Table 43 Non-functional requirements: Compliance

Compliance		
ID	**Category**	**Description**
CO.1	S	The platform must comply with accessibility standards such as W3C Content Accessibility Guidelines
C0.2	M	Personal Information storage must comply with GDPR.
C0.3	M	During the project duration the freedom of Information

Table 44 Non-functional requirements: Operations

Operations		
ID	**Category**	**Description**
OP.1	M	SLA (99.99% availability)
OP.2	M	Service levels

3.6. Outlook

A decision support system is an information system that determines the best decision, based on certain criteria. The LEAD DSS module receives simulations scenarios from DDDAS module and determines the best ones, employing statistical and machine learning models for localised predictive analysis, and Agent Based Models (ABMs) for global perspective analytics. It also displays the results in performance monitoring dashboards. LEAD DSS decides which outcome of the DDDAS successful scenario addresses the defined KPIs, and then recommends to the city operators the interventions required in the physical world to achieve the predicted outcome. In doing so, DSS also builds its knowledge base of input parameters, model outputs, predicted outcome, and real-world readings. Outputs and recommendations from the DSS will be shown on a dashboard displaying the alignment of the simulations to the actual real-world conditions, as well as progress towards the scenario's value goals. This will enable monitoring of flows, KPIs, and will help identifying processes with discrepancies or structural faults.

The DSS of LEAD platform for last mile urban logistics consists of 3 main components as described in D2.1:

1. the configuration system that defines the models, scenarios, and devices parameters
2. the decision system that selects predictive models to make decisions regarding with the best scenarios/strategy among those that have been provided by DDDAS
3. the decision archived that is a database that keeps all the relative information regarding with the best scenario that has been chosen by Decision System

as depicted in Figure 7.

LEAD DSS acts as a strategic decision support system. The configuration system defines the models' parameters and provides them to the decision system. The results of the execution of the decision models are then stored in the decision archive that the users can ask through the dashboard and the DSS API (see Section 2.2.4).

As a first step, each of those components will be developed separately. The configuration system defines the models, scenarios, and possible devices parameters through python scripts. The decision system consists of a Cognitive Engine that is based on Bayesian Inference techniques. The engine selects the best simulation scenario and the best strategy by evaluating the various outcomes that emerge from the LL scenarios, calculating the models' externality, and training modules using learning algorithms with validated ground truths. Finally, the decision archive saves the decision system recommendations and passes them to a historical database as depicted in Figure 7. For the decision archive component, document-oriented databases such as MongoDB will be used. Later, all those components will be connected using call functions as shown in Figure 7 and JSON files will be used to transfer data related to meta-modelling.

The DSS is still in its design phase and the implementation details and timeline are being set. The development is led by INLECOM, and more details are to be provided in the D2.4 deliverable.

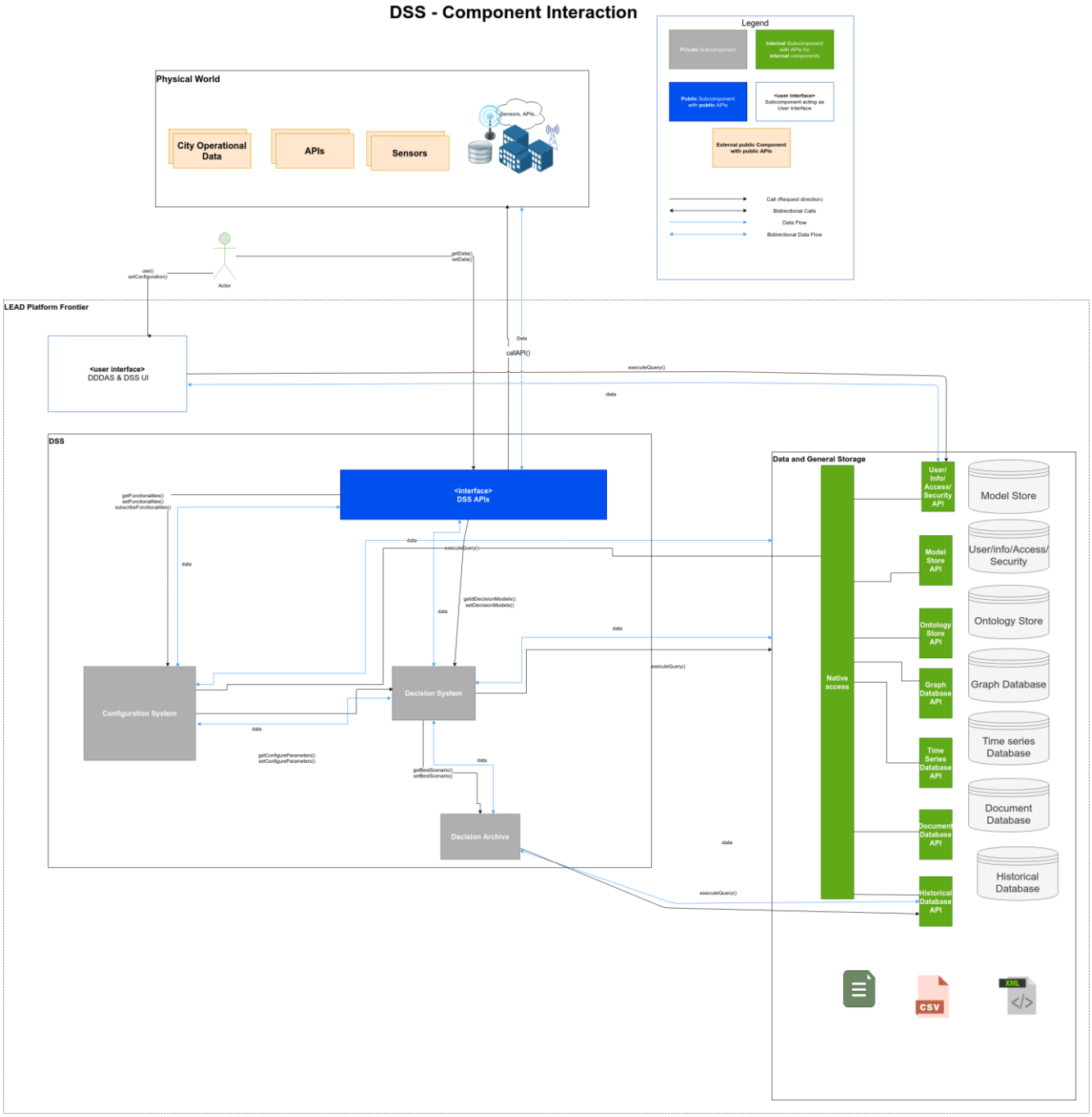

Figure 7 DSS - Component Interaction[18]

[18] Annex B, D2.1" Technical Requirements – Solution Architecture", LEAD project.

4. User Interfaces & Dashboards

4.1. Interface Design Principles and Frameworks

The trend towards a digital society provides users with new ways of accessing information and services. The providers of information and services rely increasingly on the internet to produce, collect, and provide a wide range of information and services online which are essential to the public[19]. Accessibility of information or services in this context should be understood as principles and techniques to be observed when designing, constructing, maintaining, and updating websites and mobile applications to make them more accessible to users.

Initiatives such as the Web Accessibility Directive (EU) 2016/2102[20] of the European Parliament and of the Council of 26 October 2016 on the accessibility of the websites and mobile applications of public sector bodies are proposed to improve the Web to be more accessible and available to everyone including the people with disabilities.

In this context, the World Wide Web Consortium (W3C) has proposed in 2018 in its latest version 2.1, the Web Content Accessibility Guidelines (WCAG)[21] that cover a wide range of recommendations for making Web content more accessible. Following these guidelines will make content more accessible to a wider range of people with disabilities. These guidelines and success criteria are organized around four principles (perceivable, operable, understandable, and robust), which lay the foundation necessary for anyone to access and use Web content.

To meet those accessibility requirements in accordance with the WCAG, a suite of evaluation tools exists on the Web. It helps authors make their Web content more accessible to individuals with disabilities. Automated tools such as Achecker or Wave can identify many WCAG errors, but also facilitate human evaluation of Web content. Such tools will be leveraged in the LEAD development process to improve or rectify accessibility errors and make the produced content in compliance with WCAG guidelines.

Regarding the Digital Twin Interface design principle, the W3C has published the Web of Things (WoT) Architecture[22] that defines functional and technical requirements on the Web of Things standards that enable easy integration across Internet of Things platforms and applications. This initiative is also created to collect new IoT use cases from various domains that have been contributed by various stakeholders.

---

[19] *T. E. P. A. O. T. COUNCIL, «DIRECTIVE (EU) 2016/2102», 26 October 2016. [En ligne].* https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32016L2102&from=EN

[20] *D. (. 2. o. t. E. P. a. o. t. C. o. 2. O. 2016, «Accessibility of the websites and mobile applications of public sector bodies,» 05 June 2018. [En ligne].https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=uriserv:OJ.L_.2016.327.01.0001.01.ENG.*

[21] *W3C, «Web Content Accessibility Guidelines, » 05 June 2018. [En ligne].https://www.w3.org/TR/WCAG21/*

[22] *W. I. Group, «Web of Things (WoT): Use Cases and Requirements, » 18 May 2021. [En ligne]. Available: https://www.w3.org/TR/wot-usecases/. [Accès le 2021].*

Several specific domains have been proposed including a Digital Twin use case that is comparable to the objectives of LEAD project and enables what-if analysis to easily interact with models to simulate unique machine conditions and perform what-if analysis using well-designed interfaces.

To list some common principles described in the proposed WoT Architecture, W3C recommends enabling mutual interworking of different ecosystems using Web technology and should be based on the Web architecture using RESTful APIs. Regarding the description mechanism of the twins and related models, the recommendation given in the WoT Architecture is to support a common description mechanism which enables describing things and their functions and be able to exchange this description using a format which is commonly used in the Web.

In the context of a Digital Twins system, it needs to generate programming interfaces internally based on metadata (descriptions), and to represent virtual devices by using those programming interfaces[23]. A twin must produce a description for the virtual device and make it externally available. Identifiers of virtual devices need to be newly assigned, therefore, are different from the original devices. This makes sure that virtual devices and the original devices are clearly recognized as separate entities. Regarding the transport, security mechanisms and settings of the virtual devices, it can be different from the original devices if necessary. Virtual devices are required to have descriptions provided either directly by the twin or to have them available at external locations. In either case it is required to make the descriptions available so that other components can find and use the devices associated with them.

In our work carried out in tasks 2.2, 2.3 and 2.4, we aim to comply at best with the recommendations of the W3C regarding the usage of Web technologies and Restful APIs as it is already presented in our first deliverables D2.1 Technical Requirements – Solution Architecture. A JSON-based description of the model library and the created virtual twins enables to exchange this description using a format that is an open standard and commonly used in the Web. Finally, thanks to the usage of an open-source framework such as Eclipse DITTO, it enables us to follow the guidelines provided by W3C for creating and managing digital twins in the IoT for the Context Entity Manager component of the LEAD Platform.

Finally, to analyse, display key performance indicators or monitor the health of a system or specific process for different users and stakeholders, information management tools such as data dashboards are of great value. They are customizable tools enabling them to meet the specific needs of exposing valuable information. Behind the scenes, a dashboard can connect to different files, attachments, services, and APIs, but on the surface displays all this data in the form of tables, line charts, bar charts and gauges. A data dashboard is the most efficient way to track multiple data sources because it provides a central location to monitor and analyse performance or real-time data. Several open-source tools exist in the market such as Kibana or Grafana that are powerful Frameworks. The development of the LEAD Dashboard (end-user portal side) will leverage this kind of solution to monitor and visualize digital twin data, support the integration with different data sources and create a dashboard that supports a variety of graphs and visualizations for different users and stakeholders.

[23] *David J. C. Mackay, Information Theory, Inference, and Learning Algorithms, CUP, 2005, ISBN: 0521642981,9780521642989*

4.2. User Requirements

To design a user interface that would provide the functionality and meet the expectations of the users, feedback from the project partners has been taken to form a set of user requirements that would help defining the dashboards.

The main guideline for the design is that the dashboard needs have the graphical elements necessary so that the user can compute and optimize KPIs for the use cases of a LL. This should be done by testing different scenarios and considering many combinations of parameters.

Therefore, based on the existing models, the available KPIs must be presented to the user as a drop-down list. Based on the user selection, the model selection interface must be enabled, and a drop-down list should display the models that calculate the selected KPI. The graphical widgets for the models should be dynamic and contain elements that reflect the models' metadata enabling the seamless employment of a wide variety of different models. The user should be then given the capability to fill the model's parameters and data sources through a set of appropriate inputs. The model selection interface should provide the option to add further models in the pipeline and have their input linked to the output of the previous model based on the user's selection while providing a comprehensive representation of the process.

Furthermore, a user should be able to register a new context entity/device providing their descriptions and functionalities. Entities and devices can also be modified and deleted from the system along with their configurations. Similarly, the user should be able to add, modify and remove models, their descriptions, and parameters. The simulation could be automatically deducted based on the selected models in the execution pipeline but the option to provide the user with an interface for simulation management is still discussed. Finally, a graphical interface for basic CRUD actions on what-if scenarios should be available to the user. The integration of updated versions of scenarios and models should be also supported and available to the user for selection.

The system should also be able to provide suggestions to the user in different use cases. The user should be able to query the system for appropriate models based on expected model outputs. Similarly, the system should suggest models that can be executed based on provided inputs. The interfaces should support the decision-making process, which implies the need for a comprehensive visualisation of KPIs metrics.

The LEAD user interfaces and dashboards will be designed and implemented considering all the above requirements that will be further elaborated as the project evolves.

4.3. User Journey - Scenario Creation

In this section, we provide a user journey related to the creation of a scenario as designed for the LEAD digital twin platform. To execute this model, model inputs should be added based on the metamodel JSON file of the COPERT [24] model. Two main inputs for the COPERT model are the number of vehicles needed and the total delivery distances per vehicle. To find the optimal/minimum

[24] *https://www.emisia.com/utilities/copert/*

number of vehicles needed and minimize the delivery distances, the Echelon[25] model for optimizing delivery routing is needed. The user defines the what-if scenario and the expected outputs to the interfaces. The suggested model is the Echelon model that is also integrated in Model Library and its JSON metamodel file is used to connect the model library to the DDDAS component of the LEAD platform. The echelon model outputs are the number needed and the delivery distances per delivery route. To compute the total delivery distance per vehicle more data processing is needed. A transformation function is implemented to compute the delivery distances per vehicle and then provide them as inputs to the COPERT together with other inputs such as vehicle specifications that will be provided by the user through the interface. After having set all the above parameters, the user can run the scenario by pressing the corresponding button. The scenario execution is scheduled for execution by the task scheduler. The user can observe the progress and the logs of the scenario execution from the task scheduler's web interface. Finally, he receives notifications regarding the scenario execution status and can access a history of previously executed scenarios from a dedicated view. LEAD LL scenarios are currently under development (WP3) and the next version of this deliverable (D2.4) will present representative scenarios creation and user journeys for each LL.

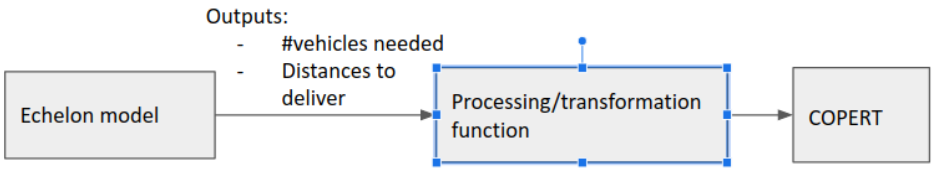

Figure 8 Example Scenario

A user journey of a scenario creation using two models and their connections is presented in the following paragraphs. The aim of this example scenario (Figure 8) is to calculate the optimal ecological KPIs and specifically the GHG emission that is a function of the number of vehicles and the total delivery distances.

Once a user is logged into the LEAD platform, a drop-down list enables the user to choose the ecological, economical, and/or social KPI that he would like to optimize, such as GHG emission, delivery cost or delivery time. The user selects the KPI related to the optimisation of the GHG emission, and the system provides a list of the models that calculate the selected KPI along with their brief description.

The chosen model is the COPERT, which is an EU standard vehicle emissions calculator. The system retrieves the meta data of the model from the Model Library and presents to the user the input fields for the model's parameters and data sources. Two main inputs for the COPERT model are the number of vehicles needed and the total delivery distances per vehicle.

To provide these inputs to the COPERT model, the system suggests to the user the echelon model for optimal delivery routing. The model finds the optimal/minimum number of vehicles needed and minimizes the delivery distances for each route. The model's meta data are also provided by the

[25] *Teodor Gabriel Crainic, Nicoletta Ricciardi, and Giovanni Storchi. 2009. Models for Evaluating and Planning City Logistics Systems. Transportation Science 43, 4 (November 2009), 432–454. DOI:https://doi.org/10.1287/trsc.1090.0279*

Models Library component. To compute the total delivery distance per vehicle further data processing is needed. A transformation function is implemented to compute the delivery distances per vehicle and then provide them as inputs to the COPERT together with other inputs such as vehicle specifications that will be provided by the user through the interface.

The user then defines the what-if scenario, adds the necessary model parameters, and selects the input data to the interfaces. He then starts the scenario execution by pressing the corresponding button.

The execution pipeline is managed by the task scheduler and can be monitored through the scheduler's web interface. The user is also notified regarding the scenario execution status and can access a history of previously executed scenarios from a dedicated view.

LEAD LL scenarios are currently under development (WP3) and the next version of this deliverable (D2.4) will present representative user journeys for each LL.

4.4. Dashboard and Interfaces Further Work

A dashboard is a visual display that shows a combination of elements (menus, buttons, graphs etc) to clearly display information to the users and facilitate their understanding. Unlike visual analytics (e.g., the VA-tools), dashboards focus more on presenting what is known (i.e., the results) rather than enabling users to reveal unknowns by means of analytical inference.

The LEAD Digital Twin platform user interface will be implemented using web technologies and can be therefore accessed from web browsers maximizing accessibility by the project partners. It will act as a strategic dashboard providing an overview of (both historical and real-time) aggregated statistics of urban logistics data (e.g., traffic data, delivery data) as well as a display of the analytical results generated from the execution of scenarios.

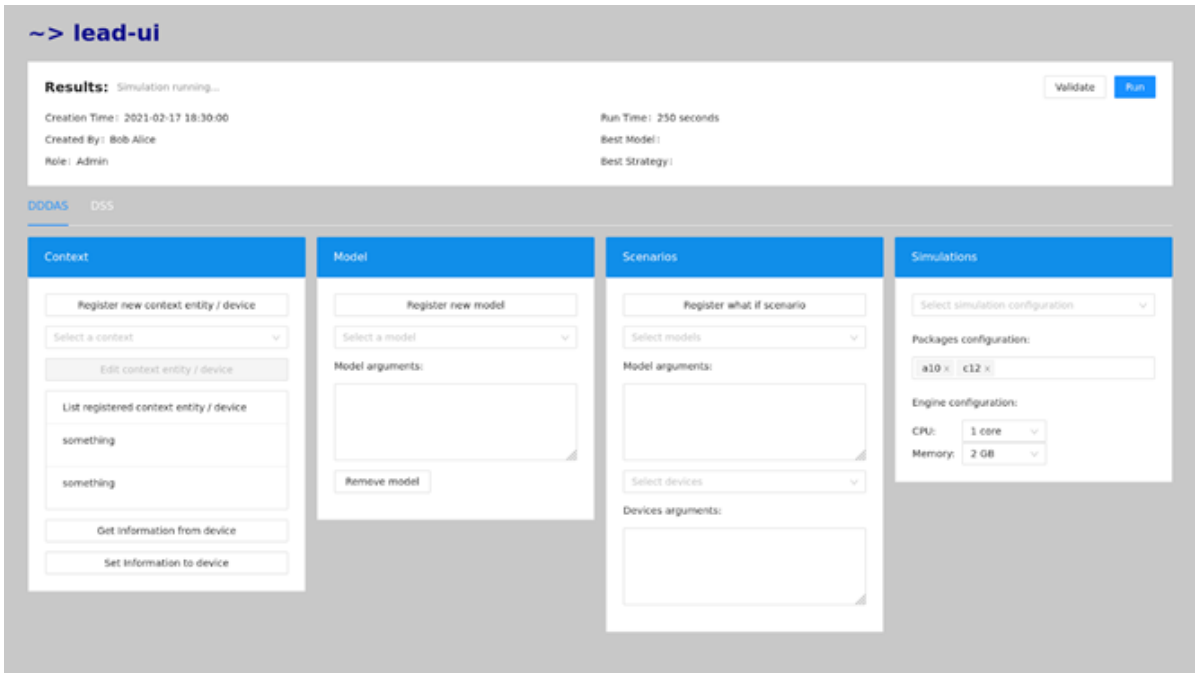

Figure 9 DDDAS Dashboard mock-up

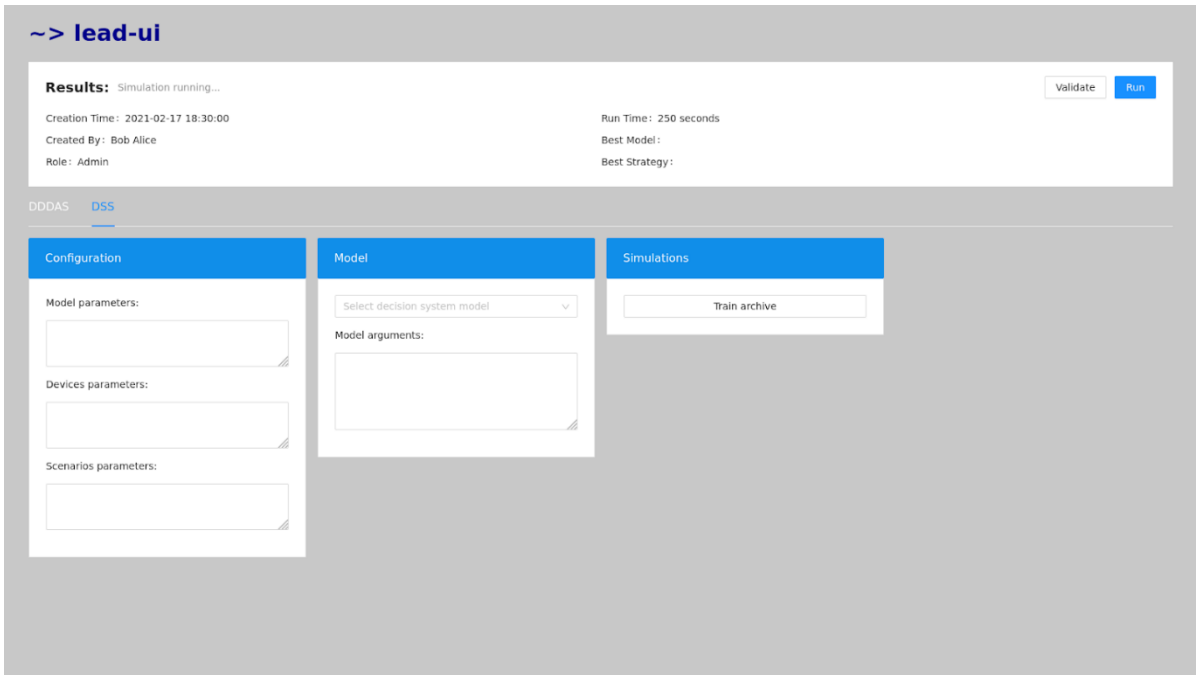

Figure 10 DSS Dashboard mock-up

The design of the dashboard also takes data privacy issues as one of the major considerations, to guarantee its compliance with GDPR. All the adopted variables and results are statistics of aggregated data, being displayed in geographic coordinates and temporal (e.g., each hour, day, or month) resolutions, making it impossible for individual traces to be re-identified. In addition, all project partners that access the dashboards have signed agreements adherent to the data privacy policy, providing anonymization data to the platform in accordance with GDPR.

This dashboard will provide simple interactions, allowing users to configure profiles, select KPI and models, define scenarios, explore the data (and the analytical results) and start a simulation. The dashboard interface is composed of a set of dynamic widgets for displaying different types of statistical variables and aggregated variables (e.g., numerical, spatial, and temporal variables). The dashboard is currently under development in close cooperation with LL stakeholders to capture and configure customized domain variables. This section is an early overview of the intended work and early mock-ups can be seen in Figure 9 and Figure 10; details are to be provided in deliverable D2.4.

5. Conclusions

LEAD aims to develop a digital twin platform to enable the composition of replicas of urban logistics networks and encompass last mile operation processes, key actors, and interrelationships. The deliverable D2.3 has designed and developed the technological core of the LEAD Digital Twin Platform, focusing on decision support system and the APIs for selecting the best scenarios for urban logistics network and last mile delivery operations.

The LEAD platform supports scalable and reliable data distribution in a secure manner. The LEAD data infrastructure consists of the Kafka subsystem for real-time data and both document-based and relational database management systems are offered for further data storage. LL data definitions include fleet information, facilities, service details, delivery areas, parcel and delivery information, order and store information that will be used to model urban logistics operations and optimize KPIs.

The LEAD platform ensures data security providing its services to authenticated users over HTTPS using its SSL certificate. Secure protocols (e.g., FTPS) are also used for data collection from other resources.

Furthermore, the LEAD platform architecture is presented, and the inter-module communication is defined. Its modular approach provides the flexibility needed by the project and enforces the LEAD platform components' reusability.

The technical specifications for the DSS are also presented focusing on functional and non-functional requirements that are crucial to provide a decision system that conforms to the needs of the project's partners. The DSS outlook regarding the development of its components, its connections and technologies is also discussed. To meet the objectives above, we will use technologies such as MongoDB for the storage management, restful APIs for DSS interfaces, and techniques such as Bayesian inference and machine learning will be considered for the decision system.

The LEAD platform user interfaces will follow design principles and frameworks according to the WCAG. The Web of Things Architecture provides also useful insights and ideas for a functional and successful design of a DT user interface. A few initial mock-ups of the DDDAS and DSS have been developed reflecting the basic needs of the users regarding the selection of KPIs (e.g., GHG emissions, delivery cost), the definition of the model sequence and models' parameters and the scenario execution. To elaborate on the user interaction with the dashboard a user journey regarding a scenario creation involving two models has been presented.

The LEAD platform is still in its first steps and the design presented here will be enhanced and expanded based on iterative discussions with the project's partners. Future developments, platform features and design modifications will be discussed in future versions of this deliverable.

6. References

1. A Requirements Driven Digital Twin Framework: Specification and Opportunities, June 2020IEEE Access PP(99):1-1, DOI:10.1109/ACCESS.2020.3000437
2. https://www.cerema.fr/en
3. https://openrouteservice.org/
4. https://swagger.io/specification/
5. D2.1 – LEAD Solution Architecture
6. Richard Mcelreath, Statistical Rethinking: A Bayesian Course with Examples in R and STAN, CRC Press, 2020, ISBN: 036713991X, 9780367139919
7. *https://cloudify.co/*
8. *https://www.openstack.org/*
9. *https://www.cloudfoundry.org/*
10. *https://cloudstack.apache.org/*
11. *https://www.linux-kvm.org/page/Main_Page*
12. *https://www.docker.com/*
13. *https://kubernetes.io/*
14. *https://www.ansible.com/*
15. *https://slurm.schedmd.com/documentation.html*
16. *https://www.linux-kvm.org/page/Main_Page*
17. https://grafana.com/grafana/dashboards
18. Annex B, D2.1" Technical Requirements – Solution Architecture", LEAD project.
19. T. E. P. A. O. T. COUNCIL, «DIRECTIVE (EU) 2016/2102», 26 October 2016. [En ligne]. https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32016L2102&from=EN
20. D. (. 2. o. t. E. P. a. o. t. C. o. 2. O. 2016, «Accessibility of the websites and mobile applications of public sector bodies,» 05 June 2018. [En ligne].https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=uriserv:OJ.L_.2016.327.01.0001.01.ENG.
21. W3C, «Web Content Accessibility Guidelines, » 05 June 2018. [En ligne].https://www.w3.org/TR/WCAG21/
22. W. I. Group, «Web of Things (WoT): Use Cases and Requirements, » 18 May 2021. [En ligne]. Available: https://www.w3.org/TR/wot-usecases/. [Accès le 2021].
23. David J. C. Mackay, Information Theory, Inference, and Learning Algorithms, CUP, 2005, ISBN: 0521642981,9780521642989