D6.8 - Updated Interfaces for Professional and Residential Users

Deliverable ID	**D6.8**
Deliverable Title	**Updated Interfaces for Professional and Residential Users**
Work Package	**WP6**
Dissemination Level	**PUBLIC**
Version	**1.0**
Date	**23/08/2018**
Status	**final**
Type	**Prototype**
Lead Editor	**Fraunhofer FIT**
Main Contributors	**FIT (Veronika Krauß, Sarah Leon Rojas), ISMB (Teodoro Montanaro)**

Published by the Storage4Grid Consortium

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731155.

Document History

Version	Date	Author(s)	Description
0.1	2018-05-07	Veronika Krauß (FIT)	Initial TOC
0.2	2018-08-02	Veronika Krauß (FIT)	Added Introduction, removed Professional GUI for Grid Operators
0.3	2018-08-06	Veronika Krauß (FIT)	Added content for Section 3
0.4	2018-08-20	Teodoro Montanaro (ISMB)	Added content for Section 2
0.5	2018-08-22	Veronika Krauß (FIT)	Added Executive Summary and Conclusions
1.0	2018-08-23	Veronika Krauß (FIT)	Final version, ready for submission to the EC

Internal Review History

Review Date	Reviewer	Summary of Comments
2018-08-22 (v0.5)	Vasco Delgado-Gomes (UNINOVA)	Approved: • Minor corrections.
2018-08-22 (v0.5)	Yini Xu (LIBAL)	Approved: • Minor changes in the text

Legal Notice
The research work leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731155 - Storage4Grid project. The information in this document is subject to change without notice. The Members of the Storage4Grid Consortium make no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Members of the Storage4Grid Consortium shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material. The European Union and the Innovation and Networks Executive Agency (INEA) are not responsible for any use that may be made of the information contained therein.

Table of Contents

Executive Summary

This document summarizes the work of Task T6.3 and describes the developed software prototypes for professional and residential users as GUIs. The prototypes are developed following the requirements collected during end-user workshops to ensure their relevance as well as their applicability in the corresponding domains. While the residential GUI is developed as a web application which is mainly displayed on mobile devices, the professional GUI will mostly be accessed via desktop PC. The residential GUI empowers private house owners to monitor their production, consumption, state of charge of their storage system and – in case they own one –additional information about their electric vehicle. Further information about the residential GUI is provided in Section 2.

The professional GUI which is detailed in Section 3 addresses the needs of DSO grid planners who are investigating in future storage solutions to secure the correct voltage levels on low voltage grids. Those grid planners can simulate the impact of storage virtually placed in the current grid topology by running a simulation using the professional GUI.

1 Introduction

D6.8 describes the "Updated Interfaces for Professional and Residential Users" prototype, developed by the Storage4Grid project. Similar to other prototypes, this document provides a minimal technical documentation necessary to understand the functionalities, structure and deployment instructions for the residential and the professional GUI prototype. Information that is more detailed can be retrieved from related documents summarized in Section 1.2.

1.1 Scope

This prototype deliverable has been developed by Task 6.3 – "Interaction with Residential and Professional Users" and updates D6.7 "Initial Interfaces for Professional and Residential Users". Further update of this prototype is to be released as D6.9 "Final Interfaces for Professional and Residential users" (M33). The implementation is based on the end-user requirements collected by WP2.

1.2 Related documents

ID	Title	Reference	Version	Date
[RD.1]	Final Storage Scenarios and Use Cases	D2.2	V1.0	31/05/2018
[RD.2]	Updated Requirements and Lessons Learned Report	D2.5	V1.0	31/05/2018
[RD.3]	Updated S4G Components, Interfaces and Architecture Specification	D3.2	V1.0	31/08/2018
[RD.4]	Initial Interfaces for Professional and Residential Users	D6.7	V1.0	31/08/2017
[RD.5]	Updated User-side ESS control system	D4.2	V1.0	14/06/18
[RD.6]	Initial Cooperative EV charging station control algorithms	D4.6	V1.0	31/08/2018
[RD.7]	Initial DSF Hybrid Engine	D5.1	V1.0	28/02/2018
[RD.8]	POPD Requirement No.1	D8.1	V1.0	31/03/2017

2 Residential Interfaces

2.1 Overview

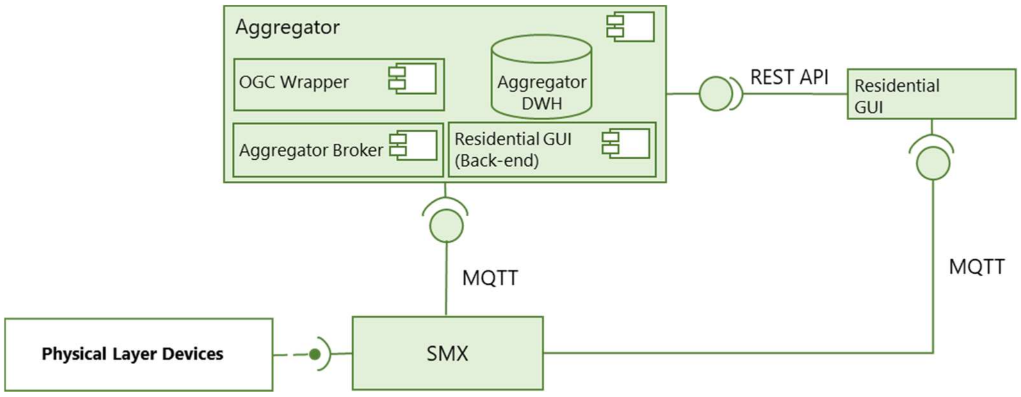


Figure 1 – The prototypal structure of the system that provides the residential GUI (extracted from D3.2)

The Residential Graphical User Interface (GUI) enables users who are hosting a residential ESS and/or a controllable EV charging system to monitor the behaviour of their smart energy management system through a local (i.e., installed and accessible only within their private Local Area Network) dedicated graphical interface. Figure 1 summarizes the high-level structure of the prototype of the system that provides the residential GUI to end-users.
As depicted in Figure 1, various types of sensors continuously send updated data to the Smart Meter eXtension (SMX) framework that exposes such data through some dedicated MQTT topics.
The Aggregator and the Residential GUI are then registered to such dedicated MQTT topics to receive real-time data with the aim of providing their services. One of the services provided by the Aggregator is used by the Residential GUI to access historical information.

The sub-components of the prototype are shortly summarized in the following subsections.

2.1.1 Aggregator

When a new OGC formatted data (e.g., the current power consumption) is received by the "Aggregator Broker", the Aggregator parses it through the "OGC Wrapper" and stores the contained information into the "Aggregator Data Warehouse (DWH)".
Consequently, the received information are exposed though the "Residential GUI (Back-end)" and its REST API.

2.1.2 Residential GUI

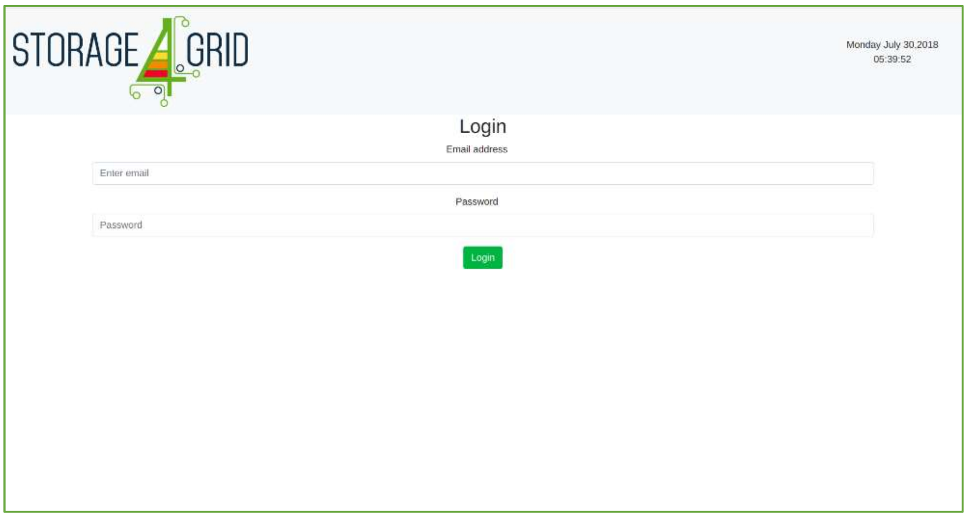
The Residential GUI gets real-time data from the SMX and historical data from the Aggregator and acts as dashboard for users. Specifically, it provides a responsive website (i.e., that can be used on different devices and screen sizes) to:

a) Log in into the system through personal credentials (a screenshot of the view is reported in Figure 2)
b) Show in a summary view (a screenshot of the view is reported in Figure 4):
    - The health status of the system
    - The current level of the house independency (i.e., a percentage value that indicates how and if the current production covers the current consumption)
    - The current consumption of the household in real time
    - The current production of the household in real time
    - The current charging status of the household battery SOC (i.e., plugged/unplugged)

- The current charging status of the available car (i.e., plugged/unplugged)
- The current behavioural mode selected by the user (e.g., "maximize the battery life" mode)

c) Present a data analysis of consumption and production over various time frames in a dedicated view (reported in Figure 3)
d) Set and update user preferences about email updates
e) Set and update user personal information (e.g., username and password)

(a)



(b)

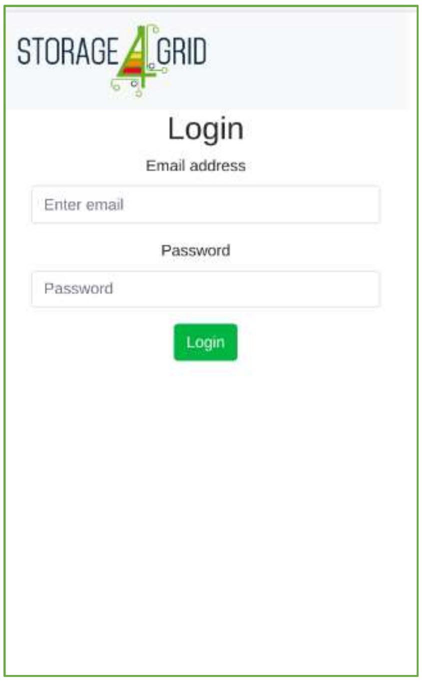
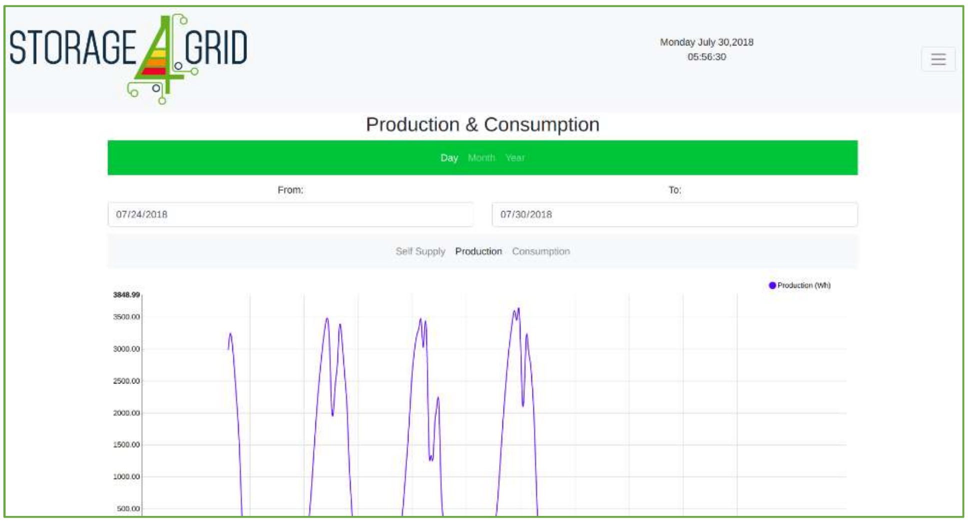


Figure 2 – Login view: a) desktop and b) mobile version

(a)



(b)

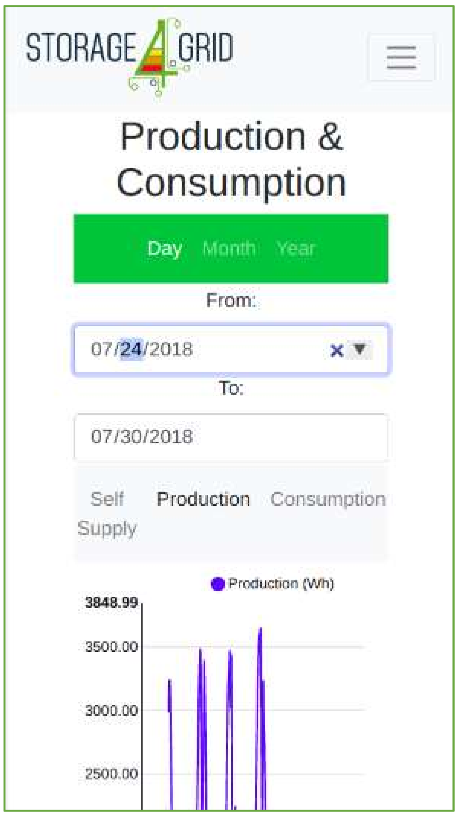
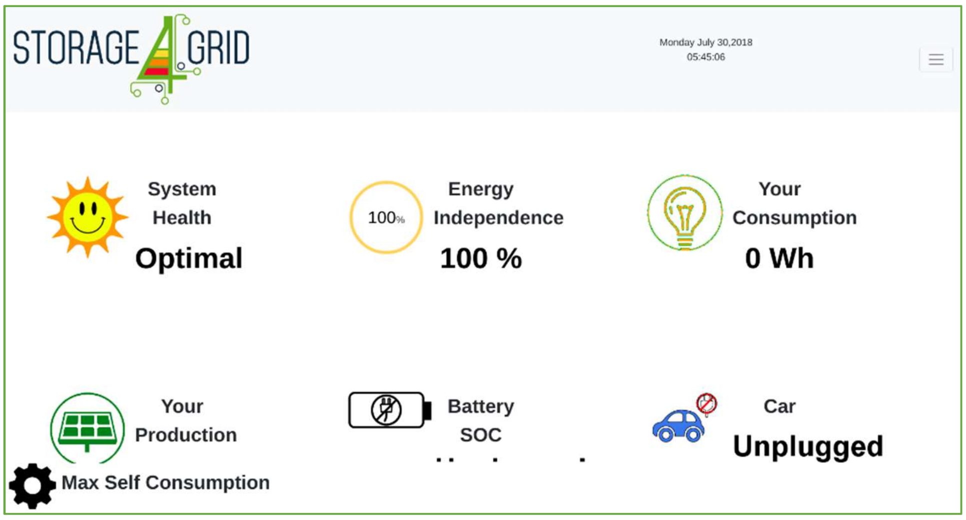


Figure 3 – Production & Consumption view: a) desktop and b) mobile version

(a)



(b)

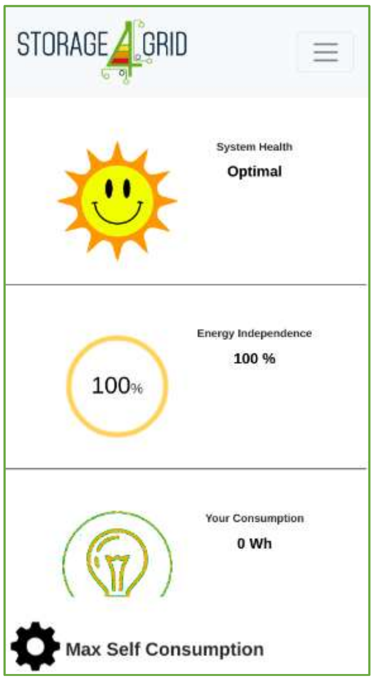


Figure 4 – Home view: a) desktop and b) mobile version

2.2 Installation/Deployment instructions

The Residential GUI can be served on a simple http-server e.g. Apache (https://httpd.apache.org/). It should be installed on the same LAN in which the SMX is installed to receive the data through the MQTT topics.

2.3 Software dependencies and requirements

No special hardware requirements are needed to use and/or deploy the first prototype of the Residential GUI. The prototype is based on the JavaScript framework AngularJS[i] and uses the Firebase Auth[ii] service offered by Firebase[iii] (a cloud mobile[iv] and web application[v] development platform) for authentication.

Table 1 – Software dependencies of the residential GUI

Dependency	License	Role
Apache httpd, version 2.4.26	Apache version 2.0	Used to register and expose web-pages for the GUI
AngularJS V. 1.5.5, https://angularjs.org/	MIT License	Framework on which the residential GUI is based
Firebase Auth, https://firebase.google.com/docs/auth/	Google Cloud Platform License Agreement	Authentication service

2.4 API Reverence

The Residential GUI prototype gets real-time data from the SMX and historical data from the Aggregator. In addition, it interacts with the Firebase Auth service to authenticate users.

- The interactions with the SMX are based on the MQTT protocol, thus the data are intercepted on the dedicated topics. In the current prototype, there is only one available **MQTT topic**: **RESIDENTIAL/GUI**.
  No special requirements are specified by the MQTT protocol for the data format, thus the following text format is used:

```
<SENSOR_ID> KEY1=<value1>,KEY2=<value2>,...,KEYn=<valuen>
datatype=<dataType> <timestamp>
```

Figure 5 – Data format used on the MQTT topic

- The communication with the Aggregator is guaranteed by the REST API exposed by the Aggregator itself. An overview of the API exposed by the current prototype version of the Aggregator is depicted in the following Table 2.

Table 2 – API exposed by the aggregator

Resource path	Available methods
/query	**GET**: Get historical data from the aggregator. **Parameters** accepted in the request: • **q**: influxDB query to select the needed data

- The authentication is guaranteed by the Firebase Auth service through its exposed API. An overview of the API exposed by the Firebase service is available at the following URI: https://firebase.google.com/docs/reference/rest/auth/.

3 Professional Interfaces Grid Planner

3.1 Overview

The overall, high-level structure of the professional GUI is summarized in Figure 6.

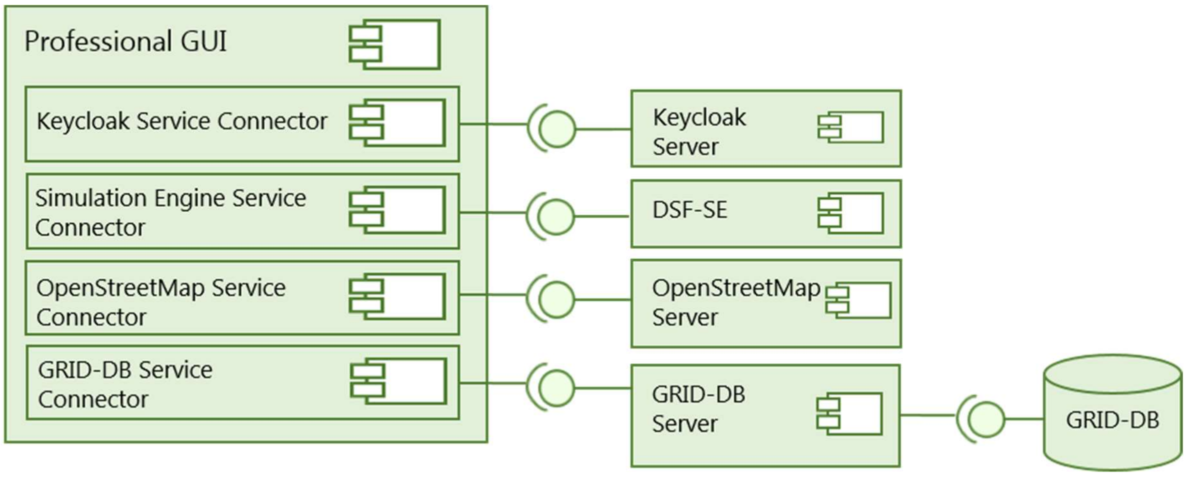

Professional GUI
Keycloak Service Connector
Simulation Engine Service Connector
OpenStreetMap Service Connector
GRID-DB Service Connector
Keycloak Server
DSF-SE
OpenStreetMap Server
GRID-DB Server
GRID-DB

Figure 6 – The structure of the professional GUI prototype

The professional interface enables grid planners to observe the impact of storage in the grid by simulating the voltage levels over a user-defined time frame in a radial selected by the grid planner. Therefore, a connection to the available data sources such as a dedicated GRID-DB storing and providing information about the grid topology, as well as the simulation engine (DSF-SE) for running the simulation of storage elements etc. needs to be established. Whereas the DSF-SE API for the communication with the simulation engine is already defined and in-use (see Section 3.4), the GRID-DB is future work and still needs to be defined based on data structures actually used by the DSOs.
The professional GUI inclusion in the overall S4G software environment is further being described in D3.2 [M21], especially in D3.2 – Figure 13. The screenshots presented in Figure 7 and Figure 8 show how the GUI supports the grid planner by displaying the grid topology and allowing the installation of new storage models to set up the simulation environment.

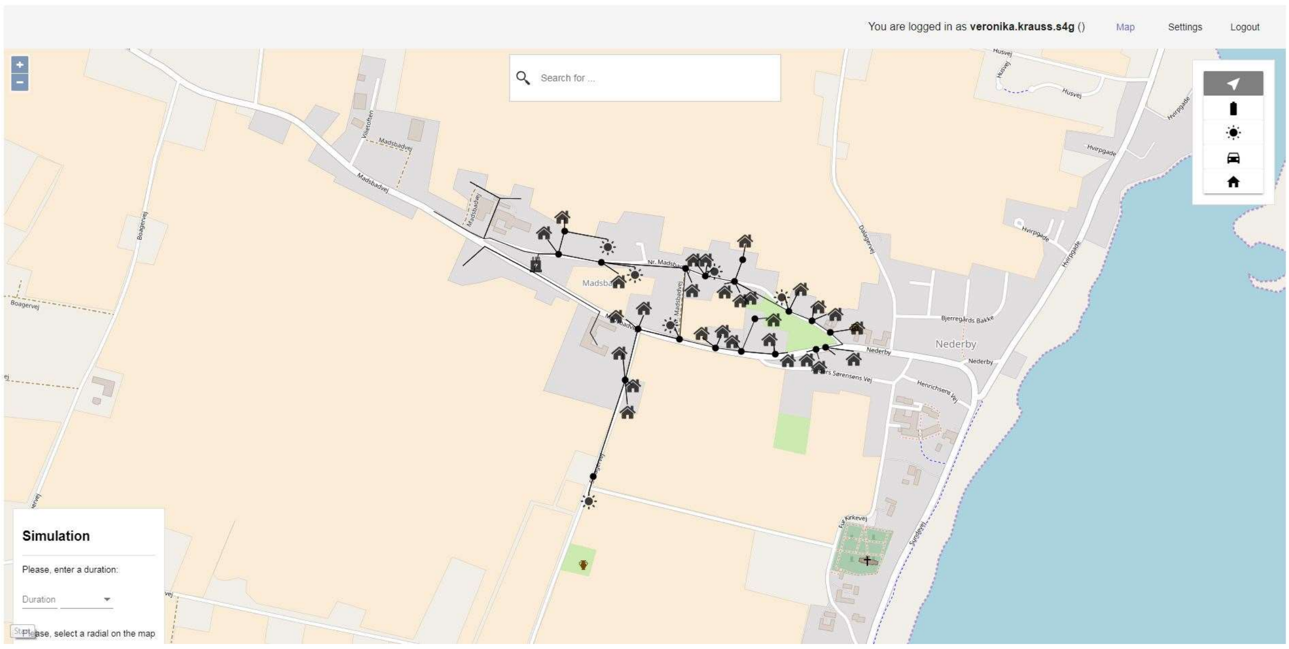

You are logged in as veronika.krauss.s4g ()
Map
Settings
Logout
Search for ...
Simulation
Please, enter a duration:
Duration
Please, select a radial on the map
Nederby

Figure 7 – Screenshot of the professional GUI displaying the grid topology of Fur

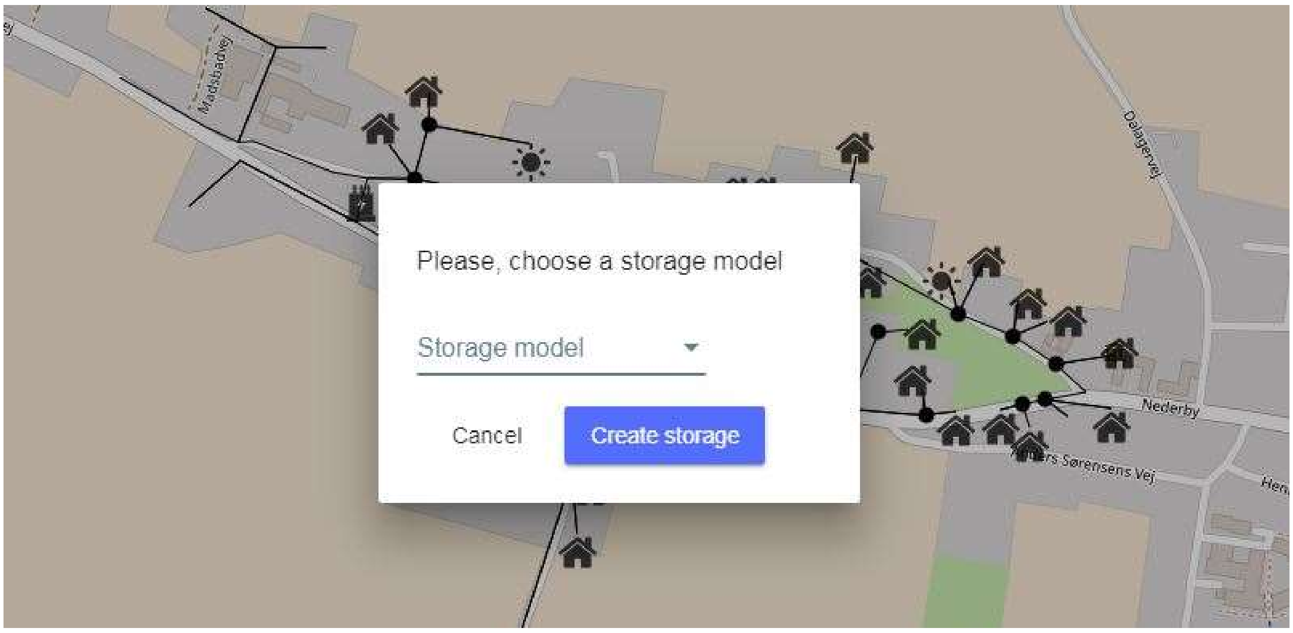


Figure 8 – Storage selection mask for determining the storage type used in the simulation

The sub-components of the prototype are shortly summarized in the following subsections.

3.1.1 Keycloak Service Connector

The Keycloak Service connector is established based on the Keycloak JavaScript Adapter listed in Table 3. It allows for end-user authentication using a REST API and an access/ ID token approach.

3.1.2 Simulation Engine Service Connector

The Simulation Engine Service connector establishes data exchange between the professional GUI and the DSF-SE based on a REST API. Hereby, the GUI acts as visual component for operating the DSF-SE by enabling the end-user to specify data that should be used in the simulation, start/stop the simulation as well as manipulate the current grid topology and query/download simulation results.

3.1.3 GRID-DB Service Connector

A REST API connector for communicating with the GRID-DB service; the GRID-DB holds information about the grid topology of interest. The API still needs to be defined based on the information provided by the DSOs regarding data models and sources.

3.1.4 OpenStreetMap Service Connector

A connector implemented using Openlayers; it connects to OpenStreetMap via REST API to get map data.

3.2 Installation/Deployment instructions

The application can be served on a simple http-server e.g. Apache (https://httpd.apache.org/).

3.3 Software dependencies and requirements

No special hardware requirements are needed to use and/or deploy the professional GUI. The prototype is based on the JavaScript framework Angular[vi] and uses Keycloak[vii] for authentication.

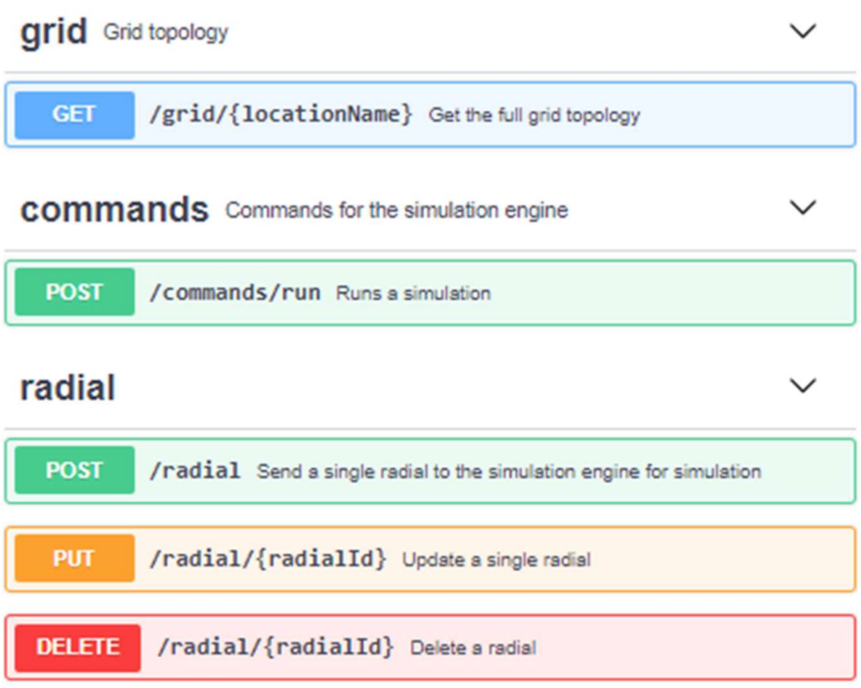
Table 3 – Software Dependencies of the professional GUI

Dependency	License	Role
Angular V. 5.2.11, https://angular.io/	MIT License	Framework on which the professional GUI is based
Keycloak V. 4.1.0, https://www.keycloak.org/index.html	Apache version 2.0	Authentication service
Keycloak JavaScript Adapter V. 4.1.0, https://www.npmjs.com/package/keycloak-js	Apache version 2.0	Connects the professional GUI with the Keycloak authentication service
Openlayers V. 5.0.0, https://github.com/openlayers	2-Clause BSD	Communicates with OpenStreetMap and enhances the visualization of map data
Node Package Manager npm, https://www.npmjs.com/	Artistic License 2.0	Package manager

3.4 Server connection

The professional GUI prototype connects to both the GRID-DB keeping grid-relevant data such as topology information as well as the DSF-SE via a dedicated REST API specified for each component. More information about the GRID-DB and the DSF-SE is documented in the related documents D3.2 [M21] and D5.1 [M15]. An overview of the API for the DSF-SE in its latest version 0.0.3 is depicted in Figure 9. An API for connecting to the GRID-DB will be provided in D6.9 [M33].
For accessing data, user authentication is required using a Keycloak-based service, which offers a dedicated API for authentication requests. The authentication is established via access & ID token with a validity of 30 minutes.

Map data is provided via OpenStreetMap[viii] and accessed using the API offered by the library Openlayers listed in Table 3.



grid Grid topology

GET /grid/{locationName} Get the full grid topology

commands Commands for the simulation engine

POST /commands/run Runs a simulation

radial

POST /radial Send a single radial to the simulation engine for simulation

PUT /radial/{radialId} Update a single radial

DELETE /radial/{radialId} Delete a radial

Figure 9 – API for the communication between the professional GUI and the DSF-SE in version 0.0.3

4 Conclusions

This deliverable presented the current implementation state of the two GUI software prototypes for residential and professional users. Further development will be conducted following the user needs and requirements collected in WP2 and documented in the issue-tracking software JIRA. Additionally, the prototypes will be evaluated with actual end-users to ensure a high quality and applicability of the proposed solutions in the problem domain addressed by S4G. Further updates will be presented in D6.9 Final interfaces for professional and residential users (M33).

Acronyms

Acronym	Explanation
API	Application Programming Interface
DSF-DWH	Decision Support Framework Data Warehouse
DSF-SE	Decision Support Framework Simulation Engine
DSO	Distributed System Operator
GUI	Graphical User Interface
GRID-DB	Grid Data base, a data base providing information about the grid topology
MQTT	Message Queue Telemetry Transport
OGC	Open Geospatial Consortium
REST	Representational State Transfer
SMX	Smart Meter eXtensions
SOC	State of charge

List of figures

List of tables

References

i AngularJS V. 1.5.5, https://angularjs.org/, last accessed 2018/08/20
ii Firebase Auth service, https://firebase.google.com/docs/auth/, last accessed 2018/08/20
iii Firebase, https://firebase.google.com/, last accessed 2018/08/20
iv Mobile App, https://en.wikipedia.org/wiki/Mobile_app, last accessed 2018/08/23
v Web Application, https://en.wikipedia.org/wiki/Web_application, last accessed 2018/08/23
vi Angular, https://angular.io/, last accessed 2018/08/08
vii Keycloak, https://www.keycloak.org/index.html, last accessed 2018/08/08
viii OpenStreetMap, https://www.openstreetmap.de/, last accessed 2018/08/08