



H2020-LC-SC3-EE-2019

EUROPEAN COMMISSION

European Climate, Infrastructure and Environment Executive Agency

Grant agreement no. 893857



frESCO

New business models for innovative energy services bundles for residential consumers

Project acronym	frESCO
Full title	New business models for innovative energy service bundles for residential consumers
Grant agreement number	893857
Programme	H2020-EU.3.3.1. - Reducing energy consumption and carbon footprint by smart and sustainable use
Topic identifier	LC-SC3-EE-13-2018-2019-2020 - Enabling next-generation of smart energy services valorising energy efficiency and flexibility at demand-side as energy resource
Call	H2020-LC-SC3-EE-2019
Funding scheme	IA – Innovation Action
Project duration	42 months (1 June 2020 – 30 November 2023)
Project adviser	Rebecca Kanellea - CINEA
Coordinator	CIRCE – Fundacion Circe Centro de Investigacion de Recursos y Consumos Energeticos
Consortium partners	CIRCE, S5, EI-JKU, CARTIF, UBITECH, UBE, KONCAR KET, KRK, COSMA, LCTE, VOLT, VERD, IOSA, RINA-C
Website	http://fresco-project.eu
Cordis	https://cordis.europa.eu/project/id/893857

DISCLAIMER OF WARRANTIES

This document has been prepared by frESCO project partners as an account of work carried out within the framework of the EC-GA contract no. 893857.

Neither Project Coordinator, nor any signatory party of frESCO Project Consortium Agreement, nor any person acting on behalf of any of them:

- (a) makes any warranty or representation whatsoever, expressed or implied,
 - (i). with respect to the use of any information, apparatus, method, process, or similar item disclosed in this document, including merchantability and fitness for a particular purpose, or
 - (ii). that such use does not infringe on or interfere with privately owned rights, including any party's intellectual property, or
 - (iii). that this document is suitable to any particular user's circumstance; or
- (b) assumes responsibility for any damages or other liability whatsoever (including any consequential damages, even if the Project Coordinator or any representative of a signatory party of the frESCO Project Consortium Agreement has been informed of the possibility of such damages) resulting from your selection or use of this document or any information, apparatus, method, process, or similar item disclosed in this document.

ACKNOWLEDGMENT



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement n. 893857

Disclaimer: *The European Commission is not responsible for any use made of the information contained herein. The content does not necessarily reflect the opinion of the European Commission.*

Deliverable D5.8

frESCO Multi-Service Package Toolkit

Deliverable number	D5.8
Deliverable name	frESCO Multi-Service Package Toolkit
Lead beneficiary	UBITECH
Description	This report describes the interfaces and the technical requirements as well as input and output parameters of the frESCO multi-service package toolkit. This version is a resubmission including 2 new annexes.
WP	WP5
Related task(s)	T5.1, T5.2, T5.3, T5.4, T5.5
Type	Report
Dissemination level	Public
Due Delivery date	31.01.2023
Main author	Kostas Mylonas (UBITECH)
Contributors	CIRCE, CARTIF, S5

Document history

Version	Date	Changes	Author
V0.1 – first draft of ToC	29.09.2022	First version of Table of Contents	UBITECH
V0.2 – second draft of ToC	29.09.2022	Second version of Table of Contents	UBITECH
V1 – first draft	31.01.2023	First draft of the deliverable	UBITECH, CIRCE, CARTIF, S5
V1 – revised CIRCE	02.02.2023	Revision by CIRCE	CIRCE
V1 – revised CARTIF	02.02.2023	Revision by CARTIF	CARTIF
V2 – consolidated version	06.02.2023	Editing of the deliverable, after the feedback provided from the review process	UBITECH
V3 – Final version	23.02.2023	Final draft of the deliverable	UBITECH
Submission	23.02.2023	Submission to EC	CIRCE

V4 – annex first draft	07.07.2023	First draft of the annex of the deliverable	UBITECH, CIRCE, CARTIF, S5
V4 – revised CIRCE	11.07.2023	Revision by CIRCE	CIRCE
V4 – revised CARTIF	18.07.2023	Revision by CARTIF	CARTIF
V5 – consolidated version	18.07.2023	Editing of the deliverable, after the feedback provided from the review process	UBITECH
V6	02.08.2023	Second version of deliverable ready with two new annexes included	CIRCE

ABBREVIATIONS

Abbreviation	Name
ANN	Artificial Neural Network
API	Application Programming Interfaces
BDP	Big Data Platform
CIM	Common Information Model
DER	Distributed Energy Resource
DHW	Domestic Hot Water
DOA	Description of the Action
DR	Demand Response
EE	Energy efficiency and self-consumption optimisation services
EMS	Energy Management System
ESCO	Energy Service Company
FL	Flexibility services
GBR	Gradient Boosting Regression
GDM	Global Demand Manager
GUI	Graphical User Interface
GUI	Graphical User Interface
HVAC	Heating Ventilation and Air Conditioning
KPI	Key Performance Indicator
LDM	Local Demand Manager
LP	Linear Program
MIQP	Mixed-Integer Quadratic Programming
NE	Non-energy services
PV	Photovoltaic
RT	Sensing and smart equipment retrofitting
SRI	Smart Readiness Indicator
TSO	Transmission System Operator
UC	Use Case
VPP	Virtual Power Plant
WP	Work package

TABLE OF CONTENTS

Abbreviations	5
Executive Summary	9
2 Introduction.....	1
2.1 Purpose and Target Group	2
2.2 Scope of the Document.....	3
2.3 Structure of the Document	3
3 Multi-Service Package Toolkit for Service Providers Overview.....	3
3.1 Complete frESCO Architecture Overview.....	3
3.2 Local Demand Manager Modules Overview	5
3.2.2 Asset Energy Performance Forecasting/Monitoring Module	5
3.2.2.1 Introduction.....	5
3.2.2.2 Input Data and Technology Stack.....	6
3.2.2.3 Methodology and Requirements Overview based on frESCO Use Cases	9
3.2.2.4 Outputs.....	12
3.2.2.5 Relationship with other frESCO Modules.....	13
3.2.3 Personalized Energy Analytics Module	13
3.2.3.1 Introduction.....	13
3.2.3.2 Input Data and Technology Stack.....	13
3.2.3.3 Methodology and Requirements Overview based on frESCO Use Cases	15
3.2.3.4 Outputs.....	16
3.2.3.5 Relationship with other frESCO Modules.....	17
3.2.4 Human Centric Automation Module	17
3.2.4.1 Introduction.....	17
3.2.4.2 Input Data and Technology Stack.....	18
3.2.4.3 Methodology and Requirements Overview based on frESCO Use Cases	19
3.2.4.4 Outputs.....	20
3.2.4.5 Relationship with other frESCO Modules.....	20
3.3 Global Demand Manager Modules Overview	21
3.3.2 Energy Management Analytics Module	21
3.3.2.1 Introduction.....	21
3.3.2.2 Input Data and Technology Stack.....	22
3.3.2.3 Methodology and Requirements Overview based on frESCO Use Cases	23
3.3.2.4 Outputs.....	25
3.3.2.5 Relationship with other frESCO Modules.....	25
3.3.3 Self-Consumption Optimization Module.....	25
3.3.3.1 Introduction.....	25
3.3.3.2 Input Data and Technology Stack.....	27
3.3.3.3 Methodology and Requirements Overview based on frESCO Use Cases	27
3.3.3.4 Outputs.....	29
3.3.3.5 Relationship with other frESCO Modules.....	29
3.3.4 Smart Contract Monitoring/Handling Module.....	30
3.3.4.1 Introduction.....	30
3.3.4.2 Input Data and Technology Stack.....	30
3.3.4.3 Methodology and Requirements Overview based on frESCO Use Cases	34
3.3.4.4 Outputs.....	36
3.3.4.5 Relationship with other frESCO Modules.....	39

3.3.5	Flexibility Analytics Module.....	41
3.3.5.1	Introduction.....	41
3.3.5.2	Input Data and Technology Stack.....	42
3.3.5.3	Methodology and Requirements Overview based on frESCO Use Cases	43
3.3.5.4	Outputs.....	46
3.3.5.5	Relationship with other frESCO Modules.....	46
3.3.6	VPP Optimal Configuration Module	46
3.3.6.1	Introduction.....	46
3.3.6.2	Input Data and Technology Stack.....	48
3.3.6.3	Methodology and Requirements Overview based on frESCO Use Cases	48
3.3.6.4	Outputs.....	50
3.3.6.5	Relationship with other frESCO Modules.....	50
4	UIs overview	51
4.1	UI for ESCOs.....	51
4.1.2	Energy Management Analytics and Self-Consumption Optimization Dashboard.....	51
4.1.3	Login	51
4.1.4	Clustering.....	52
4.1.5	Self-Consumption	52
4.2	UI for Aggregators	53
4.2.2	Flexibility Analytics and Optimal VPP configuration Dashboard.....	53
4.2.2.1	Login	53
4.2.2.2	DR Events.....	54
4.2.2.3	Flexibility Results	55
4.2.2.4	Flexibility Analytics	56
4.2.3	Smart Contract Monitoring/Handling Dashboard.....	57
4.2.3.1	Login	57
4.2.3.2	Assets.....	57
4.2.3.3	Contracts	58
4.3	UI for Prosumers/Consumers.....	62
4.3.2	LDM's Dashboard	62
4.3.2.1	Login	63
4.3.2.2	Monitored Data and Savings	63
4.3.2.3	Recommendations.....	64
4.3.2.4	Automations.....	65
4.3.2.5	Notifications	67
4.3.2.6	Generation and Demand	69
4.3.2.7	Settings.....	73
4.3.2.8	Help	75
4.3.3	Smart contract monitoring/handling Dashboard	75
4.3.3.1	Login	75
4.3.3.2	Prosumer's Perspective	75
5	Future Development & Improvements	77
6	Conclusions.....	78
	References.....	79
	ANNEX A: Data Integration and Modules Updates	81
A.1	Local Demand Manager Modules	81
A.1.1	Asset Energy Performance Forecasting/Monitoring Module	81
A.1.1.1	Data integration actions.....	81
A.1.1.2	Module updates	81
A.1.2	Personalized Energy Analytics Module	84
A.1.2.1	Data integration actions.....	84

A.1.2.2 Module updates	86
A.1.3 Human Centric Automation Module.....	88
A.1.3.1 Data integration actions.....	88
A.1.3.2 Module updates	89
A.2 Global Demand Manager Modules	93
A.2.1 Energy Management Analytics Module	93
A.2.1.1 Data integration actions.....	93
A.2.1.2 Module updates	95
A.2.2 Self-Consumption Optimization Module	97
A.2.2.1 Data integration actions.....	97
A.2.2.2 Module updates	99
A.2.3 Smart Contract Monitoring/Handling Module.....	99
A.2.3.1 Data integration actions.....	99
A.2.3.2 Module updates	103
A.2.4 Flexibility Analytics Module	104
A.2.4.1 Data integration actions.....	104
A.2.4.2 Module updates	105
A.2.5 VPP Optimal Configuration Module.....	107
A.2.5.1 Data integration actions.....	107
A.2.5.2 Module updates	110
ANNEX B: Energy Solvers Comparison and Analysis	112
B.1 Introduction.....	112
B.2 Methodology	112
B.2.1 Optimization solvers	112
B.2.2 KPIs	114
B.2.3 Data sets.....	114
B.3 Implementation.....	116
B.3.1 Results	118
B.4 Conclusions.....	121

EXECUTIVE SUMMARY

Deliverable D5.8 describes the interfaces and the technical requirements as well as input and output parameters of the frESCO multi-service package toolkit and together with the deliverable D5.7 act as the final product of the modules of WP5. Using the frESCO conceptual design, as presented in D2.5 “Report on the frESCO conceptual architecture” [1], all modules and their corresponding features are specified in detail in this document, as well as which functional and end-user requirements they satisfy.

Deliverable D5.8 summarizes the deliverables D5.1 [2], D5.2 [3], D5.3 [4], D5.4 [5], D5.5 [6] and D5.6 [7], which presented the different modules of the frESCO multiservice package toolkit, and it provides the final architecture and the interconnection between them. The current deliverable explains the final form, the input and output and the technology stack of the modules, as well as the methodology used for each module. To that end, appropriate screenshots of the dashboards that support the different modules are provided to show the functionalities that are included in the final release of the frESCO multiservice package toolkit. The most important features are thoroughly displayed and discussed, and they are complemented by indicative guidelines, so that the users of the frESCO multiservice package toolkit have a clear picture of what is made available to them.

Deliverable D5.8 emphasizes on the results of deliverables D5.1, D5.2, D5.3, D5.4, D5.5 and D5.6 and focuses on summarizing the final version of the modules within the frESCO multiservice package toolkit and the function and end-user requirements that each module has implemented. The involved stakeholders and users, such as ESCOs, aggregators and prosumers can get a basic understanding of the benefits they can enjoy through the developed dashboards.

This document, prepared in January 2023, has been completed in July 2023 with an Annex A that updates the multi-service package toolkit with additional developments for data integration from the demo sites and fine tuning of the modules carried out during the last period. In Annex B, the conclusions of an energy management solver comparison are presented as a justification of frESCO solvers’ final path forward.

As a final note, we have to mention that the referenced documents D5.2, D5.3, D5.4, D5.5 and D5.6 are confidential, but this new deliverable D5.8 is not disclosing any piece of confidentiality.

2 INTRODUCTION

This report describes the interfaces and the technical requirements as well as input and output parameters of the final version of the frESCO multi-service package toolkit. It is the outcome of the five tasks in WP5 – multi-service package toolkit for service providers. D5.8 is related to other deliverables, tasks and WPs as explained in Table 1.

Deliverable/ Task/ WP	Title	Comment
D2.5 / T2.5	Report on the frESCO conceptual architecture / Specifications, Architecture Design and Communication Interfaces	D2.5 / T2.5 introduced the whole frESCO platform structure, which includes T5.3 and T5.4 modules
D3.1 / T3.1	Definition of the novel energy services for residential consumers / Design of the novel energy services for residential consumers	All the services of the frESCO project have been designed in D3.1 / T3.1
D5.1 / T5.1 / T5.2 / T5.3	Monitoring, forecasting, energy management analytics, flexibility analytics and optimization mechanisms / Advanced Performance Monitoring/Forecasting Module for Generation/Storage/Demand Assets / Energy Management Analytics and Self-Consumption Optimization Tool for ESCOs / Advanced Flexibility Analytics and Optimal VPP configuration tool for Consumer-Centric Demand Response Optimization	D5.1 / T5.1 / T5.2 / T5.3 features will be implemented in the GUI developed in T5.4
D5.3 / T5.4	Algorithms for Human-Centric automation tool	D5.3 introduces the algorithms that operate two of the LDM modules, which are the Personalized Energy Analytics and the Human-Centric Automation.

Deliverable/ Task/ WP	Title	Comment
D5.4 / T5.4	Release of the Human-Centric automation module	The algorithms presented in D5.3 were implemented in a tool, which was be delivered in D5.4
D5.5 / T5.5	Definition of blockchain mechanisms enabling smart contract monitoring	D5.5 provided the specifications for the Smart Contract monitoring and management module.
D5.6 / T5.5	Release of the module for Smart Contract monitoring and management	D5.6 outlines the mechanisms and tools that have been developed to enable local flexibility to participate in flexibility market transactions as part of aggregator services to overlay electricity grid optimization.
D5.7 / D5.8	Fine-tuning and final release of frESCO multi-service package toolkit / frESCO multi-service package toolkit	The final integrated version of the services and algorithms are reported in D5.7 and D5.8

Table 1: Relation of D5.8 to other frESCO deliverables, tasks, and WPs.

2.1 Purpose and Target Group

The current deliverable, D5.8 "frESCO Multi-Service Package Toolkit" builds on D5.1, D5.2, D5.3, D5.4, D5.5 and D5.6 and provides the final version of the modules that make up the frESCO multiservice package toolkit. It gives an overview of the methodology and technology stack used per module and the input and output parameters. It presents an extensive analysis of each module based on UI examples from the dashboards of LDMs and GDMs and it also gives an exact understanding of which requirements are satisfied by which modules.

2.2 Scope of the Document

This document constitutes a user manual for the modules of the frESCO multiservice package toolkit. It addresses the contents and the features of the modules, as well as the interfaces and the relationship between them. The toolkit's high-level architecture is also defined, as well as the technology stack that was used towards the implementation. It also explains some future steps that need to be done regarding the complete testing and the integration of the toolkit with the BDP.

2.3 Structure of the Document

The document is structured as follows. Section 2 presents the final integrated version of all the modules that constitute the frESCO multi-service package toolkit. It gives a detailed description of the final version of all the components and their respective modules, that comprise the platform solution. Special attention is given to how each module addresses the functional and the end user requirements specified in D2.5. Section 3 provides an overview of the various user interfaces implemented for the different modules of the toolkit. Section 4 identifies possible improvements and areas of future development. Finally, Section 5 gives some concluding remarks.

The original document submitted in early 2023, is complemented with an Annex A that updates the multi-service package toolkit with additional developments for data integration from the demo sites and fine tuning of the modules carried out until July 2023. Annex B shows the conclusions of an energy management solver comparison analysis made to support the frESCO solvers' final path forward decision.

3 MULTI-SERVICE PACKAGE TOOLKIT FOR SERVICE PROVIDERS OVERVIEW

3.1 Complete frESCO Architecture Overview

One of the main objectives of the frESCO project is the development of novel energy services that ESCOs and Aggregators can offer to building owners and users. Four main groups of services have been identified throughout the project as defined in D3.1 [8]: Sensing and

smart equipment retrofitting (RT), Energy efficiency and self-consumption optimization services (EE), Flexibility services (FL) and Non-energy services (NE). All of them help to increase the Smart Readiness Indicator (SRI) of buildings, as they empower users when it comes to managing and getting informed about their energy usage.

In order to develop and test all frESCO features, a complete platform architecture has been designed. There are two components in the users' side, the Local Demand Manager (LDM), which is focused on consumers/prosumers, and the Global Demand Manager (GDM), designed to help ESCOs and aggregators.

The complete frESCO architecture, followed by the detailed description of the system's components and modules, is presented in Figure 1.

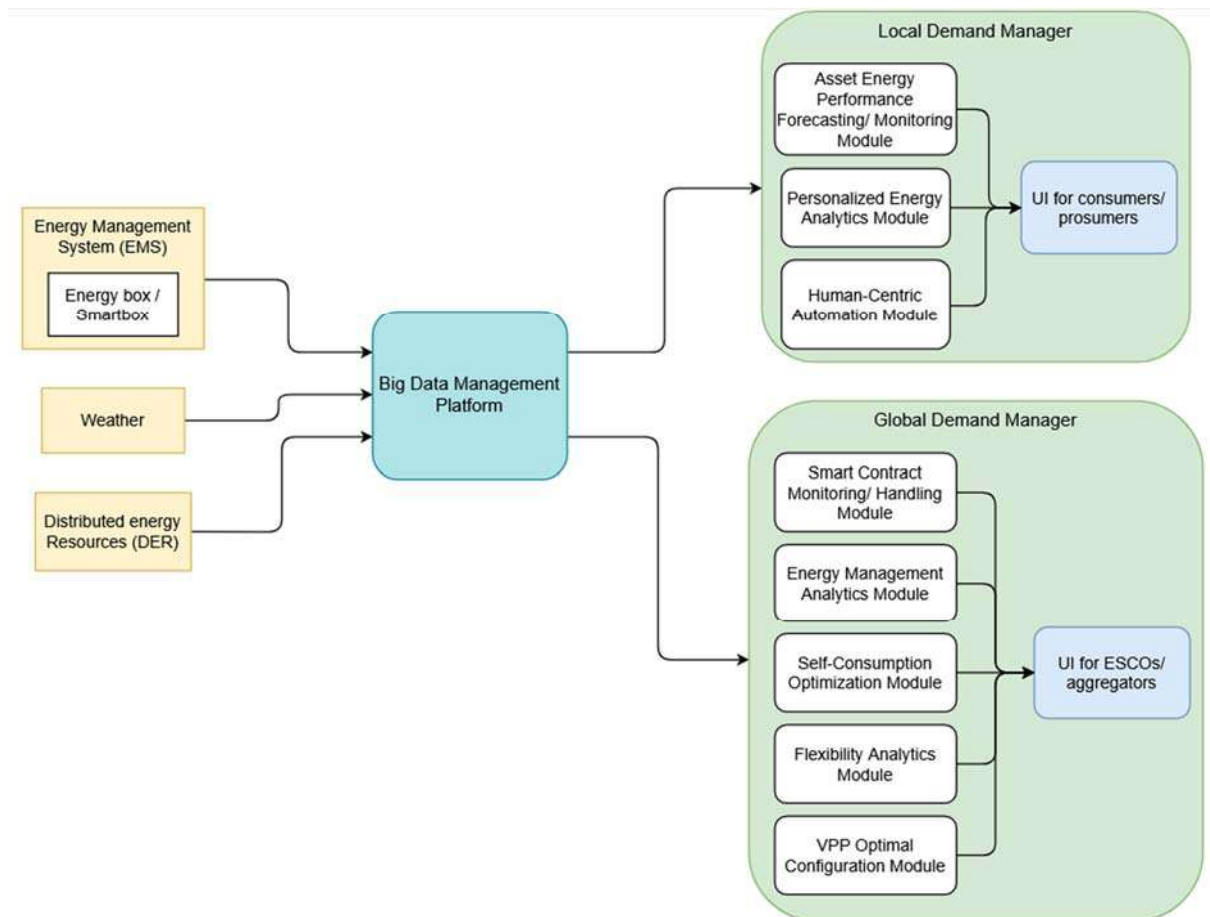


Figure 1: The complete frESCO architecture.

3.2 Local Demand Manager Modules Overview

The LDM modules include the following modules:

- Asset energy performance/forecasting monitoring module
- Personalized energy analytics module
- Human centric automation module

The results of these modules will be communicated with the prosumers with a UI as can be seen in Figure 2.

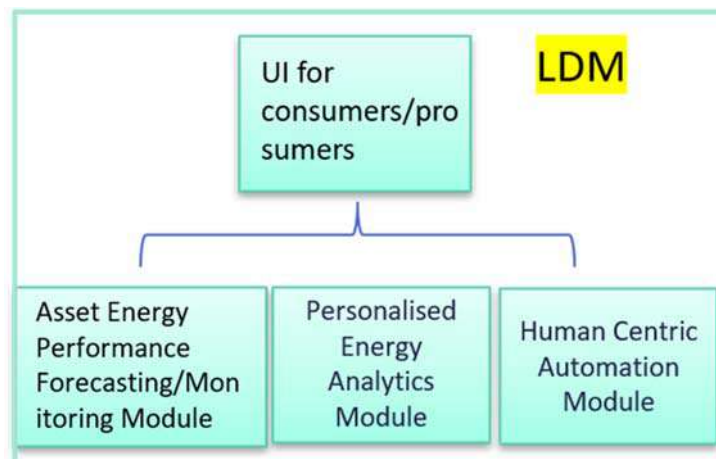


Figure 2: Modules of the Local Demand Manager.

3.2.2 Asset Energy Performance Forecasting/Monitoring Module

3.2.2.1 Introduction

This document describes the set of functionalities delivered under the Asset “Energy Performance Forecasting/ Monitoring Module” of the Local Demand Manager, according to the frESCO system architecture D2.5. The objective is to provide building residents with the required information to understand generation and demand evolution at home, and with the generation and demand forecasts in the mid and long terms. These forecasts will be a valuable insight to understand the effect of energy efficiency decisions in the long term.

Firstly, data are accumulated and stored in the Big Data Platform's Data Storage Module. Later, the data of demand and generation at the dwelling level (mainly metering data) and other related variables (such as internal and external temperatures, local irradiation, etc.) will provide the information to the generation of forecasting models on mid-term/long-term time windows and for monitoring the use of the energy in the dwelling.

From the user's point of view, the combination of monitoring and forecasting on the same tool will provide the user with the necessary information for a correct understanding of the home's energy usage. It will provide present and future tracking of the generation and demand and historical behavioural trends. In addition, it will provide knowledge about the impact of the user's decisions on the final energy balance of the house (consumption-generation).

After the first release version of the algorithms, developed based on dummy data, several improvements have been done to the algorithms. One of the main is the tuning of the algorithms based on aggregated historical data (available at the building level in the Spanish demo and the Greek demo, but not at the dwelling level), to catch the real behaviour of the assets in the site in terms of PV generation and demand in the dwelling and taking into account exogenous meteorological variables.

In this section, the Monitoring and forecasting services of the final version of the algorithms are described. Data needs, methodology and algorithms are explained for the main objectives of task 5.1 "Monitoring and forecast of generation and demand at dwelling level".

The "*final version*" will be mentioned throughout the chapter, but it should be noted that this "*final version*" is subject to modifications due to the delays suffered in the project, especially regarding the real data availability as described in the next subsection.

3.2.2.2 Input Data and Technology Stack

The whole list of desirable variables was cited in D5.1. To develop the **final version** described on this deliverable the variables cited below have been considered. These variables were selected considering the available data in the historical datasets and the influence of them over the model response considered for the training of this final version.

Considered datasets in the final version:

- Historical Generation data provided by demo partners by external sources to the platform (csv files, excel files)
- Historical demand data provided by demo partners by external sources to the platform (csv files, excel files)
- Meteorological data collected from reanalysis nodes according to the coordinates of the site [9].

Considered variables in the final version:

Data from dwellings:

- Building Energy generation (PV) (kWh, time series)
- Building Energy consumption (kWh, time series)
- Measured Indoor temperature (°C, time series) ¹
- Period of expected Occupancy (Boolean, time series).

Meteorological variables from reanalysis nodes:

- Ambient temperature (°C, time series): “2m_temperature (t2m)”
- Global horizontal radiation (GHI) (W/m², time series):
“surface_solar_radiation_downwards (ssrd)”
- Direct solar radiation on earth surface (DNIh) (W/m², time series):
“total_sky_direct_solar_radiation_at_surface (fdir)”
- Cloud coverage (%): “total_cloud_cover (tcc)”

The final data granularity has been adapted to be hourly from the cited different sources.

About the technology stack, the architecture considered to interact with the UI and the big data platform in the LDM follows this schema:

¹ According to IDEA recommendations, models have been trained with a pattern of comfortable temperatures. These values can be replaced by real measurements in a future version [10].

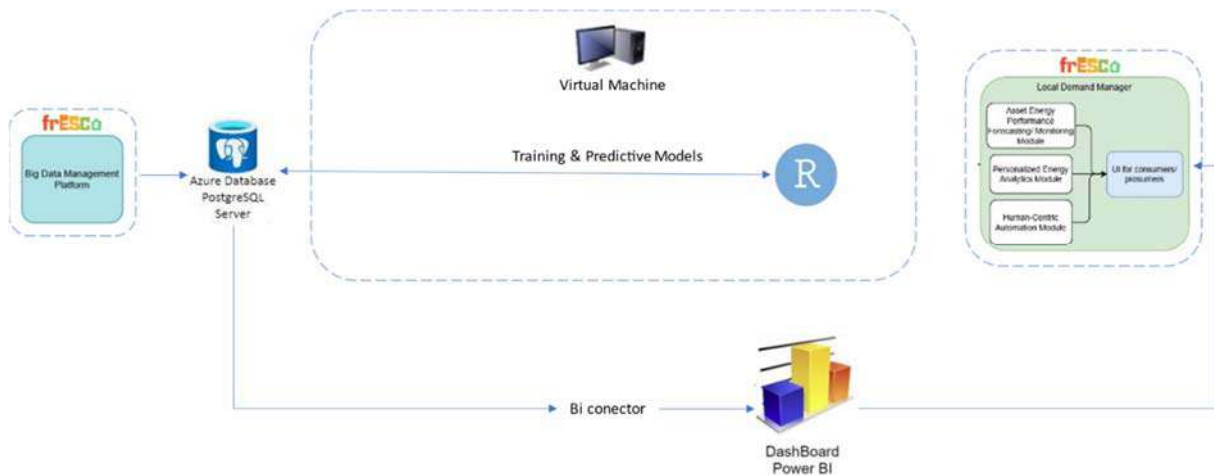


Figure 3: Architecture machine learning environment (Energy Performance Forecasting/Monitoring Module).

The final version of the developed algorithms is described in the following section.

Regarding the visualisation mock-up development, the dashboards related to the module “Asset Energy Performance Forecasting/Monitoring Module” are described in the subsection 4.3.

On this architecture, a Machine learning environment is created (Figure 3). It includes:

- Database: It stores the input data of models and writes the outputs of the algorithms. This database will feed the visualisation toolkit and the frESCO Big Data platform with the shared results needed in other modules. An Azure Database for PostgreSQL is configured for this issue.
- Dashboard environment: It will show the main results of the algorithms, and it will be connected to the UI. The Power Bi tool is connected to the database in the cloud to catch the results and the dashboards are embedded into the main User interface (see section 4).
- Machine learning models: the models are generated in R programming language and based on R and Python machine learning libraries. It runs on a virtual machine whose specifications were scaled to the data and modelling requirements.

This module is deployed on the cloud. It is delivered as a web-based environment. Analytics runs on a virtual machine in the cloud. The considered tech stack is as follows:

- Backend: Power Bi Service (back-end)
- Frontend: Power Bi service (front-end web (WFE))
- Databases: Azure Database for PostgreSQL (PostgreSQL v11, 2 virtual cores, 100 Gb)

- Analytics: R and Python programming and respective libraries like Keras, reticulate, and Tensorflow.
- Virtual machine: Windows virtual machine for Azure (D2s v3, 2vcpu, 8Gb RAM, Windows 10 pro).

This technology stack will ensure the operation of the whole system in the cloud.

3.2.2.3 Methodology and Requirements Overview based on frESCO Use Cases

This section will explain the methodology and algorithms programming details used for the monitoring and the generation and demand forecasts in the mid and long-term.

This module addresses some user requirements and functional requirements provided in D2.5. The following tables list the requirements and the implementation status ². Regarding the end user requirements, they are listed below:

Req_id	Description	Implementation status
Req_002	The system shall allow near-real-time monitoring of the energy consumption and performance and their visualisation for ESCOs and consumers/prosumers.	Partially implemented-missing training with demo sites data
Req_006	The system shall provide forecasting tools for energy efficiency assessment and verification.	Fully implemented

Table 2: End user requirements and implementation status.

About the functional requirements fulfilled by this module, they are listed below:

² In case of Req_006 and Req_025 they are considered as “Fully implemented”, but they are implemented in Spanish demo and Greek demo based on historical data at building level. In case of Croatian demo they are based on dummy data. In case on Req_002 and Req_013, they are considered as “Partially implemented” because although architecture is done and the service is active, the real-time data is not included into the platform at the moment of writing this deliverable.

Req_id	Description	Implementation status
Req_013	The system shall provide near real-time monitoring of energy demand, generation, and consumption at dwelling level	Partially implemented-missing training with demo sites data
Req_025	The system shall provide forecasts of energy demands and generations (mid-term and long-term) at dwelling level	Fully implemented

Table 3: Functional requirements and implementation status.

These requirements have been addressed and can be tested in the UI.

In the case of End-user requirement Req_002 and Functional requirement Req_013, the main graphs allow to monitor of the generation and the demand on an hourly basis. The user can select the period to be displayed and get the generation or demand estimation for the selected period. Daily curves have been provided too.

In the case of End-user requirement Req_006 and Functional requirement Req_025, machine learning models have been tested and feature selection procedures have been applied to be able to obtain long-term forecasting of generation and demand based on the typical performance of the dwelling (based on historical data from dwelling measurements and meteorological data from reanalysis). The results of these forecasts are shown in the UI on an hourly and monthly base.

The main objective of this section is to describe the developed algorithms in terms of generation and demand. They will provide prosumers with a general view of the dwelling generation and the dwelling demand. The first tasks performed by the algorithms are data collection, data standardisation, data synchronization and data storage in the database. Historical energy generation and demand data have been uploaded from CSV files at this stage³, but this task will be updated in the future. The real-time raw data will be collected from the

³ Data collected from historical data provided in Spanish demo and Greek demo.

big data platform. Necessary meteorological data from reanalysis nodes have been collected by the cited API [10].

Regarding the **monitoring algorithms**, the main calculations are focused on the aggregations (hourly, monthly) of the main influencing variables such as active power, radiation in the plane of the array, global horizontal radiation, ambient temperature or thermal difference from external temperature. Furthermore, the calculation of the typical year of PV generation and demand for each dwelling has been done, where the yearly behaviour of the dwelling is cached into an hourly dataset (8,760 hours per year). Additionally, the daily average curve has been calculated for each month in this final version. This calculation can be used as a qualitative comparison of the influence of dependent variables over PV generation and demand.

Regarding the **forecasting algorithms**, the methods tested include forecasting methods for dealing with seasonal patterns and multivariate models. After the tests carried out on the release version, an Artificial Neural Network model (ANN [11]) and a Gradient Boosting Regression Model (GBR) have been selected for the forecast. Depending on the site and the intrinsic characteristics of the generation and demand, ANN or GBR models are selected for this task. Both, the ANN and GBR can estimate the expected demand or generation for the site. As many models have been generated as there are dwellings for demand and generation.

The main steps and calculations included in the forecasting algorithms include:

- Feature selection based on Spearman [12] correlation to preselect the predictor variables of the models.
- Construction of models' architecture and tuning of parameters and hyperparameters (GBR and ANN).
- Training of models and model validation based on historical data considering the predictor variables (indoor temperature, day of week, month, ambient temperature, thermal difference between indoors and outdoors, hour, occupancy, radiation in the plane of array estimated by a model transposition [13] and total cloud covering).
- Typical year estimation of the demand and PV generation dataset. This will be used as input in the forecasting phase.

- Demand and PV generation forecast for the long-term selected period, by default 12 months ahead based on the selected predictor variables.

3.2.2.4 Outputs

The main outputs are cited below and shown through the User interface (see subsection 4.3). Regarding the **monitoring of generation and demand** at the dwelling level, these are the outputs:

Main graphics on the dashboard

- Historical evolution of the generation and the demand
 - Time evolution of generation and demand on an hourly basis.
 - Time evolution of generation and demand per month.
- Daily patterns in terms of generation and demand
 - Daily Generation and Demand curves calculation per month.

Main metrics on the dashboard:

- Sum of generation and consumption at dwelling level.
 - In the selected period by the user.
 - On a monthly basis.

About the results related to **forecasting of generation and demand** at the dwelling level (mid and long-term forecasts):

Main graphics on the dashboard

- Historical evolution of the generation and the demand forecasting
 - Time evolution of forecasted generation and forecasted demand on an hourly basis.
 - Time evolution of forecasted generation and forecasted demand per month.

Main metrics on the dashboard:

- A monthly forecast of the generation and the demand according to historical working conditions compared to the measured historical ones.

3.2.2.5 Relationship with other frESCO Modules

The Asset Energy Performance Forecasting/Monitoring Module, Personalized Energy Analytics Module and Human Centric Automation Module are part of the *Local Demand Manager* (LDM) component.

“Energy Performance Forecasting/Monitoring Module” takes the data inputs from the *Big Data Management Platform* where all user data are stored. After data processing, the results will be stored in the cloud. The module will communicate with the final users through the *UI for consumers/prosumers* (GUI), which summarises all the LDM information.

3.2.3 Personalized Energy Analytics Module

3.2.3.1 Introduction

The Personalized Energy Analytics module focuses on the implementation of those frESCO services giving users recommendations that, if followed, will carry along energy an economic savings. This module implements eleven services, including three non-energy services that only consider users’ comfort.

3.2.3.2 Input Data and Technology Stack

Services have different inputs. In **Table 4** the specific data is explained, but as a summary, the used data is monitored data on HVAC and all dwelling energy consumption, data from temperature, humidity and air quality sensors and PV energy generation and user comfort information.

Most services’ algorithms are coded and tested, but not implemented as data was not available yet. Some algorithms are still on work, as are new services that surged recently due to demo specific needs.

Technologies used for services implementation are listed below:

- API: API is deployed in Python using FastAPI framework. API is responsible of exposing data to frontend (GUI) and other services.
- Schedule Tasks: A Python framework called Celery is used to manage the Schedule tasks for update services information after a defined period of time.
- Redis Database: An In-memory database to store intermediate information of Celery Tasks.
- Postgres Database: A relational database for storing the information data of the services.
- BDP API: Exposed API for interacting with the Big Data Platform, to store and recover information data of the services.

	Services' short description	Input data
Service 1	Recommendation on the usage of non-automatable devices (example: washing machine, dish washer)	Expected PV generation Expected energy consumption Battery SOC and capacity Energy prices
Service 1.2	When PV is available and generation is higher than consumption, the surplus energy will be shown, and it would be recommended to shift loads to those hours.	PV solar generation Energy consumption
Service 2	When CO2 levels are above the indicated threshold (UP1) a notification will be sent	CO2 concentration level Recommended CO2 concentration threshold
Service 6	At night, if the HVAC is ON, a notification will be sent suggesting turning it OFF	HVAC consumed energy
Service 14	When temperature is outside the indicated values (UP3), a notification is sent	Room temperature User's comfort temperature
Service 14 - hotel	When a room is not rented and temperature is outside the indicated boundaries (UP3), user will receive a notification informing of the situation	Room temperature User's comfort temperature Hotel room booking information
Service 15	When humidity levels are above the specified threshold (UP5), a notification will be sent	Room humidity level Recommended humidity level

Service 15 - hotel	When humidity levels are above the specified threshold (UP5), and room is not rented, a notification will be sent.	Room humidity level Recommended humidity level Hotel room booking information
Service 17 - hotel	When temperature is outside of the specified boundaries (UP3) plus a small margin (UP6), and room is rented, a notification is sent	Room temperature User's comfort temperature Hotel room booking information
Service 18 - hotel	When battery SOC is under 50% and PV generation is under 70% of the total energy consumption, a notification is sent	Battery SOC PV generation Energy consumption
Service 20 - Croatia	When a water leak is detected, a notification is sent	Water leak sensor information

Table 4: Personalized Energy Analytics Module services and input data.

3.2.3.3 Methodology and Requirements Overview based on frESCO Use Cases

The working methodology is as follows: sensors and meters will compile the necessary information that will be stored in the BDP. The services will read the data and calculate its respective outputs. These outputs will be shown in the GUI.

Within task T2.5 Requirements 5 and 2 were elicited (the requirements can be seen in the following table).

Req_id	Description	Implementation Status
Req_002	The system shall allow near-real-time monitoring of the energy consumption and performance and their visualisation for ESCOs and consumers/prosumers.	Fully implemented
Req_005	The system shall allow recommendations for energy usage to promote user participation through a reward process.	Fully implemented

Table 5: End user requirements and implementation status.

These requirements specify that the system shall allow recommendations for energy usage to promote user participation through a reward process and that it shall allow near-real-time monitoring of the energy consumption and performance and their visualization for ESCOs and consumers/prosumers. These requirements are fulfilled by the algorithms in this module as

the algorithms are designed to operate continuously, although they have to be tested in the real environment.

The functional requirements directly related with the Personalized Energy Analytics Module are listed in the next table. Some of the requirements are partially implemented and a validation in a real environment is needed.

Req_id	Description	Implementation Status
Req_013	The system shall provide near real time monitoring of energy demand, generation, and consumption at dwelling level.	Partially implemented–missing integration with the BDP
Req_039	The system shall provide the option to the user to set their preferences (thermal, IAQ etc. comfort profiles) on a user interface.	Fully implemented
Req_040	The system shall allow the dynamic update of user’s comfort profiles.	Partially implemented – without validation in real environment
Req_042	The system should be able to automatize energy management based on the users’ comfort profiles.	Partially implemented–missing connection with the demo sites.
Req_043	The system shall provide notifications and recommendations to the user on their energy behaviour profile through a user interface.	Fully implemented
Req_044	The system shall allow prosumers and consumers to setup rules for the control of devices through a user interface.	Partially implemented–without validation in real environment
Req_045	The system shall show the price signals and the expected energy savings through a user interface.	Partially implemented–without validation in real environment
Req_047	The system shall provide visualization of EE/flexibility information at consumer level through a user interface.	Partially implemented–without validation in real environment
Req_048	The system shall increase the user awareness regarding EE and flexibility patterns through an informative user interface.	Partially implemented–without validation in real environment

Table 6: Functional requirements and implementation status.

3.2.3.4 Outputs

There are two kinds of outputs. The first one is a notification, and the second, recommendations shown in the GUI.

Notifications are sent through email and can be seen in the GUI, and include information regarding the situation that had triggered them (for example: room temperature under the comfort temperature) and give recommendations (example: turn off the HVAC).

The information that is shown in the GUI is regarding the recommendation of shifting loads or information about peak solar energy generation.

3.2.3.5 Relationship with other frESCO Modules

The Personalized Energy Analytics Module interacts with three other frESCO modules:

- Local Demand Manager (LDM):
 - Graphical User Interface (GUI): The GUI is the interface through which operation modes and set-points used by the algorithms are indicated. In addition, the notifications generated by the algorithms are displayed there (see subsection 4.3).
- Global Demand Manager (GDM):
 - Energy Management Analytics Module: This module provides temperature analytics which can be used as set-points in case the user wants to.
- Big Data Management Platform: The Personalized Energy Analytics Module reads from the platform all the metering and sensor information from demo sites that are required by the algorithms.

3.2.4 Human Centric Automation Module

3.2.4.1 Introduction

The Human Centric Automation module focuses on the implementation and automation of those frESCO services in charge of establishing the best control signals for managing controllable devices, mainly the HVAC and DHW tank. For doing this, it is necessary to automatically send the signals to the actuators responsible of the devices control. The signals are sent directly from the Automation module to the gateway via an MQTT broker.

This module implements a total of four services. Moreover, within this module it is implemented the automation of the flexibility analytics module. Due to the possibility of signal overlapping, the Human Centric Automation module establishes a priority on the signals proceeding from the flexibility analytics module. This is due the contractual responsibilities

the user has assumed with the aggregator and the possible consequences that could carry not abiding by them.

Services' algorithms are coded and tested, but not implemented as data was not available yet.

3.2.4.2 Input Data and Technology Stack

In Table 7 the services and their data needs are described. Apart from the services' input data, for automation it is necessary all the information that allows the connexion with the actuators.

Most services' algorithms are coded and tested, but not implemented as data was not available yet. Algorithms of new services, that surged recently due to demos specific needs, are still pending.

Technologies used for services implementation are listed below:

- API: API is deployed in Python using FastAPI framework. API is responsible of exposing data to frontend (GUI) and other services.
- Schedule Tasks: A Python framework called Celery is used to manage the Schedule tasks for update services information after a defined period of time.
- Redis Database: An In-memory database to store intermediate information of Celery Tasks.
- Postgres Database: A relational database for storing the information data of the services.
- BDP API: Exposed API for interacting with the Big Data Platform, to store and recover information data of the services.

	Services' short description	Input data
Service 21	When user specifies a specific schedule through the GUI, the HVAC will turn on/off automatically	User's HVAC schedule
Service 10	At night, if the HVAC is ON, it will automatically be turned off	HVAC consumed energy
Service 19	When temperature is outside of the specified boundaries (UP3) plus a small	Room temperature HVAC consumed energy

	margin (UP6), the HVAC will be automatically turned ON/OFF	User's comfort temperature
Service 16 - Greece	When not necessary, the boiler will be turned off and when it is necessary again, it will be turned on. Both actions will be automatically executed with a margin of 1-3 hours.	Hotel room booking information

Table 7: Human Centric Automation Module and data inputs.

3.2.4.3 Methodology and Requirements Overview based on frESCO Use Cases

The working methodology is as follows: sensors and meters will compile the necessary information that will be stored in the BDP. The services will read the data and calculate its respective outputs, these outputs, if necessary, will trigger an automatic reaction in the actuators for corrective actions. All the automatization is still pending as the protocol that will allow the connexion with the actuators is still under deployment and no data is available.

In the case of the Flexibility Analytics Module, it is expected for the on/off signals to be sent though an API during DR events. These signals will overwrite the ones from Human Centric Automation Modules, and only the final signals will be sent to the actuators. The communication with the module is still pending.

Within task T2.5 Requirements 1 and 2 were elicited (the requirements can be seen in the following table).

Req_id	Description	Implementation Status
Req_001	The system shall allow automatic operation of DERs.	Partially implemented – without validation in real environment
Req_002	The system shall allow near-real-time monitoring of the energy consumption and performance and their visualisation for ESCOs and consumers/prosumers.	Fully implemented

Table 8: Table 7 End User requirements and implementation status.

These requirements specify that the system shall allow the automatic operation of DERs and that it shall allow near-real-time monitoring of the energy consumption and performance and their visualization for ESCOs and consumers/prosumers. These requirements are fulfilled by

the algorithms in this module as the algorithms are designed to operate continuously, but a test in real environment is still pending.

The functional requirements directly related with the Human Centric Automation Module are listed in the next table.

Req_id	Description	Implementation Status
Req_020	The system shall allow automated device control in respect to the received DR signal at dwelling level.	Partially implemented-missing integration with the demo sites
Req_021	The system shall allow the identification and categorization of asset control properties (interrupted, shiftable and uncontrolled).	Partially implemented – without validation in real environment

Table 9: Table 8 Functional requirements and implementation status.

3.2.4.4 Outputs

There are two kinds of outputs. The first one is the trigger of the automatic action. The second is the display in the GUI of the information of the automatic action triggered, this way the user will be informed of every deployed action.

3.2.4.5 Relationship with other frESCO Modules

The Human Centric Automation Module interacts with five other frESCO modules:

- Local Demand Manager (LDM):
 - Graphical User Interface (GUI): The GUI is the interface through which operation modes and set-points used by the algorithms are indicated (see subsection 4.3).
- Global Demand Manager (GDM):
 - Management Analytics Module: This module provides temperature analytics which can be used as set-points in case the user wants to.
 - VPP Optimal Configuration Module: This module communicates with the Human Centric Automation Module whenever it calculates an on/off command needed to provide the agreed flexibility. VPP commands have priority over Human Centric Automation Module ones, as flexibility contracts requirements are mandatory.

- Big Data Management Platform: The Human Centric Automation Module reads from the platform all the metering and sensor information from demo sites that are required by the algorithms.
- Energy Management System (EMS): the automation signals are sent to the corresponding devices directly through the Energy Boxes, using the MQTT protocol.

3.3 Global Demand Manager Modules Overview

The GDM modules include the following modules:

- Energy Management Analytics module
- Self-consumption optimization module
- Smart contract monitoring/handling module
- Flexibility Analytics module
- VPP optimal configuration module

The results of these modules will be communicated with the aggregators and ESCOs with two UIs, one for aggregators and one for ESCOs, as can be seen in Figure 4.

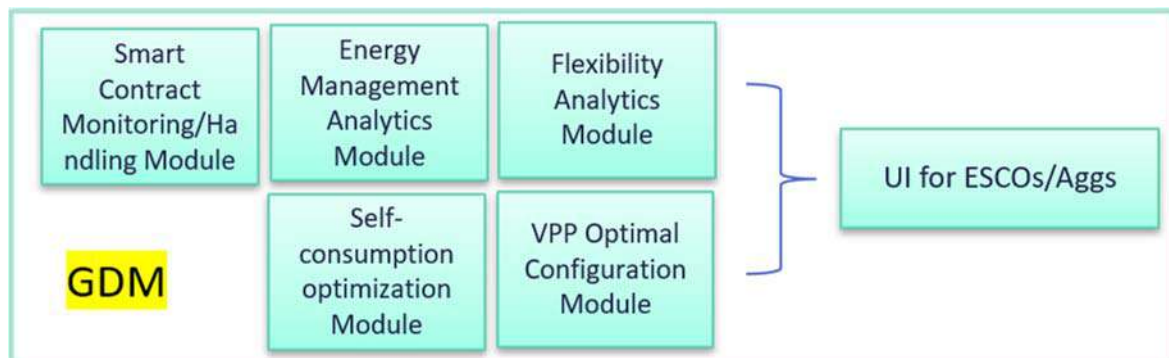


Figure 4: Modules of the Global Demand Manager.

3.3.2 Energy Management Analytics Module

3.3.2.1 Introduction

Energy management analytics is a module that is developed for ESCOs to help building occupants to monitor and analyse their energy usage. This can include tracking energy consumption across different departments and buildings, identifying trends and patterns, and

identifying opportunities for energy savings. ESCOs by performing aggregated energy analytics on their portfolio, have the opportunity to gain useful insights on the road to new business models and improve existing ones for their customers.

Specifically, ESCOs and prosumers can gain several benefits from the development of energy management analytics modules, including:

- **Increased efficiency:** By analysing energy data and identifying patterns, ESCOs can identify inefficiencies and opportunities for energy savings. This can help them to make more informed decisions about energy usage and reduce energy costs for their clients.
- **Enhanced performance:** Energy management analytics can help ESCOs to monitor and optimize the performance of energy-efficient equipment and practices, which can help them to improve their overall energy performance and reduce greenhouse gas emissions.
- **New business opportunities:** Energy management analytics can help ESCOs to identify new business opportunities, such as identifying new areas for energy savings, and providing energy-efficient solutions to clients.
- **Better decision-making:** By having data and insights from energy management analytics, ESCOs can make better decisions and improve their operations.

3.3.2.2 Input Data and Technology Stack

Since data uploading in the BDP has not finished yet, we propose here a set of input data in order to fulfill the user requirements:

1. Historical data for PV generation per building/dwelling
2. Historical data for aggregated energy consumption per building/dwelling
3. Historical data for energy consumption per device and per building/dwelling
4. Real-time data for PV generation per building/dwelling
5. Real-time data for aggregated energy consumption per building/dwelling
6. Real-time data for energy consumption per device and per building/dwelling

If any of the above input data cannot be available, then we need to modify our development accordingly.

The energy management analytics module is going to be delivered as a web-based application with the following tech stack:

- Backend: Django framework
- Frontend: Angular framework
- Database: Postgres
- Analytics: Python and respective libraries like Pandas, Sklearn
- Deployment: Kubernetes
- Login/Authentication: Keycloak [14]

3.3.2.3 Methodology and Requirements Overview based on frESCO Use Cases

This module will serve as the calculation engine of portfolio-wide energy efficiency and self-consumption strategies that will be applied to appropriate clusters of consumers, that effectively participate in the energy efficiency remuneration programs.

The ESCOs are particularly interested in aggregated analytics at building level and at portfolio level. The quantities of interest are usually the energy generation and energy consumption at building and at device level, self-consumption rate, energy savings and cost savings. Comparative analytics provide useful insights to the ESCOs as they can compare energy performances of different buildings regarding the self-consumption and derive the optimal hours for better exploiting energy generation peaks at each building.

Another interesting feature that we investigate here is finding and comparing similar peers regarding the energy generation and consumption, as well as extracting typical generation and demand profiles per cluster. We use machine learning algorithms, such as k-means and DBSCAN and statistical analysis. Because similar peers (peers that belong to the same cluster e.g., with low/mid/high energy consumption profiles) can be found either at a single-building level or at an inter-building level, a two-level K-means clustering is designed and implemented. After we have identified clusters of prosumers, then we can extract typical profiles for different temporal characteristics, such as season, month, day, etc. We are going to modify or implementation and our dashboards according to the available data from the demo sites that we will have at the end. This means that we might not have multiple buildings, or some buildings might not have PV generation.

Based on the user requirements provided in D2.5, this module is directly related to the End user requirements listed below:

Req_id	Description	Implementation status
Req_002	The system shall allow near-real-time monitoring of the energy consumption and performance and their visualisation for ESCOs and consumers/prosumers.	Partially implemented-missing demo sites data
Req_012	The system shall support different clustering criteria (type of DERs, historic participation etc.).	Partially implemented-missing demo sites data

Table 10: End user requirements and implementation status.

These requirements have been addressed and can be tested in the UI. Specifically, apart from prosumers themselves, also ESCOs need to monitor the energy consumption and generation of their customers. For this reason, in the case of Req_002, the dashboard will provide graphs for the generation and the demand of specified intervals as they come from the sensors installed in the demo sites. The user will be able to select a certain period in the past to visualize historical data. In the case of Req_012, unsupervised machine learning models have been used to perform clustering of the ESCOs' portfolio according to specified criteria, such as similar peers. The user will be able to provide the temporal characteristics, such as daily, monthly, or seasonally based on which the analytics will be computed.

Based on the user requirements provided in D2.5, this module is directly related to the functional requirements listed below:

Req_id	Description	Implementation status
Req_013	The system shall provide near real time monitoring of energy demand, generation, and consumption at dwelling level.	Partially implemented-missing demo sites data
Req_018	The system shall allow the energy service providers to perform consumers portfolio segmentation /clustering according to different parameters.	Partially implemented-missing demo sites data

Table 11: Functional requirements and implementation status.

These requirements have been addressed and can be tested in the UI. Specifically, apart from prosumers themselves, also ESCOs need to monitor the energy consumption and generation of their customers. For this reason, in the case of Req_013, the dashboard will provide graphs for the generation and the demand of 15 minutes intervals as they come from the sensors installed in the demo sites. The user will be able to select a certain period in the past to visualize historical data. In the case of Req_018, unsupervised machine learning models have been used to perform clustering of the ESCOs' portfolio according to specified criteria. The user will be able to provide the temporal characteristics, such as daily, monthly, or seasonally based on which the analytics will be computed.

3.3.2.4 Outputs

The energy management analytics module provides all the outputs as graphs in the UI. It does not compute any new quantity, but rather collects, aggregates, analyses and visualizes data coming either from the analytics or the sensors from the demo sites.

3.3.2.5 Relationship with other frESCO Modules

The energy management analytics module that resides in the GDM, takes input from the Big Data Management Platform, where all the available sensor and user data are stored. Any identified strategies for ESCOs that come as results from the Energy Efficiency Analytics, will be targeted to their respective consumers, so an interaction with the Personalised Energy Analytics Module is necessary. Moreover, the energy management analytics module will get input from the results of self-consumption optimization module to provide aggregated analytics about energy saving and cost savings.

3.3.3 Self-Consumption Optimization Module

3.3.3.1 Introduction

Self-consumption optimization is a module that is developed by ESCOs to help building occupants optimize their use of on-site renewable energy sources, such as solar panels and wind turbines. The application helps to control the energy consumption from the grid, and it

optimizes the use of the energy generated from on-site renewable energy sources. This can include real-time monitoring of energy production and consumption, forecasting energy production and consumption, and optimizing the use of energy storage systems. The goal is to maximize the amount of energy that is consumed on-site, and minimize the amount of energy that is sent back to the grid or purchased from the grid. This application can help businesses and organizations to reduce their energy costs, increase their energy independence, and decrease their carbon footprint.

Specifically, ESCOs and prosumers can gain several benefits from the development of self-consumption optimization modules, including:

- **Increased energy independence:** By optimizing the use of on-site renewable energy sources, businesses and organizations can reduce their reliance on the grid and increase their energy independence. This can help them to reduce their energy costs and decrease their carbon footprint.
- **Improved energy performance:** Self-consumption optimization modules can help ESCOs to monitor and optimize the performance of on-site renewable energy sources, which can help them to improve their overall energy performance.
- **Increased customer satisfaction:** By helping businesses and organizations to reduce their energy costs and decrease their carbon footprint, self-consumption optimization can improve customer satisfaction and loyalty.
- **New business opportunities:** Energy service companies can offer self-consumption optimization as a service to businesses and organizations, which can create new business opportunities and increase revenue.
- **Better decision-making:** By having data and insights from self-consumption optimization, energy service companies can make better decisions and improve their operations.
- **Improved reliability:** Self-consumption optimization can improve the reliability of energy supply and reduce dependence on the grid, which can help to avoid power outages or blackouts.

3.3.3.2 Input Data and Technology Stack

Since data uploading in the BDP has not finished yet, we propose here a set of input data in order to fulfill the user requirements:

1. Day ahead forecasted data for PV generation per building/dwelling
2. Day ahead forecasted data for aggregated energy consumption per building/dwelling
3. Battery static properties, such as capacity, maximum charging and discharging rate, efficiency, and minimum and maximum state of charge.

If any of the above input data cannot be available, then we need to modify our development accordingly.

The self-consumption optimization module is going to be delivered as a web-based application with the following tech stack:

- Backend: Django framework
- Frontend: Angular framework
- Database: Postgres
- Optimization: Python and respective libraries Pyomo, IPOPT, etc.
- Deployment: Kubernetes
- Login/Authentication: Keycloak

3.3.3.3 Methodology and Requirements Overview based on frESCO Use Cases

Self-consumption optimization module promotes energy self-sufficiency, using DERs (PV, battery, HVACs, etc.) by exploiting as much as possible the power generated on the building (maximising the self-consumption) and minimising the excess of energy given back to the grid. By adopting this energy usage behaviour, the prosumer maximises energy savings and minimises energy cost savings. We can divide the self-consumption scenarios into two categories: energy storage scenario and load-shifting scenario.

The energy storage scenario provides recommendations to the prosumers on how to dispatch a battery for the next day in order to achieve maximization of self-consumption. This constitutes a Mixed Integer Quadratic Program (MIQP), in which the objective function is to minimize the amount paid to import energy from the grid and the decision variables are the

charging and discharging power of the battery, the energy of the battery and the amount of energy imported from the grid. The constraints include the limits for the static variables of the battery, such as minimum and maximum state of charge, minimum and maximum charging and discharging power, the power balance equation, and the equation of energy evolution of the battery.

Based on the user requirements provided in D2.5, this module is directly related to the End user requirements listed below:

Req_id	Description	Implementation status
Req_004	The system shall allow energy savings maximization for prosumers without compromising their indoor comfort.	Partially implemented-missing demo sites data
Req_005	The system shall allow recommendations for energy usage to promote user participation through a reward process.	Partially implemented-missing demo sites data

Table 12: End user requirements and implementation status.

These requirements have been addressed and can be tested in the UI. Specifically, for the Req004 the system outputs a plan for the dispatch of the battery, which guarantees to maximize self-consumption or equivalently to minimize the amount of money that the prosumers will pay for their energy usage by exploiting the energy coming from PV generation. This schedule will be communicated with personalized energy analytics module of the LDM to act as recommendations for the prosumers aiding the Req_005.

Based on the user requirements provided in D2.5, this module is directly related to the functional requirements listed below:

Req_id	Description	Implementation status
Req_028	The system should maximize the self-consumption considering all dynamic and static parameters involved and without compromising optimum overall energy usage.	Partially implemented-missing demo sites data
Req_045	The system shall show the price signals and the expected energy savings through a user interface.	Partially implemented-missing demo sites data

Table 13: Functional requirements and implementation status.

These requirements have been addressed and can be tested in the UI. Specifically, the Req_028 is satisfied by the self-consumption optimization algorithms, which provide optimum schedules as recommendations for the usage of different devices. The dashboard also depicts the day-ahead wholesale energy prices and the energy savings satisfying the Req_045.

3.3.3.4 Outputs

The self-consumption optimization module provides all the outputs as graphs in the UI. The outputs of the module are suggestions about the charging and discharging commands of the battery for the next day.

3.3.3.5 Relationship with other frESCO Modules

The self-consumption optimization module that resides in the GDM, takes input from the Big Data Management Platform, where all the available sensor and user data are stored. It also gets the results from the analytics that are trained on the platform and include forecasts of the day ahead energy consumption and energy generation. Moreover, the self-consumption optimization module will send the results to the energy management analytics module, as well as to the personalized energy analytics module.

3.3.4 Smart Contract Monitoring/Handling Module

3.3.4.1 Introduction

Smart grid management is facing new challenges related to the introduction of the renewable energy sources. To be able to optimize energy flows and achieve the energy balance, local energy flexibility from assets originating from the participation of active prosumers can be exploited by Aggregators and the existence of smart contracts that can be signed between those parties poses an enormous motivation factor to trade available flexibility in a peer-to-peer fashion.

Turning a settlement between two parties into a contract written in lines of code, is the definition of a smart contract. And to turn the signing of the contract, along with all other transactions, into unique and irreversible facts, smart contracts are tracked across a distributed, decentralized blockchain network that allows agreements to be settled between parties, without the need of a central authority, legal system, or external enforcement mechanism.

Having performed a thorough comparison between five distributed ledger technologies, the Ethereum blockchain is chosen as the optimal solution for the needs of the frESCO project, to accommodate the smart contracting functionalities, based on rewarding computations via digital assets. Utilizing the functionalities offered by the Ethereum smart contracts, the implemented Blockchain-enabled Smart Contract Monitoring, Handling, Settlement and Remuneration module, enables an impartial and transparent settlement and remuneration of each contract being signed between Aggregators and prosumers, posing the main parties involved in a flexibility contract.

3.3.4.2 Input Data and Technology Stack

The primary target of Aggregators is to be able to moderate the electricity consumption of a group of consumers, gather available flexibility and sell it to the energy market. To be able to create a single entity out of multiple prosumers, contracts need to be established with each one of them, with contact details on commercial terms and conditions for the procurement and control of flexibility. Allowing the creation of smart contracts, the Smart Contract Monitoring and Handling module allows Aggregators, through the provision of an appropriate

User Interface, to introduce the contract and its parameters in a blockchain infrastructure. Contract details, such as the contract duration, flexibility capacity, amount of nominal power to be delivered, number of activations and price are part of the input activities and the blockchain implementation ensures both transparency over the contractual process, as well as the establishment of an advanced Settlement and Remuneration process. Through this process and based on the terms that are agreed in the Smart Contract, the activated flexibility during a DR event can be verified and eventually respective remunerations can be calculated and attributed to the prosumers.

The Smart Contract Monitoring and Handling module utilizes as input the datasets collected from the Big Data Management Platform, that offer real-time data originating from flexible assets made available from the prosumers. By combining this information with the smart contract details provided for each of the contracts from the Aggregator, the execution of advanced Settlement and Remuneration processes to reward flexibility provision in a transparent and objective manner, can be executed. As a baseline for those calculations, the short-term demand forecasting analytics offered from the Data Analytics Module of the Big Data Management platform are utilized. In the next paragraphs, required input to be utilized for the needs of Smart Contract Monitoring and Handling module will be further analyzed.

Real time demand data - Big Data platform

To be able to perform the necessary calculations for the needs of the PMV, with clear evidence that prosumers are offering the flexibility promised in the signed smart contract, the data streams of the related monitored assets need to find their way into the Smart Contract Monitoring and Handling module. This is realized by utilizing the API retrieval functionality offered from the Big Data platform, after searching, identifying, and creating a dedicated query including the relevant datasets. As presented in Table 14 below, per dataset, the data that is retrieved from the platform includes a timestamp defining the timeseries, the ID for each asset and the total energy consumption per record. Those fields, combined with the location of the asset and the prosumer ID that owns it, present the minimum requirements for the needs of the Smart Contract Monitoring and Handling module.

Timestamp	Asset ID	totalEnergy Consumption	location	Prosumer ID	type	title
2022-08-29T12:00:00.000Z	5a5c5cg8	215	Athens	bb8774f3R	HVAC	Kitchen HVAC
2022-08-29T12:15:00.000Z	5a5c5cg8	217	Athens	bb8774f3R	HVAC	Kitchen HVAC
2022-08-29T12:30:00.000Z	5a5c5cg8	218	Athens	bb8774f3R	HVAC	Kitchen HVAC
2022-08-29T12:45:00.000Z	5a5c5cg8	220	Athens	bb8774f3R	HVAC	Kitchen HVAC

Table 14: Demand asset format example.

Demand forecasting – Big Data Platform

Similar to the real time demand data, also the demand forecasting timeseries is available as a result of the execution of pre-trained analytics in the Big Data Platform and can be retrieved via the API interface offered. The demand forecasting timeseries serves as the baseline for the quantification of the flexibility potential of each asset, but also, for the verification of the flexibility provided from the Smart Contract Monitoring and Handling module. In the Table 15 below, the format of the timeseries is presented, offering hourly predictions of the asset.

Timestamp	Asset ID	Predicted_Total Consumption Hourly
2022-08-29T12:00:00.000Z	5a5cm5cd8	225
2022-08-29T13:00:00.000Z	5a5cm5cd8	230
2022-08-29T14:00:00.000Z	5a5cm5cd8	220
2022-08-29T15:00:00.000Z	5a5cm5cd8	180

Table 15: Demand forecasting format example.

Flexibility Events – Big Data Platform

Offered from the Flexibility Analytics module, specific flexibility event activations, that are triggered based on the contractual agreements between Aggregators and Prosumers, are pushed into the Big Data Management platform, and retrieved from the Smart Contract

Monitoring and Handling module via the API interface. The activation timestamp, the asset, prosumer and event IDs, combined with the event duration and type, and the maximum flexibility capacity of the asset, are the event characteristics needed to allow a precise, realistic, and objective settlement, based on the defined measurement and verification process. The format of the event records is displayed in the Table 16 below.

Activation Timestamp	Asset ID	EventID	Prosumer ID	Duration Period	Flexibility Capacity	Description
2022-08-29T14:00:00.000Z	6a5c5cd8	5	bp8774f3	60	120	downwards
2022-08-29T14:00:00.000Z	6a5c5cd8	13	bp8774f3	30	60	upwards

Table 16: Flexibility events activation format example.

In the following section, the state-of-the art technologies that are utilized to implement the functionalities of the frESCO Blockchain-enabled Smart Contract Monitoring, Handling, Settlement and Remuneration module are presented.

To implement the custom UI design that offers the module users to enter contract details and verify their statuses, the **Vue.js V2** [15] framework was utilized, which is a progressive framework for building user interfaces. Unlike other monolithic frameworks, Vue is designed from the ground up to be incrementally adoptable. The core library is focused on the view layer only and is easy to pick up and integrate with other libraries or existing projects.

For the needs of the back-end layer and the implementation of the module services, the **Nest V9** [16] framework is used. The Nest is a progressive Node.js framework for building efficient, scalable, and enterprise-grade server-side applications on top of TypeScript & JavaScript.

For data storage, the **PostgreSQL** [17] is chosen, serving as the relational database for the operational data of the application, storing contracts and settlement parameters.

Finally, as a blockchain architecture, the **Ethereum blockchain** [18] was chosen, since it poses the most mature, generic, and decentralized architecture, offering the optimal solution to accommodate the smart contracting functionalities for the creation of the blockchain wallets.

3.3.4.3 Methodology and Requirements Overview based on frESCO Use Cases

As described in deliverable D2.5 that covered the frESCO conceptual architecture, numerous project use cases described in the DoA [19] dictated the main requirements towards the implementation of the applications and their corresponding features.

The Use Case designated to portray the blockchain mechanism to enable smart contracts monitoring and assessment, that served as the basis for the definition of the Smart Contract Monitoring and Handling module, is UC.09 - “Smart contract monitoring, handling and remuneration”. The details of this use case, including a description and use case actors are presented in Table 17 below.

Title	Description	Actors
UC.09: Smart contract monitoring, handling and remuneration	Aggregators are expected to play a key role in unleashing the flexibility potential of the building sector and transforming it into a tradeable commodity in energy markets. In this context, aggregators need access to fine-grained information streams from buildings, towards being able to assess the flexibility they can offer at different temporal granularity. On the other hand, prosumers need to better understand their flexibility and get involved in transparent and trusted flexibility transactions and respective contracts. This requires the establishment of innovative Smart Contracting Mechanisms that will enable both aggregators and prosumers to be continuously aware of the parameters involved in their contractual agreements, while transparently monitoring the satisfaction of the contractual terms and the effectiveness of their bilateral contracts, thus having a clear understanding of the settlement and remuneration processes in relation to flexibility events (activation) or flexibility availability.	Aggregators, facility managers and owners, residents, energy communities

Table 17: Smart contract monitoring, handling and remuneration UC.

Based on the use case description, an Aggregator being the main actor of the Smart Flexibility Contracts Application, whose role can be undertaken also from an ESCO or a Facility Manager/Owner, should be able to perform various actions that are captured in a list of

functional and non-functional requirements as presented in the table below and are in alignment with the requirements presented in D2.5. As part of the table, multiple calculations and outputs of the module are also captured.

Req_id	Description	Implementation status
Req_030	The system should allow aggregators to introduce the Smart contract information, established with prosumers, in a blockchain infrastructure, by utilising a user interface.	Fully implemented
Req_031	The system should allow aggregators to search for flexible assets through a user interface.	Fully implemented
Req_032	The system should allow aggregators to get access to the results of the remuneration process (i.e., their contracts) through a user interface.	Fully implemented
Req_033	The system should allow calculation of prosumer revenue for flexibility capacity.	Fully implemented
Req_034	The system should allow calculation of prosumer revenue for flexibility provision.	Fully implemented
Req_035	The system should allow calculation of prosumer penalty for non-delivery of flexibility.	Fully implemented
Req_036	The system should allow flexibility settlement details based on flexibility ordered by the aggregator.	Fully implemented
Req_037	The system should provide remuneration details to settle flexibility contracts.	Fully implemented
Req_038	The system should offer flexibility remuneration details based on the contractual agreement economical terms.	Fully implemented

Table 18: Functional requirements and implementation status.

Based on the user requirements provided in D2.5, this module is directly related to the end user requirements listed in the table below:

Req_id	Description	Implementation status
Req_008	The system shall support a fair and transparent measurement and verification methodology for pay for performance in flexibility and energy efficiency events.	Fully implemented
Req_010	The system shall allow demand forecast, generation forecast, incoming market flexibility requests, market bid prices monitoring, DER management and VPP configuration, information about connection status, localization, clustering of DERs	Fully implemented

	under smart contracts, energy dispatch, final flexibility delivery, remuneration parameters.	
Req_011	The system shall allow the identification of available and appropriate flexibility assets.	Fully implemented
Req_013	The system shall support a secure environment for the introduction, visualization, and signature of flexibility contracts.	Fully implemented

Table 19: End user requirements and implementation status.

3.3.4.4 Outputs

Aggregators, utilizing the functionalities offered by the Smart Contract Monitoring and Handling module, should be ensured that the contract signed with the prosumers and their corresponding details, are properly introduced in the blockchain enabled wallets and flexibility activations are settled based on accurate, realistic, and objective measurement and verification processes.

Towards this direction, an appropriate User Interface, to allow the aggregator to enter all necessary information into a form is being offered from the module, allowing the execution of the necessary blockchain calls to introduce the contract details into the blockchain as a new block, turning the normal contract between the aggregator and the prosumer, into a smart contract. Such contract details include the unique contract number, prosumer details, such as the first and the last name, asset details, with name, type, location, flexibility capacity and nominal power as attributes, duration details of the contract, as well as schedule and financial details. For each contract, multiple assets can be selected to become part of the agreement and appropriate schedules can be defined based on the needs of the contract.

Figure 5 below, provides a screenshot of the contract creation UI page, demonstrating all contract details and their attributes.

Figure 5: Contract creation page.

The contractual details, introduced in the Smart Contract Monitoring and Handling module, present input requirements for modules that base their internal calculations on them, such as the Flexibility Analytics module, that requires assets, corresponding schedules and remuneration agreements to calculate and trigger flexibility events. To allow the easy retrieval of the contract details, the Smart Contract Monitoring and Handling module is pushing that information into the Big Data Platform, to take advantage of the available API retrieval method offered by the platform. The table below, demonstrated an example of the format and the various fields used to capture and push the contracts information in the Big Data Platform. The status of the contract, start and end dates, the Prosumer ID and the Aggregator created the contract, accompanied with schedule and remuneration information, are all parts of the information that is offered per contract.

ContractID	6t4c5cd8	6t4c5ct0
Contract Status	Active	Inactive
ContractStartPeriod	21/11/2022	21/11/2022
ContractEndPeriod	21/11/2023	25/11/2023
Prosumer UUID	bb8774f3	bb8774f3
contractAssets	5a5c5cd8, 5a5c5cd9	5dty5d9, 5dtyjur
contractAggregator	aggregator@suite5.eu	aggregator@suite5.eu
ContractFlexibilityCapacity	10	5
Contract ScheduleDay	Wednesday	Saturday
Contract ScheduleStart	23:00	08:00

Contract ScheduleEnd	05:00	12:00
ContractUsageRenumeration	10	20
ContractAvailabilityRenumeration	5	7
Contract Penalty	4	5

Table 20: Contract details example

Following the smart contract creation, the impartial and transparent settlement and remuneration of each contract being signed between Aggregators and Prosumers is being enabled. With the contractual agreements serving as a basis, different sources of flexibility are monitored and remunerated for providing their flexibility when requested. Settlements are calculated based on actual metering data streams which are collected and stored in the Big Data Platform from various data points, acting as flexibility sources. Compared with baseline calculations offered via short-term demand forecasting, quick and accurate settlement for the provided flexibility is ensured, followed by remuneration activities based on the contract details.

Also in that case, Aggregators are offered with a UI to be able to have an overview of the performance of all signed contract. Per asset, the page displays a graphical representation of the flexibility timeline, combined with numerous details about the baseline, the requested and the actual demand, the offer, and the power deficiency, as well as monetary details, covering the utilization, availability, penalty and settlement values. This information is illustrated in the Figure 6 below.

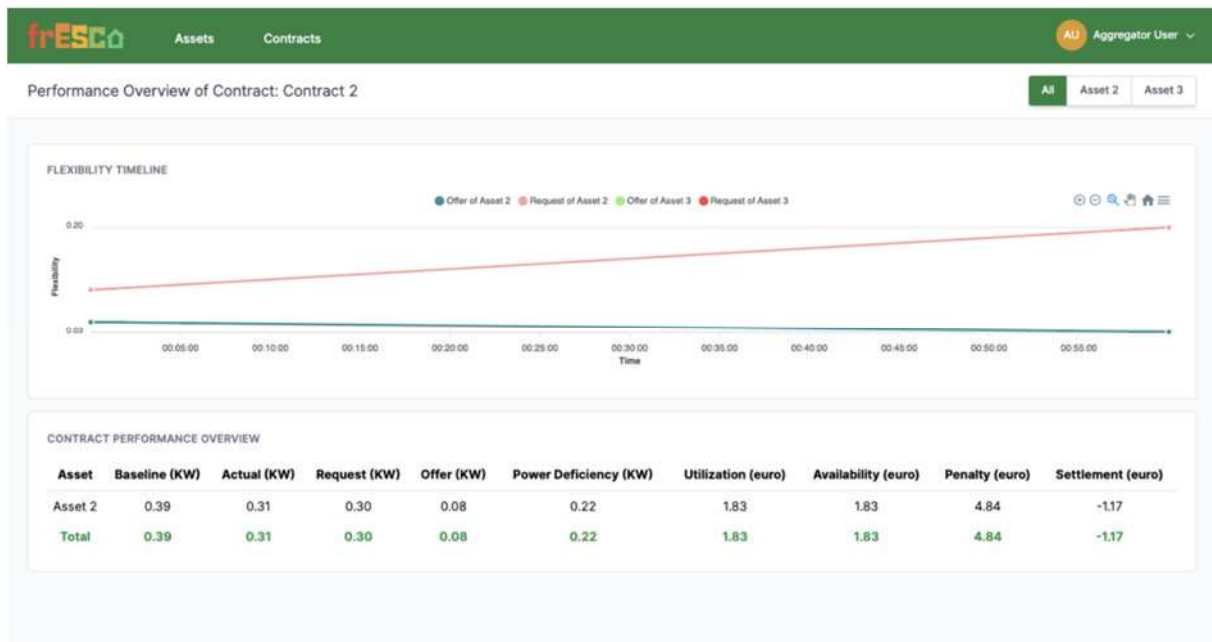


Figure 6: Contract performance overview.

3.3.4.5 Relationship with other frESCO Modules

Being part of the Global Demand Manager, the Smart Contract Monitoring and Handling module is communicating with the Big Data Management Platform and with the Ethereum blockchain infrastructure, as presented in the Figure 7 below.

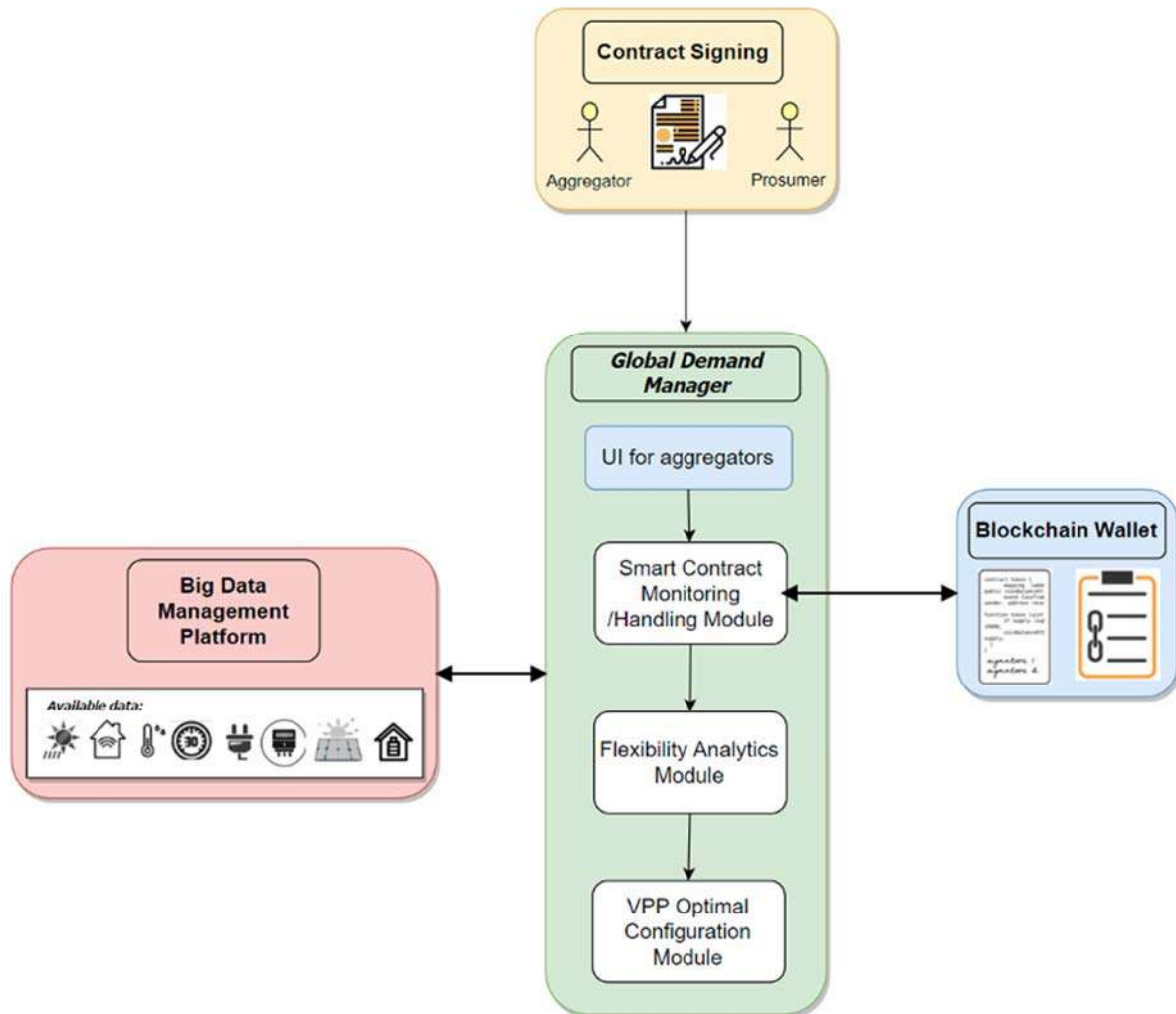


Figure 7: Global Demand Manager modules related to Smart Contract Monitoring/Handling module.

Blockchain Wallet: After the successful introduction of the contract details from the Aggregator into the Smart Contract Monitoring and Handling module, an appropriate collection of code and data is sent to a specific address on the Ethereum Blockchain Wallet, capturing the agreement between Aggregators and Prosumers in a highly secured and transparent manner.

Big Data Management Platform: The datasets created from heterogeneous data demand sources and are stored into the Big Data Management platform, are retrieved from the Smart Contract Monitoring and Handling module for the needs of real-time monitoring. In addition, baseline data and flexibility activation events are also retrieved from the Big Data platform and utilized from the module for the settlements of the smart contracts. Utilizing the data

ingestion methods offered by the Big Data Management platform, contract details are pushed from the module into the platform, to serve the needs of the Flexibility Analytics module.

3.3.5 Flexibility Analytics Module

3.3.5.1 Introduction

Flexibility analytics is an application that is developed by aggregators to help businesses, organizations, and individuals participate in demand response programs and optimize the use of their flexible energy assets. The application helps to analyse the energy consumption data, forecast the energy consumption and production, and optimize the use of flexible energy assets such as electric vehicles, battery storage, and thermal storage systems.

The goal of flexibility analytics is to help the participants to provide balancing services to the grid by adjusting their energy consumption, or by providing energy from their flexible assets, in order to balance the supply and demand of energy. This can include real-time monitoring of energy production and consumption, forecasting energy production and consumption, and optimizing the use of energy storage systems.

By participating in demand response programs, businesses and organizations can earn revenue by providing balancing services to the grid, and can also help to increase the integration of renewable energy sources into the grid. Additionally, the flexibility analytics application can help to improve the overall energy performance and reduce the carbon footprint of the participants.

Aggregators can use the flexibility analytics to bundle different flexible assets together, and sell the flexibility to grid operators as a single, flexible resource.

Aggregators and prosumers can gain several benefits from the development of flexibility analytics modules, including:

- **Increased revenue:** By participating in demand response programs and providing balancing services to the grid, businesses and organizations can earn revenue by providing balancing services to the grid, and can also help to increase the integration of renewable energy sources into the grid.

- Improved customer service: Flexibility analytics modules can provide real-time energy data and alerts to aggregators, which can help them to quickly identify and respond to issues with energy usage. This can improve customer service and satisfaction.
- Enhanced performance: Flexibility analytics can help aggregators to monitor and optimize the performance of flexible energy assets, such as electric vehicles, battery storage, and thermal storage systems, which can help them to improve their overall energy performance and reduce greenhouse gas emissions.
- New business opportunities: Flexibility analytics can help aggregators to identify new business opportunities, such as identifying new areas for energy savings, and providing energy-efficient solutions to clients.
- Better decision-making: By having data and insights from flexibility analytics, aggregators can make better decisions and improve their operations.
- Increased participation: Flexibility analytics can help to increase the participation of different flexible assets in the demand response programs by providing the information needed for the assets to participate.
- Improved grid integration: By providing balancing services to the grid with the help of flexibility analytics, aggregators can help to improve the integration of renewable energy sources into the grid and make the grid more resilient.

3.3.5.2 Input Data and Technology Stack

Since data uploading and finalization of analytics in the BDP has not finished yet, we propose here a set of input data in order to fulfill the user requirements:

1. Static properties of devices taking part in DR events per prosumer, such as nominal power and capacity
2. Flexibility contract static properties per prosumer, such as capacity, penalties, and payment
3. Historical flexibility settlement information per prosumer per DR event, such as settlement date, requested power, available power, delivered flexibility power, penalty and price
4. Forecasted short term available flexibility per device and per prosumer

If any of the above input data cannot be available, then we need to modify our development accordingly.

The flexibility analytics module is going to be delivered as a web-based application with the following tech stack:

- Backend: Django framework
- Frontend: Angular framework
- Database: Postgres
- Analytics: Python and respective libraries like Pandas, Sklearn
- Deployment: Kubernetes
- Login/Authentication: Keycloak

3.3.5.3 Methodology and Requirements Overview based on frESCO Use Cases

The extraction of demand flexibility profiles from prosumers as well as their flexibility registry nominal characteristics will be used in order to assess the amount of available flexibility based on very short (15 minutes) time forecasts.

The grade of flexibility of a DER system depends on how quickly it can adapt generation or demand in response to external forces. This knowledge is very useful for the flexibility service providers such as Aggregators, as they are in charge to prepare and offer smart flexibility contracts to their prosumers, negotiate their rules and thus attract customers from their portfolio into participating in DR programs. Customer segmentation is a very powerful tool to find patterns that will help the flexibility service providers to improve their DR strategies into more targeted and personalized campaigns.

Here as a customer segmentation tool, we implement a clustering engine based on Kmeans and DBSCAN. The reason behind choosing also a second clustering method is that Kmeans in order to work well, the following assumptions need to be true:

- the variance of the distribution of each attribute (feature) is spherical

- all variables have the same variance
- the prior probability for all k clusters is the same, i.e. each cluster has roughly equal number of observations.

By features in this case, we mean user flexibility settlement information, demographics, contract criteria, types of flexibility assets etc. So, in case the clustering of some of the aforementioned features produces unevenly sized clusters like for example if the aggregator's portfolio has a large cluster of residential consumers and a small cluster of commercial ones, the DBSCAN method is preferred. As other advantages, DBSCAN does not require a-priori specification of the number of clusters, and it is able to identify noise in data.

The produced clusters of prosumers provide the aggregator with useful insights, so in case of a new DR event they can identify targeted clusters that can act as Virtual Power Plants, to respond promptly in this DR event.

Moreover, some aggregated time series analytics will also be calculated here either at a cluster level or at a whole portfolio level. Examples are:

- Total available flexibility per device
- Total delivered flexibility per device
- Total available flexibility per prosumer
- Total delivered flexibility per prosumer

Based on the user requirements provided in D2.5, this module is directly related to the End user requirements listed below:

Req_id	Description	Implementation status
Req_012	The system shall support different clustering criteria (type of DERs, historic participation etc.).	Partially implemented-missing demo sites data

Table 21: End user requirements and implementation status.

These requirements have been addressed and can be tested in the UI. The flexibility analytics module is going to give an overview of the aggregators' customers available flexible assets

that can be used to meet the DR events. In the case of Req_012, unsupervised machine learning models have been used to perform clustering of the aggregators’ portfolio according to specified criteria. The user will be able to observe statistics of their customers’ participation in DR response events and clusters of customers according to these results.

Based on the user requirements provided in D2.5, this module is directly related to the functional requirements listed below:

Req_id	Description	Implementation status
Req_016	The system shall allow user flexibility monitoring.	Partially implemented-missing demo sites data
Req_019	The system shall allow the aggregators to optimize their DR strategies considering DR attributes and dynamic electricity prices.	Partially implemented-missing demo sites data
Req_046	The system shall provide visualization of information at different levels: individual, clusters, portfolio through a user interface for the aggregators.	Partially implemented-missing demo sites data
Req_049	The system shall provide information to the aggregator on ongoing/completed DR campaigns through a user interface.	Partially implemented-missing demo sites data

Table 22: Functional requirements and implementation status.

These requirements have been addressed and can be tested in the UI. Specifically, regarding Req_016, Req_019 and Req_046, the dashboard provides the ability to the aggregator to monitor the flexibility delivered by each participant and the flexibility analytics algorithms clusters the participants based on the amount of delivered flexibility allowing the aggregator to optimize the DR strategies. Also, the aggregator can check how well the participants have responded to the DR event by having a historical visualization of requested and delivered flexibility satisfying Req_049.

3.3.5.4 Outputs

The flexibility analytics module provides all the outputs as graphs in the UI. It does not compute any new quantity, but rather collects, aggregates, analyses and visualizes data coming either from the analytics trained on the BDP or the smart contract monitoring/handling module.

3.3.5.5 Relationship with other frESCO Modules

The flexibility analytics module that resides in the GDM, takes input from the Big Data Management Platform, where all the available sensor and user data are stored, as well as the results of the analytics. An interaction with the smart contract monitoring/handling module is necessary in order to retrieve static contract details as well as details about the procurement and settlement of DR events. Moreover, the flexibility analytics module will get input from the results of the optimal VPP configuration module to provide aggregated analytics about DR events participation.

3.3.6 VPP Optimal Configuration Module

3.3.6.1 Introduction

An optimal VPP configuration application is a tool developed by aggregators to help design, operate and control Virtual Power Plant (VPP) systems. A VPP is a system where multiple distributed energy resources (DERs) such as solar panels, wind turbines, energy storage systems, and smart appliances are connected to the grid and operated together as a single, centralized power plant. The application helps to optimize the performance of the VPP by analyzing the energy consumption data, forecasting the energy consumption and production, and optimizing the use of the different DERs.

The goal of optimal VPP configuration application is to help the aggregators to provide balancing services to the grid by adjusting the power production or consumption of the different DERs in the VPP, in order to balance the supply and demand of energy. This can

include real-time monitoring of energy production and consumption, forecasting energy production and consumption, and optimizing the use of energy storage systems.

By optimizing the performance of the VPP, aggregators can increase the revenue from selling energy to the grid, improve the reliability of the energy supply, and reduce the carbon footprint of the VPP. Additionally, the optimal VPP configuration application can help to improve the overall energy performance and reduce the costs of the VPP.

Aggregators and prosumers can gain several benefits from the development of optimal VPP configuration modules, including:

- **Increased revenue:** By optimizing the performance of the Virtual Power Plant (VPP) and providing balancing services to the grid, aggregators can increase the revenue from selling energy to the grid, and also help to increase the integration of renewable energy sources into the grid.
- **Improved energy performance:** Optimal VPP configuration modules can help aggregators to monitor and optimize the performance of the VPP, which can help them to improve their overall energy performance and reduce greenhouse gas emissions.
- **Enhanced reliability:** By optimizing the performance of the VPP, aggregators can improve the reliability of the energy supply and reduce the dependence on the grid, which can help to avoid power outages or blackouts.
- **New business opportunities:** Optimal VPP configuration can help aggregators to identify new business opportunities, such as identifying new areas for energy savings, and providing energy-efficient solutions to clients.
- **Better decision-making:** By having data and insights from optimal VPP configuration, aggregators can make better decisions and improve their operations.
- **Improved Grid integration:** By providing balancing services to the grid with the help of optimal VPP configuration, aggregators can help to improve the integration of renewable energy sources into the grid and make the grid more resilient.
- **Cost reduction:** By optimizing the configuration of the VPP, aggregators can reduce the costs of the VPP and make it more cost-effective, which can make the VPP more attractive to potential customers.

3.3.6.2 Input Data and Technology Stack

Since data uploading and finalization of analytics in the BDP has not finished yet, we propose here a set of input data in order to fulfill the user requirements:

1. Forecasted short term available flexibility per device and per prosumer
2. Flexibility contract static properties per prosumer, such as capacity, penalties, and payment
3. Short term forecasted energy consumption per prosumer and per device

If any of the above input data cannot be available, then we need to modify our development accordingly.

The optimal VPP configuration module is going to be delivered as a web-based application with the following tech stack:

- Backend: Django framework
- Frontend: Angular framework
- Database: Postgres
- Optimization: Python and respective libraries Pyomo, IPOPT, etc.
- Deployment: Kubernetes
- Login/Authentication: Keycloak

3.3.6.3 Methodology and Requirements Overview based on frESCO Use Cases

The optimal VPP configuration module is responsible for maximizing the potential of total available flexibility at a given timeslot, given a specific DR request from the grid. The procedure is the following:

Initially, the aggregator creates a DR event by specifying the DR notice period length, the DR duration and the DR reduction request. After that, the optimization algorithm, which is the core of the optimal VPP configuration module solves the problem of choosing how much energy will be reduced by which prosumer and device and then it sends the results to the human centric automation module for actuation of the respective devices.

The optimization problem is formulated as Linear Program (LP), in which the objective function is to maximize the delivered flexibility from the prosumers, who make up the VPP and the decision variables are the active power that each device is going to deliver during the

DR event. The constraints include the limits respecting the contract details that each prosumer has signed with the aggregator and also the fact that each device has a maximum amount of available flexibility that can offer at each timestamp in order to remain within comfort boundaries and this comes from the analytics trained on the platform.

Based on the user requirements provided in D2.5, this module is directly related to the end user requirements listed below:

Req_id	Description	Implementation status
Req_009	The system shall support demand side aggregation with about 5 seconds' delay.	Partially implemented-missing demo sites data
Req_010	The system shall allow demand forecast, generation forecast, incoming market flexibility requests, market bid prices monitoring, DER management and VPP configuration, information about connection status, localization, clustering of DERs under smart contracts, energy dispatch, final flexibility delivery, remuneration parameters.	Partially implemented-missing demo sites data

Table 23: End user requirements and implementation status.

These requirements have been addressed and can be tested in the UI. Specifically, the Req_009 and Req_010 are addressed by the optimization algorithm, which outputs a plan for the aggregation of the devices that take part in the DR event.

Based on the user requirements provided in D2.5, this module is directly related to the functional requirements listed below:

Req_id	Description	Implementation status
Req_022	The system shall allow the schedule of future flexibility activations.	Partially implemented-missing demo sites data

Req_023	The system shall allow the update and optimization of consumer flexibility clusters (dynamic VPPs) to balance user comfort and flexibility remuneration.	Partially implemented-missing demo sites data
Req_029	The system should allow configuration of flexibility assets into virtual power plants based on assets flexibility availability.	Partially implemented-missing demo sites data

Table 24: Functional requirements and implementation status.

These requirements have been addressed and can be tested in the UI. Specifically, regarding Req_022 the aggregator can trigger DR events from the dashboard and regarding Req_023 and Req_029 the optimal VPP configuration module allows the calculation of the flexibility that has to delivered from each device and thus creates flexibility clusters.

3.3.6.4 Outputs

The optimal VPP configuration module provides all the outputs as graphs in the UI. The outputs of the module are the set points for the devices that are going to be actuated through the human centric automation module.

3.3.6.5 Relationship with other frESCO Modules

The optimal VPP configuration module that resides in the GDM, takes input from the Big Data Management Platform, where all the available sensor and user data are stored, as well as the results of the analytics. An interaction with the smart contract monitoring/handling module is necessary to retrieve static contract details. Moreover, there will be communication with the human centric automation module to actuate the devices based on the results. Finally, the optimal VPP configuration module will send the outputs to the flexibility analytics module, which in turn will use the outputs for the calculation of the analytics and the statistics about the DR events participation.

4 UIs OVERVIEW

4.1 UI for ESCOs

4.1.2 Energy Management Analytics and Self-Consumption Optimization Dashboard

4.1.3 Login

The login page (Figure 8) redirects the user into the Keycloak page to sign in with the credentials provided by the administrator. The ESCOs of a specific demo site will only be able to see the specific demo site's related data and visualizations. We have to note that the EU logo and the acknowledgment have been included in all the following ESCOs dashboards screenshots, and those images presented in the deliverable without the EU logo and the acknowledgement are only due to focus the image on a specific part of the screen.

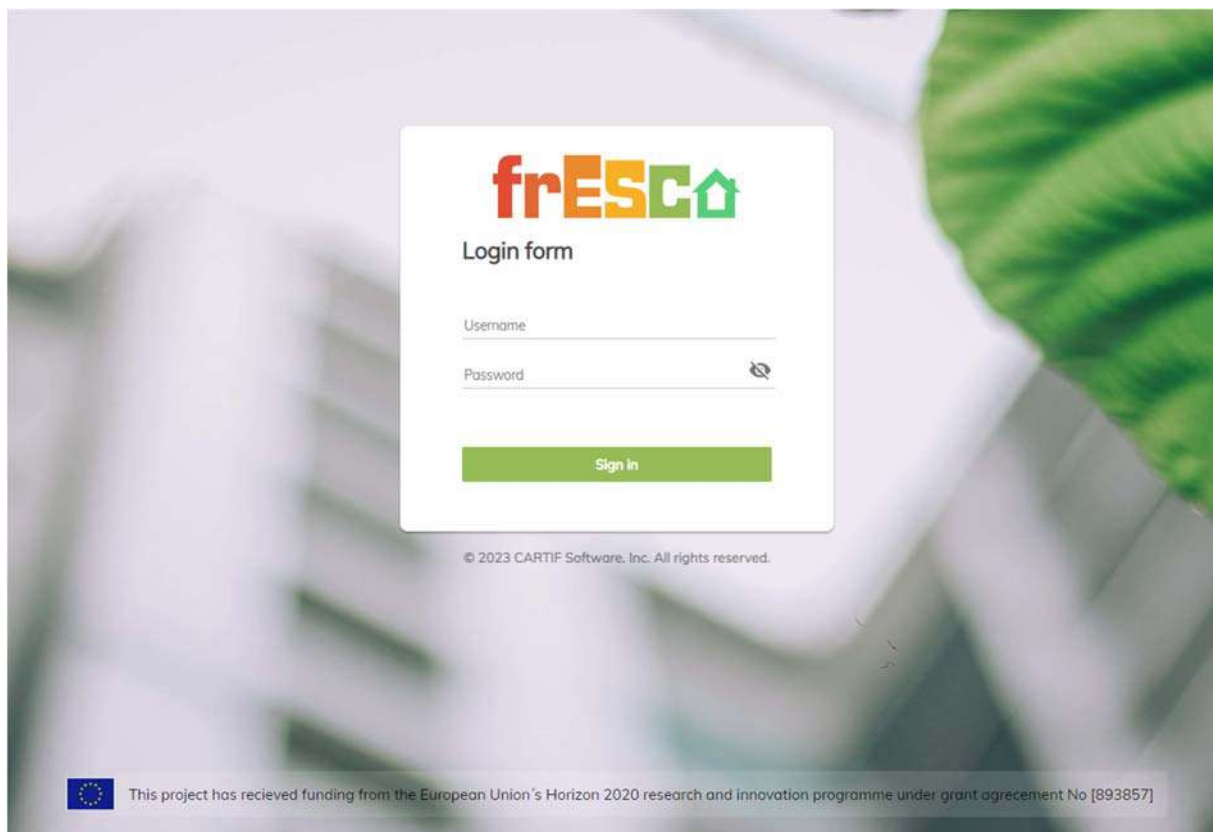


Figure 8: Login page for ESCOs dashboard.

4.1.4 Clustering

The clustering page provides an overview of different clustering criteria, such as creating clusters based on energy consumption of the different buildings (low, mid and high energy consumption), as shown in Figure 9.

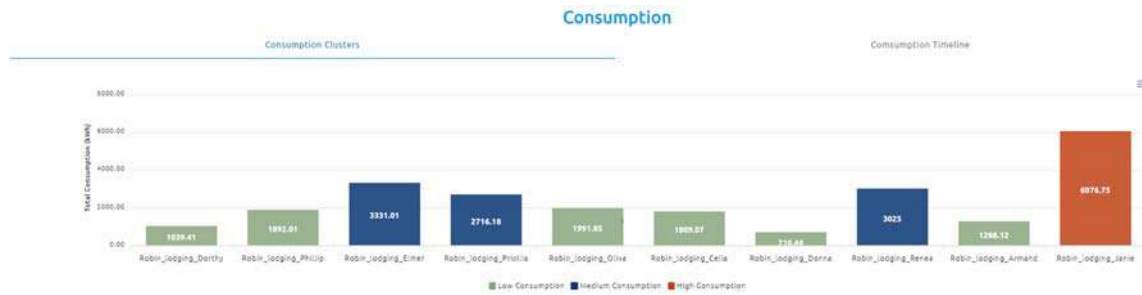


Figure 9: Clustering of buildings based on monthly energy consumption.

The dashboard offers monitoring of historical and real-time energy consumption monitoring based on the user needs (Figure 10).

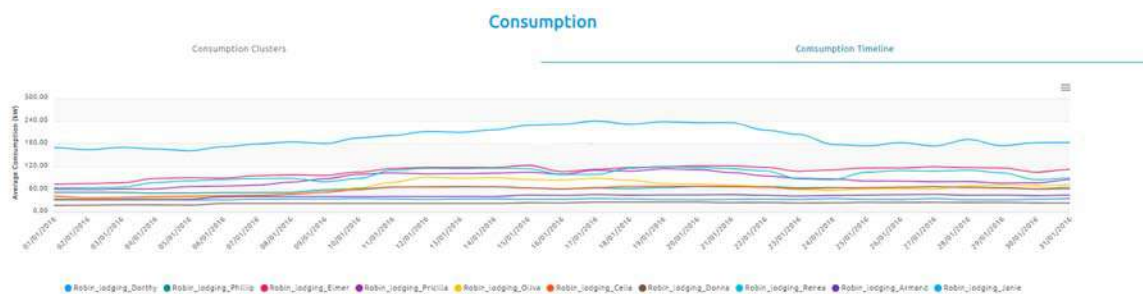


Figure 10: Monitoring of daily energy consumption of different buildings for a month.

4.1.5 Self-Consumption

The self-consumption page provides the results of the self-consumption optimization module. Specifically, in Figure 10, ESCOs can monitor the day-ahead energy demand, energy generation and imported grid energy by taking into account the schedule for charging/discharging of the battery, which can be seen in Figure 11. Finally, Figure 12 shows the day-ahead wholesale prices.

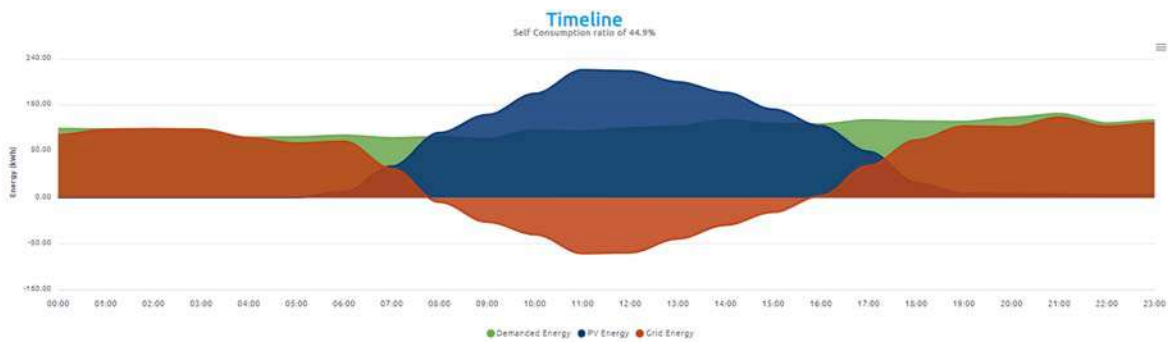


Figure 11: Day-ahead generation, demand and grid import for a building taking into account the battery schedule.



Figure 12: Day-ahead battery energy according to the schedule of self-consumption optimization.



Figure 13: Day-ahead wholesale energy prices

4.2 UI for Aggregators

4.2.2 Flexibility Analytics and Optimal VPP configuration Dashboard

4.2.2.1 Login

The login page (Figure 14) redirects the user into the Keycloak page to sign in with the credentials provided by the administrator. The aggregators of a specific demo site will only be able to see the specific demo site's related data and visualizations. We have to note that the EU logo and the acknowledgment have been included in all the following aggregators dashboards screenshots, and those images presented in the deliverable without the EU logo and the acknowledgment are only due to focus the image on a specific part of the screen.

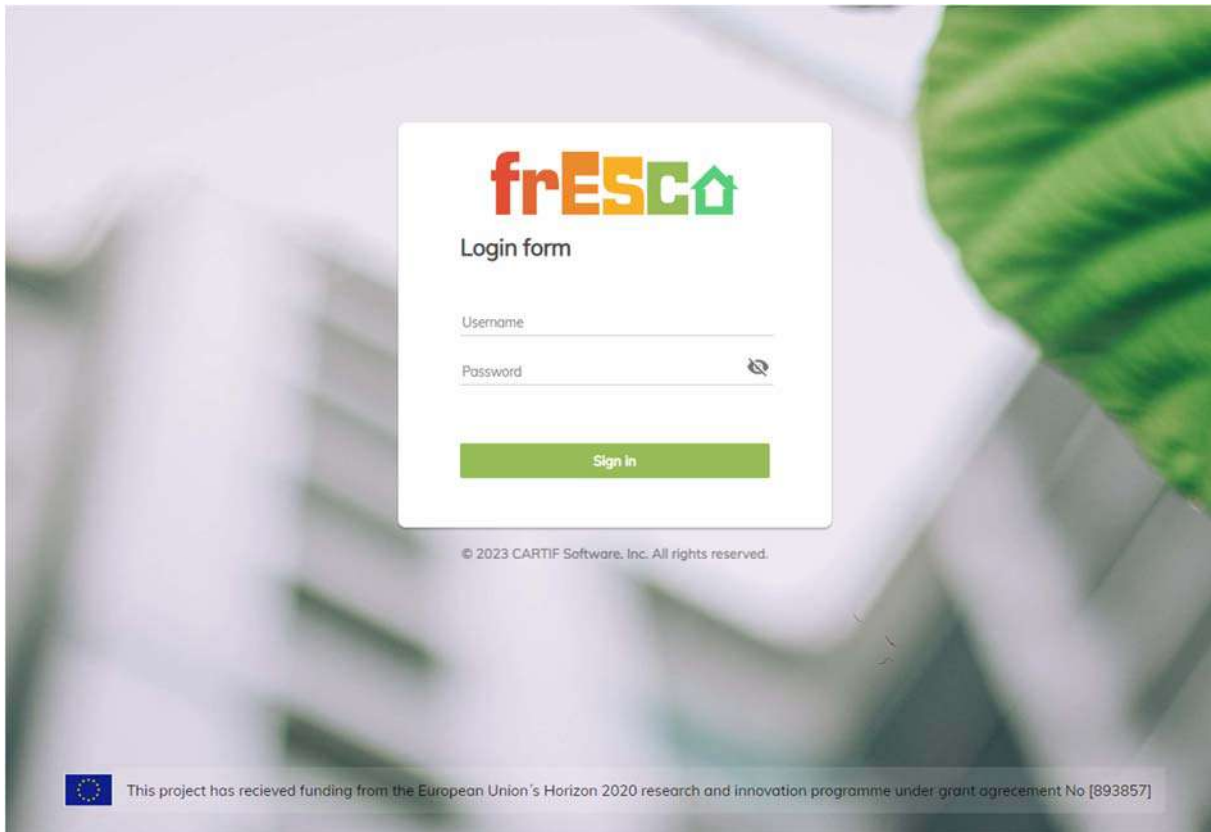


Figure 14: Login page for aggregators dashboard.

4.2.2.2 DR Events

The dashboard gives the ability to the aggregators to trigger DR events as they come from a Transmission System Operator (TSO). Specifically, the aggregator specifies the notice period in minutes, the duration of the DR event in minutes and the amount of flexibility demanded in kW, as shown in Figure 15.

Create a new Event

Cancel
Create

Figure 15: Triggering of DR event.

The aggregator is able to view a list of all the DR events for the demo site for which is responsible in a table sorted by the date that the DR event was created, as shown in Figure 16.

Events				
No	Duration (min)	Notice Period (min)	Flexibility Requested (kW)	Date created
1	15	5	300	13-09-2022 08:37:15
2	30	10	200	15-09-2022 08:49:06

[New Event](#)
Items per page: 5 1 - 2 of 2

Figure 16: Table including all the DR events with chronological order.

4.2.2.3 Flexibility Results

After the triggering of the DR event by the aggregator, the optimal VPP configuration module calculates the optimal dispatch of the DERs belonging to the customers of the aggregator taking part in the DR event for the whole duration of the DR event. The dashboard provides the timeline of DR events with the amount of delivered flexibility and the amount of non-delivered, if any, aggregated from every DER of each prosumer taking part in the DR event fulfilment, as can be seen in Figure 17.

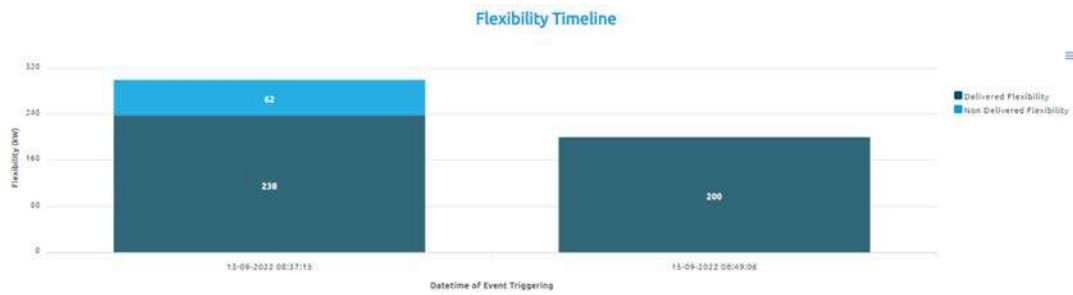


Figure 17: Flexibility timeline with delivered and non-delivered amount of flexibility per DR event.

Detailed results per DR event show how much each device has contributed to the fulfilment of the specific event, as shown in Figure 18.

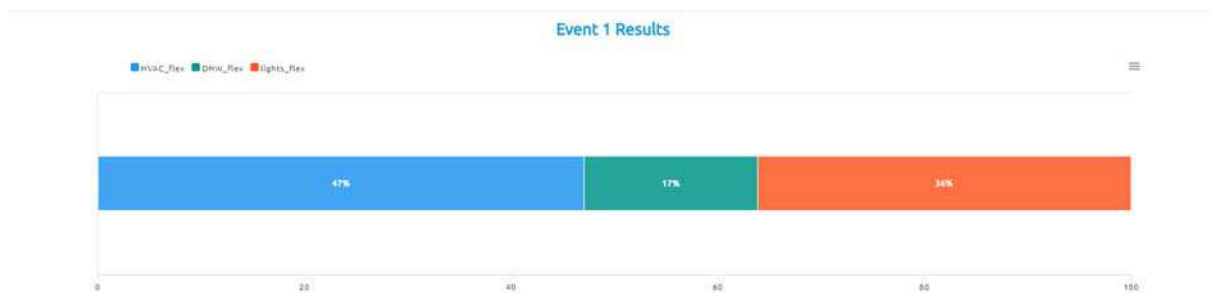


Figure 18: Percentage of DERs participation in specific DR event.

4.2.2.4 Flexibility Analytics

The dashboard provides visualization of useful analytics to the aggregators based on the results of the optimal VPP configuration. Figure 19 show the percentage of the available and the delivered flexibility per device for all the DR events.



Figure 19: Total percentage of each DER's available and delivered flexibility in DR events.

4.2.3 Smart Contract Monitoring/Handling Dashboard

4.2.3.1 Login

Utilizing the credentials provided from the project administrator, Aggregators can log in to the application by accessing the Login page that is presented in the Figure 20 below. We have to note that the EU logo and the acknowledgment have been included in all the following smart contracts dashboards screenshots, and those images presented in the deliverable without the EU logo and the acknowledgement are only due to focus the image on a specific part of the screen.

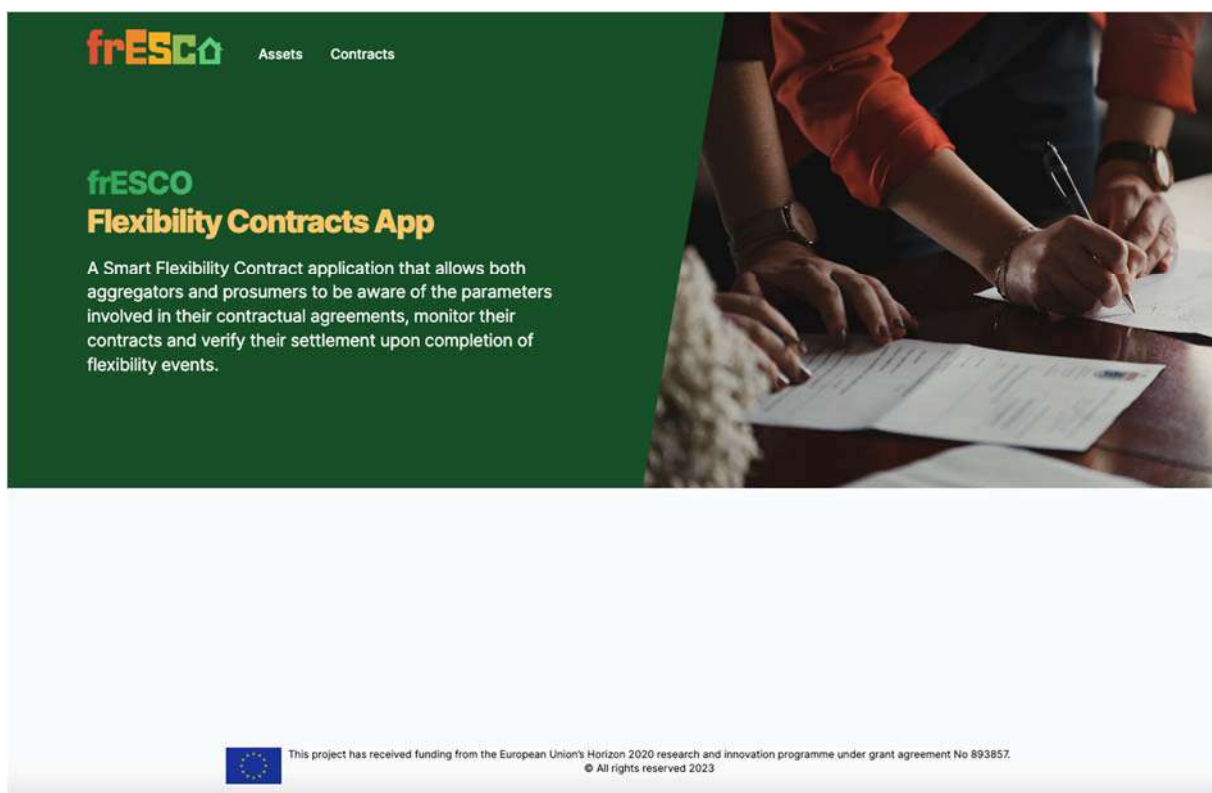


Figure 20: Login page for flexibility contracts application.

4.2.3.2 Assets

After logging in, a list of all flexibility assets is presented to Aggregators. This list contains only assets that each Aggregator has access to and can be utilized for the creation of the smart contracts. For each asset, additional attributes are presented to the Aggregator, such as name, type, location, flexibility capacity, and nominal power, as presented in the Figure 21 below.

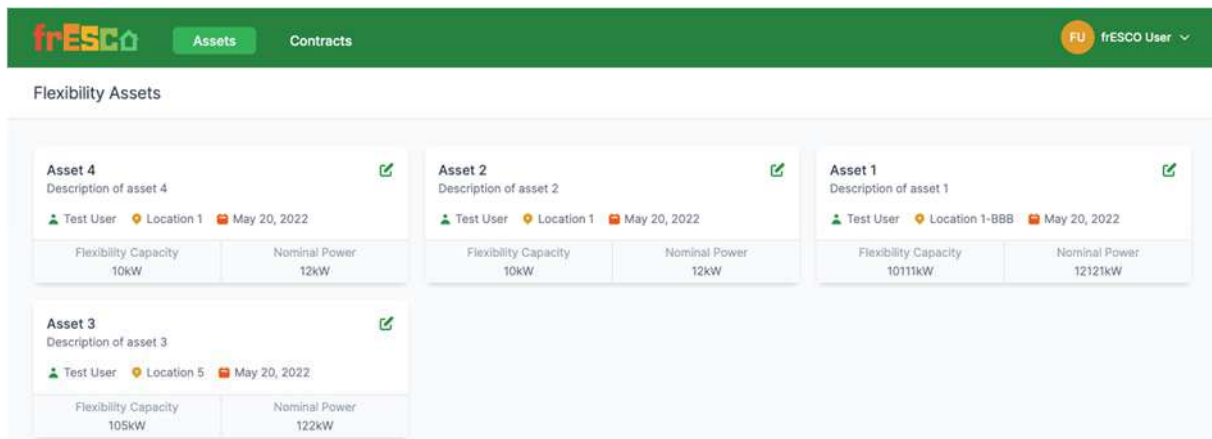


Figure 21: Flexibility Assets overview.

Selecting an asset, additional information such as the title, a description of the flexibility asset, the date that it was created in the app, as well as the asset provider is presented to the Aggregator. Editing the asset’s metadata is possible via the UI, with Aggregators being able to define the type of the asset, its location together with its flexibility capacity and nominal power, as shown in Figure 22.

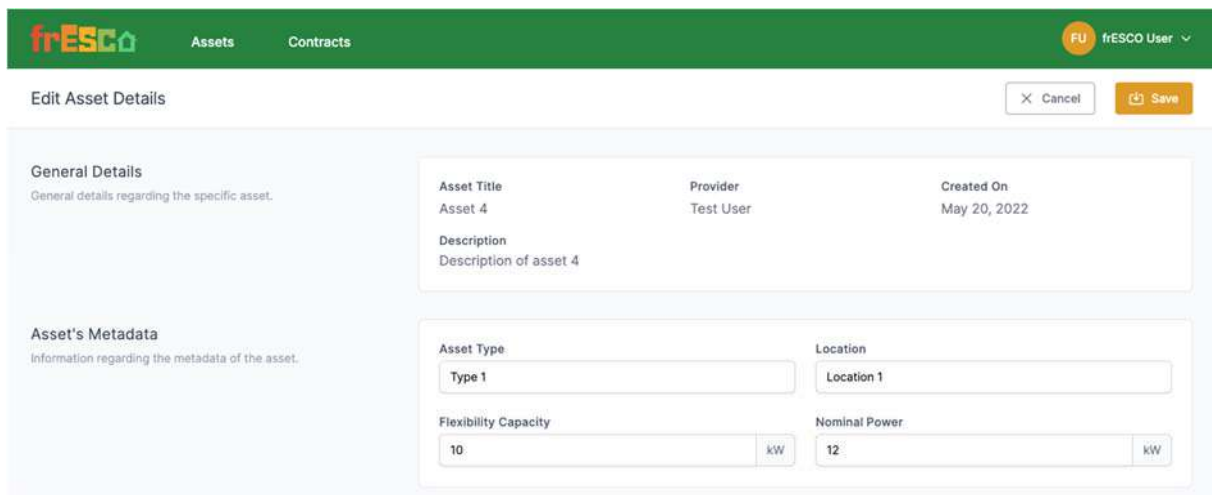


Figure 22: Asset details.

4.2.3.3 Contracts

Moving to the contracts tab, Aggregators are presented with a list of all signed contracts as demonstrated in Figure 23. Additional information per contract, related to the prosumer that the contract is signed with, the date that the contract was created, the number of assets that are included in the contractual agreement and the date up until the contract is valid, are

presented to Aggregators. Finally, an icon next to each contract shows its status, being active or expired.

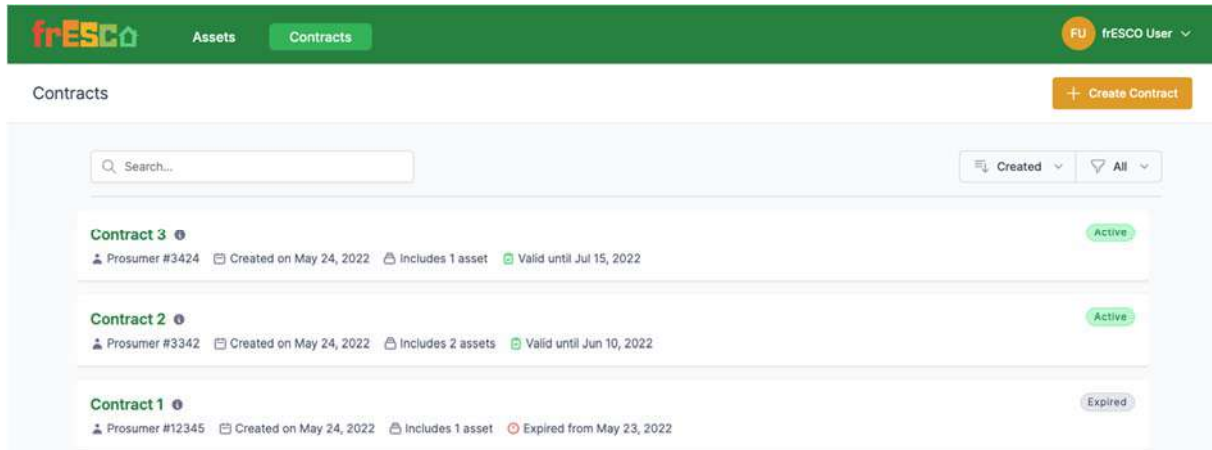


Figure 23: Contracts Overview.

To create a new contract, Aggregators can click on the “Create Contract” button and a new page is presented to them, to provide all the necessary information about the terms of the contract, as depicted in Figure 24.

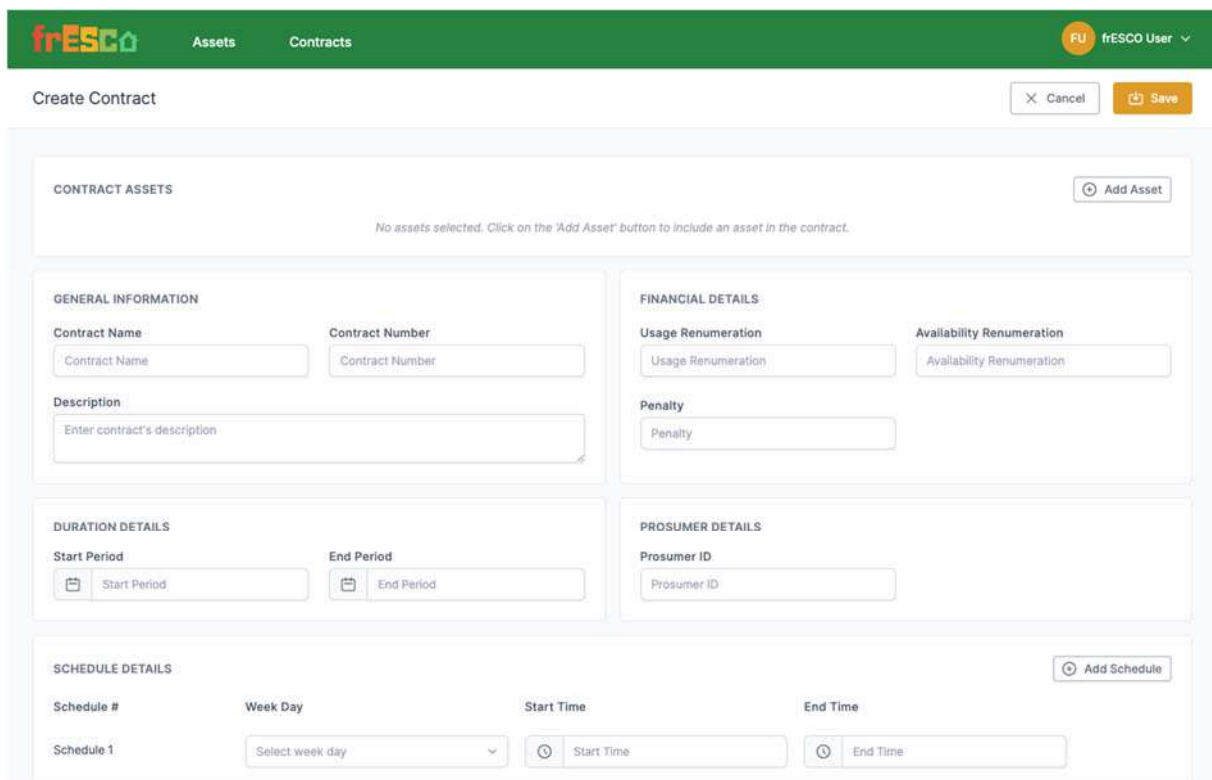
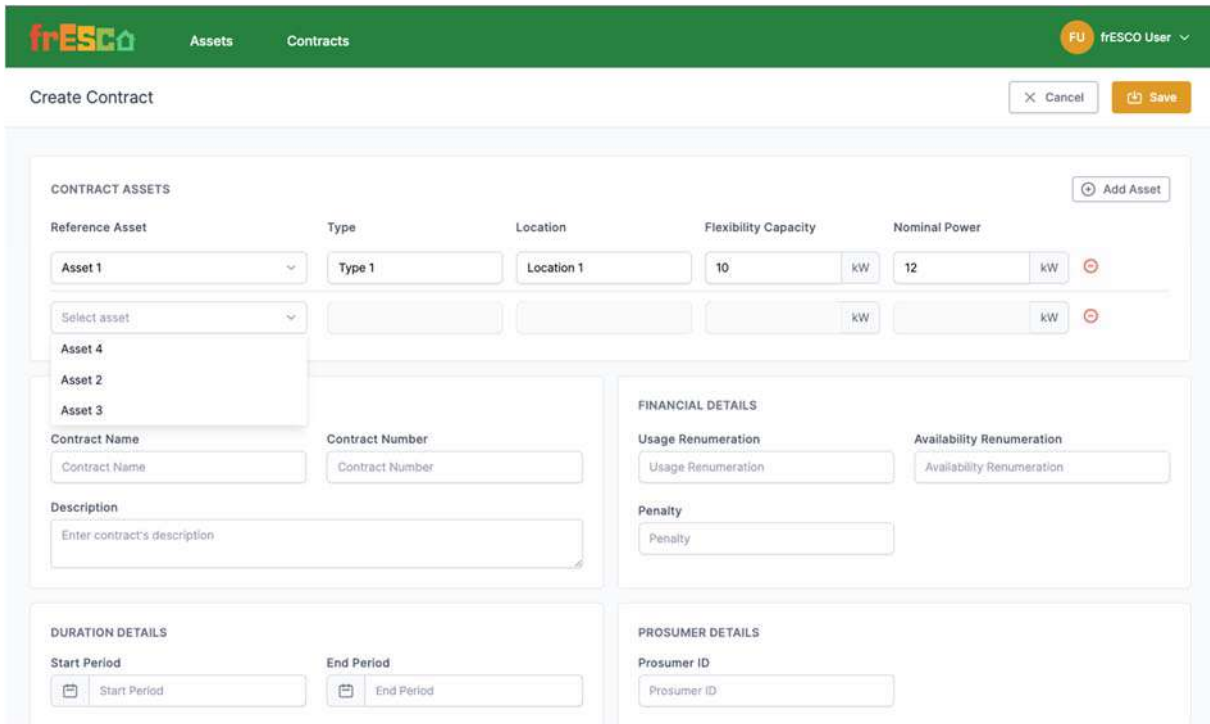


Figure 24: Contract Parameters definition.

After having selected one or multiple assets, the Aggregator can provide additional information per asset, as presented in Figure 25. The asset details (type, location, flexibility capacity and nominal power) are displayed in textboxes, so the Aggregator can easily modify them during the creation of a new contract.



The screenshot shows the 'Create Contract' interface in the frESCO system. The top navigation bar includes the frESCO logo, 'Assets', 'Contracts', and a user profile 'FU frESCO User'. The main form is titled 'Create Contract' and includes 'Cancel' and 'Save' buttons. The form is divided into several sections:

- CONTRACT ASSETS:** A table with columns for Reference Asset, Type, Location, Flexibility Capacity, and Nominal Power. It shows one asset selected: 'Asset 1' with Type 'Type 1', Location 'Location 1', Flexibility Capacity '10 kW', and Nominal Power '12 kW'. There is an 'Add Asset' button and a dropdown menu for selecting assets (Asset 1, Asset 2, Asset 3, Asset 4).
- Contract Name and Number:** Textboxes for 'Contract Name' and 'Contract Number'.
- Description:** A text area for 'Enter contract's description'.
- FINANCIAL DETAILS:** Textboxes for 'Usage Renumeration', 'Availability Renumeration', and 'Penalty'.
- DURATION DETAILS:** Textboxes for 'Start Period' and 'End Period' with calendar icons.
- PROSUMER DETAILS:** A text box for 'Prosumer ID'.

Figure 25: Assets included in the contract.

Lastly, the Aggregator provides the general details of the contract (name, number, and a short description), the duration of the contract, the financial aspects of the contract (remuneration/penalties), and the id of the respective prosumer, as shown in Figure 26. To save and created the new smart contract, the "Save" button must be pressed.

frESCO
Assets
Contracts
FU frESCO User

X Cancel
Save

Create Contract Add Asset

CONTRACT ASSETS

Reference Asset	Type	Location	Flexibility Capacity	Nominal Power
Asset 1	Type 1	Location 1	10 kW	12 kW
Asset 2	Type 2	Location 5	105 kW	122 kW

GENERAL INFORMATION

Contract Name: Contract 4
 Contract Number: 3242
 Description: Description of contract 4

FINANCIAL DETAILS

Usage Renumeration: 102
 Availability Renumeration: 243
 Penalty: 23

DURATION DETAILS

Start Period: 31/05/2022
 End Period: 14/07/2022

PROSUMER DETAILS

Prosumer ID: 4242

SCHEDULE DETAILS Add Schedule

Schedule #	Week Day	Start Time	End Time
Schedule 1	Monday	02:00	05:00

Figure 26: Information provision for the creation of the contract.

Figure 27 demonstrates the overview of a created contract, along with the indication its status (in this case “Active”), based on the duration details provided.

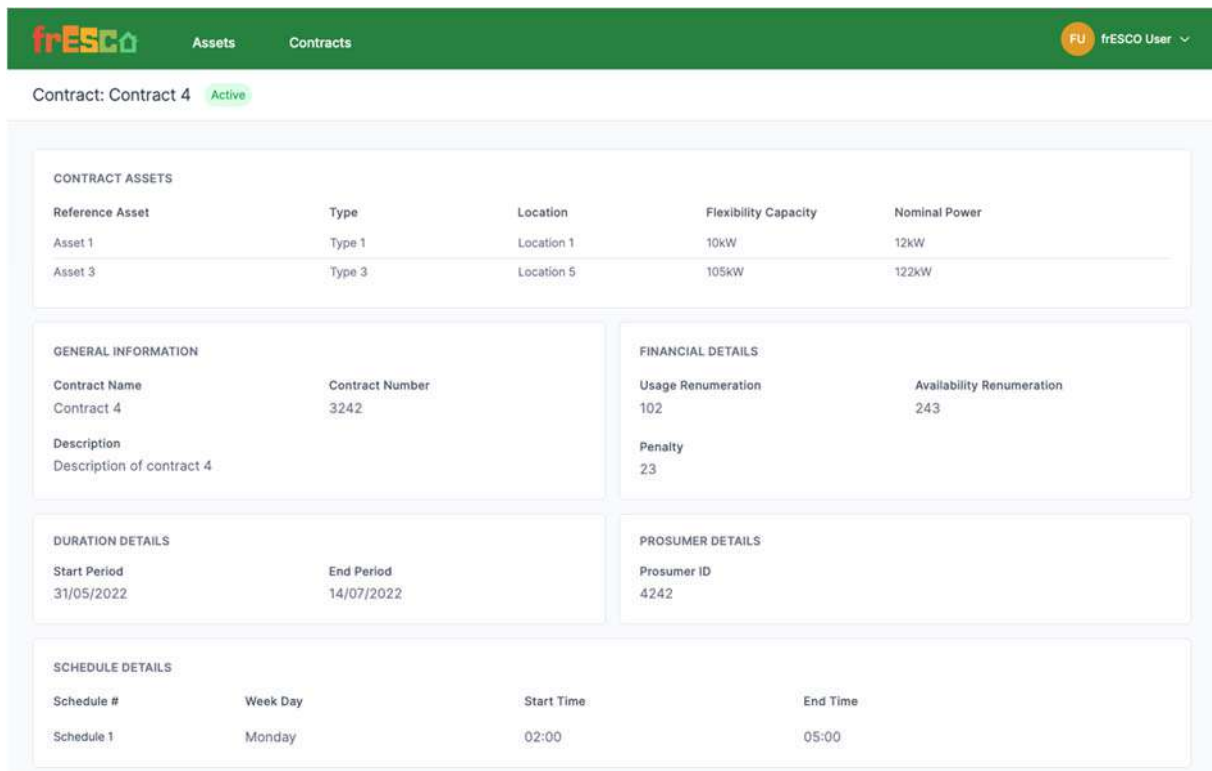


Figure 27: Contract Details overview.

4.3 UI for Prosumers/Consumers

Prosumers will have access to both LDM modules, such as asset energy performance/forecasting monitoring, personalized energy analytics and human centric automation, as well as the GDM module smart contract monitoring/handling. Thus, we distinguish between the dashboard that shows information regarding the LDM modules and the dashboard that shows information regarding the smart contract monitoring/handling module.

4.3.2 LDM's Dashboard

The following subsections show the main functionalities of the LDM's modules dashboard. In deliverable D5.4 a first version of the GUI was explained, but the new services defined for demos needs and the incapability of implement some of the services due to lack of data or some other issues, some changes were necessary. In this section, sometimes screenshots of the final GUI, and sometimes only the mock-ups of these changes will be shown. A final version of the GUI with all the screens and use cases description will be make available later on.

4.3.2.1 Login

The login screenshot is shown in Figure 28. Users will have their own user account with their own password. The accounts will be provided by the administrator as a previous configuration is needed. We have to note that the EU logo and the acknowledgment have been included in all the following prosumers dashboards screenshots, and those images presented in the deliverable without the EU logo and the acknowledgment are only due to focus the image on a specific part of the screen.

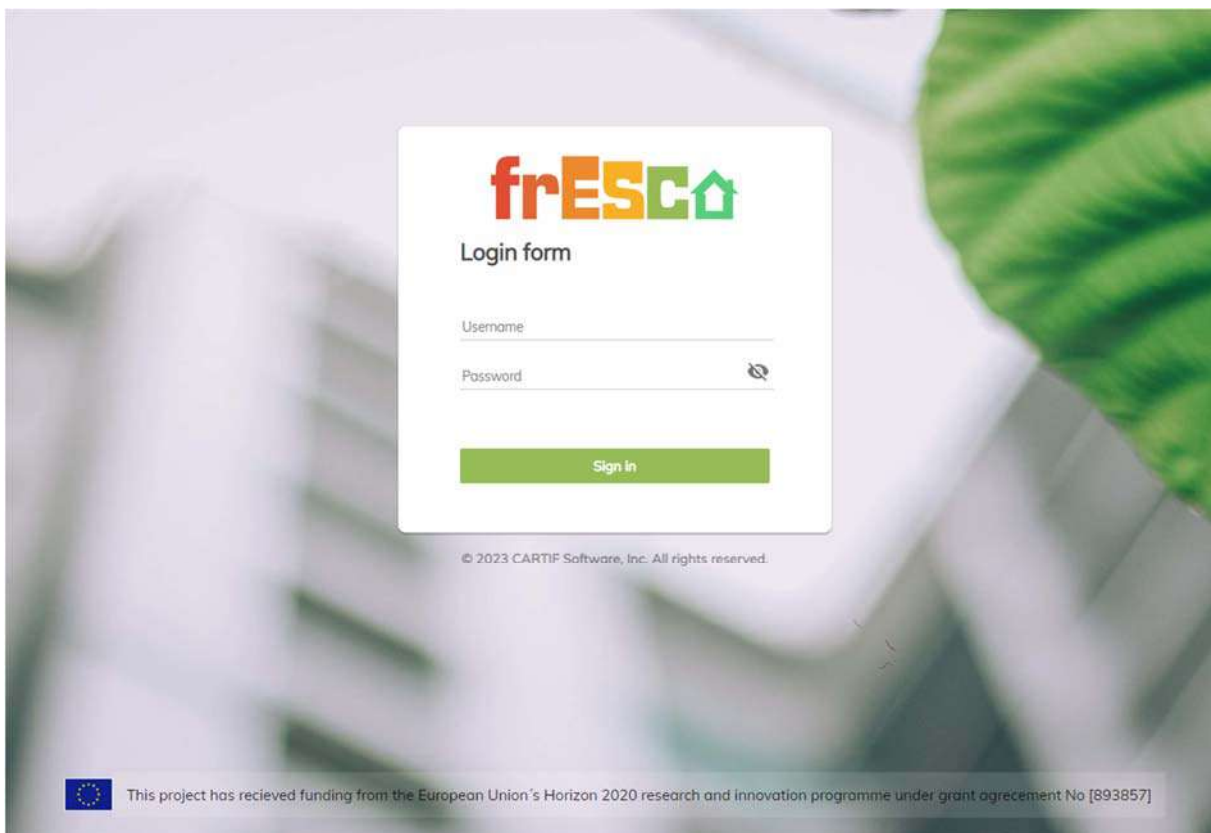


Figure 28: Login page for prosumers/consumers dashboard.

4.3.2.2 Monitored Data and Savings

One of the services the GUI provided is the representation of the monitored data (see Figure 29). The user can choose the period of time select between hourly, daily and monthly data. Moreover, information about costs and savings, both economical and of energy, are displayed.

The requirement 14 specifies that the system shall support the provision of monthly billing reports. Here, this requirement is fulfilled.



Figure 29: Monitored data screen for prosumers/consumers dashboard.

4.3.2.3 Recommendations

In the recommendation screen (see Figure 30) users will see some suggestions on behaviour change such as non-automatable devices schedule. In this regard, users will be able to configure these devices clicking in a configuration button (see Figure 31).

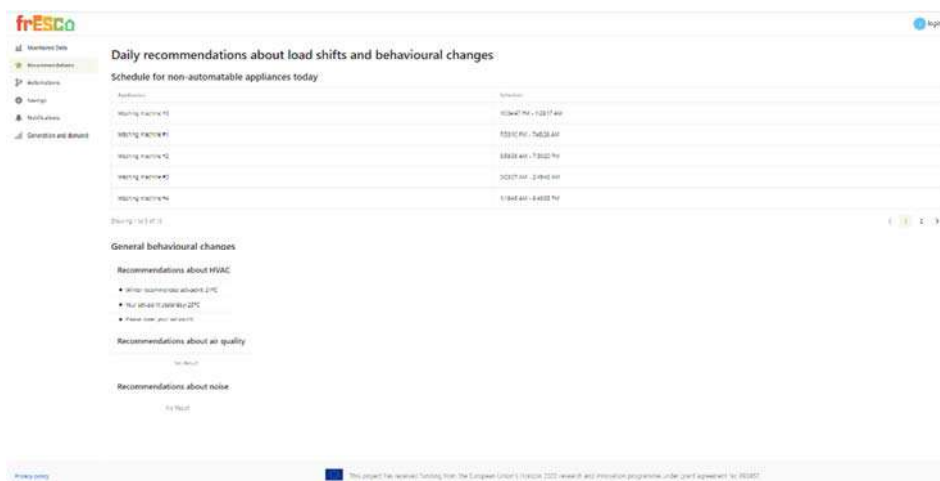


Figure 30: Recommendations screen for prosumers/consumers dashboard.

DER Configuration (non-automatable)

Appliances	Days to be used	Power (kW)	Duration (min)	Modify
Washing machine	Sun (1), Thu (2)	1.234	10	
Dishwasher	Sun (1), Mon (2), Tue (1), Fri (1)	5.23	55	
Iron	Sat (1), Mon (1), Tue (2), Wen (2), Thu (2)	3.322	29	
Electric kitchen	Sat (2), Sun (1), Mon (1), Thu (2), Fri (1)	3.17	3	

[Add DER](#)

Showing 1 to 4 of 4 < 1 >

Figure 31: Settings screen for non-automatable devices recommendation for prosumers/consumers dashboard.

4.3.2.4 Automations

In the automation screen three functionalities will be available. The first one, the user schedule (see mock-up in Figure 32). Here the user will be able to schedule his/her own controllable devices (see Figure 33) that will turn on/off accordingly.

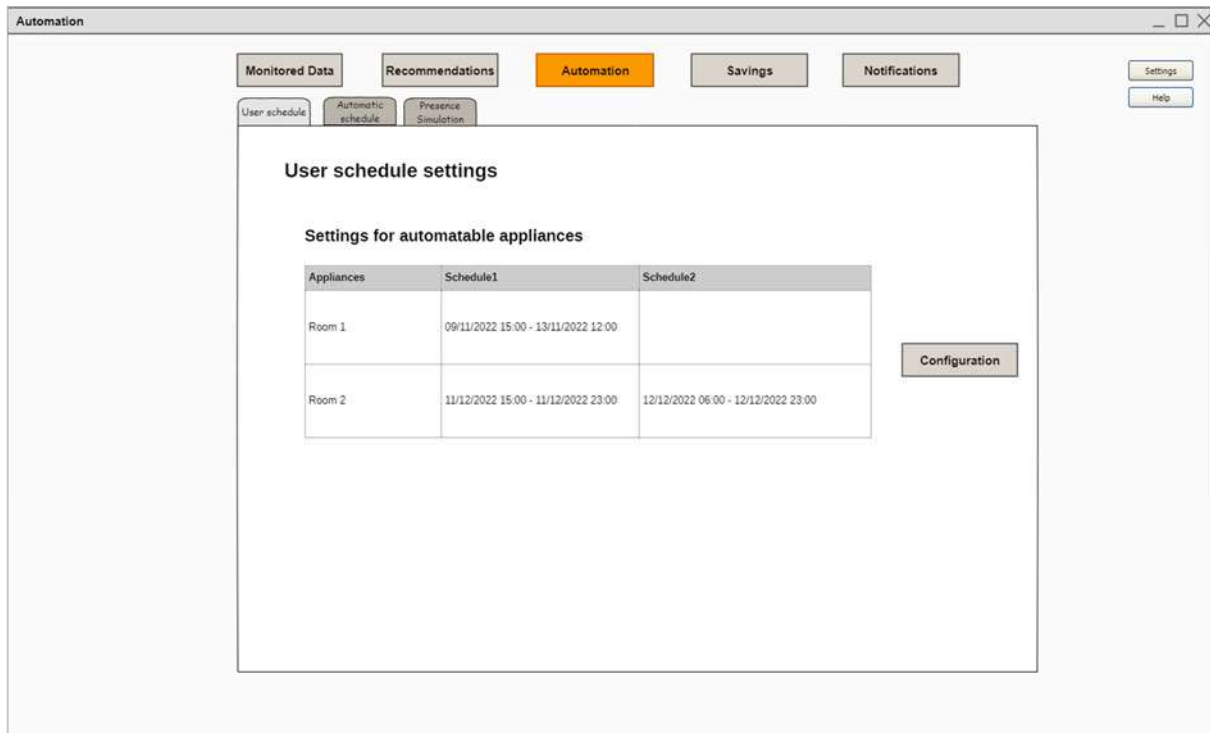


Figure 32: User schedule screen for automation for prosumers/consumers dashboard.

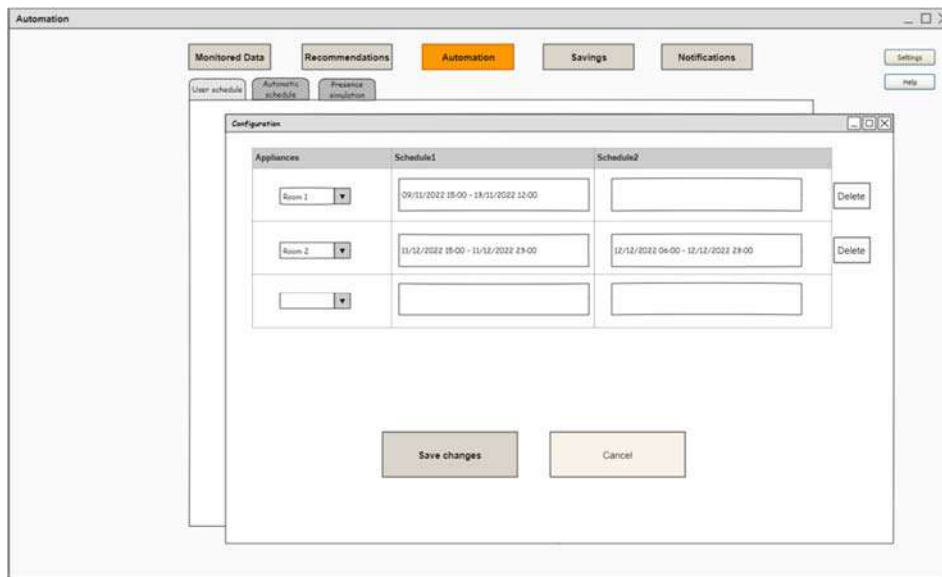


Figure 33: User schedule configuration screen for automation for prosumers/consumers dashboard.

The second functionality is a screen as in Figure 34 where users can see the automations Schedule for the day.

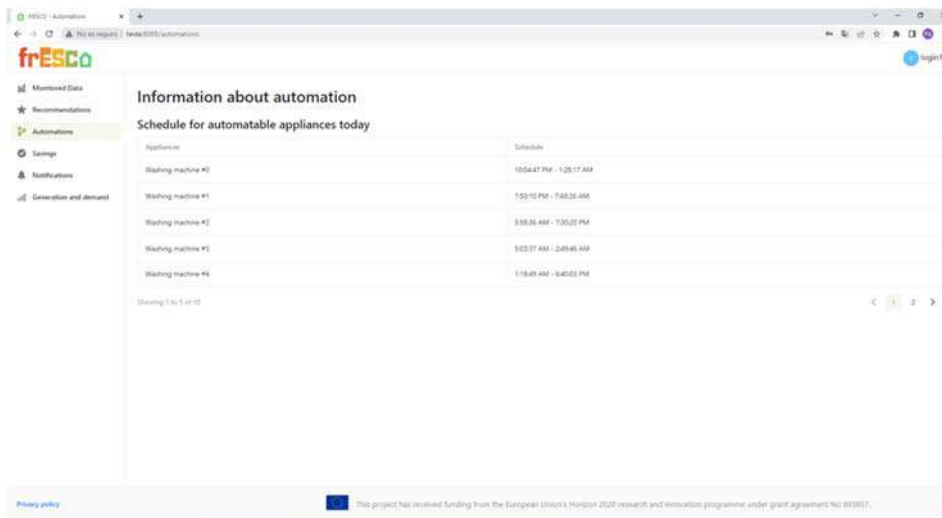


Figure 34: Schedule for automation screen for prosumers/consumers dashboard.

The last functionality is the setting of the presence simulation service, where the user will be able to include the parameters of the presence simulator (see Figure 35).

Simulation profiles

Simulation status Date range * 1970-01-01 - 1970-01-01

Profile name *	Time Range *	Min. number of activations *	Max. number of activations *	Min. duration (min) *	Max. duration (min) *
Dawn	AM 07:25 - AM 09:55	1	3	15	35
Dust	PM 08:25 - PM 10:55	2	4	15	35
Happy potato	AM 08:00 - PM 05:00	90	90	3	3
Juan	PM 10:58 - PM 11:52				

Figure 35: Presence simulation screen for prosumers/consumers dashboard.

4.3.2.5 Notifications

In the notifications screen (see Figure 36) users can see the notification they have received from the Personalized Energy Analytics Module services. If they click on one of the notifications listed, they will see a screen like the one shown in Figure 37, with basic information, the problem that triggered the notification and some recommendations to correct the situation, displayed.

The screenshot shows the frESCO web interface. At the top left is the frESCO logo. At the top right is a user profile icon with the text 'login1'. A vertical sidebar on the left contains navigation items: 'Monitored Data', 'Recommendations', 'Automations', 'Savings', 'Notifications' (highlighted in green), and 'Generation and demand'. The main content area is titled 'Notifications' and contains a 'List of notifications' section. This list includes two entries: 'Air Quality' and 'Noise', each with a magnifying glass icon. At the bottom of the page, there is a footer with a 'Privacy policy' link on the left, a small European Union flag in the center, and a text block on the right stating: 'This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 893857.'

Figure 36: Notifications screen for prosumers/consumers dashboard.

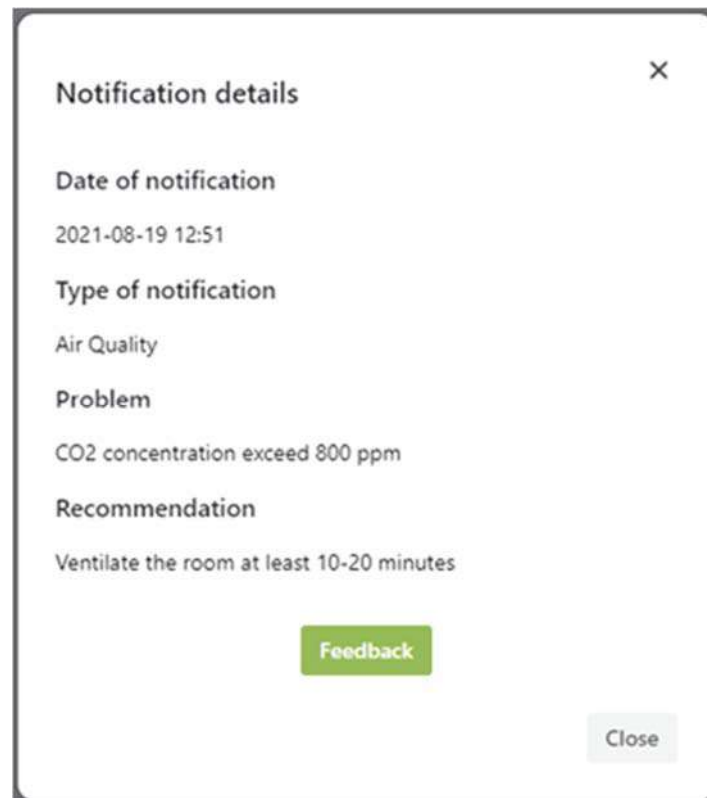


Figure 37: Notifications details from the prosumers/consumers dashboard.

4.3.2.6 Generation and Demand

This section shows the dashboards related to the developed module in terms of generation and demand at the dwelling level.

These dashboards are developed under Power BI platform, and they are integrated into the main GUI of the LDM.

The next figure shows the main panel of this app. Here, the user can press the related button and navigate through the panels.

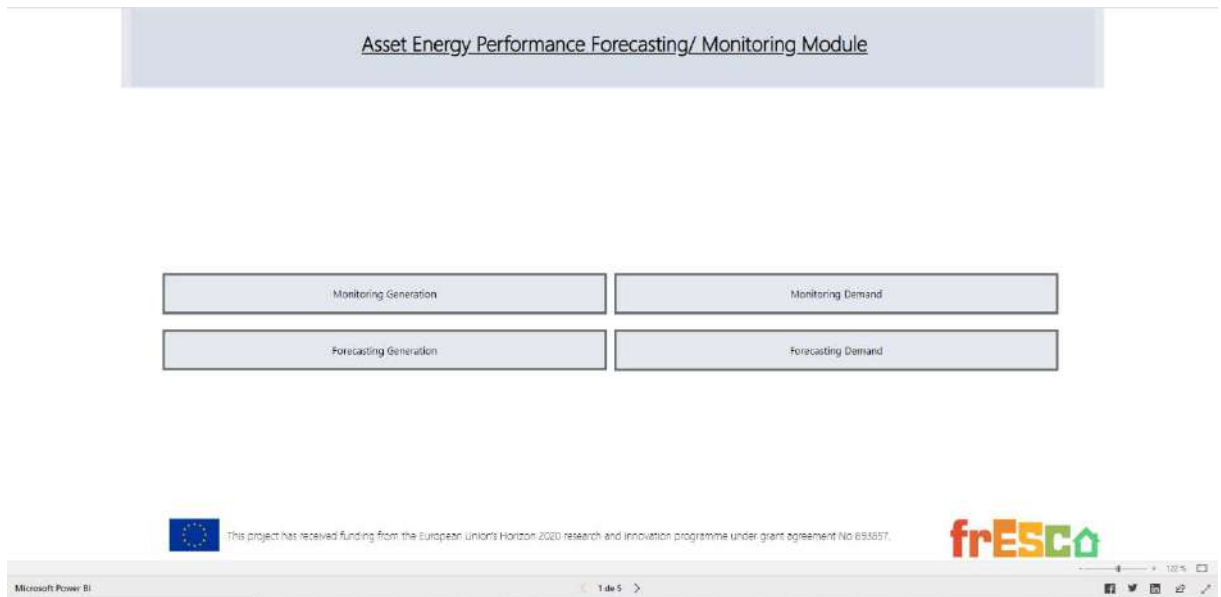


Figure 38: Main Panel.

The next figure shows the main dashboard related to the monitoring of the generation at the dwelling level.

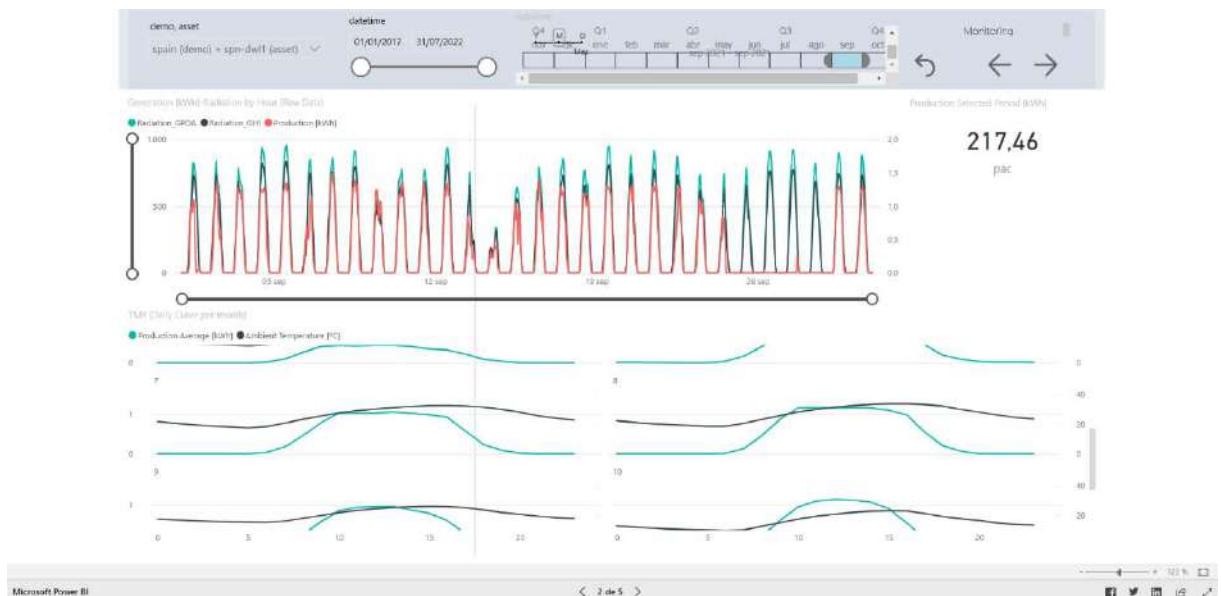


Figure 39: Generation Monitoring at dwelling level Mockup.

In the top selectors, it is possible to select the dwelling to be analysed (according to the granted access of the user) and the period to be displayed. In the lower visuals, you can see the accumulated production in the selected period, the production data's time evolution, and

other analysed variables (i.e radiation). It is also possible to see the typical daily production curve for each month and other influencing variables (i.e. ambient temperature).

The next figure shows the main dashboard related to the forecasting of the generation at the dwelling level.

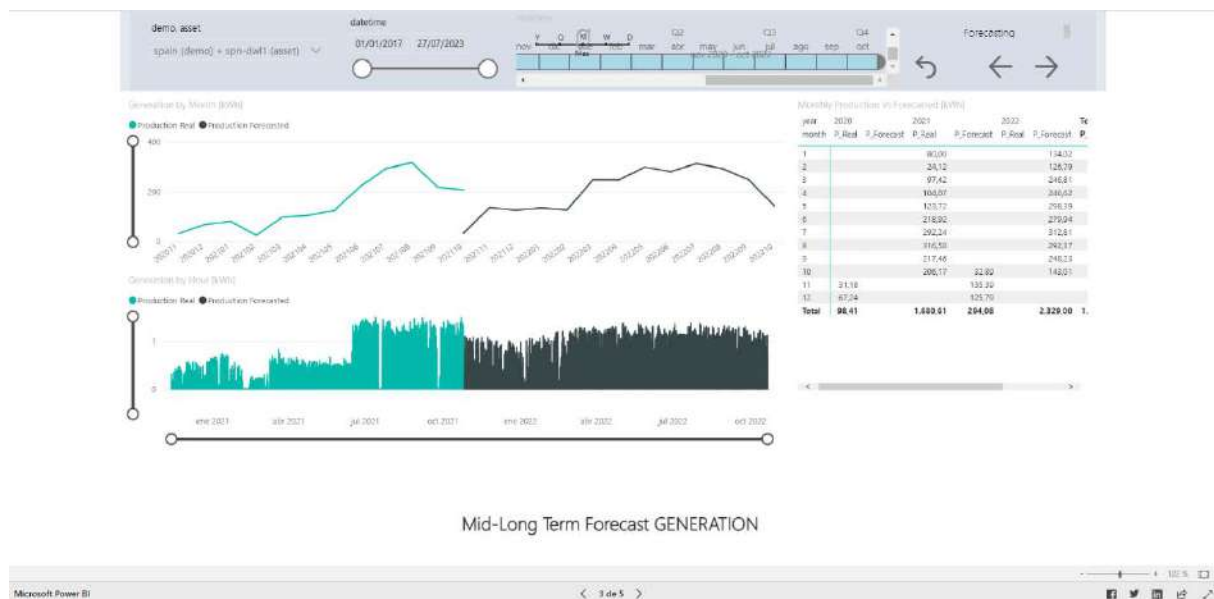


Figure 40: Generation Forecasting at dwelling level Mockup.

In the top selectors, it is possible to select the dwelling to be analysed (according to the granted access of the user) and the period to be displayed. The lower visuals show the production in the selected period and the forecast 12 months ahead, the production data's time evolution compared with the forecast per month. It is also possible to display the production and forecast on an hourly basis.

The next figure shows the main dashboard related to the monitoring of the demand at the dwelling level.

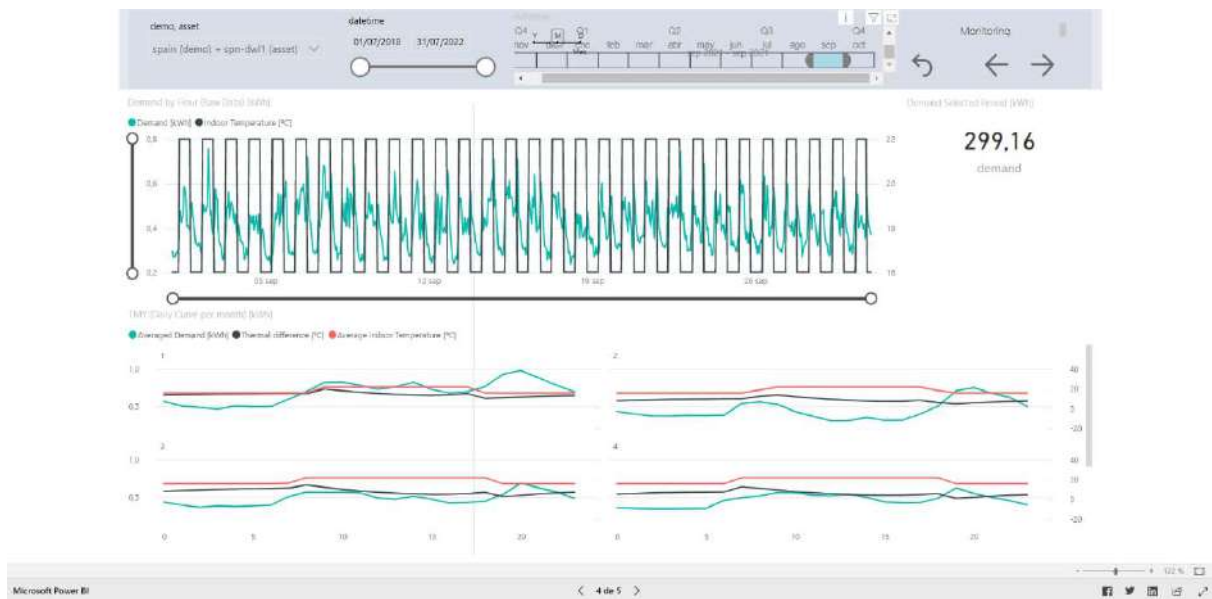


Figure 41: Demand Monitoring at dwelling level Mockup.

In the top selectors, it is possible to select the dwelling to be analysed (according to the granted access of the user) and the period to be displayed. In the lower visuals, you can see the accumulated demand in the selected period, the demand data's time evolution, and other analysed variables (i.e., Indoor temperature). It is also possible to see the typical daily demand curve for each month and other influencing variables (i.e., indoor temperature and thermal difference from external temperature).

The next figure shows the main dashboard related to the forecasting of the demand at the dwelling level.

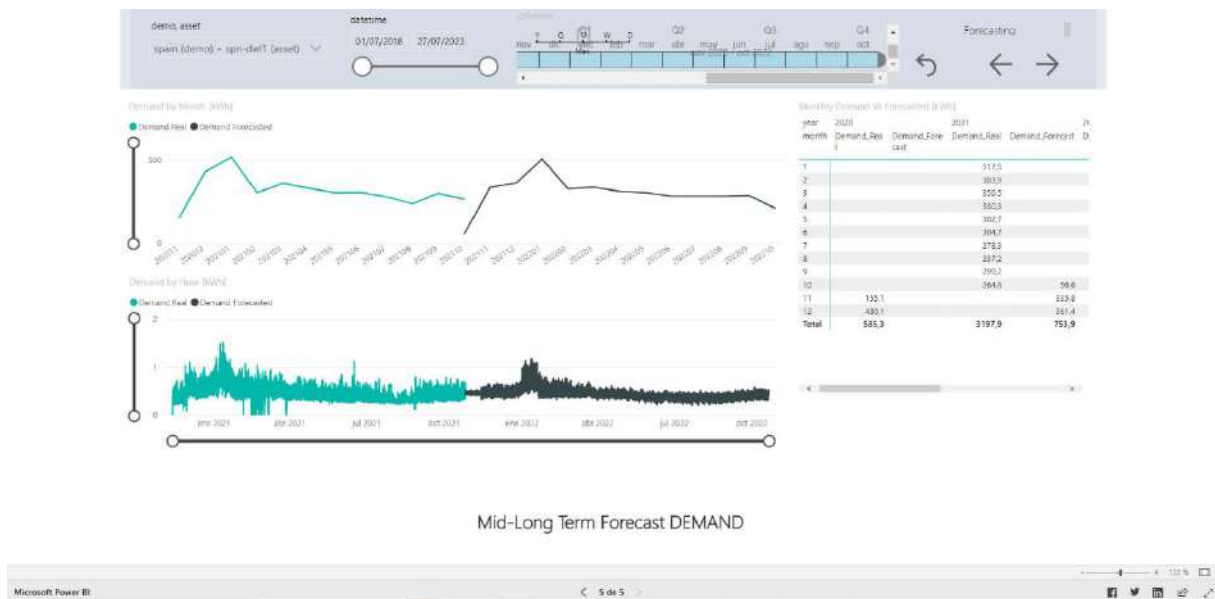


Figure 42: Demand Forecasting at dwelling level Mockup.

In the top selectors, it is possible to select the dwelling to be analysed (according to the granted access of the user) and the period to be displayed. The lower visuals show the demand in the selected period and the forecast 12 months ahead, the demand data's time evolution compared with the forecast per month. It is also possible to display the demand and forecast on an hourly basis.

4.3.2.7 Settings

By clicking on settings, users can access the configuration of the notification, where they will include the e-mail address, they will receive the notification information (see Figure 43).

Name	E-Mail	Verified	Actions
Sail	sdh@...	Yes	Delete
Page	vdh@...	Yes	Delete

Figure 43: Notification setting screen from the prosumers/consumers dashboard.

Moreover, users can set their own comfort preferences (Figure 44) and manage the services they are subscribed at (see mock-up in Figure 45).

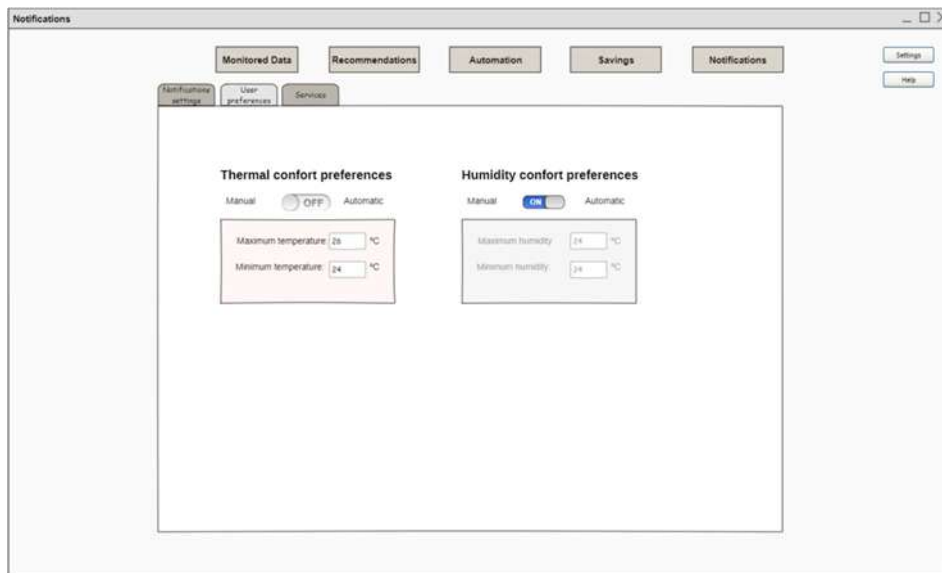


Figure 44: User preference screen from the prosumers/consumers dashboard.

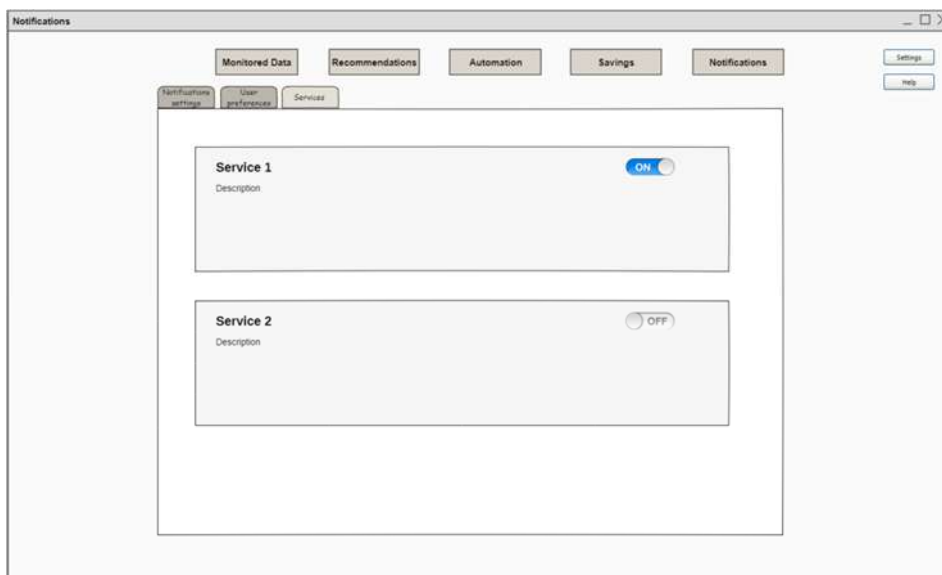


Figure 45: Service management screen from the prosumers/consumers dashboard.

4.3.2.8 Help

A help button will be available, where users can find useful information about how the web works, and an explanation of the different functionalities the GUI provide.

4.3.3 Smart contract monitoring/handling Dashboard

The following subsections show the main functionalities of the prosumer's perspective for the smart contract monitoring/handling dashboard.

4.3.3.1 Login

To login to the application, Prosumers need to utilize the credentials they received by the project administrator in the Login Page, as can be seen in Figure 20. We have to note that the EU logo and the acknowledgment have been included in all the following smart contracts dashboards screenshots, and those images presented in the deliverable without the EU logo and the acknowledgement are only due to focus the image on a specific part of the screen.

4.3.3.2 Prosumer's Perspective

As soon as they successfully login, they are presented with an overview of the contracts that are relevant to them, as shown in Figure 46, an identical view to what the Aggregator sees; prosumers do not have the option to modify the contract's parameters though.

Contract: Contract 4 Active

CONTRACT ASSETS				
Reference Asset	Type	Location	Flexibility Capacity	Nominal Power
Asset 1	Type 1	Location 1	10kW	12kW
Asset 3	Type 3	Location 5	105kW	122kW

GENERAL INFORMATION		FINANCIAL DETAILS	
Contract Name	Contract Number	Usage Renumeration	Availability Renumeration
Contract 4	3242	102	243
Description		Penalty	
Description of contract 4		23	

DURATION DETAILS		PROSUMER DETAILS	
Start Period	End Period	Prosumer ID	
31/05/2022	14/07/2022	4242	

SCHEDULE DETAILS			
Schedule #	Week Day	Start Time	End Time
Schedule 1	Monday	02:00	05:00

Figure 46: Contract details overview for prosumers.

5 FUTURE DEVELOPMENT & IMPROVEMENTS

As already mentioned, the integration of data from demo sites has not finished yet, so the modules have not been tested with real data. Moreover, some data availability is still under discussion, which means that minor changes can be done from the modules developers to deal with potential data unavailability per demo site case. Although the architecture of the frESCO multi-service package toolkit, as well as the basic methodology described in this deliverable will remain the same in the future, the dashboards will be enhanced with further features and functionalities when the data become available.

Specifically, the exact future improvements of the modules will be the following when the data from the demo sites become available in the platform and the integration between the modules reaches the final level:

- The modules will be tested and debugged with historical and real-time data coming from specific buildings of the demo sites. This means that modules, such as energy management analytics will have available data for monitoring from different buildings of the demo sites and the clustering will be enhanced with real data rather than mock data.
- With the utilisation of the uploaded demos' datasets, the fine-tuning of the frESCO analytics, as presented in D4.5 [20] and D4.6 [21], can be triggered. This means that all modules that get as inputs the results of analytics will be updated with real data and the results of the modules will be realistic. An example is the optimal VPP configuration module, which is going to be applied with real short term available flexibility forecasts.
- When the integration with demo sites is done, then the human centric automation module will be able to send automation commands to available DERs for explicit DR participation and DER scheduling and efficiency strategies. Specifically, DHW heaters in the Greek demo bungalow and HVAC systems in the Spanish demo dwellings will be available for automatic operation. Actuation commands will be sent directly from the automation module to the Energy Boxes using the platform MQTT broker.

6 CONCLUSIONS

To summarize, this deliverable, covering tasks 5.1, 5.2, 5.3, 5.4, 5.5 and 5.6 has presented the methodologies towards the final form of the energy efficiency and flexibility services and the visualizations through the dashboards for the frESCO stakeholder, namely prosumers, ESCOs and aggregators. The final architecture of the LDM and the GDM and the dependencies between the modules give a clear picture and a basic understanding to the end-users of the benefits that they can get if they adopt these services. However, we have to mention that as the data integration progresses, modifications to the modules might be done.

The added value of this deliverable is that its results can be used in conjunction with the results from WP3, Novel Performance-based Business Models and Verification Method for bundled energy services, towards the design, development, and deployment of new business P4P services, generating even more significant benefits for all parties involved.

In the next period, demo leaders and supporting technical partners should utilise the BDP and the full spectrum of its functionalities and features to upload data from the demo sites, so that the platform's database can be expanded with datasets that the modules can utilise to accomplish the frESCO project goals.

REFERENCES

- 1 frESCO Consortium. (2020). frESCO D2.5 “Report on the frESCO conceptual architecture”
- 2 frESCO Consortium. (2020). frESCO D5.1 “Monitoring, forecasting, energy management analytics, flexibility analytics and optimisation mechanisms”
- 3 frESCO Consortium. (2022). frESCO D5.2 “Release of the set of monitoring, forecasting, analytics and optimisation modules”
- 4 frESCO Consortium. (2022). frESCO D5.3 “Algorithms for Human-Centric automation tool”
- 5 frESCO Consortium. (2022). frESCO D5.4 “Release of the Human-Centric automation module”
- 6 frESCO Consortium. (2022). frESCO D5.5 “Definition of blockchain mechanisms enabling smart contract monitoring”
- 7 frESCO Consortium. (2022). frESCO D5.6 “Release of the module for Smart Contract monitoring and management”
- 8 frESCO Consortium. (2021). frESCO D3.1 “Definition of the novel energy services for residential consumers”
- 9 H. Hersbach et al., “ERA5 hourly data on single levels from 1979 to present,” Copernicus Climate Change Service (C3S) Climate Data Store (CDS), 2018. <https://cds.climate.copernicus.eu/cdsapp#!/dataset/reanalysis-era5-single-levels?tab=overview> (accessed May 26, 2022)
- 10 https://www.idae.es/uploads/documentos/documentos_BIENESTAR_TERMICO_EN_UN_ESPACIO_CLIMATIZADO_2_articulo_ASV_3725727c.pdf
- 11 Y. E. UNUTMAZ, A. DEMİRCİ, S. M. Tercan, and R. Yumurtaci, “Electrical Energy Demand Forecasting Using Artificial Neural Network,” in *2021 3rd International Congress on Human-Computer Interaction, Optimisation and Robotic Applications (HORA)*, 2021, pp. 1–6, doi: 10.1109/HORA52670.2021.9461186.

- 12 L. S. Cohen, J., Cohen, P., West, S. G., & Aiken, *Applied multiple regression/correlation analysis for the behavioral sciences*, 3rd ed. Routledge, 2002.
- 13 Perez, R., Ineichen, P., Seals, R., Michalsky, J., Stewart, R., 1990. Modeling daylight availability and irradiance components from direct and global irradiance. *Solar Energy* 44 (5), 271-289
- 14 <https://www.keycloak.org/>
- 15 <https://vuejs.org/v2/guide/>
- 16 <https://nestjs.com/>
- 17 <https://www.postgresql.org/about/news/postgresql-14-released-2318/>
- 18 <https://ethereum.org/en/developers/docs/smart-contracts/>
- 19 frESCO Consortium. (2020). “frESCO Grant Agreement”
- 20 frESCO Consortium. (2022). frESCO D4.5 “frESCO Baseline Data Analytics – Draft Release”
- 21 frESCO Consortium. (2022). frESCO D4.6 “frESCO Integrated Platform – Beta Release”

ANNEX A: DATA INTEGRATION AND MODULES UPDATES

This chapter provides the updates on the different modules of the frESCO architecture since the first submission of the document in January 2023. More specifically, the following sections include the description of the actions that took place for the adaptation of the different applications to cover the needs of the different pilot sites, and adapt to the specificities of the various and heterogeneous data streams from the frESCO demonstration sites.

A.1 Local Demand Manager Modules

A.1.1 Asset Energy Performance Forecasting/Monitoring Module

A.1.1.1 Data integration actions

Regarding data integration actions, a great effort has been made to collect real-time data from all measuring devices included in the project demo sites at dwelling level.

To this end, the following actions have been carried out:

- Generation of ad hoc algorithms for each of the type of datasets existing in the BDP and from external sources (live data and static data). The main functions of these algorithms include:
 - Identification of the fields of interest in the complex structure of the data.
 - Signal conditioning tasks mainly related to unit conversion and the creation of hourly aggregations.
 - Merging and synchronizing data from different data sources (measuring devices from each dwelling from the BDP, meteorological data from external APIs, external files including historical data)
 - Periodic data collection and storage in a local database associated with the specific application.

A.1.1.2 Module updates

The main updates to the module in the last stage of the project have focused on the following points:

- Final fine-tuning of machine learning models.
- Task automation and synchronization with the UI.
- Update of the UI.

Regarding the last adjustments of machine learning models oriented to generation and demand forecasting, ANN and GBR in regressive mode have been selected to estimate generation and demand from the related variables. For this purpose, the models have been trained with all the historical data available provided by demo partners in each demo (for each dwelling) together with the meteorological data obtained externally.

For the correct integration of the data in the UI and the generation of both monitoring and forecasting results, it has been necessary to encapsulate all the tasks of the module in separate functions. These functions have been converted into automated tasks within the virtual machine defined in the architecture described in Section 3.2.2.2. These tasks are executed daily and include:

- Data collection from demos.
- Generation of new forecasts from the last data received (live data).
- Storage in the cloud database associated with this module.
- Close to real-time connection with the UI.

In addition to what has already been described in Section 4.3.2.6, some modifications have been made to the UI. Firstly, some changes have been done to improve the navigation and integration into the main UI at a single-user level as it is requested in the UI for prosumers/consumers (see Section 4.3).

The developed UI oriented to show generation and demand at the dwelling level was modified according to the feedback from the end users. In this sense, a new dashboard is added (see Figure 47) that allows the user to compare generation and demand for the dwelling combined in the same control panel.

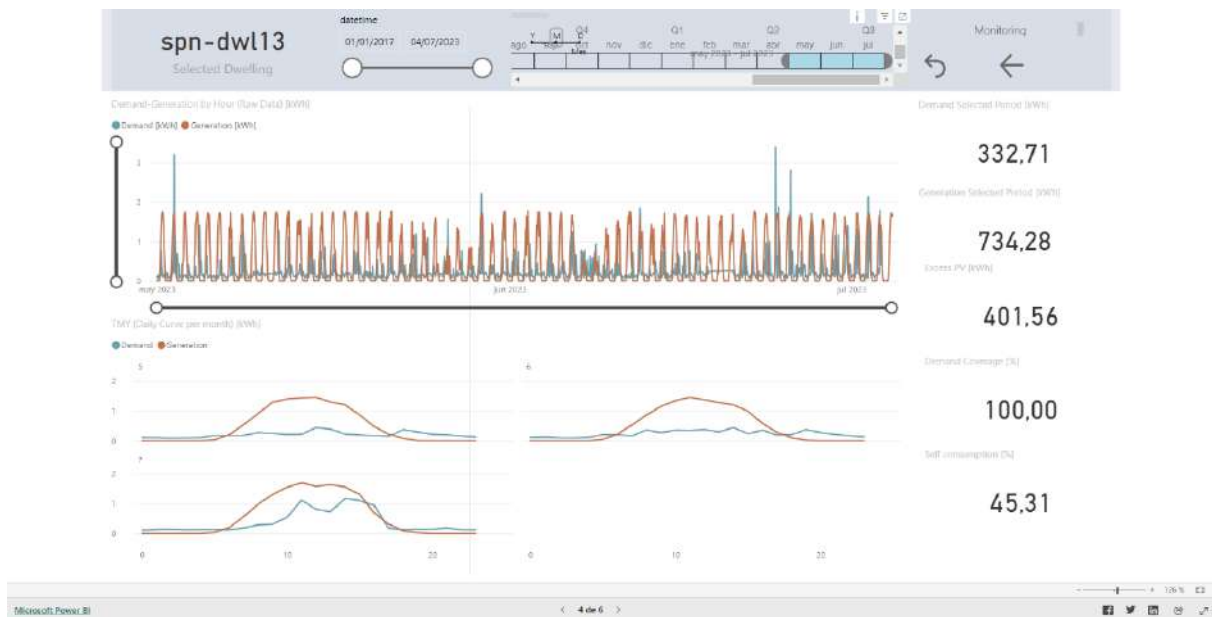


Figure 47: Demand and Generation Monitoring at dwelling level.

In this new control panel, the user can visualize:

- Comparative graph of generation and demand on an hourly basis.
- Comparative graph of generation and demand as a daily average curve for each month.
- Generation selected period: Total sum of generation for the period analyzed.
- Demand selected period: Total sum of demand for the period analyzed.
- PV surplus: Difference between generation and demand for the selected period.
- Demand coverage: Percentage of demand that can be covered by the PV generation (in an ideal scenario).
- Self-consumption: Percentage of PV generation that can be used to cover the demand.

Taking into consideration these updates, we can consider that the user requirements and the functional requirements provided in D2.5 and cited in Section 3.2.2.3 (Req_002,Req_006, Req_013 and Req_025) are “Fully Implemented” and as a result, the functionalities related to the monitoring and long term forecast of generation and demand at the dwelling level are fully addressed.

A.1.2 Personalized Energy Analytics Module

A.1.2.1 Data integration actions

The personalized energy analytics module includes three types of functionalities: monitoring, recommendations, and notifications. The recommendations, in its turn, are divided in several services, already explained in Section 3.2.3.2.

All data required in this module needs to be gathered from each of the users. Here, we understand users as each of the residents that will be using the UI in which all these functionalities are integrated (see sections 4.3 and A.1.2.2). This includes individual prosumers and consumers (dwellings from Croatia and Spain) and a specific user that includes common areas in a Spanish pilot building, and a 5-room bungalow in a hotel (Greek demo site user). Even if all data is required from all users, not all of them have all the data available. For those users that have not the required information, whether it is due to missing or failing equipment, such as sensors or energy meters, or due to any other reason, there will not be the possibility to have access to those services or functionalities that use this data. Below, there is a summary of all data required for the model:

- Historical energy consumption: retrieved once a day from the BDP to be used for monitoring.
- Energy prices: gathered from each demo responsible to be used for monitoring and recommendations.
- Energy consumption 24-hour forecasts: to be gathered from the BDP and yet to be available for some of the users. To be used for recommendations.
- PV energy generation 24-hour forecasts: to be gathered from the BDP and yet to be available for some of the users. To be used for recommendations.
- Current battery SoC: retrieved from the BDP to be used for recommendations and one of the notification services.
- Static data, such as installed battery capacity, and grid contractual limits (both consumption and selling energy): retrieved from static data from the BDP or through each demo responsible.

- Information about non-automatable appliances: to be retrieved from the consumers/prosumers UI to be used for recommendations.
- Air quality information: retrieved from the BDP for the air quality related notification services.
- Current HVAC energy consumption: to be retrieved from the BDP to be used for those notification services that require the status (ON/OFF) of the HVAC.
- Current indoor temperature: to be retrieved from the BDP to be used for some notification services.
- Information about room occupancy status (only for Greek user): retrieved directly from the consumers/prosumers UI and to be used for some notification services.
- Current humidity levels: retrieved from the BDP to be used for some notification services.
- Current dwelling energy consumption: retrieved from the BDP used for one of the notification services.
- Current PV generation: retrieved from the BDP used for one of the notification services.
- Comfort parameters: to be collected from the users through the consumers/prosumers UI and from the BDP to be used for the notification and automation services.

All data retrieved from the consumers/prosumers UI is introduced by the user through a user-friendly interface and stored in an internal database to be used by the algorithms associated to each functionality and service. The same happens with all static information provided by the demo responsible partners.

As for data retrieved from the BDP, it is accessed through an API that get the necessary attributes filtered by the necessary parameters (typically, user ID and timestamp) to gather the information once a day, or with a specific periodicity (currently, from 5 to 15 minutes) depending on the functionalities and services' needs. All monitored related information is stored in the local database for a fast access when a display in the interfaces is required (see Figure 29).

A.1.2.2 Module updates

Most of the developed functionalities within the consumers/prosumers UI were already described in section 4.3, but some of them were still under development and only mock-ups were included.

In this section, we update the progress in the UI. As both personalized energy analytics module and human centric automation module are implemented in the same interface, the reader should know that only the first module updates are specified here, and the rest are included in Section A.1.3.

Apart from minor esthetic changes and minor error corrections, the most remarkable changes are:

- Removing the “Savings” section as, finally, no data will be shown there (see Figure 48).

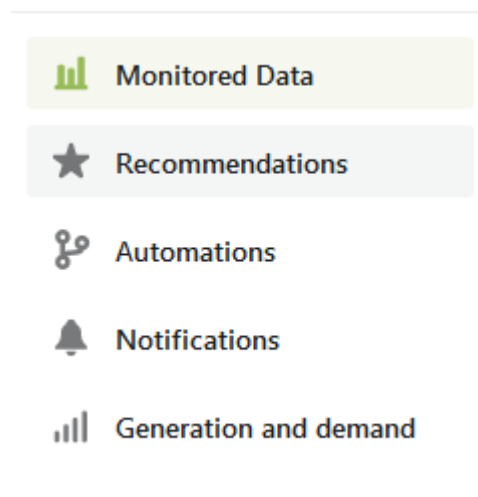


Figure 48: Lateral menu in prosumers/consumers dashboard.

- Replacing mock data from users’ consumption and energy prices with real data (see previous version with no real data in Figure 29).
- Showing the actual non-automatable appliances included by the user in the recommendations section with a link to the configuration page (see Figure 49).

Schedule for non-automatable appliances today

Name	Schedule
washing machine	
aire acondicionado	
wash machine	

Showing 1 to 3 of 3

< 1 >

Configuration →

Figure 49: Screenshot of the non-automatable schedule recommendations in prosumers/consumers dashboard.

- Removing the “General behavior Changes” from the Recommendations section (see Figure 30).
- Configuration of non-automatable appliances is now fully operational updating data in real time in the database (Figure 31).
- Notifications are fully operational showing real data and suggesting the appropriate corrective actions when necessary.
- Services activation and deactivation is now available for the user (corresponding to mock-up in Figure 45). A screenshot is available in Figure 50.

Services

Air quality alarm	<input checked="" type="checkbox"/> on +
Solar energy surpluses detection	<input checked="" type="checkbox"/> on +
AC/heating reminder at night	<input checked="" type="checkbox"/> on +
Comfort temperature check	<input checked="" type="checkbox"/> on +
Air Conditioner user schedule	<input checked="" type="checkbox"/> on +
Boilers automatic control	<input checked="" type="checkbox"/> on +
AC/heating device comfort temperature control	<input checked="" type="checkbox"/> on +
Humidity alarm	<input checked="" type="checkbox"/> on +

Figure 50: Screenshot of the service configuration in prosumers/consumers dashboard.

A.1.3 Human Centric Automation Module

A.1.3.1 Data integration actions

The human centric automation module includes both, energy efficiency automatic actions (the specific services and its data needs can be seen in Section 3.2.4) and the integration with the optimal VPP configuration module.

Each automation service, plus the optimal VPP configuration module, determines some automation signals that need to be sent to the corresponding relays. This connection is established through an MQTT broker, but prior to sending the signal, a unification process is

required, as the different services and the flexibility events will generate different signals that could affect the same relay. The unification was made defining priorities within the services, being the flexibility events a priority.

A summary of the data needed in this module is listed below:

- Information regarding MQTT connection with relays: this information is retrieved once from the BDP and stored in a local database for easy and fast access.
- Current HVAC energy consumption: to be retrieved from the BDP and to be used for those automation services that require the status (ON/OFF) of the HVAC.
- Users' HVAC schedules: retrieved directly from the prosumers/consumers UI and stored in a database locally. To be used for HVAC automation signals.
- Rooms occupancy status information (only for Greek user): retrieved directly from the prosumers/consumers UI and stored in a database locally.
- Current temperature: retrieved from the BDP to be used in those automation services that required them.
- Information for flexibility events: The information is gathered through an API made available for the aggregator and stored in a local database.
- Comfort temperature: to be retrieved from the consumers/prosumers UI and from the BDP to be used for the HVAC automatic control.

All data required in this module needs to be gathered from each of the users (i.e. dwellings from Croatia and Spain and common facilities from Spanish demo building and Greek bungalow manager).

All data retrieved from the consumers/prosumers UI is introduced by the user through a user-friendly interface and stored in an internal database to be used by the algorithms associated to each functionality and service.

A.1.3.2 Module updates

In this section, we update the progress on the UI, already described in section 4.3, even though some minor changes are to be expected. As both personalized energy analytics module and human centric automation module are implemented in the same interface, the reader needs

to notice that only the second module updates are specified here, and the rest are included in Section A.1.2.

The most remarkable changes are:

- Creation of two APIs that enable communication with the optimal VPP configuration module to retrieve the control signals deriving from the flexibility events. The first one is to obtain an access token from Keycloak using the aggregator's credentials. This token will give access to the aggregators to the second endpoint (see Figure 51).

```
{
  "access_token": "TOKEN",
  "expires_in": 3600,
  "refresh_expires_in": 1800,
  "refresh_token": "not needed",
  "token_type": "Bearer",
  "not-before-policy": 0,
  "session_state": "not needed",
  "scope": "not needed"
}
```

Figure 51: Example of the API response for the token obtention from the keycloak for aggregators.

Once the token is obtained, a query using the second API is made. In this second query it is necessary to include the previous access token and, in the body, the information regarding the flexibility event and the automation signals (see Figure 52).

```
\ "eventinitialdate\":\ "01-06-2023 14:00\",
\ "eventenddate\":\ "01-06-2023 14:15\",
\ "event\":[
  {
    \ "device\":\ "RELAY01.output6\",
    \ "state\":[
      1,
      0
    ]
  },
  {
    \ "device\":\ "RELAY01.output7\",
    \ "state\":[
      0,
      0
    ]
  }
]
```

Figure 52: Example of the body to be included in the API query for flexibility events signals communication from the VPP Optimal Configuration Module to the Human Centric Automation Module.

- Creation of a module that unifies all automation signals deriving from the different automation services and the flexibility events. This module always prioritizes the flexibility events' signals due to users' contractual agreements with the aggregators.
- Successfully controlling the relays remotely according to the signal's unification submodule.
- Creation of an interface where users can introduce their own schedules (Figure 53). The schedules will be transformed on automation signals that (after the corresponding unification on the unification submodule) will turn ON/OFF the specified devices at the specified hours of the day.

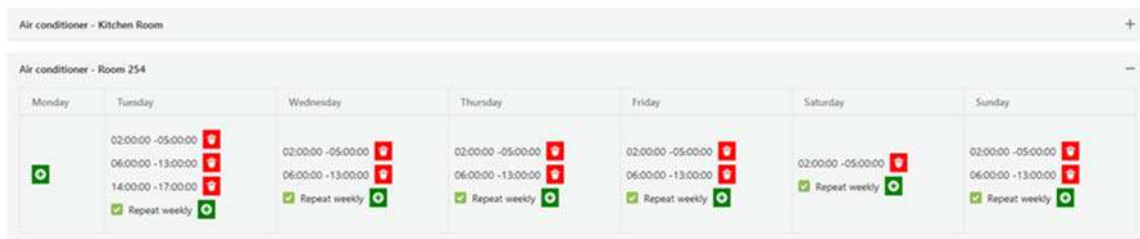


Figure 53: Schedules interface for users own automation schedules in prosumers/consumers dashboard.

- Creation of an interface where Greek user can introduce the rented periods for any of the individual rooms of the bungalow. In Figure 54, a screenshot was included.

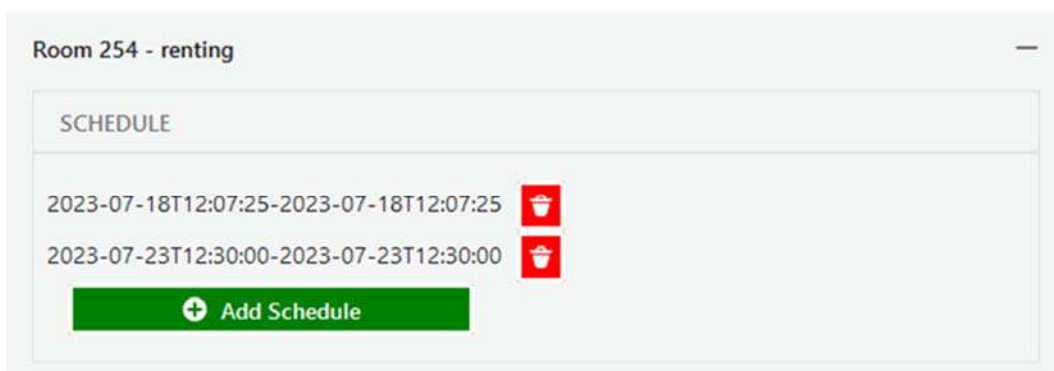


Figure 54: Screenshot form the renting interface in prosumers/consumers dashboard.

- Creation of an interface (see Figure 55) where users can monitor the current status of the relays, the next scheduled automation action with the new status and a button where the user will be able to force a change of status, if it is necessary. This button will guarantee that the user can always have control over the devices (HVACs and boilers). This is of special importance as we have control over the supply of energy to the different devices, but not over the devices per se, while the user has only access to the devices, but not the energy supply; and if we were to disconnect the power to any of them, this button would be the only way that the user can get back the control and turn in on.

Automations schedule

Appliances	Current status	Schedule next automation	Next status	Force ON/OFF
Room 254 - renting	OFF	2023-06-23T23:59:59	ON	<input type="checkbox"/> off
Air conditioner - Kitchen Room	OFF	2023-06-23T23:59:59	ON	<input type="checkbox"/> off

Showing 1 to 2 of 2

< 1 >

Figure 55: Automation schedule screen in prosumers/consumers dashboard.

A.2 Global Demand Manager Modules

A.2.1 Energy Management Analytics Module

A.2.1.1 Data integration actions

The energy management analytics module uses the following data from the BDP:

- Power consumption per apartment and per device: active power from energy meters for a) total consumption per apartment and b) consumption per device and per apartment.
- PV generation power per building: active power from energy meters for the PV generation per building.

The integration between the above-mentioned data and the module is being done through the BDP. Specifically, all the available data from the demo sites are being uploaded to the BDP and the module communicates with the BDP through API calls. The procedure of data retrieval from the BDP is explained in detail in D4.6. In our case, we collect all the historical datasets coming from a specific demo site into one data retrieval job and the real-time datasets coming from a specific demo site into one other data retrieval job. The historical data retrieval job is being called once based on the timestamp in order to retrieve from the BDP and then store all the historical data for all the devices and all the apartments in our local database. The real-time data retrieval job is being scheduled to run automatically once per day based on the

timestamp in order to retrieve from the BDP and then store all the real-time data of the previous day for all the devices and all the apartments in our local database. We store everything in our local database for speed and efficiency reasons, since there might be delays or even failures when there are a lot of API calls to the BDP. Before storing the data into our local database, we pre-process them in order to convert them from power to energy.

To sum up, we communicate with the BDP based on the URL provided to perform GET requests and the BDP responds with the requested data for the specific timestamp for all the apartments. An example that shows the json format of the results that we are getting for a period of 24 hours and for all the apartments is the following:

```
[
  {
    "EnergyMeasurements": {
      "power": [
        0.101
      ],
      "createdDateTime": "2023-05-24T00:01:10.000Z",
      "relatedDevice": {
        "nodeID": [
          "113465"
        ]
      }
    }
  },
  {
    "EnergyMeasurements": {
      "power": [
        0.053
      ],
      "createdDateTime": "2023-05-24T00:24:11.000Z",
      "relatedDevice": {
        "nodeID": [
          "113465"
        ]
      }
    }
  },
  {
    "EnergyMeasurements": {
      "power": [
        0.128
      ],
      "createdDateTime": "2023-05-24T00:37:12.000Z",
      "relatedDevice": {
```

```
    "nodeID": [
      "113465"
    ]
  }
},
{
  "EnergyMeasurements": {
    "power": [
      0.103
    ],
    "createdDateTime": "2023-05-24T00:51:10.000Z",
    "relatedDevice": {
      "nodeID": [
        "113465"
      ]
    }
  },
  {
    "EnergyMeasurements": {
      "power": [
        0.057
      ],
      "createdDateTime": "2023-05-24T01:01:12.000Z",
      "relatedDevice": {
        "nodeID": [
          "113465"
        ]
      }
    }
  }
}
]
```

In the above example, we retrieve the total consumption, and we notice that the measurements are taken every 15 minutes from the energy meters. The different apartments have different “nodeID”.

A.2.1.2 Module updates

The updates performed to the module have to do with the integration of the available data from the BDP. The ESCOs dashboard has been updated accordingly to provide insights to the ESCO for the energy consumption and generation of the building.

The first addition to the module is the at a glance page, where the ESCO can have an overview of the energy consumption of the whole building and the individual devices of the apartments and the energy generation of the whole building, as can be seen in Figure 56. Specifically, we can choose “Category”, which provides the different smart meters that can be monitored in a specific building and the “Apartment”, which allows to choose the specific apartment or the whole building. There is also a date picker for choosing the dates that the monitoring will take place. We provide, also, a summation of the energy consumed and generated for the selected period.

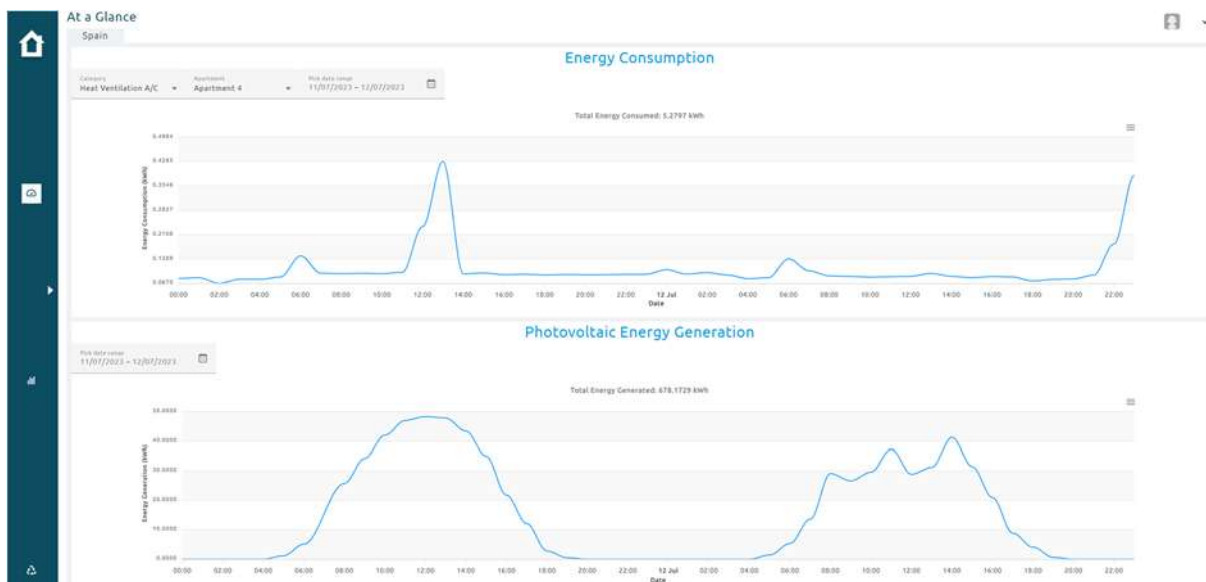


Figure 56: The "At a Glance" page of the ESCOs dashboard.

The second update was on the energy analytics page, where we have two choices, namely the “Period” and the “Choose a date” option and two different parts, namely the “Consumption Clusters” and the “Consumption Timeline” as can be seen in Figure 57. The “Period” defines the amount of time for the aggregation of the consumption and the “Choose a date” allows the ESCO to pick a date as starting point of the aggregation. In the “Consumption Clusters” part, the ESCO can have an overview of which apartment belongs to which cluster (low, mid and high energy consuming clusters). In the “Consumption Timeline” part, the ESCO can have an overview of how the consumption evolves on a daily, weekly, monthly or yearly basis.



Figure 57: The "Energy Analytics" page of the ESCOS dashboard.

A.2.2 Self-Consumption Optimization Module

A.2.2.1 Data integration actions

The self-consumption optimization module uses the following data from the BDP:

- Day-ahead PV power generation forecast at building level: 24-hour ahead PV generation forecast.
- Day-ahead total power consumption forecast at building level: 24-hour ahead total consumption forecast.
- Day-ahead power consumption forecast per device at building and apartment level: 24-hour ahead consumption forecast per device and per building/apartment.
- Retail energy prices at demo level: hourly retail energy prices for the specific demo sites.

The integration between the above-mentioned data and the module is being done through the BDP. Specifically, all the available data from the demo sites are being uploaded to the BDP and the module communicates with the BDP through API calls. The procedure of data retrieval from the BDP is explained in detail in D4.6. In our case, we have two kinds of datasets with


```

25.101232528686523,
17.98517417907715,
10.505160331726074
]
}

```

In the above example, we retrieve the day-ahead (24 values) PV power generation forecast from the current timestamp onwards.

A.2.2.2 Module updates

The updates performed to the module have to do with the integration of the available data from the BDP. The ESCOs dashboard has been updated accordingly to provide insights to the ESCO for the self-consumption of the building.

The first change is the addition of a date picker to give the ESCO the capability to check the historical results. We also compute and show the cost savings per day in addition to the self-consumption ratio per day. Figure 58 shows the above-mentioned updates.

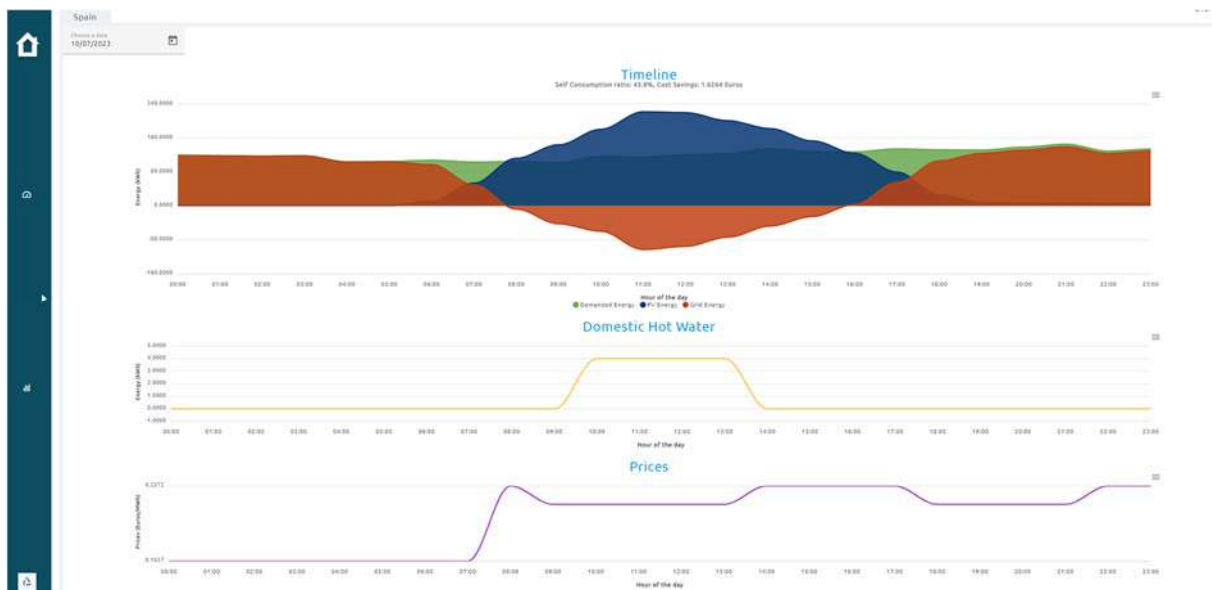


Figure 58: The "Self-Consumption" page of the ESCOs dashboard.

A.2.3 Smart Contract Monitoring/Handling Module

A.2.3.1 Data integration actions

In this section, both the datasets that are required from, as well as the datasets that are ingested to the BDP are presented in detail.

As thoroughly described in Section 3.3.4.2, there are 3 datasets that the smart contract monitoring/handling module retrieves from the BDP, namely:

- Actual HVAC and DHW demand
- Short term (24-hour) HVAC and DHW demand forecast
- Flexibility events triggered by the VPP optimal configuration module

Starting with the flexibility events uploaded in the BDP from the VPP optimal configuration module as described in Section 3.3.6 utilizing the API calls that the BDP offers, the Smart Contracts app retrieves the related event information, that include the assets participating in the event, the activation time and its duration. The form of the data retrieved is as follows:

```
{
  "event_id": 134,
  "asset_id": "113474_hvac",
  "description": "downwards",
  "flexcapacity": 0.0886802825416375,
  "duration": 15,
  "activationtime": "19-06-2023 11:03:28"
}
```

Having that as input, additional requests are constructed and pushed to the APIs of the BDP, to retrieve for each specific asset that participate in the event, both the actual demand that is being captured from the pilot sites and is in the following form:

```
{
  "EnergyMeasurements": {
    "apparentPower": [
      0
    ],
    "rmsCurrent": [
      0
    ],
    "rmsVoltage": [
      0.23188
    ],
    "power": [
      0
    ],
    "createdDateTime": "2023-01-20T20:45:37.000Z",
    "relatedDevice": {
      "nodeID": [
        "113473"
      ],
      "code": "2"
    }
  }
}
```

as well as the 24-hour demand forecasts of the same assets that are in that form:

```
{
  "device_nodeID": "113473",
  "device_code": "2",
  "datetime": "2023-01-20T20:45:37.000Z ",
  "hour": 20,
  "predictions": [
    0.10870587825775146,
    0.14304058253765106,
    0.14568787813186646,
    0.09026797115802765,
    0.04962100461125374,
    0.06790496408939362,
    0.051486819982528687,
    0.041812632232904434,
    0.06709554046392441,
    0.13238897919654846,
    0.09784982353448868,
    0.09255490452051163,
    0.11500467360019684,
    0.06476779282093048,
    0.08803898096084595,
    0.12951987981796265,
    0.03459661453962326,
    0.062109172344207764,
    0.09680020809173584,
    0.08256804943084717,
    0.08399830758571625,
    0.12699586153030396,
    0.10632068663835526,
    0.10252189636230469
  ]
}
```

The information is being requested for the specific periods that the event is taking place, based also on the information initially fetched from the BDP.

Combining that data with the contract details available, the smart contract module can verify for each event the actual and the forecasted demand and calculate based on the PMV the proper remuneration that is due. An overview of the executed calculations is offered in the UI of the smart contract app, as presented in the next section.

The contracts information that is being entered in the smart contracts module, needs also to be used from the rest of the modules in the project. For that reason, an ingestion mechanism using the MQTT method is being developed, to push this information in the BDP, so that it becomes commonly available and retrievable via the API mechanisms of the platform. The form of the data is presented below:

```
{
  "ContractualAgreement": {
    "name": "C2",
    "description": "Description",
  }
}
```

```
"id": "CN2",
"relatedDevice": [
  {
    "nodeID": [
      "122409_1"
    ]
  }
],
"scheduledPeriod": [
  {
    "weekdayIndicator": "Monday",
    "scheduleStart": [
      "21:00"
    ],
    "scheduleEnd": [
      "22:00"
    ]
  },
  {
    "weekdayIndicator": "Tuesday",
    "scheduleStart": [
      "01:00"
    ],
    "scheduleEnd": [
      "23:00"
    ]
  },
  {
    "weekdayIndicator": "Wednesday",
    "scheduleStart": [
      "02:00"
    ],
    "scheduleEnd": [
      "01:00"
    ]
  },
  {
    "weekdayIndicator": "Thursday",
    "scheduleStart": [
      "00:00"
    ],
    "scheduleEnd": [
      "00:00"
    ]
  }
],
"penalties": 3,
"price": 2,
"payment": 1,
"status": "active",
"contractPeriod": [
  {
    "startDateTime": "2023-05-08T11:34:00.000Z",
    "endDateTime": "2023-05-13T11:34:00.000Z",
    "referenceDateTime": "2023-05-08T00:00:00.000Z"
  }
],
"relatedAggregator": [
  {
    "id": "ed2a6f91-4541-46c0-9792-004be7b9210f"
  }
],
"relatedProsumer": [
  {
    "id": "a358954d-7f54-4d2d-aed4-c042ae12a070"
```

```

}
  ]
}
}

```

A.2.3.2 Module updates

Module updates are related to the performance overview of the contracts, that can capture multiple events and offer a better overview per contract as presented in Figure 59 below.

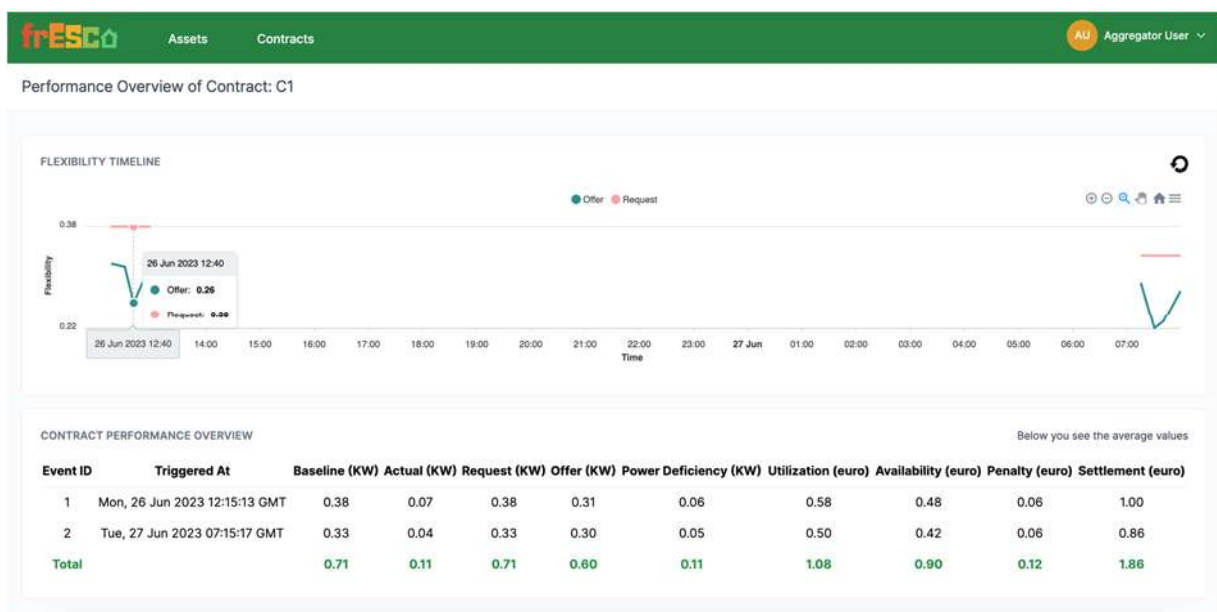


Figure 59: Performance Overview of Contract.

In the “CONTRACT PERFORMANCE OVERVIEW” part, we have the following parameters: baseline (kW), actual (kW), request (kW), offer (kW), power deficiency (kW), utilization (euro), availability (euro), penalty (euro), and settlement (euro). The baseline is the forecasted power demand for the devices during the event, the actual is the power coming from the actual metering that takes place every 15 minutes, the request is the power requested by each device coming from the VPP optimal configuration module, the offer is the difference between baseline and actual power and the deficiency is the difference between the requested power and the flexibility delivered. For the remuneration concepts, the utilization depicts the remuneration for the flexibility provided and it is based on the offer per event, the availability is the remuneration for the flexibility made available and the penalty is the fine paid for not offering the flexibility promised and it is based on the deficiency power. Regarding the

availability, the prosumers are remunerated also based on the availability they provide, even if not used at the end. This acts as a motivation for the prosumers to make their devices available for flexibility events.

Selecting a specific event, more detailed information is presented to the module user, allowing an easier understanding of the measured values for each point in time of the event, as presented in Figure 60.

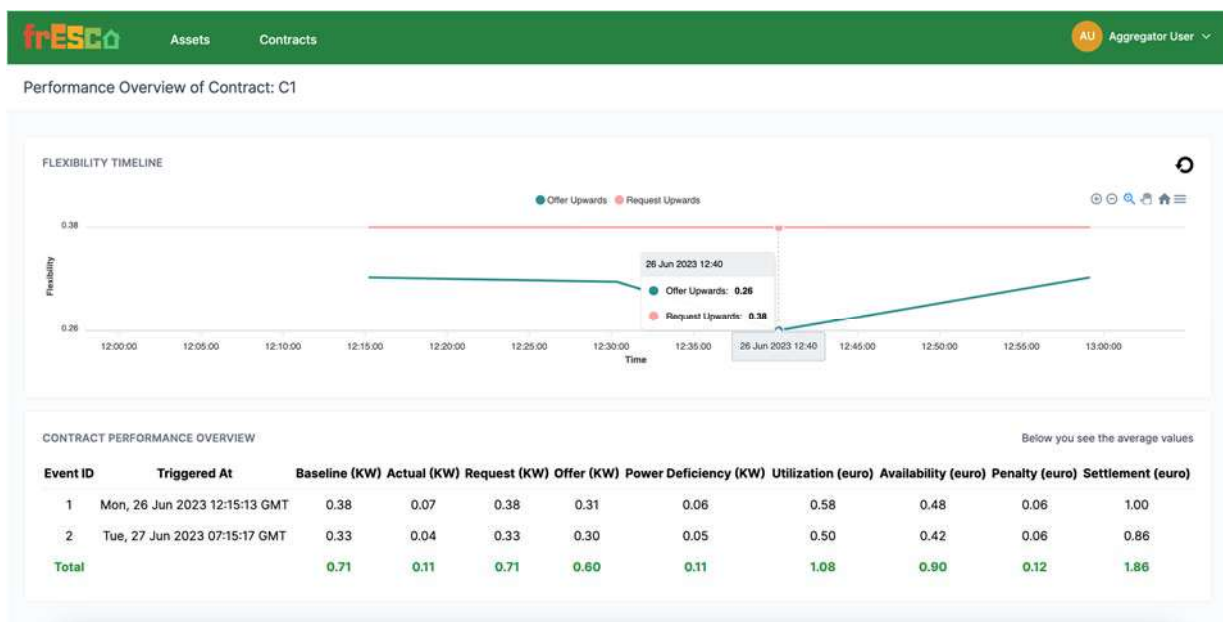


Figure 60: Performance Overview of Contract - Detailed.

A.2.4 Flexibility Analytics Module

A.2.4.1 Data integration actions

The data that we use for the flexibility analytics module come from the results of the VPP optimal configuration module and there are no external data coming from the BDP. Specifically, the flexibility analytics module uses the following data:

- Flexibility requested per device and per apartment: amount of flexibility requested per flexible device of each apartment in order to fulfill the flexibility event.
- Available flexibility forecasts per device and per apartment: 3-hour ahead available flexibility forecast per device and per apartment.

The above data are stored in our local database after the VPP optimal configuration problem is solved and this means that we retrieve them internally.

A.2.4.2 Module updates

The updates performed to the module have to do with the integration of the available data from the flexible devices performed in the VPP optimal configuration module. The aggregator dashboard has been updated accordingly to provide advanced analytics to the aggregator based on the results of the VPP optimal configuration module.

First, we made major changes in the way that we show the analytics on the flexibility analytics page. Specifically, we divided the page into “Flexibility Clusters” and “Flexibility Aggregation”. In the “Flexibility Clusters” part we included three choices with titles “Aggregation Period”, “Clustering Attribute” and “Choose a date”, as can be seen in Figure 61. The “Aggregation Period” has “Daily”, “Monthly” and “Yearly” as choices and it is used to aggregate the data for the selected period to perform the clustering. The next choice is “Clustering Attribute” and it has “Available Flexibility”, “Provided Flexibility” and “Percentage Contribution” as choices and it is used to select the attribute to perform the clustering. The last choice is the “Choose a date” and it is a date picker to select the initial date from which the aggregation will take place based on the “Aggregation Period” choice. The clusters are always three, namely high, mid and low.



Figure 61: The flexibility clusters of the "Flexibility Analytics" page of the aggregators dashboard.

In the "Flexibility Aggregation" part we included two choices with titles "Aggregation Period" and "Choose a date", as can be seen in Figure 62 and they have the same choices as the "Flexibility Clusters" part. Based on the "Aggregation Period" and the "Choose a date" choices, we perform aggregation of the available flexibility and the provided flexibility per device.



Figure 62: The flexibility aggregation of the "Flexibility Analytics" page of the aggregators dashboard.

A.2.5 VPP Optimal Configuration Module

A.2.5.1 Data integration actions

The VPP optimal configuration module uses the following data from the BDP:

- Available flexibility forecasts per device and per apartment: 3-hour ahead available flexibility forecast per device and per apartment.
- Static smart contracts information: parameters included in the smart contract signed between the aggregator and the prosumer, such as nominal power of the flexible devices, flexibility capacity, available hours for flexibility capacity, etc.

The integration between the above-mentioned data and the module is being done through the BDP. Specifically, all the available data from the demo sites are being uploaded to the BDP and the module communicates with the BDP through API calls. The procedure of data retrieval from the BDP is explained in detail in D4.6. In our case, we have two kinds of datasets with different handling. The first part of datasets is the available flexibility forecasts. In this case, we find the dataset that we need to retrieve, we choose the attributes that are of interest for us, we choose the query parameters (the timestamp), and we define a data retrieval job. Each time a flexibility event is triggered, and we need to solve the optimization problem of the VPP configuration, we communicate with the BDP based on the URL provided to perform GET request and the BDP responds with the requested data for the specific timestamp for all the apartments. The second part is the static information from the smart contracts. With the same way, we find the dataset that we need to retrieve, we choose the attributes that are of interest for us, we retrieve the whole dataset from the BDP and then we store all the retail prices for the year for all the demo sites in our local database. We store everything in our local database for speed and efficiency reasons, since there might be delays or even failures when there are a lot of API calls to the BDP. An example that shows the json format of the results that we are getting for a specific apartment is the following:

```
{  
  "datetime": "2023-05-17T15:00:00.000Z",  
  "device_nodeID": "113473",  
  "device_code": "2",
```

```

    "flexibility": [
      0.1380341455247667,
      0.1444745659828186,
      0.16842341423034668
    ]
  }
}

```

In the above example we get the available flexibility forecast for the next 3 hours for the specific timestamp and the specific apartment.

The results of the optimal VPP configuration module have to be uploaded to the BDP in order to be used by the smart contracts monitoring/handling module. The procedure of data collection is also explained in detail in D4.6. In our case, we first create an API that responds to a GET request from the BDP once per day and uploads to the BDP the results of the flexibility events that triggered the last 24 hours and then we match the fields of the json file with the CIM of the BDP. The form of the json file is the following:

```

[
  {
    "event_id": 134,
    "asset_id": "122409_dhw",
    "description": "downwards",
    "flexcapacity": 4.69616455165519,
    "duration": 15,
    "activationtime": "19-06-2023 11:03:28"
  },
  {
    "event_id": 134,
    "asset_id": "113474_hvac",
    "description": "downwards",
    "flexcapacity": 0.0886802825416375,
    "duration": 15,
    "activationtime": "19-06-2023 11:03:28"
  },
  {
    "event_id": 134,
    "asset_id": "122410_hvac",
    "description": "downwards",
    "flexcapacity": 0.0887375338537718,
    "duration": 15,
    "activationtime": "19-06-2023 11:18:28"
  },
  {
    "event_id": 133,
    "asset_id": "122409_dhw",
    "description": "downwards",
    "flexcapacity": 6.1364261008294,
    "duration": 30,
    "activationtime": "19-06-2023 10:47:13"
  },
  {
    "event_id": 133,
    "asset_id": "113467_hvac",

```

```

    "description": "downwards",
    "flexcapacity": 0.366800019702117,
    "duration": 30,
    "activationtime": "19-06-2023 10:47:13"
  },
  {
    "event_id": 133,
    "asset_id": "113468_hvac",
    "description": "downwards",
    "flexcapacity": 0.366800019702117,
    "duration": 30,
    "activationtime": "19-06-2023 10:47:13"
  },
  {
    "event_id": 133,
    "asset_id": "113466_hvac",
    "description": "downwards",
    "flexcapacity": 0.366800019702117,
    "duration": 30,
    "activationtime": "19-06-2023 10:47:13"
  }
]

```

In the above example, we notice that two events were triggered during the last 24 hours. For each event, we can observe the amount of flexibility that each device is requested to provide from the activation time and for amount of time equal to the duration.

To sum up, the overall procedure of the provision of flexibility and the communication between the different modules and the BDP can be shown in Figure 63 with a sequence diagram.

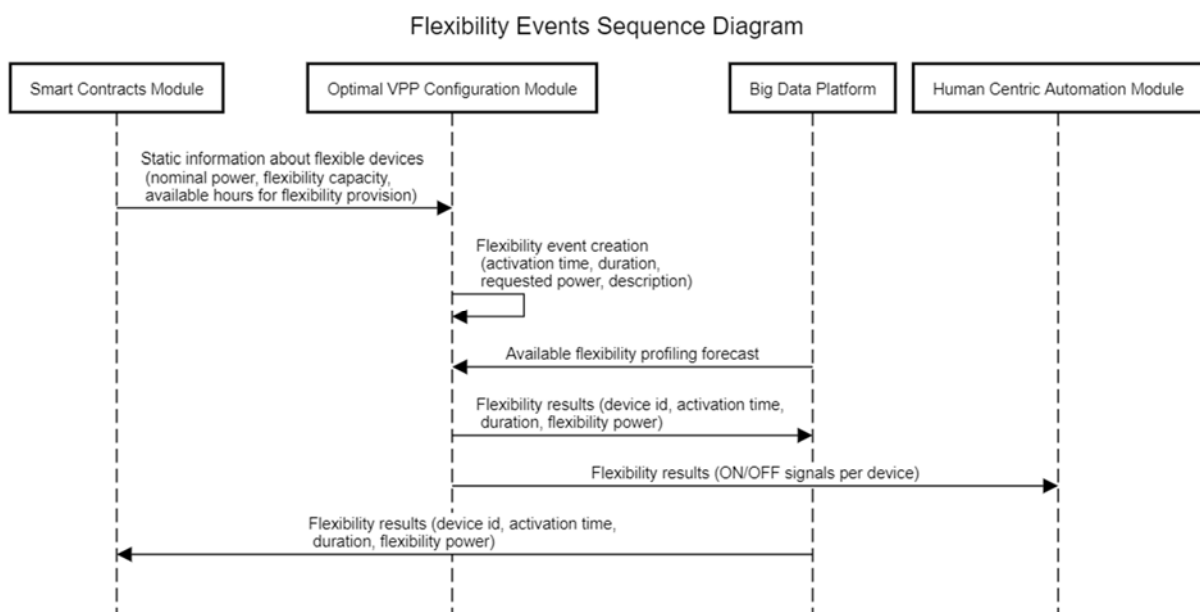
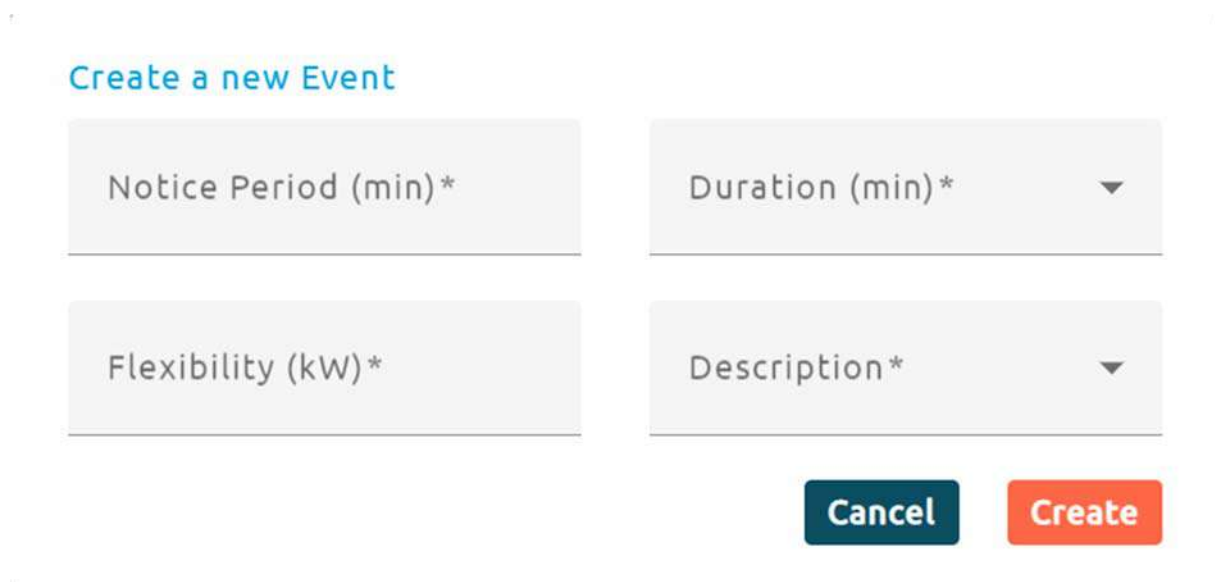


Figure 63: Flexibility events sequence diagram.

A.2.5.2 Module updates

The updates performed to the module have to do with the integration of the available data from the flexible devices and the smart contracts. The aggregators dashboard has been updated accordingly to allow the aggregator to trigger flexibility events and to provide insights to the aggregator for the flexibility events.

First, in the flexibility event triggering window a new choice “Description” was added with the available values of “Downwards” and “Upwards” and the choice “Duration” can only take values of 15, 30, 45 and 60 minutes, as can be seen in Figure 64.



The screenshot shows a form titled "Create a new Event" with the following fields:

- Notice Period (min)*
- Duration (min)*
- Flexibility (kW)*
- Description*

At the bottom right of the form are two buttons: "Cancel" (dark blue) and "Create" (red).

Figure 64: The event creation part of the "Flexibility" page of the aggregators dashboard.

When the flexibility event is triggered, the problem is being solved and we can observe the details of the flexibility event in a table and if we choose a specific event, we can obtain an overview of the amount of flexibility that each device is requested to provide, as can be seen in Figure 65. The aggregator can also use a date picker to choose to inspect flexibility events of the past.

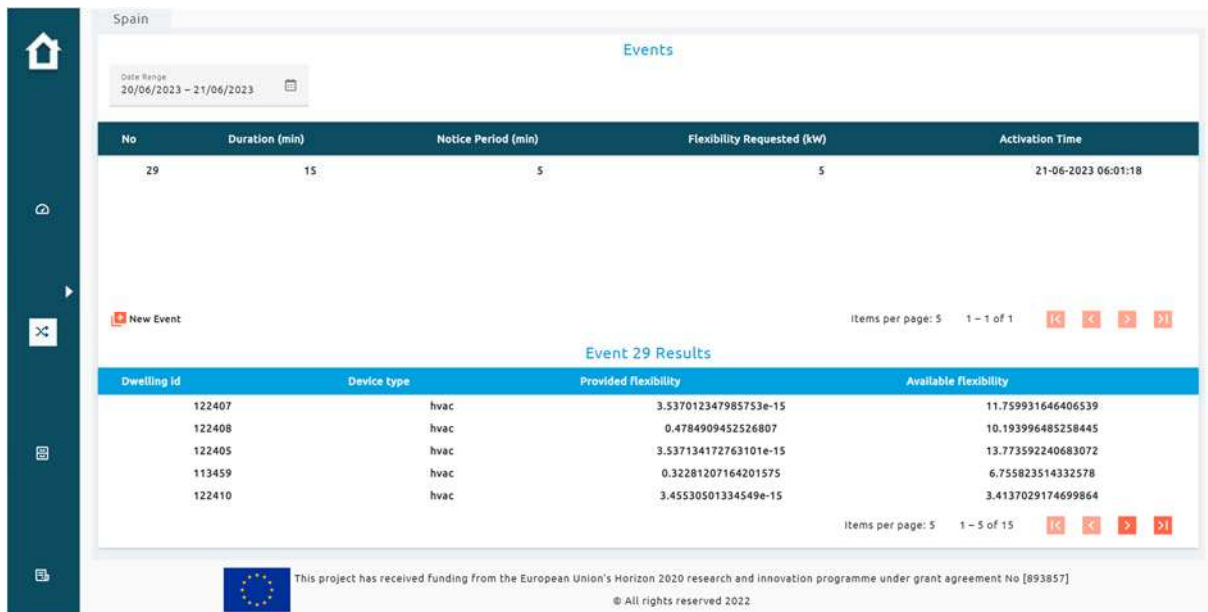


Figure 65: The details of the events of the "Flexibility" page of the aggregators dashboard.

In the at a glance page, a date picker was also added to give the aggregator the ability to check how much of the initially requested flexibility was provided through the devices, as can be seen in Figure 66.



Figure 66: The "At a Glance" page of the aggregators dashboard.

ANNEX B: ENERGY SOLVERS COMPARISON AND ANALYSIS

Task T5.2 is focused on the energy management analytics, implementing the optimization mechanisms tool for ESCOs. The first question addressed by developers was whether to analyze the use of already available commercial solvers or use an alternative platform to develop an optimization problem for the frESCO optimization issues of the ESCO portfolio energy management. These solvers may be used to program and decide the best management strategy for the renewable and variable energy resources that are installed in the residential buildings. In this line, this annex provides a benchmark analysis of commercial optimization solvers. The results have been used to decide the implementation of an own solver or the use of a commercial one for the frESCO ESCO module.

B.1 Introduction

In energy applications, commercial solvers use advanced mathematical techniques to optimize the allocation and utilization of resources, ensuring the maximum output from variable renewable energy resources and the minimum operation cost while considering constraints and operational requirements. Although they are not specifically designed to solve energy problems, they are one of the best decision-making tools in the renewable energy sector. Depending on the complexity and type of the problem, it can be solved in few milliseconds or in hours.

In the next section a benchmark is done, comparing different characteristics and possible implementations in these energy management tools. The following sections include the description of the methodology used, the solvers selected and the KPIs chosen. Finally, results and final conclusions are shown.

B.2 Methodology

B.2.1 Optimization solvers

In the benchmark carried out for this project, five commercial solvers have been chosen. In this section, the most significant characteristics of the selected libraries and solvers are summarized.

- **CPLEX** is a high-performance mathematical optimization solver developed by IBM. It is particularly well-suited for finding solutions to Linear Programs (LP), Mixed Integer Programs (MIP), Quadratic Programs (QCP) and Constrained Quadratic Programs (MIQCP). **Ref:** [CPLEX \(gams.com\)](https://www.gams.com)
- **GUROBI** is a state-of-the-art mathematical optimization solver developed by GUROBI Optimization company. This algorithm can add high complexity to the model and solve it in the shortest possible time. It works with Linear programs, Mixed-Integer Programs (MIP), Quadratic Programs (QP), and Nonlinear Programs (NLP). **Ref:** [Gurobi Optimizer Delivers - Gurobi Optimization](#)
- **PuLP** is an open-source modelling library in Python, which can be used in many different projects and platforms. It provides a user-friendly interface for constructing and its default solver (CBC) is able to solve optimization problems using Linear programming (LP) and mixed-integer linear programming (MILP) techniques. **Ref:** [PuLP · PyPI](#)
- **Google OR-Tools** is an open-source operations research library developed by Google. It offers a wide range of optimization algorithms and tools for solving combinatorial optimization problems, its principal solver, GLOP, includes linear programming (LP) and mixed-integer programming (MIP). **Ref:** [OR-Tools | Google for Developers](#)
- **SCIP** (Solving Constraint Integer Programs) is a powerful optimization solver developed by Zuse Institute Berlin and other research institutions. It is designed to solve a wide range of optimization problems, including MILP and mixed-integer nonlinear programming (MINLP). **Ref:** [SCIP \(scipopt.org\)](https://scipopt.org)

B.2.2 KPIs

In this section, the key Performance Indicator (KPI) that are considered to compare and benchmark the optimization programs are analyzed. These KPIs are the reference to propose the best optimization solver to use in energy management operation optimization solution.

1. **KPI1 Accuracy:** This KPI is calculated comparing the best value between the solutions found by all the solvers and each of the solutions found by each solver.
2. **KPI2 Speed:** Speed is measured in terms of execution time and number of iterations.
3. **KPI3 Ease of use:** It considers the number of lines and the number of different equivalent equations needed to program a concrete problem. These software need a specification method to introduce and need all the data inputs.
4. **KPI4 Flexibility:** This section counts the number of different kinds of problems the solvers can solve.
5. **KPI5 Availability and accessibility:** It considers the need for a paid license, the possibility to work online, and the number of platforms where it can be used.
6. **KPI6 Technical help and support:** Aspects such as the existing official web, a technical support and a consulting community is evaluating with a Boolean variable and the final KPI value is the sum of all of them.

B.2.3 Data sets

The data used in this work are not the real ones of the frESCO project because in the moment this analysis was carried out no robust data was available. The data used has been selected from the BENEfICE project. Three users have been selected, in two seasons of the year (winter and spring). **Ref:** <http://www.gecad.isep.ipp.pt/BeNeFiCE/home/>

- User 1:

A single-family user with moderate consumption who does not return home for lunch in the middle of the day is considered. Therefore, on school days, this user leaves early in the morning and returns in the evening, where the highest consumption of the day is observed. The energy profile is very similar throughout the week, except the weekend. In spring analysis, flexible loads are introduced. This user has nonflexible loads (light, fridge and continuous loads,

for example), PV production, EV, battery energy storage, and flexible loads like dishwasher and washing machine (only in spring).

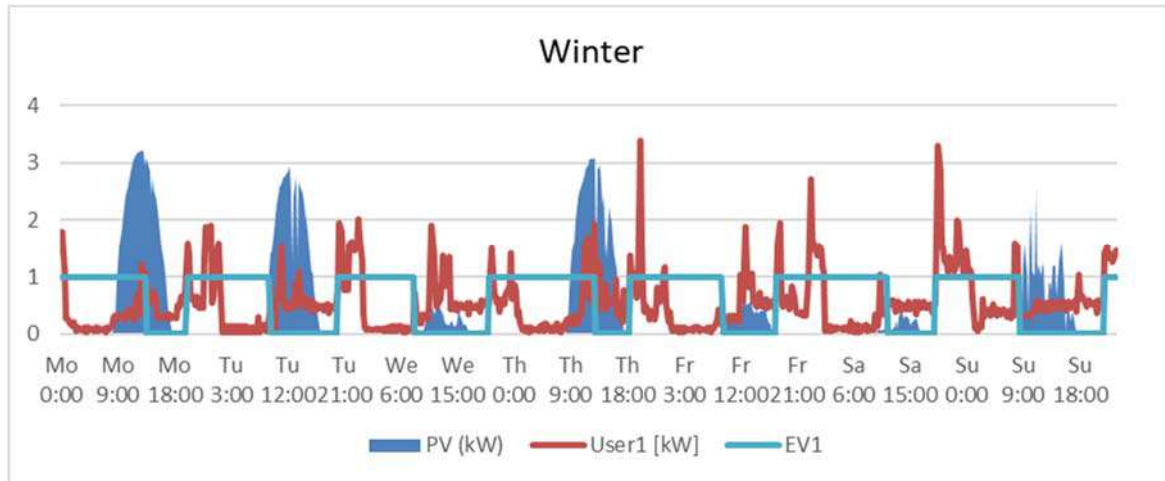


Figure 67: User 1 winter load consumption, PV generation and EV availability profiles.

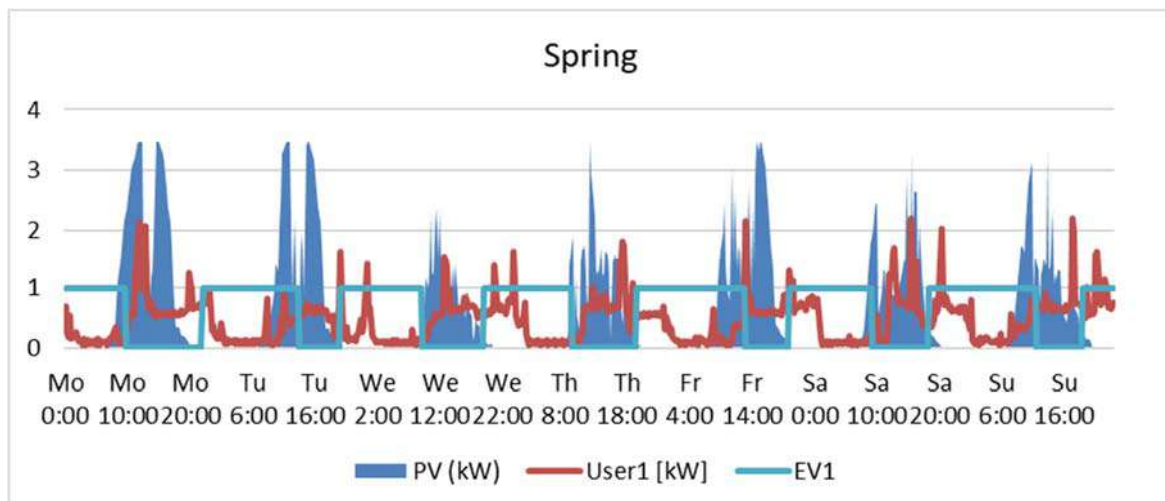


Figure 68: User 1 spring load consumption, PV generation and EV availability profiles.

- User 2

This second user has a large consumption in the evening and at midday. In this case, part of the required energy is consumed at midday, in the peak period of solar production. The profile of this user is quite irregular, consuming at different times of the day, both during the day and at night. Despite having a similar consumption on working days, on certain days there is a higher peak power. In addition, the weekend has a very irregular profile. This user has nonflexible loads, PV production, EV, and battery energy storage.

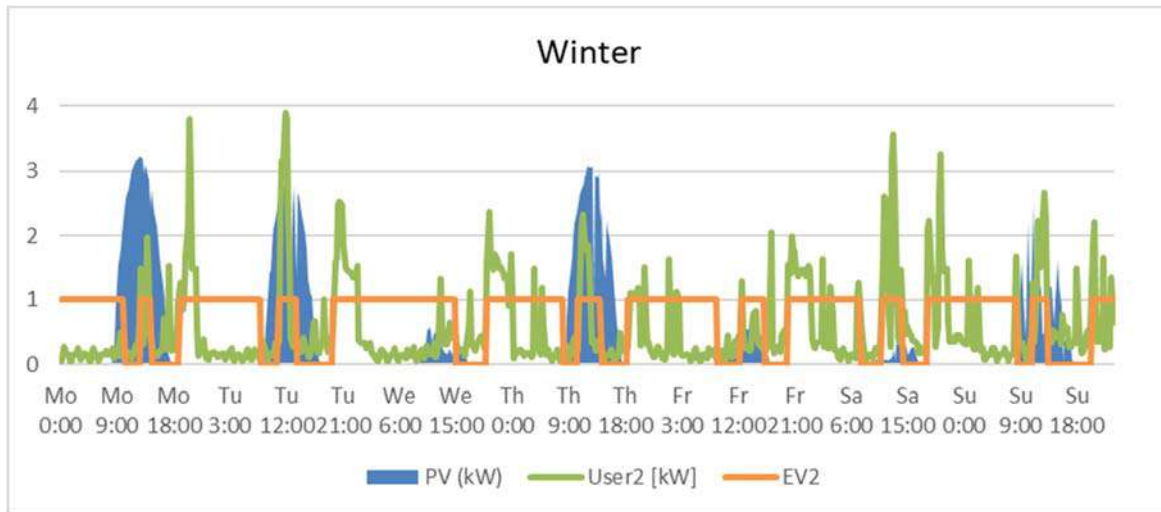


Figure 69: User 2 winter load consumption, PV generation and EV availability profiles.

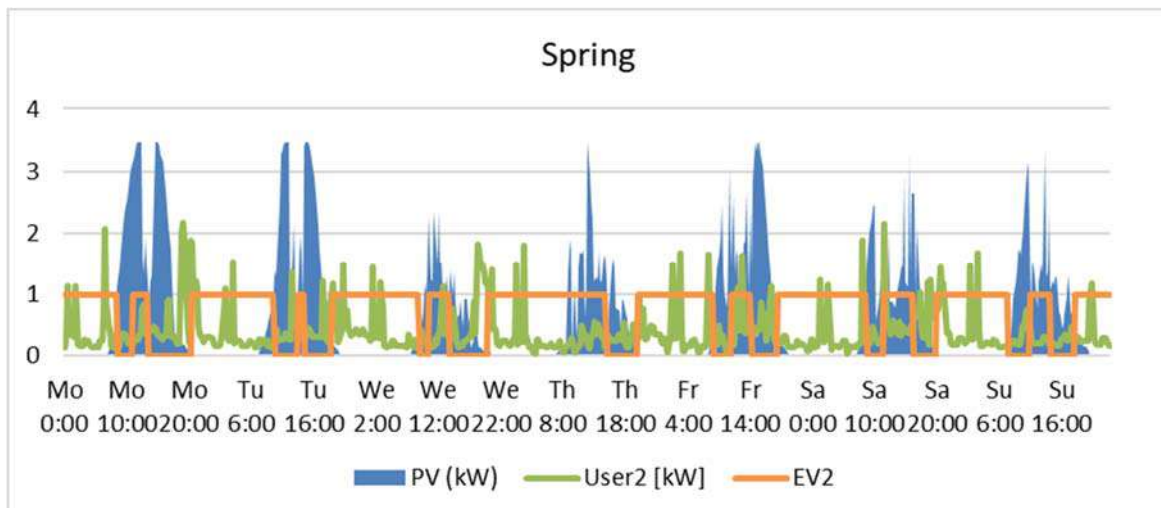


Figure 70: User 2 spring load consumption, PV generation and EV availability profiles.

B.3 Implementation

Following section defines the characteristic of the cases analyzed. Although various scenarios are tested, only the results of two cases are presented. Since KPI factors such as availability and technical support are the same in each solver regardless of the case studied, the differences between the cases are shown in results of the accuracy, speed, and code length. The characteristics of the two cases analyzed are the next ones:

- **Case 1:** Base Case. The base case has been carried out with a system with PV, batteries, EV, and grid connection. For the analysis, user 2 was used in winter analysis period, where the energy managed by the solar resource is the smallest.
- **Case 2:** Flexible loads. In this second case, in addition to the previous elements, flexible loads have been included. In this case, the data of user 1 for the spring season has been used, where the solar energy provided by the panels is higher than that compared during the winter season.

Similar problem has been done with other scenarios, changing the inputs data and the characteristics of the users. However, the objective is the same in all scenarios: minimizing the cost of energy operation. For this calculation, only the cost of energy from the grid is considered, not the maintenance cost of the panels or loads.

$$\text{Cost} = \min \left(\sum P_{grid} C_{grid} - P_{EV} C_{EV} + P_{Bat} C_{Bat} \right)$$

With this objective, the main constraints requirements from the problem are the ones related to the power flow of all the elements and branches of the electric circuit of a building. In the Figure 71, there is show the basic electric distribution of a building with renewable resources.

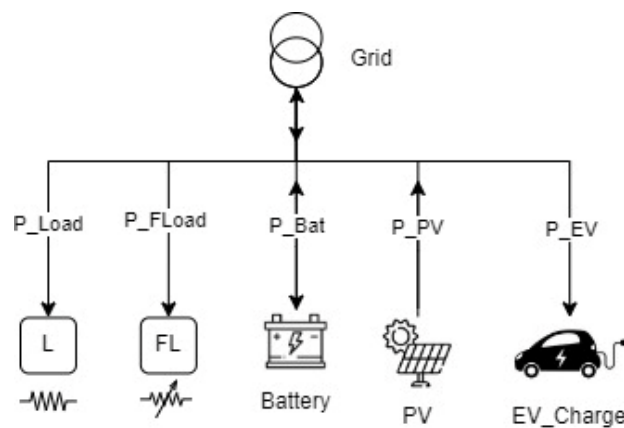


Figure 71: General graph of the electric power circuit of a user.

The problem create for the benchmark is a MIP problem. It has continuous variables such as the power consumption of each branch and the state of charge of the batteries and the EV, and binary variables that connect and disconnect the branches of the electric circuit. The main constrains of this circuit are the next ones:

- Power balance between the energy input from the grid and the energy consumed and generated through the branches.
- Photovoltaic generation power. The sub-objective of the circuit is obtaining the maximum power from renewable resources.
- Electric vehicle power consumption. The power needed to charge the vehicle can be variable, depending on the final state of charge of the vehicle and the availability time.
- State of charge of the battery. The power the system can extract or inject from the batteries depends on the state of charge of them in each time step.
- Operation limits of microgrid component equipment: maximum and minimum operating power of components, battery storage capacity, operating costs, ...

B.3.1 Results

Using the cases defined in the previous section, there have been calculated all the KPI value for the selected solvers. The results in the Table 25 corresponding to the case 1 where no flexible loads are involved and the results in Table 26 corresponds to the case 2, where flexible loads are included in the simulation variable elements.

	KPI1	KPI2	KPI3	KPI4	KPI5	KPI6
Cplex	6.18390 Best Option	409 iter. t = 276 ms	183 lines	6 LP, MIP, QCP MIQCP, RMIP, RMIQCP	0+5 GAMS, C++, Java, Python & MATLAB	3 Official Web, tec. Support & big commu.
GUROBI	6.18398 (7.8682 e-5)	1383 iter. t = 30 ms	121 lines	6 LP, MILP, QP, MIQP, QCP, MIQCP	0+5 Python, C++, Java, MATLAB & R	3 Official Web, tec. Support & commu.
PuLP	6.18398 (7.8680 e-5)	446 iter. t = 170 ms	116 lines	2 LP, MIP	1+1 Python	1 Big commu.
OR-Tool	6.18398 (7.8682 e-5)	519 iter. t = 231 ms	126 lines	2 LP, MIP	1+4 C++, Python, Java & .NET	2 Official web & tec. Support.
SCIP	6.18398 (7.8682 e-5)	1211 iter. t = 299 ms	126 lines	4 MINLP, COP, MIP, CIP, LP	1+3 Python, C++, java	1 Official web

Case 1 optimal result: 6.1839€

Table 25: Case 1 KPIs.

Comparing the results obtained with the rest of the solvers, in this first case CPLEX and GUROBI has the best KPI results. On one hand, they have a fast resolution with a big official

web, technical support, and a big community. On the other hand, both are paid license solvers, so the used is limited. The free solvers PuLP, OR-Tool and SCIP are a good option for the problem that is done. However, if the complexity of the problem increase, these solvers are limited to the type of problems that they can solve. In conclusion, and considering all the KPI results, for a linear programming problem such the one analyzed in case 1, the best solver use is PuLP.

	KPI1	KPI2	KPI3	KPI4	KPI5	KPI6
CPLEX	2.79636 (0.007283)	506 iter. t = 323 ms	265 Lines	6 LP, MIP, MIQCP, QCP, RMIP, RMIQCP	0+5 GAMS, C++, Java, Python y MATLAB	3 Official Web, tec. Support & big commu.
GUROBI	2.78967 (2.67e-9)	1832 iter. t = 230 ms	208 Lines	6 LP, MILP, QP, MIQP, QCP, MIQCP	0+5 Python, C++, Java, MATLAB y R	3 Official Web, tec. Support & commu.
PuLP	2.78967 Best Option	2334 iter. t = 3.46 s	193 Lines	2 LP, MILP	1+1 Python	1 Big commu.
OR-Tool	2.78967 (6.05 e-7)	6292 iter. t = 6.292 s	214 lines	2 LP, MIP	1+4 C++, Python, Java & .NET	2 Official web & tec. Support.
SCIP	2.78967 (2.67e-9)	4218 iter. t = 3.166 s	214 Lines	4 MINLP, COP, MIP, CIP, LP	1+3 Python, C++, java	1 Official web

Case 2 optimal result: 2.789 €

Table 26: Case 2 KPIs.

In the second case, where the number of variables of the problem and the iterations needed has increase, the resolution time is bigger. It is seen that CPLEX and GUROBI have the best KPI results with the faster resolution and the maximum problem resolution. But, as it is said in the previous case, the price of the license can be the limitation for its real application. In this case, it can be seen that the free solvers PuLP, OR-Tool and SCIP need more than ten times more time to solve the problem. So, depending on the granularity of the problem these solvers can offer the needed answer in a reasonable time, lees than the granularity estimated. In the case studied, with a granularity of 15 minutes, the resolution in some seconds or minutes is not a problem. So, in conclusion, in this case SCIP solver is the best option because of its resolution time. However, PuLP solve has been chosen because of its big community.

Similar problem has been done changing the inputs data and the characteristics of the users. The results obtained were different but the conclusions not. The KPI comparison of the solvers is nearly identical in all the scenarios tried. GUROBI and CPLEX are the fasters ones while PuLP and SCIP could be a good solution to free implementation needs. Moreover, OR-Tool may be used in simplex problems and the results are also fast and accessible. So, it could be another solver used according to the problem to solve.

B.4 Conclusions

After the benchmark done with five optimization commercial solvers, the best solvers are GUROBI and CPLEX. Although they can solve very complex problem in a very low time, they required an expensive license. On the other hand, PuLP, OR-Tool and SCIP are very useful for the resolution of linear problems. However, they are not as fast as the other two, and do not have technical support.

The problem considered in frESCO project can be solved with these solvers. However, they are not especially focused on energy management problems and it could be difficult to program because of its mathematical language and the need to include all the constrains and all the parameters one by one. Moreover, they have specific structure, and if complexity increases, so do the required lines of the code and the constraints of the problem. Hence, in conclusion, the frESCO project has decided to use an alternative platform to create and implement the optimization problem, focused on the energy resource optimization of the project.