



BIM based fast toolkit for  
Efficient rEnovation in Buildings

# D4.7 API, Master End-User Front End



This project has received funding from  
European Union's H2020 research and innovation  
programme under grant agreement N. 820660

The content of this document reflects only the author's  
view only and the Commission is not responsible for any  
use that may be made of the information it contains.

<b>Programmes</b>	H2020
<b>Call for Proposal</b>	LC-EEB-02-2018 Building information modelling adapted to efficient renovation
<b>Project Title</b>	BIM based fast toolkit for Efficient rEnovation in Buildings
<b>Acronym</b>	BIM4EEB
<b>Project Grant Agreement</b>	820660

<b>Work Package</b>	WP4
<b>Lead Partner</b>	One Team
<b>Contributing Partner(s)</b>	One Team
<b>Dissemination Level</b>	Public
<b>Type</b>	Other
<b>Due date</b>	30/06/2020
<b>Date</b>	30/06/2020
<b>Version</b>	1.0

## DOCUMENT HISTORY

Version	Date	Comments	Main Authors
0.1	10.03.2020	TOC added	Davide Madeddu (One Team), Alessandro Valra (One Team), Diego Farina (One Team), Jacopo Chiappetti (One Team)
0.5	21.04.2020	First draft	Davide Madeddu (One Team), Alessandro Valra (One Team), Diego Farina (One Team), Jacopo Chiappetti (One Team)
0.7	26.05.2020	Second draft	Davide Madeddu (One Team), Alessandro Valra (One Team), Diego Farina (One Team), Jacopo Chiappetti (One Team)
0.8	08.06.2020	Ready for review	One Team
0.9	12.06.2020	Review Done	Kostas Tsatsakis (Suite 5), Teemu Matasniemi (VTT)
1.0 FINAL	29.06.2020	Final Version	Davide Madeddu (One Team), Alessandro Valra (One Team), Diego Farina (One Team), Jacopo Chiappetti (One Team)

### Statement of originality:

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

BIM4EEB action has received funding from the European Union under grant agreement number 820660.

The information in this document is provided "as is", and no guarantee or warranty is given that the information is fit for any particular purpose. The above referenced consortium members shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials subject to any liability which is mandatory due to applicable law.

## EXECUTIVE SUMMARY

This deliverable deals with the development of the BIM Management System (BIMMS) Application Programming Interfaces (API), the APIs documentation and the Front-End User manual.

The BIMMS is a platform built around a Common Data Environment (CDE) that stores all the data and information gathered through different sources and along the whole building life-cycle, acting a single source of truth (SSOT). The CDE allows to collaborate and to store, share and visualise BIM and GIS (Geographic Information System) models, manage data and link the data-streaming from sensors to the models. The APIs are a set of procedures that define the interaction between two applications in order to exchange data and perform tasks. The BIMMS platform integrates a set of digital tools (BIM4EEB toolkit) to support BIM-based building renovation that can exchange data through Web APIs. The APIs have become very popular and essential to connect applications thanks to the spread use of the cloud-based software.

The first section of the deliverable introduces the BIMMS services dedicated to data exchange and communication between the applications and the CDE. The second section describes the API documentation making a list of all methods with sample requests and responses. Finally, the last section provides the user manual of the BIMMS front end.

This deliverable illustrates the work carried out in the Working Package (WP) 4 - Task 4.5 started in M13 and that will end in M30, after 17 months of duration. The development plan has set two main goals: the former in M18 (June 2020) in which a preliminary toolset for integration testing with the core functionality has been released and the latter in M24 (December 2020) in which the final toolset for validating the functionalities will be released.

This deliverable provides the status of the APIs and the BIMMS front end available until M18 (June2020). Further developments are expected and the final documentation will be updated based on the new features that will be implemented in the final toolset.

## PUBLISHING SUMMARY

This deliverable provides the documentation for the BIM Management System (BIMMS) Application Programming Interfaces (API) and the BIMMS front end user manual. The BIMMS is a platform built around a Common Data Environment (CDE) that stores all the data and information gathered through different sources and along the whole building life-cycle. The APIs are a set of procedures that define the interaction between two applications in order to exchange data and perform tasks. In light of these considerations, the APIs documentation helps the developers to connect their applications within the BIMMS. The front end user manual helps the end users on how to use the BIMMS web portal.

## TABLE OF CONTENTS

1	Introduction.....	8
1.1	Scope of the document .....	8
1.2	Relevance with the other deliverables .....	8
1.3	Structure of the document .....	8
2	The BIMMS Exchange Layer Services.....	9
2.1	Development roadmap .....	11
3	API .....	13
3.1	OTMySQL .....	15
3.1.1	USER API .....	15
3.1.2	PROJECT API.....	21
3.1.3	IFC API .....	25
3.1.4	RESOURCE API .....	35
3.1.5	SENSOR API .....	39
3.1.6	ENDUSER API.....	46
3.1.7	OCCUPANCY API.....	50
3.1.8	ACTIVITY API .....	54
3.1.9	ALERT API.....	57
3.2	OTSPARQL.....	60
3.2.1	POST sparql/getSPARQLData .....	60
3.3	OTVirtuoso.....	62
3.3.1	POST api/OTVirtuoso.....	62
3.3.2	DELETE api/OTVirtuoso .....	63
3.3.3	DELETE api/OTVirtuoso/DropSilentGraph .....	64
3.4	OTGeoLinkedData .....	66
3.4.1	GET api/OTGeoLinkedData .....	66
3.5	OTRdf .....	67
3.5.1	POST api/OTRdf .....	67
4	BIM Management System (BIMMS) User Manual.....	69
4.1	BIMMS User access.....	69
4.2	BIMMS User Registration workflow .....	70
4.3	BIMMS Home page.....	74
4.4	BIMMS Interface .....	74
4.5	BIMMS Navigation Menu.....	75

4.5.1	User profile settings.....	75
4.5.2	Project area access.....	75
4.6	Main menu .....	76
4.6.1	Home Page .....	76
4.6.2	Roles and rules .....	76
4.6.3	Resource management .....	79
4.6.4	BIM models .....	92
4.6.5	BIMMS 3D IFC BIM Viewer .....	93
4.6.6	SPARQL Endpoint.....	103
4.6.7	Ontology viewer .....	107
4.6.8	Data streaming.....	108
4.6.9	API Enabled users.....	115
4.6.10	Tools .....	116
5	Conclusions .....	117

## LIST OF FIGURES

Figure 1 – The main BIMMS modules .....	9
Figure 2 – An example of API request and response.....	10
Figure 3 - Login and access form .....	69
Figure 4 – User registration form .....	70
Figure 5 – BIMMS macro-roles.....	71
Figure 6 – BIMMS registration steps.....	71
Figure 7 – Account menu .....	72
Figure 8 – Project authorization process.....	72
Figure 9 – User Top menu with authorized Project and Roles .....	73
Figure 10 – New user Invitation workflow .....	73
Figure 11 – BIMMS interface areas .....	74
Figure 12 – BIMMS Context navigation menu.....	75
Figure 13 – BIMMS homepage.....	76
Figure 14 – BIMMS user management dashboard .....	77
Figure 15 – BIMMS Project management .....	78
Figure 16 – BIMMS Ontology reference management.....	78
Figure 17 – BIMMS Resource creation wizard – step 1 .....	80
Figure 18 – BIMMS Resource creation wizard – step 2 .....	81
Figure 19 – BIMMS Resource creation wizard – step 3 .....	81
Figure 20 – BIMMS Resource creation wizard – step 4 .....	82
Figure 21 – BIMMS Resource creation wizard – step 5 .....	83
Figure 22 – BIMMS Resource creation wizard – step 6 .....	84
Figure 23 – BIMMS Resource creation wizard – step 7 .....	85

Figure 24 – BIMMS Resource creation wizard – step 8 .....	85
Figure 25 – Resource List dashboard .....	86
Figure 26 - Resource revision selector .....	87
Figure 27 – Resource classification Tabs .....	87
Figure 28 – Resource new revision step 1 .....	88
Figure 29 – GIS map of georeferenced Project data .....	89
Figure 30 - New linked data object wizard step 1 .....	90
Figure 31 - New linked data object wizard step 2 .....	90
Figure 32 - New linked data object wizard step 3 .....	91
Figure 33 – BIM Models dashboard .....	92
Figure 34 – BIM Models BIMMS viewer .....	93
Figure 35 – BIM Models viewer .....	94
Figure 36 – BIM Models viewer – hide objects .....	95
Figure 37 – BIM Models BIMMS viewer – commands and options .....	96
Figure 38 – BIM Models viewer – Data access .....	97
Figure 39 – BIM Models viewer – Hierarchy & Zones Access .....	98
Figure 40 – BIM Models viewer – Zones .....	99
Figure 41 – BIM Models viewer – Floors management .....	100
Figure 42 – BIM Models viewer – Floors management result .....	100
Figure 43 – BIM Models viewer – object Linked data access .....	101
Figure 44 – BIM Models viewer – object Linked data access result .....	101
Figure 45 – BIM Models viewer – object Linked data .....	102
Figure 46 – SPARQL Interface .....	103
Figure 47 – SPARQL sample query .....	104
Figure 48 – SPARQL sample query .....	104
Figure 49 – SPARQL sample query result HTML .....	105
Figure 50 – SPARQL sample query result Raw .....	105
Figure 51 – SPARQL sample query result Graph .....	106
Figure 52 – SPARQL sample query result – ifc linked objects .....	106
Figure 53 – RDF Import interface .....	107
Figure 54 – Ontology viewer interface .....	108
Figure 55 – Project Sensor List .....	108
Figure 56 – Project Measurements List .....	109
Figure 57 – Sensors Import interface .....	109
Figure 58 – Sensors Import interface .....	110
Figure 59 – Measurements Import interface .....	111
Figure 60 – Measurements Import interface .....	112
Figure 61 – BIMMS endpoints Help Page .....	113
Figure 62 – BIMMS endpoints Swagger test page - 1 .....	113
Figure 63 – BIMMS endpoints Swagger test page - 2 .....	114
Figure 64 – BIMMS endpoints Swagger test page - 3 .....	115
Figure 65 – API Enabled user management interface .....	115
Figure 66 – Tools page .....	116

## 1 Introduction

---

### 1.1 Scope of the document

The activities carried out in Task 4.5 to develop the BIMMS APIs have provided two outcomes: (1) the APIs documentation and (2) the BIMMS front end user manual. The APIs documentation gives heads-up to the developers for the integration of their applications in the BIMMS. Instead, the front end user manual describes how to use the BIMMS web portal.

### 1.2 Relevance with the other deliverables

The development of BIMMS APIs is based on the requirements published in previous deliverables. Indeed, the APIs have been developed considering the specifications defined in the Task 4.1 with the deliverable D 4.1 “Specifications and overall design of a BIM management system”, where it is possible to find the essential functionalities for the tools developed in WP5, WP6 and WP7, i.e., how to collect, exchange and structure the information. The WP 4 Task 4.5 works actively to support the other WPs developers’ needs and to set detailed specifications and common standards to allow the data exchange among the applications. More details about the software architecture, the technical configuration of the platform and the development of the additional services on the BIMMS are available in these deliverables:

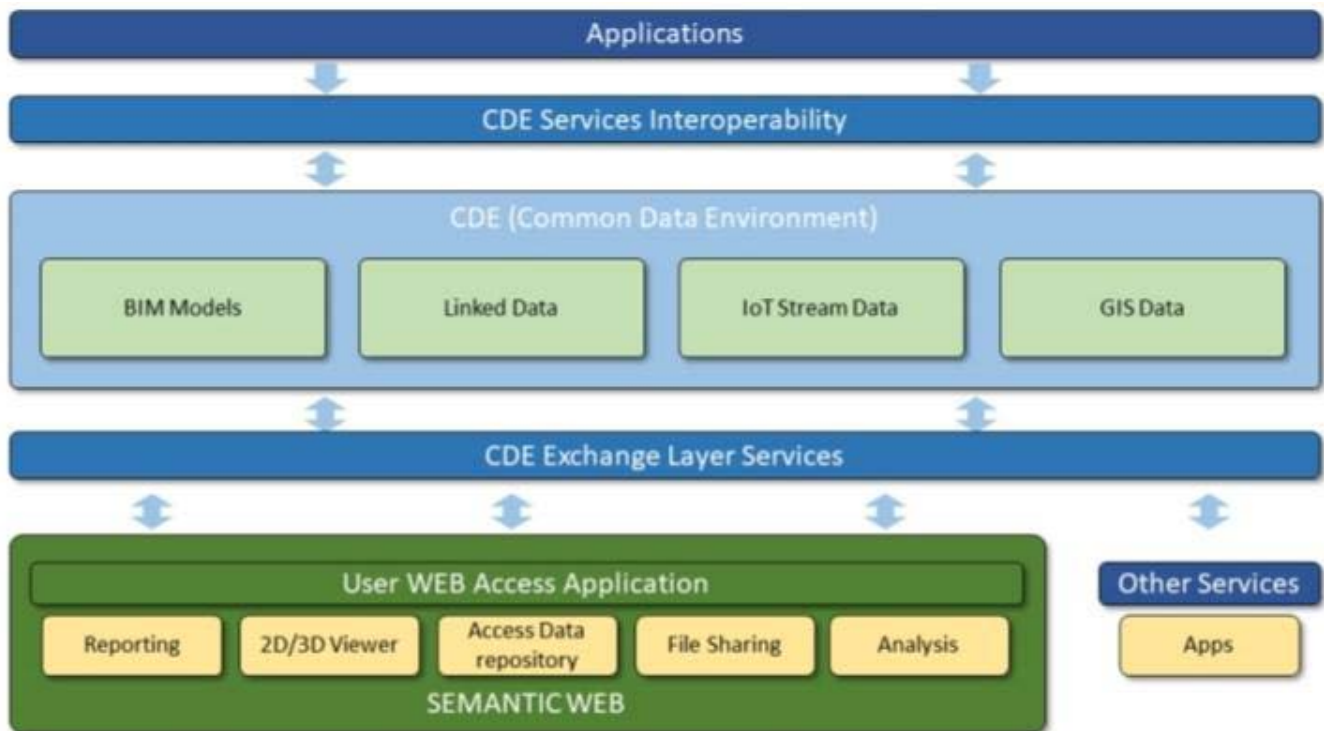
- PUBLIC DELIVERABLES:
  - Deliverable D4.8 “Guidelines for the integration of new tools in the BIM management system” describes the developers and reference documentation to integrate new tools in the BIM management system.
- CONFIDENTIAL ONLY FOR CONSORTIUM OR ADVISORY BOARD MEMBERS
  - Deliverable D4.3 “CDE Database, Core DB functionalities, Ontological representation, BIM data translation engine to ontology” describes the CDE DB, enriched with the core DB functionalities such as ontological representations of items and their relationships, a translation engine from BIM models to ontology, and etc..
  - Deliverable D4.4 “CDE Services, Interoperability Services, Exchange Layer Services, I/O Protocols and Data specification” deals with the CDE Services Interoperability Services, Exchange Layer Services, I/O Protocols and Data specification.
  - Deliverable D4.5 “Technical configuration of the platform”, as the name suggests, gives indications about hardware and software configuration, OS and DB setup.
  - Deliverable D4.6 “Guidelines for the implementation of the BIM management system” provides the documentation such as user’s manual and guidelines to define objects, relationships and data.

### 1.3 Structure of the document

The document is structured in three sections: the first one introduces the BIMMS services dedicated to data exchange and communication between the applications and the CDE. The second one provides the documentation of the API making a list of all methods with sample requests and responses. Finally, the third section provides the user manual of the BIMMS front end.



## 2 The BIMMS Exchange Layer Services



**Figure 1 – The main BIMMS modules**

The access to the data is carried out by the Exchange Layer services that works as interface between applications and the Common Data Environment where data are stored. The applications are the tools developed during the BIM4EEB Project by the Partners and all other tools and services that will be developed later and that will be used to exchange data with BIMMS.

The Exchange Layer Services integrate tools and services using REST principles. REST stands for Representational State Transfer and it is a very common and consolidated architectural model used by web applications to access and exchange data. The most known services and applications over Internet use and share their REST APIs to manage data and exploit application integration. The REST model was created by Roy Fielding that defined some principles (Fielding, 2000):

- Client-Server architecture: the REST architecture is based on the separation of the user interface from the data storage to improve the portability across multiple platforms.
- Stateless: No client context is being stored in the server among requests in the client-server communication. Each request from any client contains only the information necessary to service the request. Session state is held in the client.
- Cache: clients and intermediaries can cache responses to eliminate some client-server interactions and to further improve scalability and performance
- Uniform interface among components: creating a general component interface allows for the simplification of the overall system architecture and the visibility of interactions are improved. Each part of the system can evolve independently.
- Layered System: the architecture can be structured in hierarchical layers by constraining component behaviour. Each component cannot see beyond the immediate layer which they are interacting.

- Code on demand: the architecture allows client functionality to be extended by downloading and executing code in the form of applets or scripts.

This architecture was chosen considering some important benefits that affect the BIMMS Platform and the BIM4EEB tools development:

- REST architecture is independent from programming languages and software platforms. The developers are freely to use any kind of platform: in the context of BIM4EEB project some partners use Java, PHP, Python and NET languages and Microsoft and Linux Servers.
- REST makes independent the evolution of the front-end components (client side) and back-end components (server side), allowing a fast development without developers' constraints.
- REST allows to scale performances due to client-server separation.

The Exchange Layer Service of the BIMMS is a REST service that provides a set of APIs based on the REST architecture. The APIs (Application Programming Interface) are a set of procedures that define the interaction between two applications in order to exchange data and perform tasks. APIs for web applications have get very common and essential for various tasks thanks to the increase of cloud-based applications.

The API allows an application (a client) to access the resources and the services made available by one or more servers. The server (a server) is an application that performs some functionality and serves the resources to the clients. The client can be an application on an end user device, or an application running on another server. Both applications communicate each other through the APIs. Often APIs are called Web API because provides functionalities by means of web requests over the internet, more commonly through HTTP messages. The communication acts within requests and responses in a format documented and defined by the developer. These guidelines explain how communication happens, which requests and responses are defined in the BIMMS Exchange Layer in form of APIs.

The Web API communicates through HTTP protocol which allows the fetching of resources. Resources are defined as an identifiable item by a URL (Uniform Resource Locator) that can be served by an application server. An HTML web page shown in a browser is the most common web resource but can be also served media (images and videos), files, JSON and XML objects.

```
GET / HTTP/1.1
Host: developer.mozilla.org
Accept: *
```

```
HTTP/1.1 200 OK
Date: Sat, 09 Oct 2010 14:28:02 GMT
Server: Apache
Last-Modified: Tue, 01 Dec 2009 20:18:22 GMT
ETag: "51142bc1-7449-479b075b2891b"
Accept-Ranges: bytes
Content-Length: 29769
Content-Type: text/html

<html><body></body></html>
```

**Figure 2 – An example of API request and response**

Usually, the requests and responses are done through HTTP protocol and can be encapsulated in frames and encrypted for security reasons or left in clear message as human readable text. The request shall be made by HTTP Methods. These are also called verbs and describe the type of request from the client. The most common HTTP Methods used in the API requests are the following:

- GET: Used to request a resource and to retrieve data.
- POST: Used to send data to the server. Usually this type of request is used to create a new resource or save data on a web server. Often this request includes some content as body or payload. The different type of body requests can be indicated by the Content-Type header.
- PUT: used for updating or replacing resources.
- DELETE: Used to delete the specified resource.

There are also other methods used as PATCH, CONNECT, HEAD, OPTIONS and TRACE.

After sending the request, the server gives an answer with a status code and a response message. The status code is important because it indicates whether a specific HTTP request has been successfully completed. The status codes are defined as numbers and grouped in types: informational (1xx), successful responses (2xx), redirects (3xx), client errors (4xx), server errors (5xx). The most common statuses are:

- 200 Successful request
- 201 The resource was successfully created
- 401 Unauthorized
- 403 Forbidden; you cannot access
- 404 Not found; the requested asset does not exist, or you are not authorized to see it
- 500 The server is having issues.

## 2.1 Development roadmap

The activities described in this document regard to the Working Package (WP) 4 - Task 4.5 started in M13 and that it takes 17 months. Two main goals are set: the first one in M18 (June 2020) when a preliminary toolset for integration testing purposes with the core functionality is released, and the second one in M24 (December 2020) where the final toolset for validating the functionalities will be released. After M24 is expected the application of the BIMMS and the entire toolset in the demo cases scenarios. This document illustrates the activities until the first milestone in M18. In this phase, the technical configuration described satisfies the tool integration requirements, both for the preliminary toolset of testing purposes and for the expected final toolset of validation purposes. The APIs development is structured in two main phases:

1. the former started in M12 and closed in M18 (6 months duration) where the requirements analysis and the development of the main BIMMS Platform and the REST data interchange services were performed.
2. the latter will start in M19 and will close in M24 (6 months duration). This phase will be focused on the development of the platform and its services for testing them in the demonstration pilot sites. In this phase the focus will be in the integration of the tools within the methods and procedures defined in this Deliverable. The result in M24 will be the application and the validation of these guidelines to the integration of the new tools. The current development of the guidelines follows

the ongoing work in other tasks of WP4 and in the other WPs. These guidelines aim to satisfy the tool integration requirements for the preliminary toolset of testing purposes in M18 and for the final toolset of validation purposes in M24.

The development of the interface services has been begun before the expected roadmap regarding platform access webservice and the first interconnection functionalities with the tools of the other work packages. This Deliverable reports the current progress of the development of the interoperability REST services within the BIMMS system. The development of services has already produced about 50 methods of interaction with external systems. The APIs are expected to be further integrated in the next months and the final documentation will be updated based on the new features that will be implemented.

### 3 API

---

The BIMMS REST API Service allows to put, edit and get data from and to the BIMMS. To work with API, the user will have to register to the BIMMS.

The list of the REST APIs is available on the BIMMS Web Portal under the Endpoints in the left side menu. The Endpoints page gives a brief documentation of the methods and the access to the Swagger UI service. The Swagger UI is an open source service supported by Smartbear, that helps developers and end users to visualise and interact with the API resources without having any of the implementation logic in the place (Swagger, 2020). The Swagger UI allows to familiarise with the BIMMS API by submitting HTTP requests and getting the responses before write any line of code in development environment and make easy the backend implementation and the client-side consumption.

The APIs have been organised into homogeneous groups, the different API groups are dedicated to different development areas; users enabled to use the API, access all the interoperability features of the services at the moment.

The API groups of the BIMMS system are the following:

- OTMySQL: They are all the APIs of the BIMMS system that use the MySQL Database as the data source, the central database of the BIMMS system. The macro group is divided into sub-areas:
  - User API: these are all the APIs that allow access to authentication services, the definition of user access profiles and the projects accessible to the user who uses the API.
  - Project API: these are all the APIs that allow access to the structure of each projects and graphs associated with the project.
  - IFC API: these are all the APIs that allow you to access the IFC files and structure (remembering that BIMMS also converts IFC files into data within the SQL database tables in order to optimize access to data).
  - Resource API: these are all the APIs that interact with the resources (documents) of the BIMMS system.
  - Sensors API: these are all the APIs that allow the upload and access to raw data of the sensors and associated measurements.
  - EndUser API: these are all the APIs that allow the management of associations between EndUser (Owner or Inhabitants) and the IFC Zones, in order to associate the apartments with the individual end-users.
  - Occupancy API: these are all the APIs that allow access and management of the definition of the occupancy profiles of each Zone.
  - Activity API: these are all the APIs that allow the management (reading and creation) of the Activities related to the IFC Zones.
  - API Alerts: they are all the APIs that allow the management of the Alerts (reading and creation) of the Alerts linked to the IFC Zones.
- OTSPARQL: These are the APIs dedicated to query the data stored in Virtuoso Graph Database. The APIs are always protected because they are accessible only by the users authorized in the BIMMS system via authentication.

- OTVirtuoso: These are APIs, protected by authentication, which allow to load / delete data (Triples) within the Virtuoso Graphs Database, including the creation and deletion functions of the Graphs (created by connected user).
- OTGeoLinkedData: These are APIs, protected by authentication, dedicated to the querying of geographic data stored in Virtuoso Graph Database, associated with the resources of the BIMMS.
- OTRdf: These are APIs, protected by authentication, for the loading of an RDF file within Virtuoso Graph databases.

In the next paragraphs, the reference guides of the APIs available for authorized users are listed. The documentation will be updated as new developments come out in the next months.

### 3.1 OTMySQL

#### 3.1.1 USER API

##### 3.1.1.1 GET /user/getUserToken

<https://bim4eeb.onetteam.it/BIMMSWS/user/getUserToken?username={username}&password={password}>

Get the access token for the given user, BIMMS administrator will enable users in order to access this method. Only API Enabled users can GET their userToken to access all other methods. In this scenario API Enabled users are associated to external Application (BIMMS External Tools).

Parameters	Description	Required	Parameter Type	Data Type
username	Username	yes	query	string
password	Password	yes	query	string

A successful response returns a JSON object with the tokenUser.

Response Body Example

```
"9400dcd7-f0f3-11e9-87b6-235056920005"
```

##### 3.1.1.2 GET /user/getUserProjects

<https://bim4eeb.onetteam.it/BIMMSWS/user/getUserProjects?tokenUser={tokenUser}>

Get the projects for the given tokenUser; this method requires as input the tokenUser and returns a list of projects the user identified by tokenUser has access to.

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string

Response Parameters:

```
"TOKEN": Project token identifier,
"NAME": Project name without spaces,
"COMPANY": Name of Company associated to Project,
"CITY": City Location of Project,
"STATUS": 1 = Active,
"ORIGINAL_NAME": Project Name
```

## Response Body Example

```
[
  {
    "TOKEN": "4d9b742d-102e-11ea-9c7f-005056930005",
    "NAME": "Finnish_demonstration_site",
    "COMPANY": "One Team srl",
    "CITY": "Tampere",
    "STATUS": 1,
    "ORIGINAL_NAME": "Finnish demonstration site"
  },
  {
    "TOKEN": "4d9b742b-102e-11ea-9c7f-005656920005",
    "NAME": "Italian_demonstration_site",
    "COMPANY": "One Team srl",
    "CITY": "Monza",
    "STATUS": 1,
    "ORIGINAL_NAME": "Italian demonstration site"
  },
  {
    "TOKEN": "c4833c12-aa1e-11ea-a75c-00505695f911",
    "NAME": "MyNewProject",
    "COMPANY": "NO COMPANY",
    "CITY": "NewYork",
    "STATUS": 1,
    "ORIGINAL_NAME": "MyNewProject"
  },
  {
    "TOKEN": "4d9b742a-102e-11ea-9c7f-044056920005",
    "NAME": "One_Team_Demo",
    "COMPANY": "NO COMPANY",
    "CITY": "Milano",
    "STATUS": 1,
    "ORIGINAL_NAME": "One Team Demo"
  },
  {
    "TOKEN": "4d9b742c-102e-11ea-9c7f-005057920005",
    "NAME": "Polish_demonstration_site",
    "COMPANY": "One Team srl",
    "CITY": "Chorzow",
    "STATUS": 1,
    "ORIGINAL_NAME": "Polish demonstration site"
  },
  {
    "TOKEN": "036a79ec-a18c-115a-a73c-00505695f911",
    "NAME": "Review_Project",
    "COMPANY": "One Team srl",
    "CITY": "Milano",
  }
]
```



```

"STATUS": 1,
"ORIGINAL_NAME": "Review Project"
},
{
"TOKEN": "4d9911cf-152e-11ea-9c7f-005056920005",
"NAME": "test",
"COMPANY": "One Team srl",
"CITY": "Milano",
"STATUS": 1,
"ORIGINAL_NAME": "test"
},
{
"TOKEN": "4d9b7427-102e-11ea-9c7f-005056920005",
"NAME": "Test2",
"COMPANY": "One Team srl",
"CITY": "Brescia",
"STATUS": 2,
"ORIGINAL_NAME": "Test2"
}
]

```

### 3.1.1.3 GET /user/getUserRoles

<https://bim4eeb.oneteam.it/BIMMSWS/user/getUserRoles?tokenUser={tokenUser}&tokenProject={tokenProject}>

Get the roles for the given project and user token, this method returns only enabled Roles for a user (identified by tokenUser) on a specific Project

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string

Respose Parameters:

```

"NAME": Role Description
"INTERNAL_ROLE": BIMMS internal Role useb by system to defile Role permission

```

Response Body Example

```
[
  {
    "NAME": "Administrator",
    "INTERNAL_ROLE": 1
  },
  {
    "NAME": "Client/Owner",
    "INTERNAL_ROLE": 3
  },
  {
    "NAME": "Health and safety adviser",
    "INTERNAL_ROLE": 4
  }
]
```

**3.1.1.4 GET /user/getUserStages**

<https://bim4eeb.oneteam.it/BIMMSWS/user/getUserStages?tokenUser={tokenUser}&tokenProject={tokenProject}&role={role}>

Get the projects stages for the given user token and role, in actual BIMMS implementation. Stages are not used yet but they can be used to identify several phases (and related data) of the project.

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	query	string
role	role retrieved by user/getUserRoles method	query	string

Response Parameters:

"STAGENAME": Name of Accesible Stage

## Response Body Example

```
[
  {
    "STAGENAME": "Design - Concept design"
  },
  {
    "STAGENAME": "Design - Preliminary design"
  },
  {
    "STAGENAME": "Design - Developed design"
  },
  {
    "STAGENAME": "Design - Detailed design"
  },
  {
    "STAGENAME": "Procurement - Procurement"
  },
  {
    "STAGENAME": "Procurement - Construction contracting"
  },
  {
    "STAGENAME": "Construction - Pre-construction"
  },
  {
    "STAGENAME": "Construction - Commissioning"
  },
  {
    "STAGENAME": "Construction - Handover"
  },
  {
    "STAGENAME": "Construction - Regulatory approval"
  },
  {
    "STAGENAME": "End of life - Revamping"
  }
]
```

```

},
{
  "STAGENAME": "End of life - Dismantling"
},
{
  "STAGENAME": "Initiative - Initiative"
}
]

```

### 3.1.1.5 GET /project/getUserApplications

<https://bim4eeb.onetteam.it/BIMMSWS/user/getUserApplications?tokenUser={tokenUser}>

Get the applications associated to an API Enabled user identified by the given user token

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string

Respose Parameters:

```

"APPLICATION": Name of Application
"APPLICATION_ID": Application identifier

```

Response Body Example:

```

{
  "APPLICATION": "BIM4Occupants"
  "APPLICATION_ID": 1
}

```

### 3.1.2 PROJECT API

#### 3.1.2.1 GET /project/getProjectIFC

Get the IFC files list included in a specific BIMMS Project, for the given project and user token.

<https://bim4eeb.onetteam.it/BIMMSWS/project/getProjectIFC?tokenUser={tokenUser}&tokenProject={tokenProject}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string

Respose Parameters:

<p>"File Name": Original filename,          "Internal File Name": Internal unique Filename,          "Creation User": User that uploaded IFC file,          "Creation Date": Date of creation,          "Note": Notes,          "Building Name": Building name,          "Stage": Stage,          "Sub Stage": SubStage,          "Title": Resource title,          "Description": Resource description,          "Keywords": Resource keywords,          "Synonyms": Resource synonyms,          "Combo": Not used</p>
---

Response Body Example:

<pre>[   {     "File Name": "Palazzo_2019.ifc",     "Internal File Name": "Palazzo_2019_31563461014.ifc",     "Creation User": "Jacopo Chiappetti",</pre>
---

```
"Creation Date": "2019-07-18T14:43:33",
"Note": null,
"Building Name": "Bldg1",
"Stage": null,
"Sub Stage": null,
"Title": "Palazzo One Team",
"Description": "Palazzo One Team - via Winkelman 2",
"Keywords": "",
"Synonyms": "",
"Combo": null
},
{
"File Name": "ABC_r0.ifc",
"Internal File Name": "ABC_r01564505140.ifc",
"Creation User": "Jacopo Chiappetti",
"Creation Date": "2019-07-30T16:45:39",
"Note": null,
"Building Name": "Bldg1",
"Stage": null,
"Sub Stage": null,
"Title": "Test ABC",
"Description": "",
"Keywords": "",
"Synonyms": "",
"Combo": null
},
]
```

### 3.1.2.2 GET /project/getProjectBuildings

Get the list of Buildings associated to a specific BIMMS project, for a given project and user token

<https://bim4eeb.onetteam.it/BIMMSWS/project/getProjectBuildings?tokenUser={tokenUser}&tokenProject={tokenProject}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string

Response Parameters:

"ID": building ID "IDCOMPANY": Company ID "NAME": Building Name "STATUS": 1 - Active
---

Response Body Example:

<pre>[   {     "ID": 1,     "IDCOMPANY": 2,     "NAME": "Bldg1",     "STATUS": 1   },   {     "ID": 2,     "IDCOMPANY": 2,     "NAME": "Bldg2",     "STATUS": 1   } ]</pre>
---

### 3.1.2.3 GET /project/getProjectGraphs

Get the list of Virtuoso Graphs associated to a specific BIMMS project, for a given project and user token

<https://bim4eeb.onetteam.it/BIMMSWS/project/getProjectGraphs?tokenUser={tokenUser}&tokenProject={tokenProject}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string

Response Parameters:

"NAME": URI of Graph
----------------------

Response Body Example:

```
[
  {
    "NAME": "http://bim4eeb.onetteam.it:8890/bim4eeb-test-resources#"
  },
  {
    "NAME": "http://bim4eeb.onetteam.it:8890/bim4eeb-test-linkeddata#"
  },
  {
    "NAME": "http://bim4eeb.onetteam.it:8890/bim4eeb-test-ifcdata#"
  },
  {
    "NAME": "http://bim4eeb.onetteam.it:8890/bim4eeb-test-iotdata#"
  },
  {
    "NAME": Http://bim4eeb.onetteam.it:8890/bim4eeb-TEST-bimplanner/subLevel1/sublevel23/#
  }
]
```





"GlobalId": IFC object GlobalID  
"parentGlobalId": GlobalID of parent object in hierarchy if any (# if no parent object exist),  
"Name": Name of ifc Object  
"Type": Type of ifc Object

Response Body Example:

```
[
  {
    "GlobalId": "2bd35e07-b3db-43d8-81f5-6b8a283c9e56",
    "parentGlobalId": "#",
    "Name": "0001",
    "Type": "IfcProject"
  },
  {
    "GlobalId": "2bd35e07-b3db-43d8-81f5-6b8a283c9e57",
    "parentGlobalId": "2bd35e07-b3db-43d8-81f5-6b8a283c9e54",
    "Name": "",
    "Type": "IfcBuilding"
  },
  {
    "GlobalId": "2bd35e07-b3db-43d8-81f5-6b8ad7c36203",
    "parentGlobalId": "2bd35e07-b3db-43d8-81f5-6b8a283c9e57",
    "Name": "0 - Piano Terra",
    "Type": "IfcBuildingStorey"
  },
  {
    "GlobalId": "2bd35e07-b3db-43d8-81f5-6b8a283c9e54",
    "parentGlobalId": "2bd35e07-b3db-43d8-81f5-6b8a283c9e56",
    "Name": "Default",
    "Type": "IfcSite"
  },
  {
    "GlobalId": "33ef1112-fcc1-4f9a-9cbc-b7f3c3fdd37b",
    "parentGlobalId": "2bd35e07-b3db-43d8-81f5-6b8ad7c36203",
    "Name": "1",
```

```
"Type": "IfcSpace"
},
{
  "GlobalId": "33ef1112-fcc1-4f9a-9cbc-b7f3c3fdd376",
  "parentGlobalId": "2bd35e07-b3db-43d8-81f5-6b8ad7c36203",
  "Name": "2",
  "Type": "IfcSpace"
},
{
  "GlobalId": "33ef1112-fcc1-4f9a-9cbc-b7f3c3fdd371",
  "parentGlobalId": "2bd35e07-b3db-43d8-81f5-6b8ad7c36203",
  "Name": "3",
  "Type": "IfcSpace"
},
{
  "GlobalId": "33ef1112-fcc1-4f9a-9cbc-b7f3c3fdd3ff",
  "parentGlobalId": "2bd35e07-b3db-43d8-81f5-6b8ad7c36203",
  "Name": "4",
  "Type": "IfcSpace"
},
{
  "GlobalId": "33ef1112-fcc1-4f9a-9cbc-b7f3c3fdd3f0",
  "parentGlobalId": "2bd35e07-b3db-43d8-81f5-6b8ad7c36203",
  "Name": "1",
  "Type": "IfcSpace"
},
{
  "GlobalId": "33ef1112-fcc1-4f9a-9cbc-b7f3c3fdd3f1",
  "parentGlobalId": "2bd35e07-b3db-43d8-81f5-6b8ad7c36203",
  "Name": "2",
  "Type": "IfcSpace"
},
{
  "GlobalId": "33ef1112-fcc1-4f9a-9cbc-b7f3c3fdd3f2",
  "parentGlobalId": "2bd35e07-b3db-43d8-81f5-6b8ad7c36203",
```

```

    "Name": "3",
    "Type": "IfcSpace"
  },
  {
    "GlobalId": "33ef1112-fcc1-4f9a-9cbc-b7f3c3fdd3f3",
    "parentGlobalId": "2bd35e07-b3db-43d8-81f5-6b8ad7c36203",
    "Name": "4",
    "Type": "IfcSpace"
  }
]

```

**3.1.3.3 GET ifc/getIFCspaceObj**

Get the IFC objects (equipments, furniture, etc.) in the locationID space for the given Internal Filename, project and user token

<https://bim4eeb.oneteam.it/BIMMSWS/ifc/getIFCspaceObj?tokenUser={tokenUser}&tokenProject={tokenProject}&fileName={fileName}&locationID={locationID}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user\getUserProjects method	Query	string
Filename	Unique filename retrieved by project/getProjectIFC method	Query	string
locationID	Location (space\zone) GlobalID as retrieved by ifc/getIFCHierarchy and ifc/getIFCZones methods	Query	string

Respose Parameters:

```

"GlobalId": ID of contained Object
"parentGlobalId": paren locationID,
"Name": Name of object,
"Type": Object Type

```

## Response Body Example:

```
[
  {
    "GlobalId": "b90bb6a4-eba6-4add-8779-5b9dbc95108d",
    "parentGlobalId": "f3de0c5d-e83c-48e7-880f-359673af2df0",
    "Name": "WC:1:2104873",
    "Type": "IfcFlowTerminal"
  },
  {
    "GlobalId": "b90bb6a4-eba6-4add-8779-5b9dbc95244a",
    "parentGlobalId": "f3de0c5d-e83c-48e7-880f-359673af2df0",
    "Name": "Bath:2:2108142",
    "Type": "IfcFlowTerminal"
  },
  {
    "GlobalId": "b90bb6a4-eba6-4add-8779-5b9dbc9539b6",
    "parentGlobalId": "f3de0c5d-e83c-48e7-880f-359673af2df0",
    "Name": "Sink:1:2111250",
    "Type": "IfcFlowTerminal"
  },
  {
    "GlobalId": "b90bb6a4-eba6-4add-8779-5b9dbc955651",
    "parentGlobalId": "f3de0c5d-e83c-48e7-880f-359673af2df0",
    "Name": "Gas Boiler:1:2119925",
    "Type": "IfcBuildingElementProxy"
  },
  {
    "GlobalId": "e7b97a90-2bd2-4b98-8c24-022d1b775fc8",
    "parentGlobalId": "f3de0c5d-e83c-48e7-880f-359673af2df0",
    "Name": "Heater:1:2194196",
    "Type": "IfcBuildingElementProxy"
  },
  {
    "GlobalId": "b90bb6a4-eba6-4add-8779-5b9dbc956c4d",
```

```

    "parentGlobalId": "f3de0c5d-e83c-48e7-880f-359673af2c3a",
    "Name": "Heater:3:2122473",
    "Type": "IfcBuildingElementProxy"
  },
  {
    "GlobalId": "b90bb6a4-eba6-4add-8779-5b9dbc956ab0",
    "parentGlobalId": "f3de0c5d-e83c-48e7-880f-359673af2c3a",
    "Name": "Heater:3:2122772",
    "Type": "IfcBuildingElementProxy"
  }
]

```

### 3.1.3.4 GET /ifc/getIFCZones

Get the IFC zones for the given Internal Filename, project and user token. ifcZones and ifSpaces included are returned.

<https://bim4eeb.oneteam.it/BIMMSWS/ifc/getIFCZones?tokenUser={tokenUser}&tokenProject={tokenProject}&fileName={fileName}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user\getUserProjects method	Query	string
filename	Unique filename retrieved by project/getProjectIFC method	Query	string

Response Parameters:

```

"GlobalId": IFC object GlobalID
"parentGlobalId": GlobalID of parent object in hierarchy if any (# if no parent object exist),
"Name": Name of ifc Object
"Type": Type of ifc Object

```

## Response Body Example:

```
[
  {
    "GlobalId": "b65fe2f8-9a11-4f36-88ae-d5d6640ca069",
    "parentGlobalId": "#",
    "Name": "XXXX",
    "Type": "IfcProject"
  },
  {
    "GlobalId": "1aedfb5a-89ed-11ea-b922-005056b48219",
    "parentGlobalId": "b65fe2f8-9a11-4f36-88ae-d5d6640ca069",
    "Name": "OTZoneObj",
    "Type": "IfcZone"
  },
  {
    "GlobalId": "f3de0c5d-e83c-48e7-880f-359673af2df0",
    "parentGlobalId": "1aedfb5a-89ed-11ea-b922-005056b48219",
    "Name": "D1/2.3",
    "Type": "IfcSpace"
  },
  {
    "GlobalId": "f3de0c5d-e83c-48e7-880f-359673af2c3a",
    "parentGlobalId": "1aedfb5a-89ed-11ea-b922-005056b48219",
    "Name": "D1/2.2",
    "Type": "IfcSpace"
  }
]
```

### 3.1.3.5 POST /ifc/setIFCZone

Create new IFC zone for the given file name, project and user token.  
The request Body includes:

zoneName: Name of the new zone being created,  
zoneDescription : Description of the new zone being created,  
zoneChildGlobalIds : List of child entities (IfcSpace\IfcZone) GlobalId, separated by comma, retrieved by ifc/getIFCHierarchy and ifc/getIFCZones methods

<https://bim4eeb.oneteam.it/BIMMSWS/ifc/setIFCZone?tokenUser={tokenUser}&tokenProject={tokenProject}&fileName={fileName}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
filename	Unique filename retrieved by project/getProjectIFC method	Query	string
zoneData	zoneName : Name of the new zone being created ,zoneDescription : Description of the new zone being created ,zoneChildGlobalIds : List of child entities (IfcSpace\IfcZone) GlobalId, separated by comma, retrieved by ifc/getIFCHierarchy and ifc/getIFCZones methods	body	{ }

Request body example

```
{
"zoneName": "B1",
"zoneDescription": "DemoZDescB",
"zoneChildGlobalIds": "33ef1112-fcc1-4f9a-9cbc-b7f3c3fdd3f0,33ef1112-fcc1-4f9a-9cbc-b7f3c3fdd3f1"
}
```

Response Parameters:

No response parameter

Response Body Example:

```
1 : Zone created
0 : Zone not created
```



### 3.1.3.6 PUT /ifc/modIFCZone

Modify IFC zone for the given file name, project and user token, zoneID;  
body request includes:

zoneID : GlobalId of the zone being modified ,  
zoneDescription : Description of the new zone being created ,  
zoneChildGlobalIds : List of child entities (IfcSpace\IfcZone) GlobalId, separated by comma, retrieved by ifc/getIFCHierarchy and ifc/getIFCZones methods.

Note that all previous childs will be automatically detached

<https://bim4eeb.oneteam.it/BIMMSWS/ifc/modIFCZone?tokenUser={tokenUser}&tokenProject={tokenProject}&fileName={fileName}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
filename	Unique filename retrieved by project/getProjectIFC method	Query	string
zoneData	zoneName : Name of the zone being modified ,zoneChildGlobalIds : List of child entities (IfcSpace\IfcZone) GlobalId, separated by comma, retrieved by ifc/getIFCHierarchy and ifc/getIFCZones methods. Note that all previous childs will be automatically detached	body	{ }

Request body example

```
{
"zoneID": "8a2013d3-b464-11ea-b774-00505695f911",
"zoneDescription": "DemoZDescB",
"zoneChildGlobalIds": "33ef1112-fcc1-4f9a-9cbc-b7f3c3fdd3f0,33ef1112-fcc1-4f9a-9cbc-b7f3c3fdd3f1"
}
```

Response Parameters:

No response parameter

Response Body Example:

1 : Zone modified  
0 : Zone not modified

### 3.1.3.7 DELETE /ifc/delIFCZone

Delete IFC zone for the given file name, project and user token, specifying zoneID to be removed;

<https://bim4eeb.oneteam.it/BIMMSWS/ifc/delIFCZone?tokenUser={tokenUser}&tokenProject={tokenProject}&fileName={fileName}&zoneID={zoneID}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
filename	Unique filename retrieved by project/getProjectIFC method	Query	string
zoneID	GlobalID of the zone being deleted	Query	string

Respose Parameters:

No response parameter

Response Body Example:

1 : Zone deleted  
0 : Zone not deleted

### 3.1.4 RESOURCE API

#### 3.1.4.1 POST /resource/setIFCFile

Import the IFC file for the given project and user token creating resource with given title and description. After the import, a batch and asynchronous process will create:

- SQL database tdata from IFC and Virtuoso Data View
- ifcOWL file
- BOT file

Request body contains parameter filestr equal to Base64 encoded IFC file

<https://bim4eeb.onetteam.it/BIMMSWS/resource/setIFCFile?tokenUser={tokenUser}&tokenProject={tokenProject}&title={title}&description={description}&building={building}&stage={stage}&fileName={fileName}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
title	Title for the new resource	Query	String
description	Description of the new resource	Query	String
building	Building name as retrieved by project/getProjectBuildings method	Query	String
stage	Stage as retrieved by user/getUserStages method	Query	String
fileName	Filename of the file being uploaded	Query	String
filestr	Base64 string of the file content	Body	{ }

Request body example

```
{"filestr": "WCx5LHoNCjEsMiwzDQo0LDUsNg0KNyw4LDkNCg=="}
```

Response Parameters:

```
No response parameter
```

Response Body Example:

```
1 : IFC resource imported
0 : IFC resource not imported
```

### 3.1.4.2 POST /resource/setPCSFile

Import the Point Cloud Scan file for the given project and user token creating resource with given title and description

<https://bim4eeb.oneteam.it/BIMMSWS/resource/setPCSFile?tokenUser={tokenUser}&tokenProject={tokenProject}&title={title}&description={description}&building={building}&floor={floor}&room={room}&stage={stage}&fileName={fileName}>

Request body contains parameter filestr equal to Base64 encoded PSC file

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
title	Title for the new resource	Query	String
description	Description of the new resource	Query	String
building	Building name as retrieved by project/getProjectBuildings method	Query	String
floor	Floor name as retrieved by ifc/getIFCHierarchy method	Query	String
room	Room name as retrieved by ifc/getIFCHierarchy method	Query	String
stage	Stage as retrieved by user/getUserStages method	Query	String
fileName	Filename of the file being uploaded	Query	String
filestr	Base64 string of the file content	Body	{ }

Request body example

```
{"filestr": "WCx5LHoNCjEsMiwzDQo0LDUsNg0KNyw4LDkNCg=="}
```

Response Parameters:

```
No response parameter
```

Response Body Example:

```
1 : PSC resource imported
0 : PSC resource not imported
```

### 3.1.4.3 GET /resource/getProjectResources

Get the BIMMS resources list for the given project and user token.

<https://bim4eeb.oneteam.it/BIMMSWS/resource/getProjectResources?tokenUser={tokenUser}&tokenProject={tokenProject}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string

Response Parameters:

<p>"Creation User": User that created the resource into BIMMS</p> <p>"Creation Date": Date of creation</p> <p>"Note": Notes</p> <p>"Title": Resource Title</p> <p>"Description": Resource Description</p> <p>"Keywords": Keywords</p> <p>"Synonyms": Synonyms</p> <p>"Combo": NOT USED</p>
--

Response Body Example:

<pre>[   {     "Creation User": "Alessandro Valra",     "Creation Date": "2020-02-11T13:29:04",     "Note": null,     "Title": "Test Dresden Object",     "Description": "My Dresden Test Object",     "Keywords": null,     "Synonyms": null,     "Combo": null   },   {     "Creation User": "Jacopo Chiappetti",     "Creation Date": "2020-02-11T16:50:22",</pre>
---

```
"Note": null,  
"Title": "test for Dresden",  
"Description": "",  
"Keywords": null,  
"Synonyms": null,  
"Combo": null  
},  
{  
"Creation User": "Alessandro Valra",  
"Creation Date": "2020-02-12T09:06:48",  
"Note": null,  
"Title": "WP5FastMappingIfc",  
"Description": "",  
"Keywords": "",  
"Synonyms": "",  
"Combo": null  
}  
]
```

### 3.1.5 SENSOR API

#### 3.1.5.1 POST /sensor/setProjectSensors

Import CSV file of sensors data for the given project and user token.

If Location name match with any IFC spece (ifcZone or ifcSpace) of Project, sensor will be linked with corresponding Room.

The Body request includes a Base64 csv file with this structure (without header row):

**Sensor unique code;Sensor Location Name; Sensor Type**

SensorCode04;Data Sensing Lab;Temperature  
 SensorCode05;Registration desk;Temperature  
 SensorCode06;Back corridor;Temperature

If Sensor unique code already exist in BIMMS Database, corresponding Sensor data will be updated

<https://bim4eeb.oneteam.it/BIMMSWS/sensor/setProjectSensors?tokenUser={tokenUser}&tokenProject={tokenProject}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string

Request body example

```
{
  "filestr": "U2Vuc29yQ29kZTA0O0RhdGEgU2Vuc2luZyBMYWI7VGVTcGVyYXR1cmUNCINlbnNvckNvZGUwN
  TtSZWdpc3RyYXRpb24gZGVzaztUZW1wZXJhdHVyZQ0KU2Vuc29yQ29kZTA2O0JhY2sgY29ycmlkb3I7VG
  VtcGVyYXR1cmU="
}
```

Respose Parameters:

No response parameter

Response Body Example:

1 : CSV processed  
 0 : CSV not processed

### 3.1.5.2 POST /sensor/setProjectSensor

Create a new sensor for a given project and user token. It updates data if sensor Code exists.

If Location name match with any IFC spece (ifcZone or ifcSpace) of Project, sensor will be linked with corresponding Room.

<https://bim4eeb.onetteam.it/BIMMSWS/sensor/setProjectSensor?tokenUser={tokenUser}&tokenProject={tokenProject}&code={code}&location={location}&type={type}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	String
code	Code (name, SN, GUID) of the sensor	Query	String
location	Location (room) name as retrieved by ifc/getIFCHierarchy method	Query	String
type	Sensor type (ex. Temperature)	Query	String

Respose Parameters:

No response parameter
-----------------------

Response Body Example:

1 : Sensor created 0 : No sensor created
---

### 3.1.5.3 POST /sensor/delProjectSensors

Delete sensors for the given CSV file, project and user token. The Body request includes a Base64 csv file with this structure:

**Sensor unique code;Sensor Location Name; Sensor Type**

SensorCode04;Data Sensing Lab;Temperature  
 SensorCode05;Registration desk;Temperature  
 SensorCode06;Back corridor;Temperature

<https://bim4eeb.onetteam.it/BIMMSWS/sensor/delProjectSensors?tokenUser={tokenUser}&tokenProject={tokenProject}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string



Request body example

```
{
"filestr": "U2Vuc29yQ29kZTA0O0RhdGEgU2Vuc2luZyBMYWI7VGVtcGVyYXR1cmUNCINlbnNvckNvZGUwN
TtSZWdpc3RyYXRpb24gZGVzaztUZW1wZXJhdHVyZQ0KU2Vuc29yQ29kZTA2O0JhY2sgY29ycmlkb3I7VG
VtcGVyYXR1cmU="
}
```

Response Parameters:

No response parameter

Response Body Example:

1 : CSV processed  
0 : CSV not processed

**3.1.5.4 POST /sensor/delProjectSensor**

Delete a sensor for the given sensor code, project and user token.

<https://bim4eeb.onetteam.it/BIMMSWS/sensor/delProjectSensor?tokenUser={tokenUser}&tokenProject={tokenProject}&code={code}&location={location}&type={type}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
code	Code (name, SN, GUID) of the sensor	Query	String
location	Location (room) name as retrieved by ifc/getIFCHierarchy method	Query	String
type	Sensor type (ex. Temperature)	Query	String

Response Parameters:

No response parameter

Response Body Example:

1 : Sensor Deleted  
0 : No sensor deleted

### 3.1.5.5 POST /sensor/setSensorsMeasures

Import CSV file of sensors measurements data for the given project and user token. The Body request includes a Base64 csv file with this structure:

Code - sensor code

Date - date of measurement

Value - value of measurement

Type - measurement type (Type may be one of these values:

AirQuality;Contact;Energy;Humidity;Illuminance;Light;Motion;Occupancy;Power;Switch;Temperature;Temperature\_F;UV)

Example

402c1384;23/10/2012 19:56;20.8;Temperature

402c1384;23/10/2012 20:18;20.9;Temperature

402c1384;23/10/2012 20:24;20.6;Temperature

<https://bim4eeb.oneteam.it/BIMMSWS/sensor/setSensorsMeasures?tokenUser={tokenUser}&tokenProject={tokenProject}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string

Request body example

```
{"filestr":"U2Vuc29yQ29kZTAxOzIzLzEwLzlwMTIgMTk6NTY7MjAuODtUZW1wZXJhdHVyZQ0KU2Vuc29yQ29kZTAxOzIzLzEwLzlwMTIgMjA6MTg7MjAuOTtUZW1wZXJhdHVyZQ0KU2Vuc29yQ29kZTAxOzIzLzEwLzlwMTIgMjA6MjQ7MjAuNjtUZW1wZXJhdHVyZQ=="}
```

Response Parameters:

No response parameter

Response Body Example:

```
1 : CSV processed
0 : CSV not processed
```

### 3.1.5.6 POST /sensor/setSensorMeasure

Add sensor measurement data for the given sensor code, project and user token (Type may be: AirQuality;Contact;Energy;Humidity;Illuminance;Light;Motion;Occupancy;Power;Switch;Temperature;Temperature\_F;UV)

<https://bim4eeb.oneteam.it/BIMMSWS/sensor/setSensorMeasure?tokenUser={tokenUser}&tokenProject={tokenProject}&code={code}&vdate={vdate}&value={value}&type={type}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	String
code	Code (name, SN, GUID) of the sensor	Query	String
vdate	Date of measurement	Query	String
value	Value of the measurement	Query	String
type	Type of measurement	Query	String

Response Parameters:

No response parameter

Response Body Example:

1 : Measurement created  
0 : NO Measurement created

### 3.1.5.7 GET /sensor/getSensorsFromLocation

Get the sensors for the given location (ifczone or ifcspace globalID), project and user token. GlobalID are returned from getIFCHierarchy and getIFCZones methods.

<https://bim4eeb.oneteam.it/BIMMSWS/sensor/getSensorsFromLocation?tokenUser={tokenUser}&tokenProject={tokenProject}&locationID={locationID}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	String
locationID	Location GlobalID (space\zone) id as retrieved by ifc/getIFCHierarchy and ifc/getIFCZones methods	Query	String

Response Parameters:

"code": Sensor unique code
"type": Sensor Type

Response Body Example:

```
[
  {
    "code": "d7cea0cc-2ac6-4986-9982-668936744baa",
    "type": "SmartWeather Station Tile"
  },
  {
    "code": "e5f58da9-6e92-44b2-961a-c683dbd13c68",
    "type": "PurpleAir Air Quality Station"
  },
  {
    "code": "SensorCode07",
    "type": "Temperature"
  }
]
```

### 3.1.5.8 GET /sensor/getSensorData

Get the measurements for the given sensor code, project and user token. Filter on detes is Optional.

<https://bim4eeb.oneteam.it/BIMMSWS/sensor/getSensorData?tokenUser={tokenUser}&tokenProject={tokenProject}&code={code}&fromDate={fromDate}&toDate={toDate}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user\getUserProjects method	Query	string
code	Code (name, SN, GUID) of the sensor	Query	string
fromDate	Optional date from limit in yyyy-mm-dd hh-mm format	Query	string
toDate	Optional date to limit in yyyy-mm-dd hh-mm format	Query	string

Response Parameters:

```
"date": Datetime of measurement  
"value": Measurement value  
"UM": Unit of measurement  
"type": Type of measurement
```

Response Body Example:

```
[  
  {  
    "date": "2020-06-23T13:10:10",  
    "value": 30,  
    "UM": "C",  
    "type": "Temperature"  
  },  
  {  
    "date": "2012-10-23T20:24:00",  
    "value": 20.6,  
    "UM": "C",  
    "type": "Temperature"  
  },  
  {  
    "date": "2012-10-23T20:18:00",  
    "value": 20.9,  
    "UM": "C",  
    "type": "Temperature"  
  },  
  {  
    "date": "2012-10-23T19:56:00",  
    "value": 20.8,  
    "UM": "C",  
    "type": "Temperature"  
  }  
]
```

### 3.1.6 ENDUSER API

#### 3.1.6.1 GET enduser/getEndusersZones

Get end users connected to IFC zones for the given file name, project and user token. End User are identified by APPTOKEN inside the response data.

<https://bim4eeb.oneteam.it/BIMMSWS/enduser/getEndusersZones?tokenUser={tokenUser}&tokenProject={tokenProject}&fileName={fileName}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
fileName	Unique fileName retrieved by project/getProjectIFC method	Query	string

Response Parameters:

<p>"APPTOKEN": enduser application token identifier</p> <p>"locationID": ifcZone GlobalID</p> <p>"Name": zone name</p>
--

Response Body Example:

<pre>[   {     "APPTOKEN": "jc1234!",     "locationID": "d758e715-9115-11ea-b922-005056b48219",     "Name": "Living room"   },   {     "APPTOKEN": "jc1234!",     "locationID": "d29edc5b-9115-11ea-b922-005056b48219",     "Name": "Outside"   } ]</pre>
---

### 3.1.6.2 POST enduser/setEnduserZone

Connect end user to IFC zone for the given file name, project and user token. end user is identified by the parameter "enduserToken".

<https://bim4eeb.onetteam.it/BIMMSWS/enduser/setEnduserZone?tokenUser={tokenUser}&tokenProject={tokenProject}&fileName={fileName}&enduserToken={enduserToken}&zoneGUID={zoneGUID}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
fileName	Unique fileName retrieved by project/getProjectIFC method	Query	string
enduserToken	End user Token comes from application owner	Query	string
zoneGUID	Zone GlobalID retrieved by ifc/getIFCZones method	Query	string

Respose Parameters:

No response parameter

Response Body Example:

1 : Association succeeded  
0 : Association Failed

### 3.1.6.3 DELETE enduser/delEnduserZone

Disconnect end user from IFC zone for the given file name, project and user token. End user is identified by the EnduserToken parameter.

<https://bim4eeb.onetteam.it/BIMMSWS/enduser/delEnduserZone?tokenUser={tokenUser}&tokenProject={tokenProject}&fileName={fileName}&enduserToken={enduserToken}&zoneGUID={zoneGUID}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
fileName	Unique fileName retrieved by project/getProjectIFC method	Query	string
enduserToken	End user Token comes from application owner	Query	string

zoneGUID	Zone GlobalId retrieved by ifc/getIFCZones method	Query	string
----------	---	-------	--------

Response Parameters:

No response parameter

Response Body Example:

1 : Disassociation succeeded  
0 : Disassociation Failed

### 3.1.6.4 GET enduser/getEnduserProjects

Get the projects for the given enduser token (only those accessible also by user identified by the tokenUser parameter) and roles (inhabitant\owner)

<https://bim4eeb.oneteam.it/BIMMSWS/enduser/getEnduserProjects?tokenUser={tokenUser}&enduserToken={enduserToken}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
enduserToken	End user Token comes from application owner	Query	string

Response Parameters:

"TOKEN": Project identification Token  
 "NAME": Project Name without spaces  
 "COMPANY": Project company  
 "CITY": Project City  
 "STATUS": 1 - Active  
 "INHABITANT": 0 enduser has not INHABITANT role: 1 enduser has INHABITANT role  
 "OWNER": 0 enduser has not OWNER role: 1 enduser has OWNER role  
 "ORIGINAL\_NAME": Original Project name



Response Body Example:

```
[
  {
    "TOKEN": "4d9b742d-102e-11ea-9c7f-005056920005",
    "NAME": "Finnish_demonstration_site",
    "COMPANY": "One Team srl",
    "CITY": "Tampere",
    "STATUS": 1,
    "INHABITANT": 1,
    "OWNER": 0,
    "ORIGINAL_NAME": "Finnish demonstration site"
  },
  {
    "TOKEN": "4d9b742b-102e-11ea-9c7f-005056920005",
    "NAME": "Italian_demonstration_site",
    "COMPANY": "One Team srl",
    "CITY": "Monza",
    "STATUS": 1,
    "INHABITANT": 0,
    "OWNER": 1,
    "ORIGINAL_NAME": "Italian demonstration site"
  }
]
```

### 3.1.7 OCCUPANCY API

#### 3.1.7.1 GET occupancy/getOccupancyFromLocation

Get the occupancy data for the given location, project and user token.

<https://bim4eeb.oneteam.it/BIMMSWS/occupancy/getOccupancyFromLocation?tokenUser={tokenUser}&tokenProject={tokenProject}&locationID={locationID}&fromDate={fromDate}&toDate={toDate}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
locationID	Location (space\zone) id as retrieved by ifc/getIFCHierarchy and ifc/getIFCZones methods	Query	string
fromDate	Optional date from limit in yyyy-mm-dd hh-mm format	Query	string
toDate	Optional date to limit in yyyy-mm-dd hh-mm format	Query	string

Response Parameters:

<p>"occupancyID": Occupancy unique ID          "zoneMaxOccupancy": Max occupancy          "scheduleID": Schedule unique ID          "scheduleName": Schedule name          "annotationEntity": Annotation          "scheduleDate": Schedule DateTime          "dataID": Schedule data unique ID          "date": Date time          "value": Occupancy Value</p> <p>Response Body Example:</p>
--

<pre>[   {     "occupancyID": 6,     "zoneMaxOccupancy": 5,</pre>
---

```

    "scheduleID": 11,
    "scheduleName": "Test Schedule VAL",
    "annotationEntity": "Annotation Schedule VAL",
    "scheduleDate": "2020-06-23T15:05:23",
    "dataID": 23,
    "date": "2020-06-23T16:12:43",
    "value": 3
  }
]

```

### 3.1.7.2 POST occupancy/setOccupancy

Create occupancy data for the given location, project and user token. OccupancyID is returned in Response body.

<https://bim4eeb.oneteam.it/BIMMSWS/occupancy/setOccupancy?tokenUser={tokenUser}&tokenProject={tokenProject}&locationID={locationID}&zoneMaxOccupancy={zoneMaxOccupancy}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
locationID	Location (space\zone) id as retrieved by ifc/getIFCHierarchy and ifc/getIFCZones methods	Query	string
zoneMaxOccupancy	Zone Max occupancy value	Query	string

Response Parameters:

No response parameter

Response Body Example:

> 0 New occupancy data unique ID is returned  
 0 : Occupancy data NOT created

### 3.1.7.3 POST occupancy/setOccupancySchedule

Create occupancy schedule for the given occupancy data, project and user token. New scheduleID is returned inside response body.

<https://bim4eeb.onetteam.it/BIMMSWS/occupancy/setOccupancySchedule?tokenUser={tokenUser}&tokenProject={tokenProject}&occupancyID={occupancyID}&scheduleName={scheduleName}&annotationEntity={annotationEntity}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
occupancyID	Occupancy id as retrieved by occupancy/getOccupancyFromLocation and occupancy/setOccupancy methods	Query	integer
scheduleName	Schedule name value	Query	string
annotationEntity	Annotation entity value	Query	string

Response Parameters:

No response parameter

Response Body Example:

> 0 New occupancy schedule data unique ID is returned  
 0 : Occupancy schedule data NOT created

### 3.1.7.4 POST occupancy/setOccupancyScheduleData

Create occupancy schedule data for the given schedule, project and user token. New schedule data ID is returned inside Response Body.

<https://bim4eeb.onetteam.it/BIMMSWS/occupancy/setOccupancyScheduleData?tokenUser={tokenUser}&tokenProject={tokenProject}&scheduleID={scheduleID}&sdate={sdate}&svalue={svalue}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken	query	string

	method		
tokenProject	tokenProject retrieved by user\getUserProjects method	Query	string
scheduleID	Schedule id as retrieved by occupancy/getOccupancyFromLocation and occupancy/setOccupancySchedule methods	Query	integer
sdate	Date in yyyy-mm-dd hh-mm format	Query	string
svalue	Value (double)	Query	integer

Response Parameters:

No response parameter

Response Body Example:

> 0 New occupancy schedule data value unique ID is returned  
 0 : Occupancy schedule data value NOT created

### 3.1.8 ACTIVITY API

#### 3.1.8.1 GET activity/getActivityFromLocation

Get the activity data for the given location, project and user token

<https://bim4eeb.oneteam.it/BIMMSWS/activity/getActivityFromLocation?tokenUser={tokenUser}&tokenProject={tokenProject}&locationID={locationID}&fromDate={fromDate}&toDate={toDate}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
locationID	Location (space\zone) id as retrieved by ifc/getIFCHierarchy and ifc/getIFCZones methods	Query	string
fromDate	Optional date from limit in yyyy-mm-dd hh-mm format	Query	string
toDate	Optional date to limit in yyyy-mm-dd hh-mm format	Query	string

Respose Parameters:

<p>"ID": Activity unique ID</p> <p>"location": Zone/Room name associated to Activity</p> <p>"name": Activity name</p> <p>"startdate": Activity start date</p> <p>"enddate": Activity end date</p> <p>"duration": Activity duration</p> <p>"createdate": Activity creation date</p>
--

Response Body Example:

```
[
  {
    "ID": 10,
    "location": "Outside",
    "name": "ActivityOne",
    "startdate": "2020-07-01T00:00:00",
    "enddate": "2020-07-10T23:00:00",
    "duration": 10,
    "createdate": "2020-06-23T00:00:00"
  }
]
```

### 3.1.8.2 POST activity/setActivity

Create activity data for the given location, project and user token. New Activity id is returned into Response Body.

Request body includes:

activityName: Name of the new activity being created

activityStartTime: activity start date in yyyy-mm-dd hh-mm format

activityEndTime: activity end date in yyyy-mm-dd hh-mm format

activityDuration: activity duration (double)

<https://bim4eeb.oneteam.it/BIMMSWS/activity/setActivity?tokenUser={tokenUser}&tokenProject={tokenProject}&locationID={locationID}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
locationID	Location (space\zone) id as retrieved by ifc/getIFCHierarchy and ifc/getIFCZones methods	Query	string
Activity data	activityName: Name of the new activity being created, activityStartTime: activity start date in yyyy-mm-dd hh-mm format ,activityEndTime : activity end date in yyyy-mm-dd hh-mm format ,activityDuration : activity duration (double)	Body	string

### Request body example

```
{  
  "activityName": "ActivityOne",  
  "activityStartTime": "2020-07-01 00:00",  
  "activityEndTime": "2020-07-10 23:00",  
  "activityDuration": 10.0  
}
```

### Response Parameters:

```
"activityName": New Activity name  
"activityStartTime": New Activity start Date  
"activityEndTime": New Activity end Date  
"activityDuration": New Activity duration
```

### Response Body Example:

```
{  
  "activityName": "ActivityOne",  
  "activityStartTime": "2020-07-01 00:00",  
  "activityEndTime": "2020-07-10 23:00",  
  "activityDuration": 10.0  
}
```



### 3.1.9 ALERT API

#### 3.1.9.1 GET alert/getAlertFromLocation

Get the alert data for the given location, project and user token

<https://bim4eeb.oneteam.it/BIMMSWS/alert/getAlertFromLocation?tokenUser={tokenUser}&tokenProject={tokenProject}&locationID={locationID}&fromDate={fromDate}&toDate={toDate}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
locationID	Location (space\zone) id as retrieved by ifc/getIFCHierarchy and ifc/getIFCZones methods	Query	string
fromDate	Optional date from limit in yyyy-mm-dd hh-mm format	Query	string
toDate	Optional date to limit in yyyy-mm-dd hh-mm format	Query	string

Response Parameters:

<p>"ID": Alert unique ID</p> <p>"location": Zone/Room name associated to Activity</p> <p>"type": Alert Type</p> <p>"topic": Alert Topic</p> <p>"subject": Alert Subject</p> <p>"description": Description of Alert</p> <p>"date": Alert Datetime</p> <p>"owner": Alert Owner</p> <p>"importance": level of importance</p>
---

Response Body Example:

<pre>[   {     "ID": 8,     "location": "Outside",     "type": "Alert Type",</pre>
--

```

"topic": "Alert Topic",
"subject": "Alert Subject",
"description": "Description of Alert",
"date": "2020-07-01T00:00:00",
"owner": "One Team",
"importance": "high"
}
]

```

### 3.1.9.2 POST alert/setAlert

Create alert data for the given location, project and user token. New Alert ID is returned into Response body.

Request body includes:

alertType: Type of alert being created

alertTopic: Alert topic

alertSubject: Alert subject

alertDescription: Alert description

alertTime: Alert date in yyyy-mm-dd hh-mm format

alertOwner: Alert owner

importance: Alert importance

<https://bim4eeb.oneteam.it/BIMMSWS/alert/setAlert?tokenUser={tokenUser}&tokenProject={tokenProject}&locationID={locationID}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
locationID	Location (space\zone) id as retrieved by ifc/getIFCHierarchy and ifc/getIFCZones methods	Query	string
Alert data	alertType: Type of alert being created, alertTopic: Alert topic, alertSubject: Alert subject, alertDescription: Alert description, alertTime: Alert date in yyyy-mm-dd hh-mm format, alertOwner: Alert owner, importance : Alert importance	Body	string

### Request body example

```
{  
"alertType" : "Alert Type",  
"alertTopic": "Alert Topic",  
"alertSubject" : "Alert Subject",  
"alertDescription" : "Description of Alert",  
"alertTime" : "2020-07-01 00:00",  
"alertOwner" : "One Team",  
"importance": "high"  
}
```

### Response Parameters:

```
"alertType" : New Alert Type  
"alertTopic": New Alert Topic  
"alertSubject" : New Alert Subject  
"alertDescription" : Description of New Alert  
"alertTime" : New Alert DateTime  
"alertOwner" : New Alert Owner  
"importance": New Alert level of importance
```

### Response Body Example:

```
{  
"alertType" : "Alert Type",  
"alertTopic": "Alert Topic",  
"alertSubject" : "Alert Subject",  
"alertDescription" : "Description of Alert",  
"alertTime" : "2020-07-01 00:00",  
"alertOwner" : "One Team",  
"importance": "high"  
}
```

## 3.2 OTSPARQL

### 3.2.1 POST sparql/getSPARQLData

Get SPARQL data from graph name and query, with limit and result type (Table, Raw) format.

Request body parameters includes:

"graphURI": default Graph URI

"query": SPARQL Query

"limit": result limits

"resultType": "R" raw; "T" table

<https://bim4eeb.oneteam.it/BIMMSWS/sparql/getSPARQLData?tokenUser={tokenUser}&tokenProject={tokenProject}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
SPARQL data	"graphURI": default Graph URI "query": SPARQL Query "limit": result limits "resultType": "R" raw; "T" table	Body	string

Request body example

```
{
  "graphURI": "http://bim4eeb.oneteam.it:8890/bim4eeb-test-iotdata#",
  "query": "SELECT * from <http://bim4eeb.oneteam.it:8890/bim4eeb-test-iotdata#> WHERE {?s ?p <https://w3id.org/saref#Measurement>}",
  "limit": "1",
  "resultType": "R"
}
```

Response Parameters:

No response Parameter, SPARQL query result is included in body response

Response Body Example:

```
"[{\"s\":{\"type\":\"uri\",\"value\":\"http://bim4eeb.oneteam.it:8890/bim4eeb-test-iotdata/data_measurements/ID/1#this\"},\"p\":{\"type\":\"uri\",\"value\":\"http://www.w3.org/1999/02/22-rdf-syntax-ns#type\"}}]"
```

### 3.3 OTVirtuoso

#### 3.3.1 POST api/OTVirtuoso

Insert triple form TripleList parameter into Virtuoso Graph defined by Graph parameter, for the given project and user token.

Request body parameters

"TokenUser": insert tokenUser retrieved by user/getUserToken method

"TokenProject": insert tokenProject retrieved by user/getUserProjects method

"Graph": insert destination graph example <http://bim4eeb.onetteam.it:8890/bim4eeb-test-linkeddata#>

"TripleList:" triple separated by dot

<https://bim4eeb.onetteam.it/BIMMSWS/api/OTVirtuoso>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	Body	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Body	string
Graph	destination graph	Body	string
TripleList	Triple List, triples separated by dot	Body	string

Request body example

```
{
  "TokenUser": "9400dcd7-f0f3-11e9-87b6-005056920005",
  "TokenProject": "4d9911cf-102e-11ea-9c7f-005056920005",
  "TripleList": "<#Lean> <#price> 5000 . <#Scope> <#price> 3500",
  "Graph": "http://stageBMS.onetteam.it#"
}
```

Respose Parameters:

No response Paramenter, SPARQL query result is included in body response

Response Body Example:

```
"<sparql xmlns=\\"http://www.w3.org/2005/sparql-results#" xmlns:xsi=\\"http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation=\\"http://www.w3.org/2001/sw/DataAccess/rf1/result2.xsd\">\n  <head>\n<variable name=\\"callret-0\"/>\n </head>\n <results distinct=\\"false\" ordered=\\"true\">\n <result>\n <binding name=\\"callret-0\"><literal>Insert into &lt;http://stageBMS.oneteam.it#\&gt;, 2 (or less) quads -- done</literal></binding>\n </result>\n </results>\n</sparql>"
```

### 3.3.2 DELETE api/OTVirtuoso

Delete triples in TripleList parameter from Virtuoso Graph defined in Graph parameter, for the given project and user token.

Request body parameters

"TokenUser": insert tokenUser retrieved by user/getUserToken method

"TokenProject": insert tokenProject retrieved by user/getUserProjects method

"Graph": insert destination graph example <http://bim4eeb.oneteam.it:8890/bim4eeb-test-linkeddata#>

"TripleList:" triple separated by dot

<https://bim4eeb.oneteam.it/BIMMSWS/api/OTVirtuoso>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	Body	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Body	string
Graph	destination graph	Body	string
TripleList	Triple List, triples separated by dot	Body	string

Request body example

```
{
  "TokenUser": "9400dcd7-f0f3-11e9-87b6-005056920005",
  "TokenProject": "4d9911cf-102e-11ea-9c7f-005056920005",
  "TripleList": "<#Lean> <#price> 5000 . <#Scope> <#price> 3500",
  "Graph": "http://stageBMS.oneteam.it#"
}
```

Response Parameters:

No response Parameter

Response Body Example:

Message 200: Delete succeeded

### 3.3.3 DELETE api/OTVirtuoso/DropSilentGraph

Removes the graph defined in Graph parameter from BIMMS. Only owner (creator) can delete graph, giving user token and project.

Example body

```
{
  "TokenUser": insert tokenUser retrieved by user/getUserToken method
  "TokenProject": insert tokenProject retrieved by user/getUserProjects method
  "Graph": insert graph to be removed from system
}
```

<https://bim4eeb.oneteam.it/BIMMSWS/api/OTVirtuoso/DropSilentGraph>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	Body	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Body	string
Graph	Graph URI to be removed from system	Body	string

Request body example

```
{
  "TokenUser": "9400dcd7-f0f3-11e9-87b6-005056920005",
  "TokenProject": "4d9911cf-102e-11ea-9c7f-005056920005",
  "Graph": "http://stageBMS.oneteam.it#"
}
```



Respose Parameters:

No response Paramenter

Response Body Example:

Message 200: Delete succeded

### 3.4 OTGeoLinkedData

#### 3.4.1 GET api/OTGeoLinkedData

Get GeoLinkedData information about BIM4EEB resource, using a given user and project. Other parameters are optional.

<https://bim4eeb.oneteam.it/BIMMSWS/api/OTGeoLinkedData?TokenUser={TokenUser}&TokenProject={TokenProject}&graph={graph}&UriEndPoint={UriEndPoint}&format={format}>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	query	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Query	string
UriEndPoint	optional BIM4EEB SPARQL endpoint	Query	string
Format	optional format (default application/sparql-results+json)	Query	string
graph	Optional Graph URI	Query	string

Response Parameters:

No response Parameter, SPARQL query result is included in body response

Response Body Example:

```
"\n { \"head\": { \"link\": [], \"vars\": [\"s\", \"p\", \"o\"] },\n  \"results\": { \"distinct\": false, \"ordered\": true,\n  \"bindings\": [\n    { \"s\": { \"type\": \"uri\", \"value\": \"http://bim4eeb.oneteam.it:8890/bim4eeb-test-resources/v_geopoint/ID/443fbf96-90ea-4e46-a769-4c9fb3ab1903#this\" },\n    \"p\": { \"type\": \"uri\", \"value\": \"http://www.w3.org/1999/02/22-rdf-syntax-ns#type\" },\n    \"o\": { \"type\": \"uri\", \"value\": \"http://www.w3.org/2003/01/geo/wgs84_pos#Point\" } },\n    { \"s\": { \"type\": \"uri\", \"value\": \"http://bim4eeb.oneteam.it:8890/bim4eeb-test-resources/v_geopoint/ID/443fbf96-90ea-4e46-a769-4c9fb3ab1903#this\" },\n    \"p\": { \"type\": \"uri\", \"value\": \"http://www.w3.org/2000/01/rdf-schema#label\" },\n    \"o\": { \"type\": \"literal\", \"value\": \"Mokba\" } },\n    { \"s\": { \"type\": \"uri\", \"value\": \"http://bim4eeb.oneteam.it:8890/bim4eeb-test-resources/v_geopoint/ID/443fbf96-90ea-4e46-a769-4c9fb3ab1903#this\" },\n    \"p\": { \"type\": \"uri\", \"value\": \"http://www.w3.org/2003/01/geo/wgs84_pos#lat_long\" },\n    \"o\": { \"type\": \"literal\", \"value\": \"37.77636718749999,55.724335519718835\" } },\n    { \"s\": { \"type\": \"uri\", \"value\": \"http://bim4eeb.oneteam.it:8890/bim4eeb-test-resources/v_geopoint/ID/443fbf96-90ea-4e46-a769-4c9fb3ab1903#this\" },\n    \"p\": { \"type\": \"uri\", \"value\": \"http://www.w3.org/2003/01/geo/wgs84_pos#lat\" },\n    \"o\": { \"type\": \"literal\", \"value\": \"37.77636718749999\" } },\n    { \"s\": { \"type\": \"uri\", \"value\": \"http://bim4eeb.oneteam.it:8890/bim4eeb-test-resources/v_geopoint/ID/443fbf96-90ea-4e46-a769-4c9fb3ab1903#this\" },\n    \"p\": { \"type\": \"uri\", \"value\": \"http://www.w3.org/2003/01/geo/wgs84_pos#long\" },\n    \"o\": { \"type\": \"literal\", \"value\": \"55.724335519718835\" } },\n    { \"s\": { \"type\": \"uri\", \"value\": \"http://bim4eeb.oneteam.it:8890/bim4eeb-test-resources/v_geopoint/ID/6f06350e-0fce-49e0-8642-0a74dd3b9f5b#this\" } } ] } }
```

### 3.5 OTRdf

#### 3.5.1 POST api/OTRdf

Upload RDF base64 file into Graph (Virtuoso Database)

Request Body includes:

"TokenUser": insert tokenUser retrieved by user/getUserToken method

"TokenProject": insert tokenProject retrieved by user/getUserProjects method

"Graph": insert destination graph example <http://bim4eeb.oneteam.it:8890/bim4eeb-test-linkeddata#>

"B64": RDF file to upload base 64

<https://bim4eeb.oneteam.it/BIMMSWS/api/OTRdf>

Parameters	Description	Parameter Type	Data Type
tokenUser	tokenUser retrieved by user/getUserToken method	Body	string
tokenProject	tokenProject retrieved by user/getUserProjects method	Body	string
Graph	destination graph	Body	string
B64	RDF file to upload base 64	Body	string

Request body example

```
{
  "TokenUser": "9400dcd7-f0f3-11e9-87b6-005056920005",
  "TokenProject": "4d9911cf-102e-11ea-9c7f-005056920005",
  "Graph": "http://stageBMS.oneteam.it#",
  "b64": "QHByZWZpeCBvd2w6ICAgPGh0dHA6Ly93d3cudzMub3JnLzlwMDIvMDcvb3dsIz4gLgpAcHJIZml4IHJkZjogICA8aHR0cDovL3d3dy53My5vcmcvMTk5OS8wMi8yMi1yZGYtc3ludGF4LW5zIz4gLgpAcHJIZml4IGlmY293bDogPGh0dHA6Ly9zdGFuZGFyZHMuYnVpbGRpbmdzbWFydC5vcmcvSUZDL0RFVi9JRkMyeDMvVEMxL09XTCM"
}
```

Response Parameters:

No response Parameter, SPARQL query result is included in body response

Response Body Example:

Message 200: Import succeeded

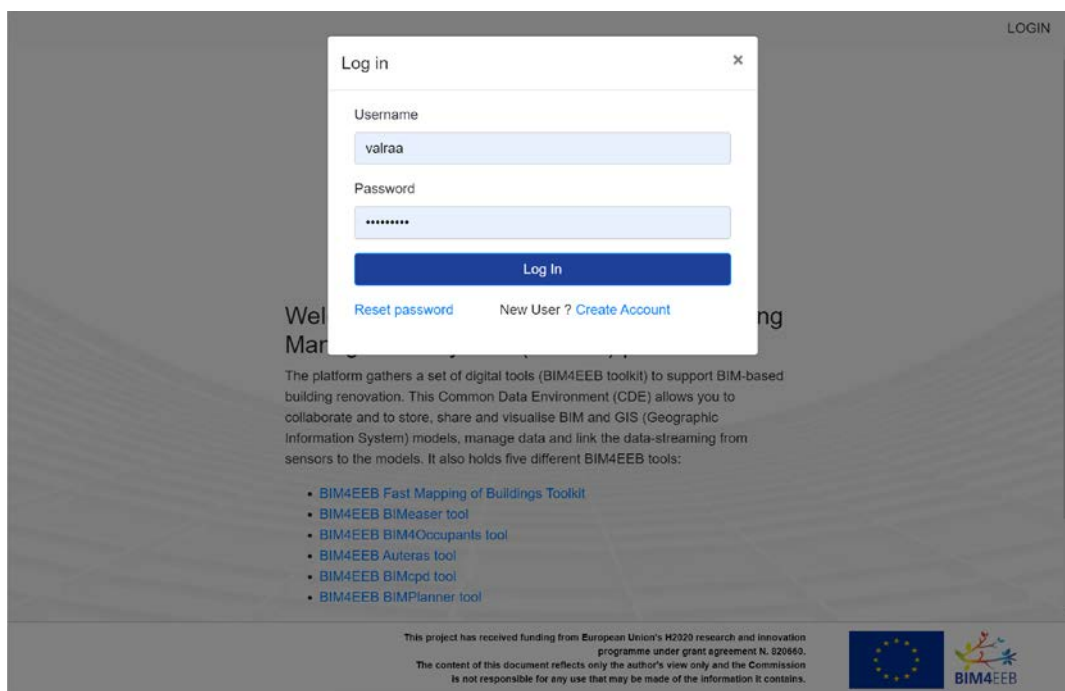
## 4 BIM Management System (BIMMS) User Manual

### 4.1 BIMMS User access

The BIMMS is accessible from a HTTPS web address in <https://bim4eeb.oneteam.it/BIMMS/>.

From this main page, the user can access to the platform using the Username and Password created during registration process. If it is the first-time access, the user can register to BIM4EEB BIMMS by creating a new account.

If a user has forgotten his/her password or username, it can be reset following the reset password link. An email will be sent to the user email with the links and the procedure to restore a password or the username used.



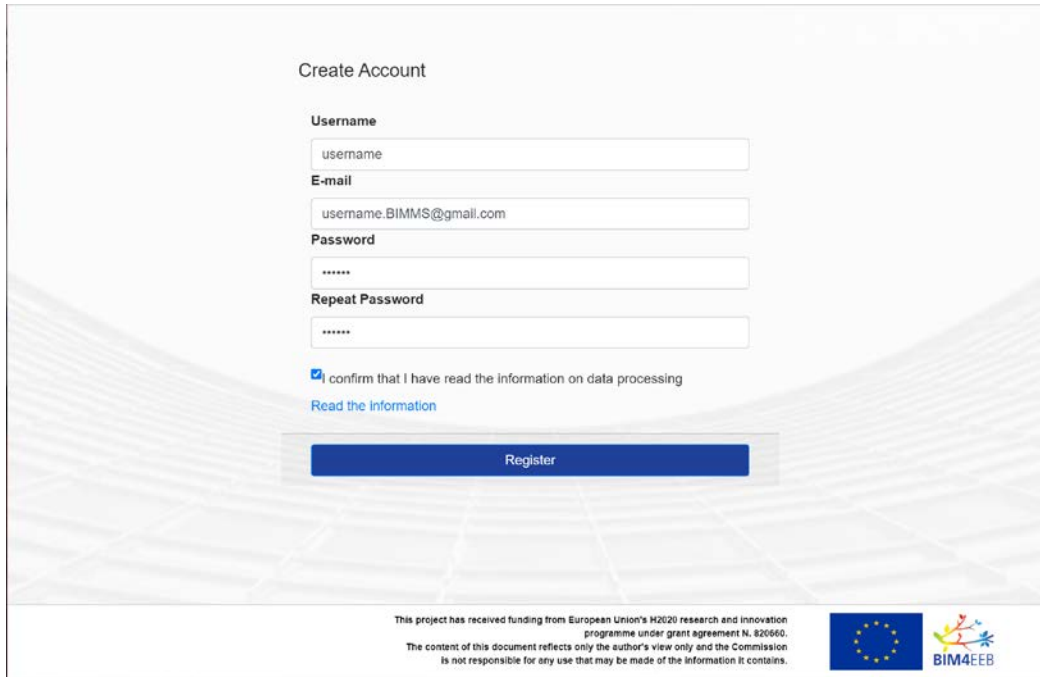
The screenshot shows a web browser window displaying the BIMMS login interface. A modal window titled "Log in" is open, featuring two input fields: "Username" with the text "valraa" and "Password" with masked characters. Below the fields is a blue "Log In" button. At the bottom of the modal, there are two links: "Reset password" and "New User ? Create Account". The background of the page is a blurred view of the BIMMS homepage, which includes a "LOGIN" button in the top right corner and a list of BIM4EEB tools at the bottom. The tools listed are: BIM4EEB Fast Mapping of Buildings Toolkit, BIM4EEB BiMeaser tool, BIM4EEB BIMOccupants tool, BIM4EEB Auteras tool, BIM4EEB BIMcpd tool, and BIM4EEB BIMPlanner tool. At the very bottom of the page, there is a small text block stating: "This project has received funding from European Union's Horizon research and innovation programme under grant agreement N. 820660. The content of this document reflects only the author's view only and the Commission is not responsible for any use that may be made of the information it contains." To the right of this text are the European Union flag and the BIM4EEB logo.

**Figure 3 - Login and access form**

A user can create a new account via registration form available at register page. The user must create a unique username and a password. The user can sign into the services by given her\his e-mail.

The user receives an e-mail with a link to confirm his/her registration.

When the sign in process is completed, the user can log-in with his/her username and password registered.

A screenshot of a web-based user registration form titled "Create Account". The form includes fields for "Username" (containing "username"), "E-mail" (containing "username.BIMMS@gmail.com"), "Password" (containing "\*\*\*\*\*"), and "Repeat Password" (containing "\*\*\*\*\*"). Below these fields is a checkbox labeled "I confirm that I have read the information on data processing" which is checked, and a link "Read the Information". A blue "Register" button is positioned below the form. At the bottom of the page, there is a disclaimer: "This project has received funding from European Union's H2020 research and innovation programme under grant agreement N. 820660. The content of this document reflects only the author's view only and the Commission is not responsible for any use that may be made of the information it contains." To the right of the disclaimer are the European Union flag and the BIM4EEB logo.

**Figure 4 – User registration form**

## 4.2 BIMMS User Registration workflow

The BIMMS user registration workflow is described in the next figures. User Roles and Accessibility matrix is in continuous development. This document reports the current development stage of BIMMS roles at M18, future implementation will be described in the next deliverables revisions.

BIMMS roles are grouped in macro-groups, as listed in Figure 5; in M18 release, accessibility to BIMSS functionalities are mapped in macro-roles.

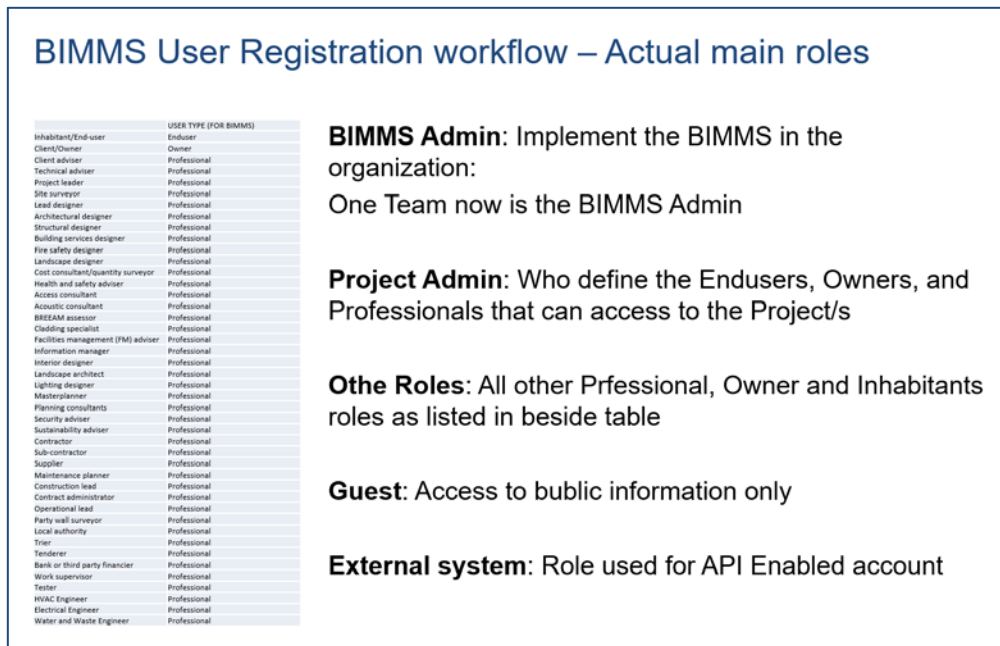


Figure 5 – BIMMS macro-roles

The user registration workflow is split in multiple steps, from the registration to the authorization and profiling. Figure 6 describes the registration steps:

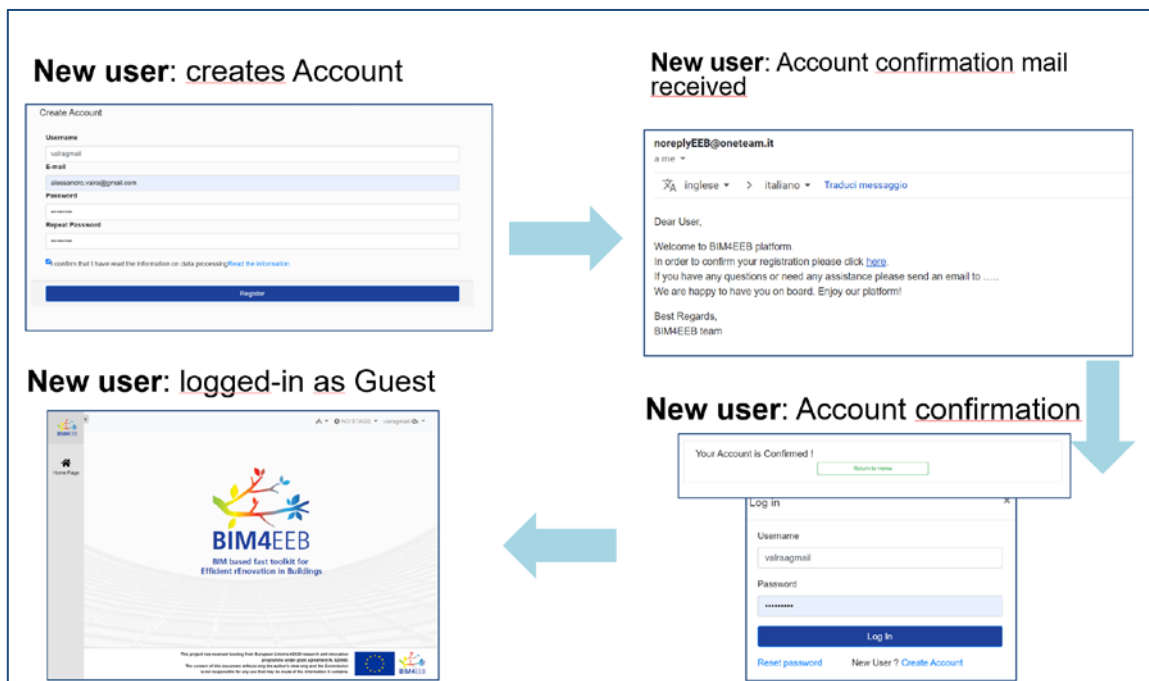


Figure 6 – BIMMS registration steps

The user registers her/his credentials into the system and confirms the information on data processing. After the registration, she/he receives an e-mail notification mandatory in order to confirm and activate the

new account. After the confirmation, the user can access BIMMS system using her/his credentials: new user access as Guest, only public information and basic functionalities are available.

After the registration, the user can request access specific BIMMS projects with specific Roles; this request is run by the “Request Projects” command in account menu

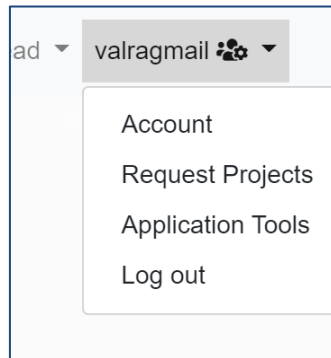


Figure 7 – Account menu

Project access authorization process is illustrated in figure 8 and starting from the Access request, the user asks for the access to a specific project and selects roles that she/he has to cover on the project.

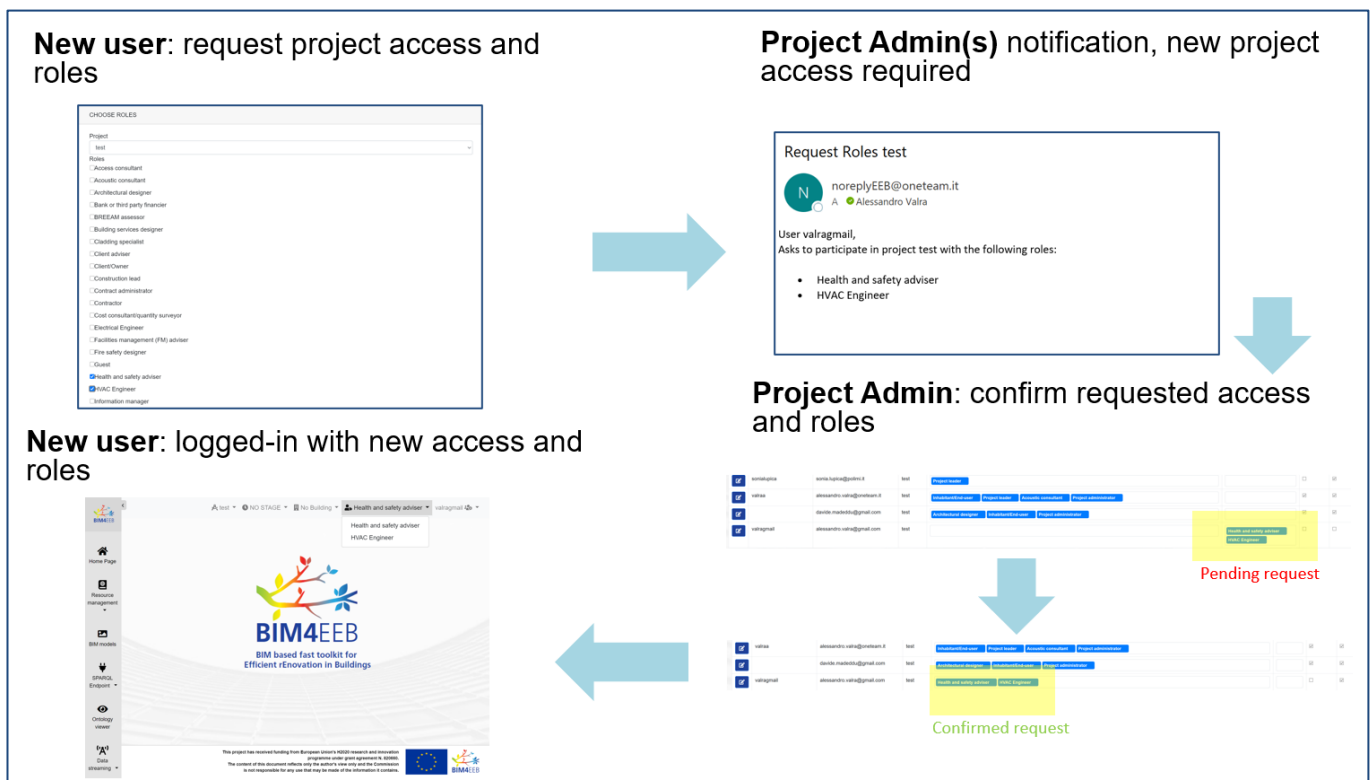
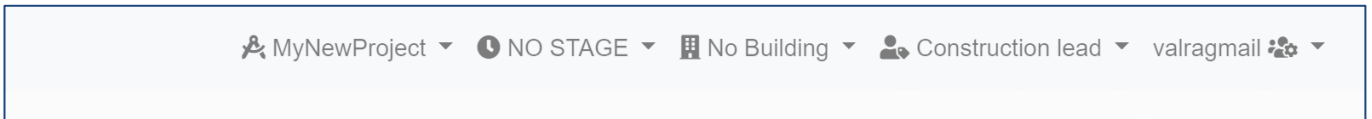


Figure 8 – Project authorization process

After the user request, an e-mail notification is sent to all Project Admins users on the requester Project.



Project Admins access the Project user management dashboard in order to confirm or deny access. When access is confirmed, the user will have access to Project with authorized roles and will see Project/Roles on top menu:



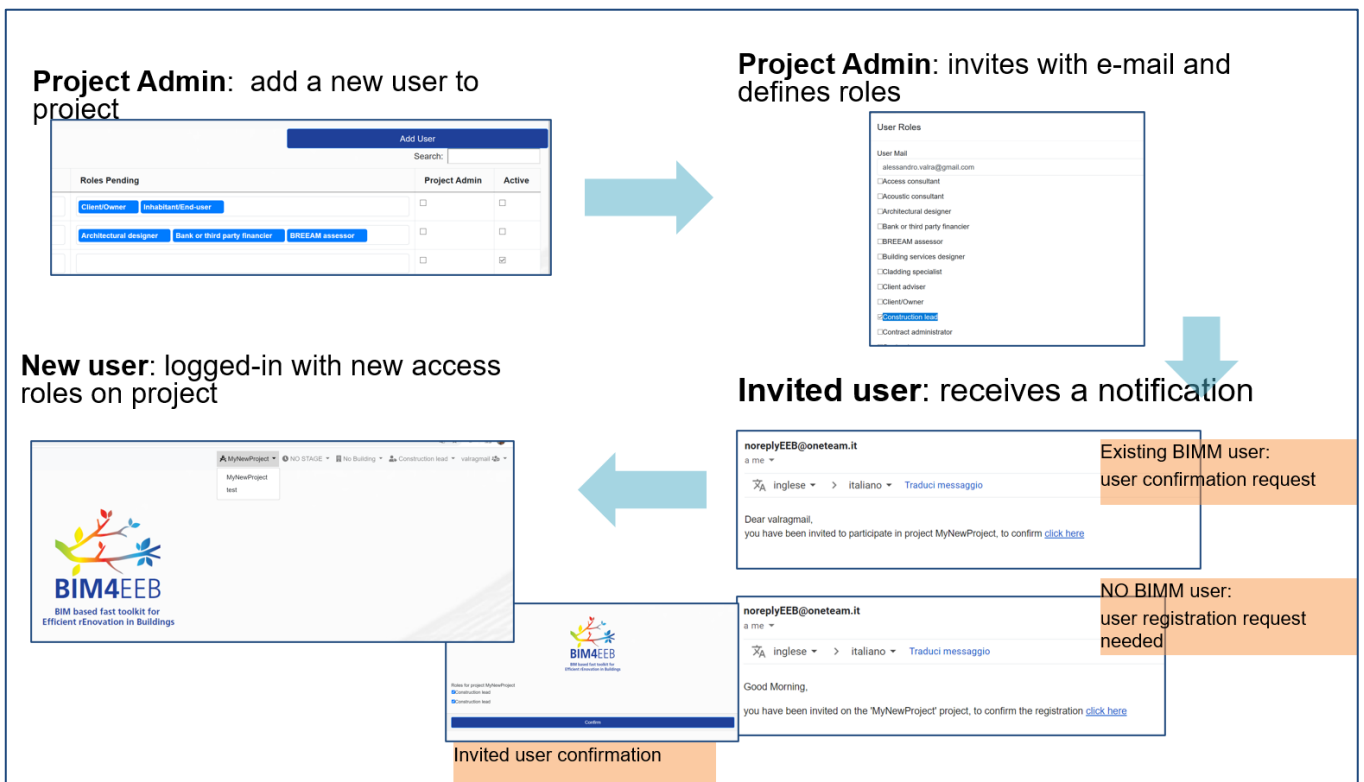
**Figure 9 – User Top menu with authorized Project and Roles**

BIMMS System includes a specific workflow starting from an invitation of a BIMMS Project Admin user. In this workflow Project Admin uses the Project User Management Dashboard to Add a new user to the project.

The Add functionality asks for the new user email and lets the Project Admin select the new user future roles.

After Project Admin Invitation, BIMMS system sends an e-mail to the new user; new user can already be a BIMMS registered user or not, depending on it:

- BIMMS registered user: the notification asks the new Project user to confirm invitation
- External user: the notification asks the new user to register into BIMMS system



**Figure 10 – New user Invitation workflow**

### 4.3 BIMMS Home page

The BIMMS Home page is composed by a welcome text that describes the BIM Management System and its role in the BIM4EEB project. The home page could contain weblinks to the BIM4EEB official website, learning videos and tutorials, and may host any news useful for the user activities in the BIM Management Web portal.

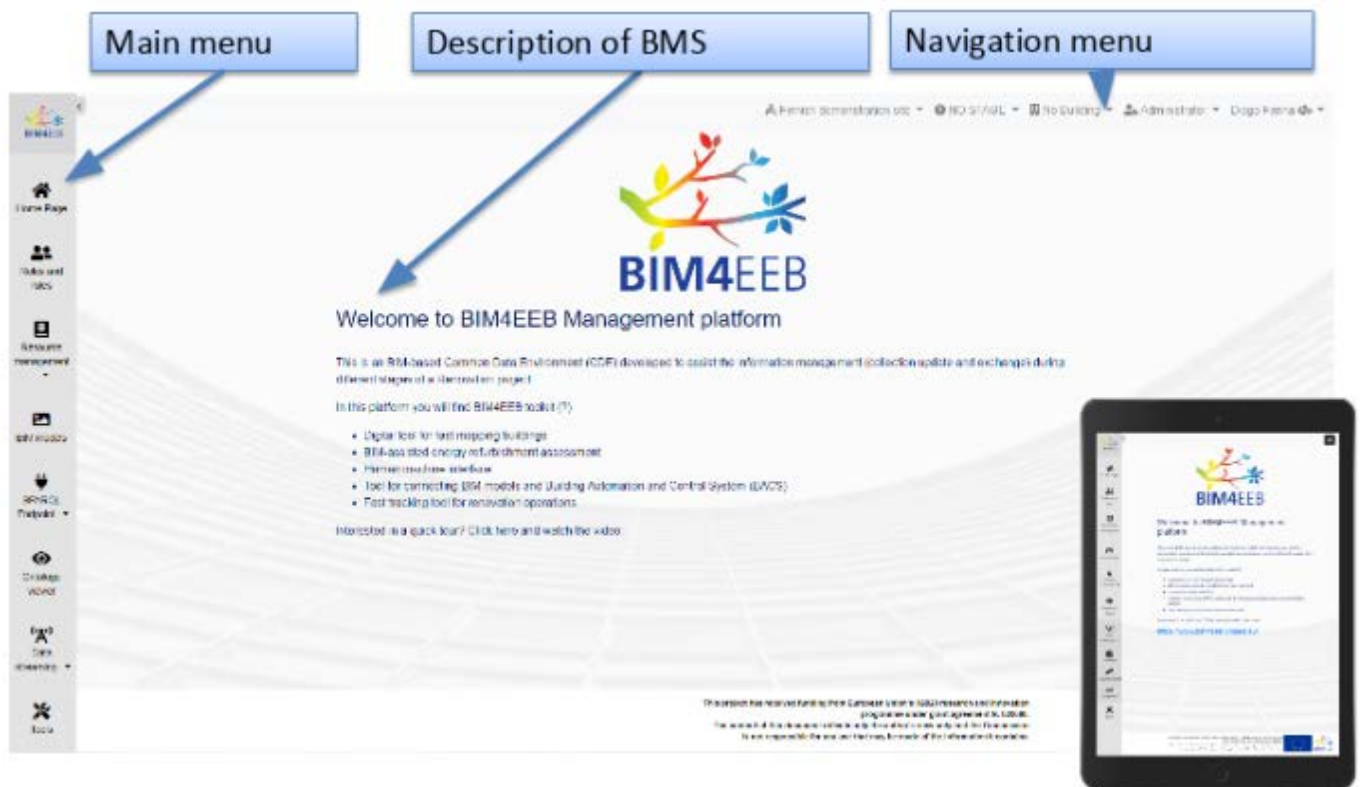


Figure 11 – BIMMS interface areas

### 4.4 BIMMS Interface

The BIM Management System web portal is available to all registered users. It has a user interface that permit to manage the user account and have access to the tools functionalities developed in the BIM4EEB project.

The web portal interface has a layout configured in three elements: a main area where the user interacts with the tools, a small header on the top of the page that contain the navigation menu, and main menu on the left side that give access to the BIMMS contents. Navigation and Main menu are always visible and allow the user to navigate through the functionalities.

The main area occupies the large part of the user interface, and it is where the user can interact with the system and the tools functionalities. In the main area, there are specific interfaces developed for every thematic area and functionalities.

The navigation menu contains the user management functionalities and allow the users to navigate through all projects they have access to. Detailed functionalities are described in 4.5.

The main menu allows the user to access the functionalities developed in the BIM4EEB Project. Functionalities access depend on user permissions defined by the user profile and may be different depending on specific permissions assigned to the user.

Navigation through the menu is possible by clicking on the icons. Some icons can reveal more buttons that could differentiate specific functionalities of the tool selected.

## 4.5 BIMMS Navigation Menu

In the navigation menu the user can manage his user profile and the data related to the user

### 4.5.1 User profile settings

This section enable user to manage his personal data, to request access to projects and roles. Her user can find also personal Enduser Token used by Ret Services to give external tool anonymous association between data and end-users.

### 4.5.2 Project area access

Using Select combo, user can change BIMMS context, available options are:

- Project
- Role
- Building
- Stage

Project and Role are always active, Building and Stage are not mandatory.

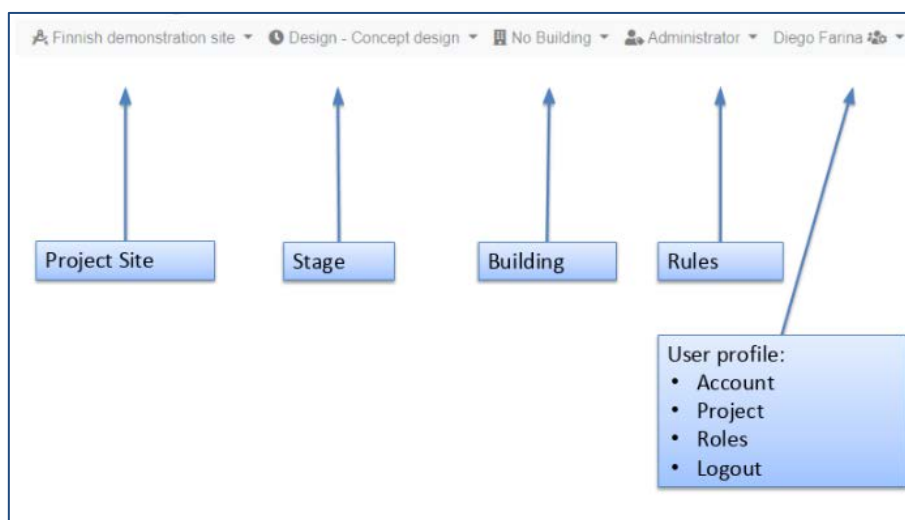


Figure 12 – BIMMS Context navigation menu

## 4.6 Main menu

Main menu, on the left side of interface, is context dependant. The following paragraphs describes the Administrator full access menu.

### 4.6.1 Home Page

Home Page button give access to the home page of the BIMMS. All users have access to this route and to the Home Page.



Figure 13 – BIMMS homepage

Home page contains only public information.

### 4.6.2 Roles and rules

The Roles and Rules button give access to the roles and rules management for the project in use. Only Administrators members (BIMMS Administrator and Project Administrator) can have access to this route and to the Roles and Rules page.

Roles and rules are defined in compliance with requirements in D4.2

The available functionalities for user and Project administration are:

### 4.6.2.1 User management dashboard

Used by BIMMS Admin and Project Admin roles to manage active users and roles on the project

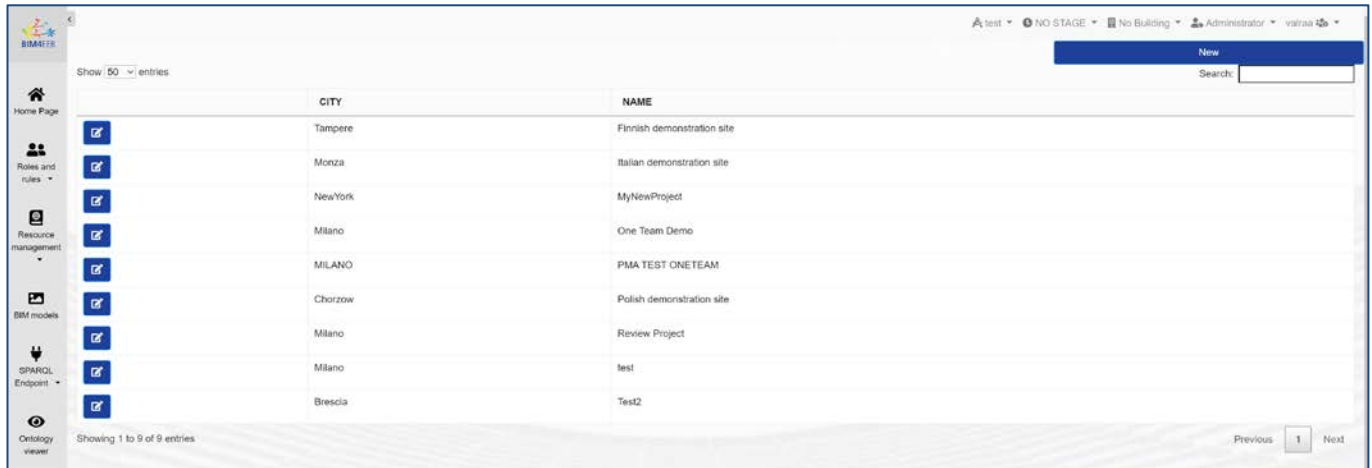


User	Email	Project	Roles active	Roles Pending	Project Admin	IFC Zones
chiappettj	jacopo.chiappetti@oneteam.it	test	Inhabitant/End-user, Client/Owner		<input type="checkbox"/>	Zones
martinop	pierfrancesco.martino2@oneteam.it	test	Inhabitant/End-user		<input type="checkbox"/>	Zones
tarinad	diego.tarina@oneteam.it	test	Client/Owner, Health and safety adviser, Project administrator		<input checked="" type="checkbox"/>	
valraa2	alessandro.valraa@libero.it	test	Client/Owner, Inhabitant/End-user, Client adviser, Project leader, Architectural designer, Structural designer, Fire safety designer, Landscape designer, Cost consultant/quantity surveyor, Health and safety adviser, Access consultant, Acoustic consultant, BREEM assessor		<input type="checkbox"/>	Zones
moniggp	pierfrancesco.martino3@oneteam.it	test	Inhabitant/End-user, Client adviser, Project leader, Client/Owner		<input type="checkbox"/>	Zones
madeddud	davide.madeddu@oneteam.it	test	Client/Owner, Project leader, Local authority, Project administrator		<input checked="" type="checkbox"/>	
teemu.m	teemu.matasniemi@vti.fi	test	Project administrator, Supplier, Inhabitant/End-user, Technical adviser, Architectural designer, Structural designer, Building services designer, Tester		<input checked="" type="checkbox"/>	Zones
mlehm	matthias_lehmann@tu-dresden.de	test	Project administrator, Supplier, Inhabitant/End-user, Client/Owner, Client adviser, Technical adviser, Project leader, Site surveyor, Lead designer, Architectural designer, Structural designer, Building services designer, Fire safety designer, Landscape designer, Cost consultant/quantity surveyor, Health and safety adviser, Access consultant, Acoustic consultant, BREEM assessor, Cladding specialist, Facilities management (FM) adviser, Information manager, Interior designer, Landscape architect, Lighting designer, Masterplanner, Planning consultants, Security adviser, Sustainability adviser, Contractor, Sub-contractor, Maintenance planner, Construction lead, Contract administrator, Operational lead, Party wall surveyor, Local authority, Tiler, Tiler/er, Tiler/er, Bank or third party financier, Work supervisor, Tester, HVAC Engineer, Electrical Engineer, Water and Waste Engineer		<input checked="" type="checkbox"/>	Zones
LarissaDR	larissa.derosso@ace-cae.eu	test	Architectural designer, Project administrator		<input checked="" type="checkbox"/>	
sto	seppo.torma@visualynk.com	test	Supplier, Inhabitant/End-user, Client/Owner, Client adviser, Project leader, Cost consultant/quantity surveyor, Information manager, Masterplanner, Planning consultants, Contractor, Sub-contractor, Contract administrator, Tenderer, Work supervisor, Tester		<input type="checkbox"/>	Zones

Figure 14 – BIMMS user management dashboard

### 4.6.2.2 Projects list

Used by BIMMS Admin roles to manage and create new projects into the system

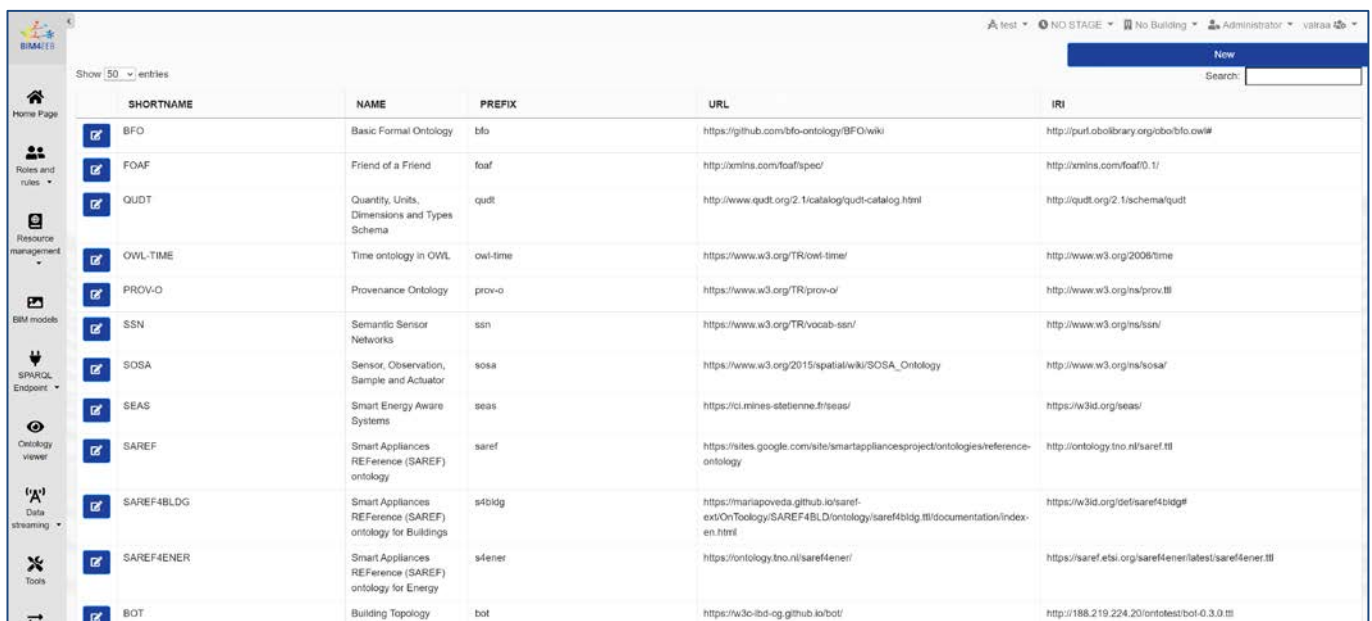


	CITY	NAME
	Tampere	Finnish demonstration site
	Monza	Italian demonstration site
	NewYork	MyNewProject
	Milano	One Team Demo
	MILANO	PMA TEST ONETEAM
	Chorzow	Polish demonstration site
	Milano	Review Project
	Milano	test
	Brescia	Test2

Figure 15 – BIMMS Project management

### 4.6.2.3 Ontology reference list

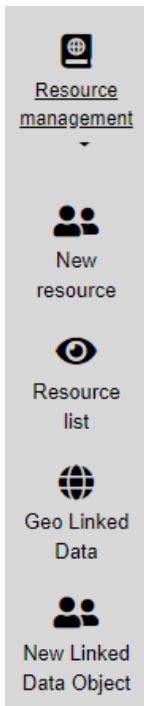
Used by BIMMS Admin roles to manage and create new ontology reference into the system; ontology references are used to implement parametric classification based on linked data for BIMMS resources.



	SHORTNAME	NAME	PREFIX	URL	IRI
	BFO	Basic Formal Ontology	bfo	https://github.com/bfo-ontology/BFO/wiki	http://purl.obolibrary.org/obo/bfo.owl#
	FOAF	Friend of a Friend	foaf	http://xmms.com/foaf/pec/	http://xmms.com/foaf/0.1/
	QUDT	Quantity, Units, Dimensions and Types Schema	qudt	http://www.qudt.org/2.1/catalog/qudt-catalog.html	http://qudt.org/2.1/schema/qudt
	OWL-TIME	Time ontology in OWL	owl-time	https://www.w3.org/TR/owl-time/	http://www.w3.org/2006/time
	PROV-O	Provenance Ontology	prov-o	https://www.w3.org/TR/prov-o/	http://www.w3.org/ns/prov.ttl
	SSN	Semantic Sensor Networks	ssn	https://www.w3.org/TR/vocab-ssn/	http://www.w3.org/ns/ssn/
	SOSA	Sensor, Observation, Sample and Actuator	sosa	https://www.w3.org/2015/spatial/wiki/SOSA_Ontology	http://www.w3.org/ns/sosa/
	SEAS	Smart Energy Aware Systems	seas	https://ci.mines-st-etienne.fr/seas/	https://w3id.org/seas/
	SAREF	Smart Appliances REference (SAREF) ontology	saref	https://sites.google.com/site/smartappliancesproject/ontologies/reference-ontology	http://ontology.tno.nl/saref.ttl
	SAREF4BLDG	Smart Appliances REference (SAREF) ontology for Buildings	s4bdg	https://mariapoveda.github.io/saref-ext/OnToology/SAREF4BLD/ontology/saref4bdg.ttl/documentation/index-en.html	https://w3id.org/def/saref4bdg#
	SAREF4ENER	Smart Appliances REference (SAREF) ontology for Energy	s4ener	https://ontology.tno.nl/saref4ener/	https://saref.etsi.org/saref4ener/latest/saref4ener.ttl
	BOT	Building Topology	bot	https://w3c-lbd-cg.github.io/bot/	http://188.219.224.20/ontotest/bot-0.3.0.ttl

Figure 16 – BIMMS Ontology reference management

### 4.6.3 Resource management



Resource Management button give access to the functionalities to manage the resources in BIMMS. Only specific user roles can access to this route.

The Resources are the data stored in the BIMMS. A resource can be types of entities such as a 3D model, a building element, a user, a task, a sensor, or a document.

The Resource Management enable to define, upload, and map these entities as resource that will be referenced in the CDE by a universal identifier. This unique reference will be used to retrieve the representation of the resource and to establish links among other resources stored in BMS CDE and across the Web. The universal identifier is automatically assigned when resource is created.

The Resource Management menu contains the submenus that give access to functionalities to create a new resource (New Resource button), list the existing resources in the project (Resource List button), create a Geo Linked Resource Data (Geo Linked Data button), and create a new linked data object (New Linked Data Object button). For a complete detailed description of each functionalities, please refer to the next paragraphs.

#### 4.6.3.1 New Resource

This menu button allows the user to create a new resource in the BIMMS. A new resource shall be created following the actions steps presented in a wizard. The wizard is composed in different steps that ask the user to compile a list of items that define the new resource.

The wizard has 8 steps, as follows.

The first step of the wizard is called Element. In this step the user defines the type and main categorization of the new resource. Users have to choose which Scope, Category and Typology belong to the resource.

Scope, Category and Typology shown different items according to the user roles and permissions. Not all users are able to create any kind of resource because they have no permissions to put this kind of data. Permissions are defined and managed by Administrator Roles.

The Scope define the root category of the resources. The scopes are:

- The “BIM4EEB Scope”: used to define a generic resource in BIMMS;
- The Logbook Scope: used to define and record the data associated to the building.

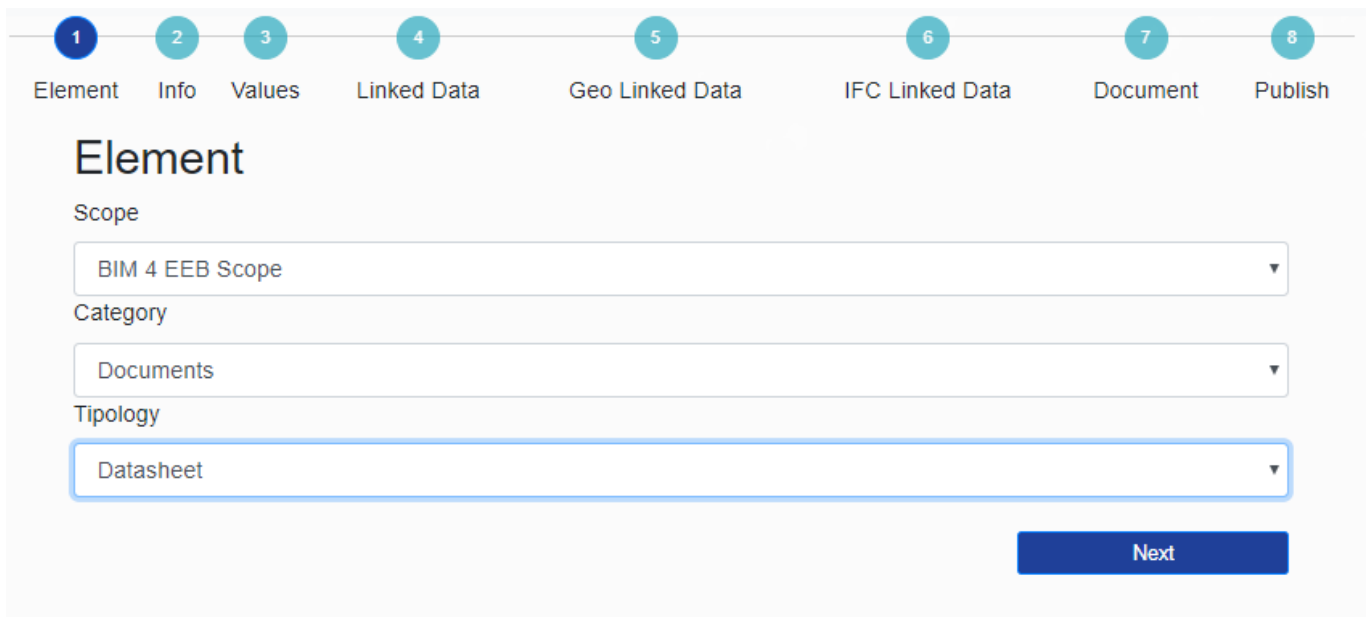
The Category shall define the type of Resource. The Categories are:

- Documents: any resource like Microsoft Office Document, PDFs, Images, Videos, TXTs, CSVs. Shall be used for Essays, Datasheets, QTO files and any other document that are not categorizable in the categories below;
- Drawings: any resource that contains drawings created by hand or in electronic format like DWGs, DXFs, PDFs;
- Models: any resource that contain 3D models in native or interoperability file format like IFCs. It

can also include 3D Point Clouds models;

The Typology specifies in detail the type of the resource. The Typologies shall be:

Datasheet, QTO, Specification, Contracts, 3D Models Native, 3D IFC Models, 3D Point Cloud



1 2 3 4 5 6 7 8

Element Info Values Linked Data Geo Linked Data IFC Linked Data Document Publish

## Element

Scope

BIM 4 EEB Scope

Category

Documents

Tipology

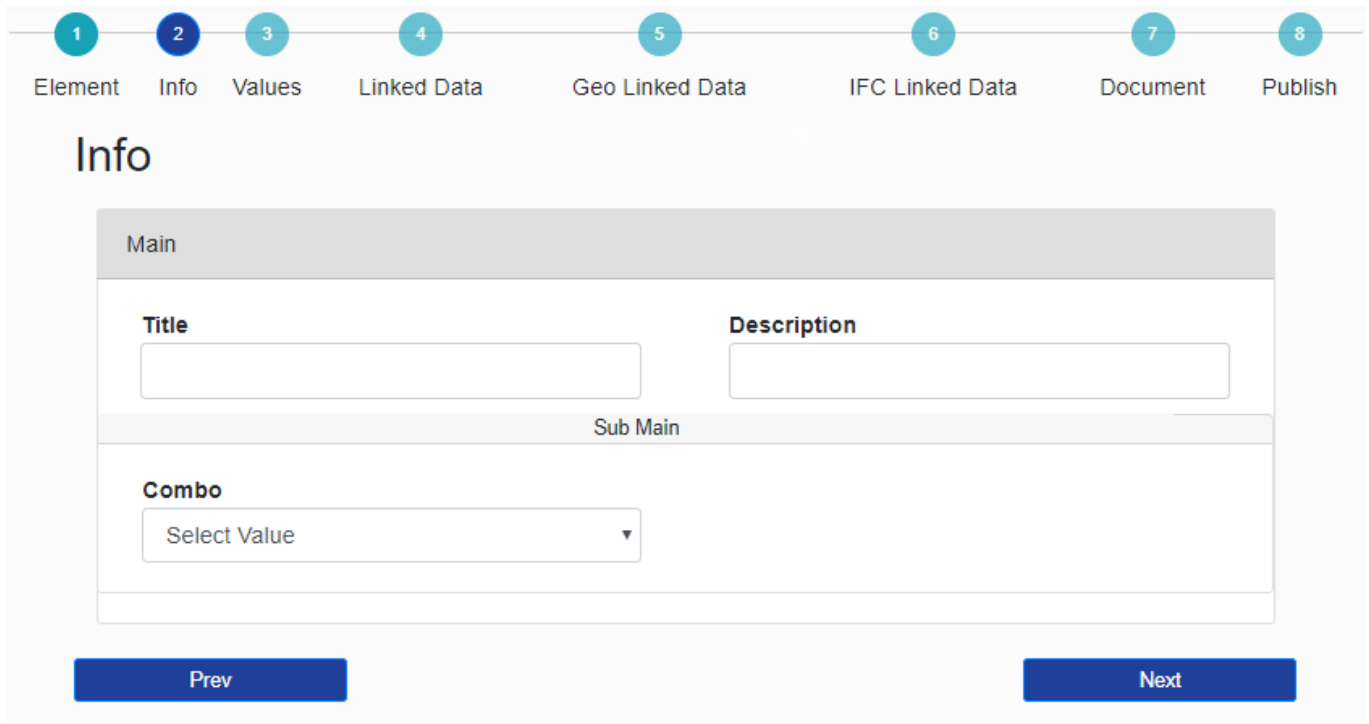
Datasheet

Next

**Figure 17 – BIMMS Resource creation wizard – step 1**

The second step of the wizard is called Info. In this step the user can add the Title and the Description to the new Resource. Some categories of resources might have additional fields that could be compiled. This step is not mandatory but is recommended to be completed. This information could be useful to later recognize the resource in the BIM Management and to query and find easily the data uploaded.





1 Element 2 Info 3 Values 4 Linked Data 5 Geo Linked Data 6 IFC Linked Data 7 Document 8 Publish

## Info

Main

Title

Description

Sub Main

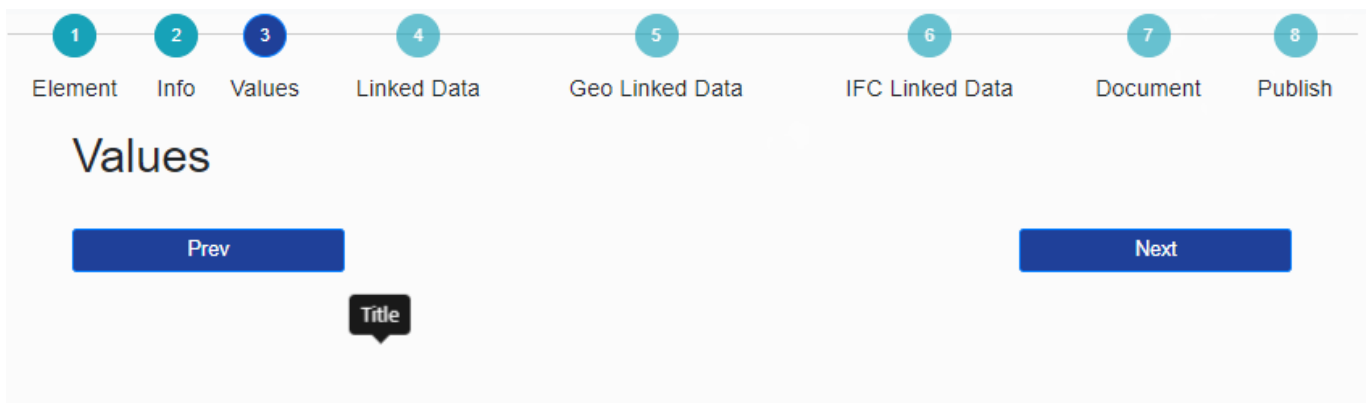
Combo

Prev Next

**Figure 18 – BIMMS Resource creation wizard – step 2**

### Values

The third step of the wizard is called Values. This step is useful only when the user is editing an existing resource. When a resource is edited, the values already provided will be shown in this step.



1 Element 2 Info 3 Values 4 Linked Data 5 Geo Linked Data 6 IFC Linked Data 7 Document 8 Publish

## Values

Prev Next

Title

**Figure 19 – BIMMS Resource creation wizard – step 3**

### Linked Data

The fourth step of the wizard is called Linked Data. This step allows the user to define a mapping with one or more ontology related to the resource domain. This step is not mandatory but is recommended to give a mapping to extend and enrich the linked data connections between the resources in the BIMMS. The

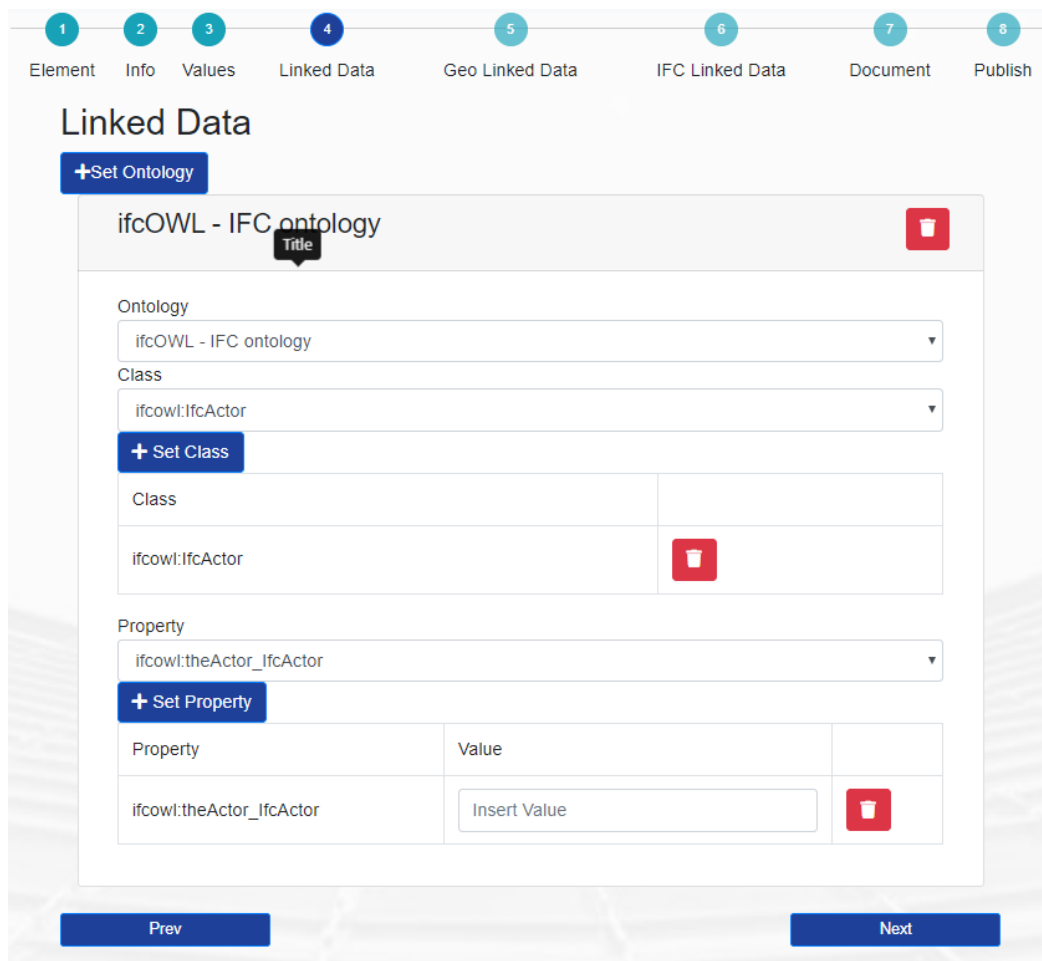
Linked Data mapping done in this step allows to retrieve additional resource information useful in SPARQL queries to find hidden relationships between data.

The ontology shall be selected from a selector menu. This menu shown the ontologies selected for the BIM4EEB Project as requirement defined in the WP3. The ontologies are read in real time from their official repository and inherit their updates. The BIMMS is not responsible of missing information or unavailability of certain ontologies reference at the time of mapping.

The users can select the ontology, then select the class and the property where to map the resource data. They can add mapping with more properties per class, more mapping classes per ontology and also more mapping ontologies per resource. Adding properties, classes and ontologies are possible clicking in the + buttons.

To complete and confirm the mapping, the users shall click the button Set Class and Set Property. After setting the property, users could define the values of the property. A trash icon allows to delete the mapping already done.

All menus are updated in real time. The users can experience some waiting time for the system to update the description menu, during the selection of ontologies, classes and properties. Please, wait a few seconds to get the data from repositories. Waiting time could be different depending on the ontology dimension and relationships complexity.



1 Element 2 Info 3 Values 4 **Linked Data** 5 Geo Linked Data 6 IFC Linked Data 7 Document 8 Publish

### Linked Data

**+Set Ontology**

ifcOWL - IFC ontology 🗑️

Ontology  
ifcOWL - IFC ontology

Class  
ifcowl:IfcActor **+ Set Class**

Class		
ifcowl:IfcActor		🗑️

Property  
ifcowl:theActor\_IfcActor **+ Set Property**

Property	Value	
ifcowl:theActor_IfcActor	Insert Value	🗑️

**Prev** **Next**

Figure 20 – BIMMS Resource creation wizard – step 4

### Geo Linked Data

The fifth step of the wizard is called Geo Linked Data. This step allows the user to define the positioning of the resource in the World giving its Latitude and Longitude. This step is not mandatory but is recommended for resources that could be positioned and could be searchable by its location.

The users could navigate in the world map, panning, and zooming. Clicking on a specific point on the map shall define the positioning. The positioning could be named and numbered. Users can define multiple locations clicking more than one time. All positioning is recorded and listed in a table with Latitude and Longitude data.

The positioning functionality relies on Open Layer library and OpenStreetMap services. Positioning data is mapped in WGS84 Geo Positioning Ontology.

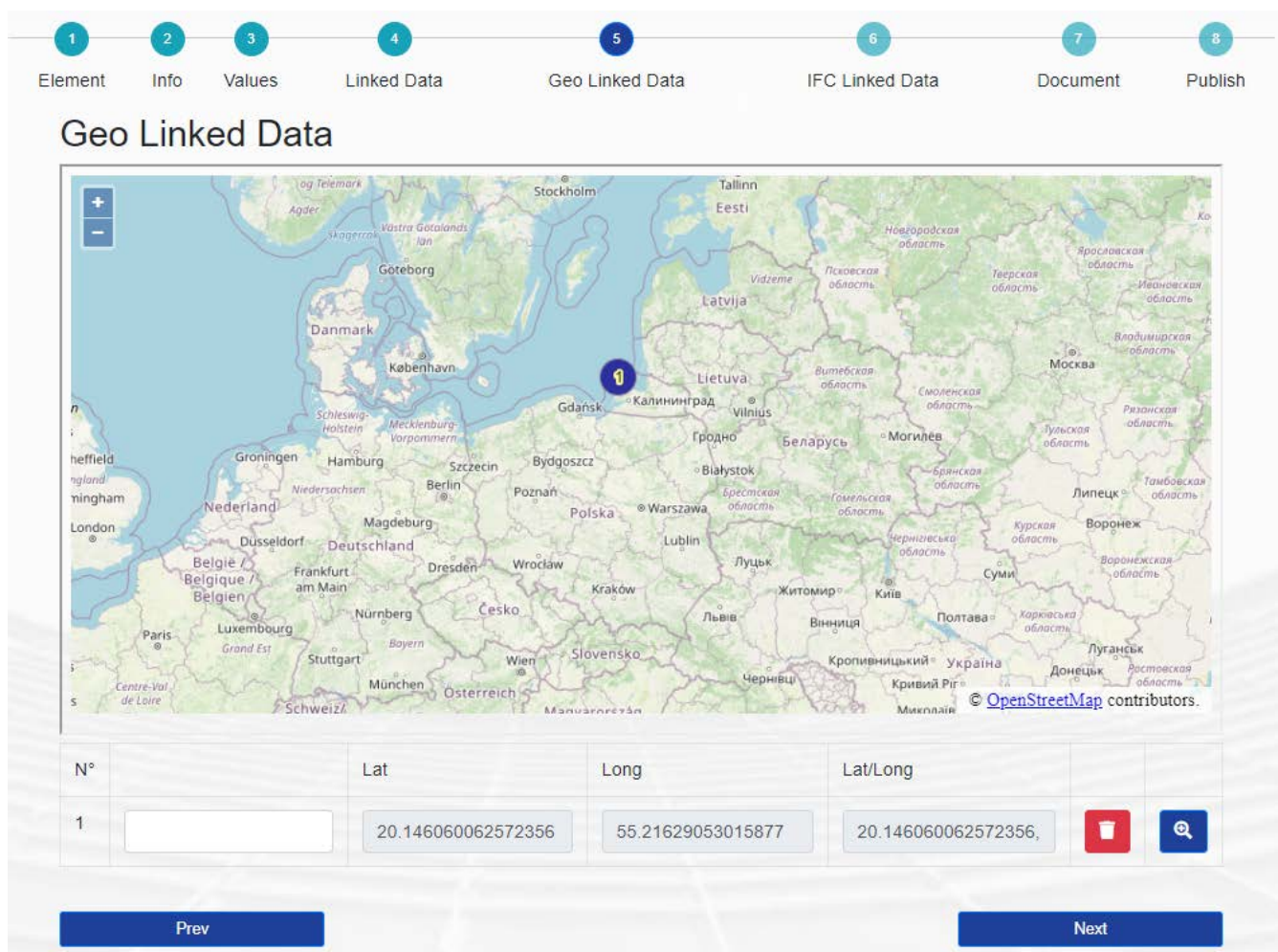


Figure 21 – BIMMS Resource creation wizard – step 5

### IFC Linked Data

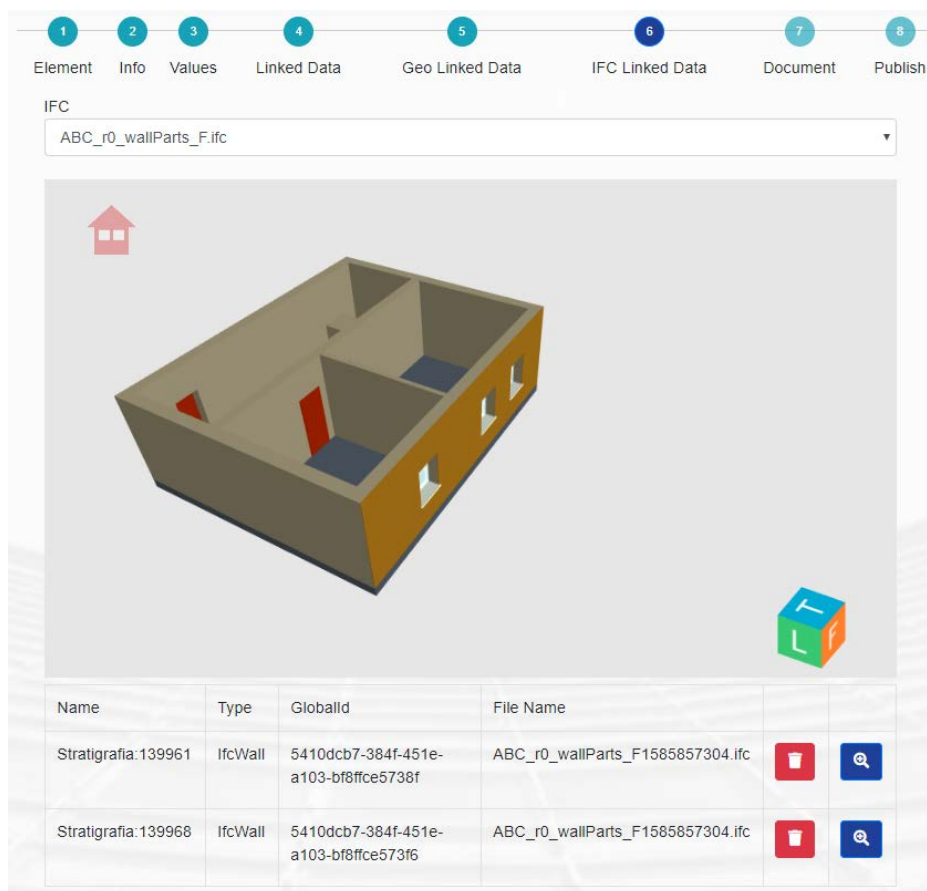
The sixth step of the wizard is called IFC Linked Data. This step allows the user to define a relationship to specific items in an IFC File already uploaded in the BIMMS. This step is not mandatory but is recommended for all resources that are related to a specific IFC BIM Model. The users shall upload the

3D IFC Model before getting this relationship selectable and available.

The users can choose the IFC File from the list of files shown in the selector.

After selecting the IFC File, the interface shows the BIM Viewer with the file selected. Then the users can double click on a specific item of the model. The item will be shown in a table below the viewer with the name, type, GlobalID and IFC File name.

The users can add multiple mapping to the resource by double clicking in other items in the BIM Viewer. All mapping will be listed in the table. A trash icon allows to delete the mapping done, and a lens icon allow to zoom to the specific item mapped.



**Figure 22 – BIMMS Resource creation wizard – step 6**

### Document

The seventh wizard step is called Document. This step allows the user to define the document related to the resource. This step is not mandatory, but it is recommended for resources as contracts, documents and so on.

The users can select the file from their computer. The file will be uploaded to the BIMMS and will be available there.

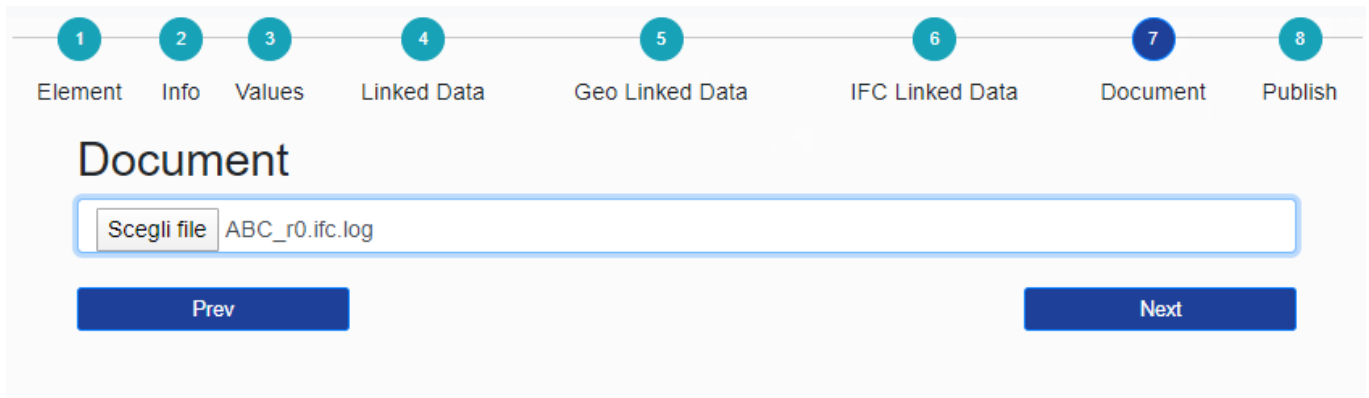


Figure 23 – BIMMS Resource creation wizard – step 7

### Publish

The eighth and the last wizard step is called Publish. This step allows the user to publish the new resource to the BIMMS. This step is required.

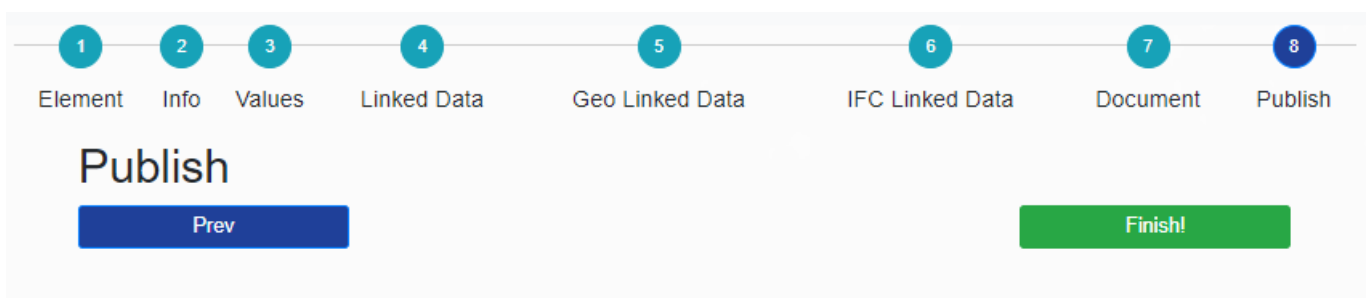


Figure 24 – BIMMS Resource creation wizard – step 8

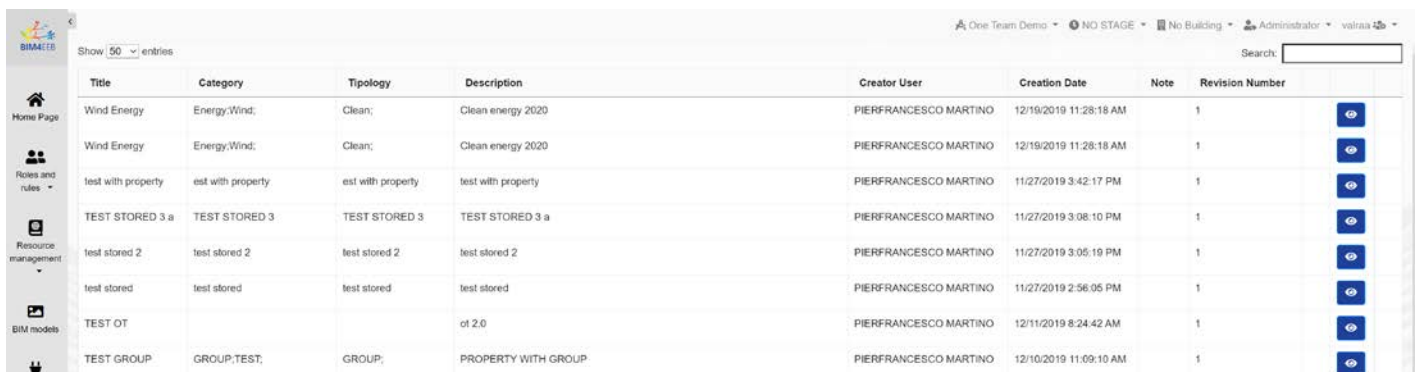
### 4.6.3.2 Resource List

Resource List button shows the page where are listed all resource available in the project that the user can have permission to access. All users have access to this route and to the resource list.

Access to the resources could be different depending on the permissions and the user role and user permissions assigned.

The resources are listed in a table that shows:

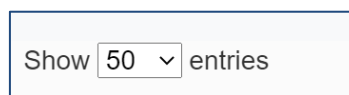
- Title of the resource: assigned by the user during the creation of the new resource;
- Category: assigned by the user during the creation of the new resource;
- Typology: assigned by the user during the creation of the new resource;
- Description of the resource: assigned by the user during the creation or the new resource;
- Creator User;
- Creation Date;
- Note;
- A button to edit the resource;
- A button to view the resource details;
- A button to set a new revision of the resource;
- All columns could be ordered in ascending or descending order clicking on the header name on top of the table.



Title	Category	Tipology	Description	Creator User	Creation Date	Note	Revision Number	
Wind Energy	Energy;Wind;	Clean;	Clean energy 2020	PIERFRANCESCO MARTINO	12/19/2019 11:28:18 AM		1	
Wind Energy	Energy;Wind;	Clean;	Clean energy 2020	PIERFRANCESCO MARTINO	12/19/2019 11:28:18 AM		1	
test with property	est with property	est with property	test with property	PIERFRANCESCO MARTINO	11/27/2019 3:42:17 PM		1	
TEST STORED 3 a	TEST STORED 3	TEST STORED 3	TEST STORED 3 a	PIERFRANCESCO MARTINO	11/27/2019 3:08:10 PM		1	
test stored 2	test stored 2	test stored 2	test stored 2	PIERFRANCESCO MARTINO	11/27/2019 3:05:19 PM		1	
test stored	test stored	test stored	test stored	PIERFRANCESCO MARTINO	11/27/2019 2:56:05 PM		1	
TEST OT			ot 2.0	PIERFRANCESCO MARTINO	12/11/2019 8:24:42 AM		1	
TEST GROUP	GROUP;TEST;	GROUP;	PROPERTY WITH GROUP	PIERFRANCESCO MARTINO	12/10/2019 11:09:10 AM		1	

Figure 25 – Resource List dashboard

Above the table header, on the left side there is a selector to page the list of the resources by 10, 25, 50 (set by default), or 100 items.



Above the table header, on the right side there is a field to search items on the list of the resources.

Search:



**Edit Resource**

Edit Resource button allow to edit the selected resource. The access to this feature depending on the users' role and permissions.

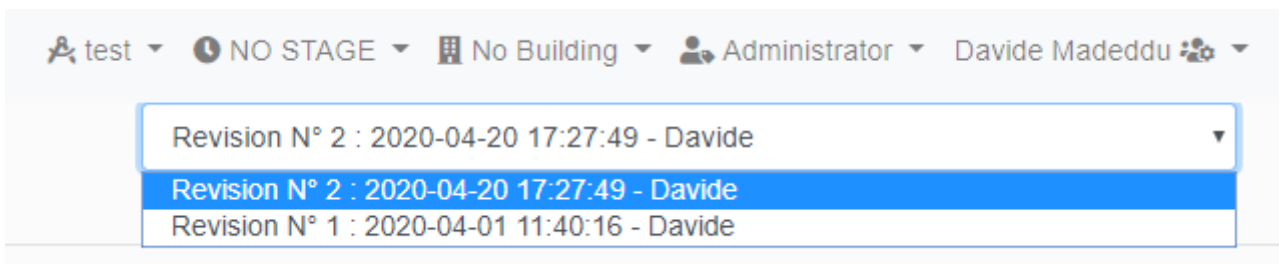
The editing feature opens the new resource wizard creation described in 4.6.3.1 where the user can change the data input following the same steps. All fields previously compiled during the creation phase of the resource are shown and the user can modify their values and choices. Detail about the steps of the creation wizard are described in 4.6.3.1.



**View Resource**

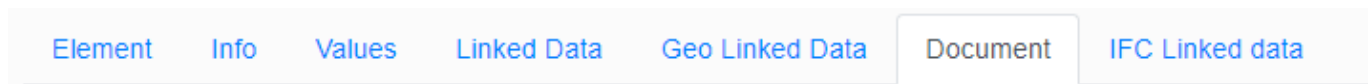
View Resource button allow to view resource details. The access to this feature depending on the users' role and permissions.

On top right side of the page is shown the resource version selector. This item allows to choose a resource version and once selected, view the details. The selector shown the revision number, date and time of the revision and the revision author.



**Figure 26 - Resource revision selector**

The details are grouped in tabs, with the same name and order defined in the new resource wizard. If no information was provided during resource creation, no data were shown in the tabs.



**Figure 27 – Resource classification Tabs**

Element tab: shows the scope, category and typology;

Info tab: shows title and description;

Values: shows al properties and values associated to resource;

Linked data: shows the ontology mappings;

Geo Linked Data: shows the Geo Linked Data with the Latitude/longitude positioning;  
Document: shows the documents of this resource. Documents could be downloaded;  
IFC Linked Data: shows the BIM 3D IFC models linked to this resource;



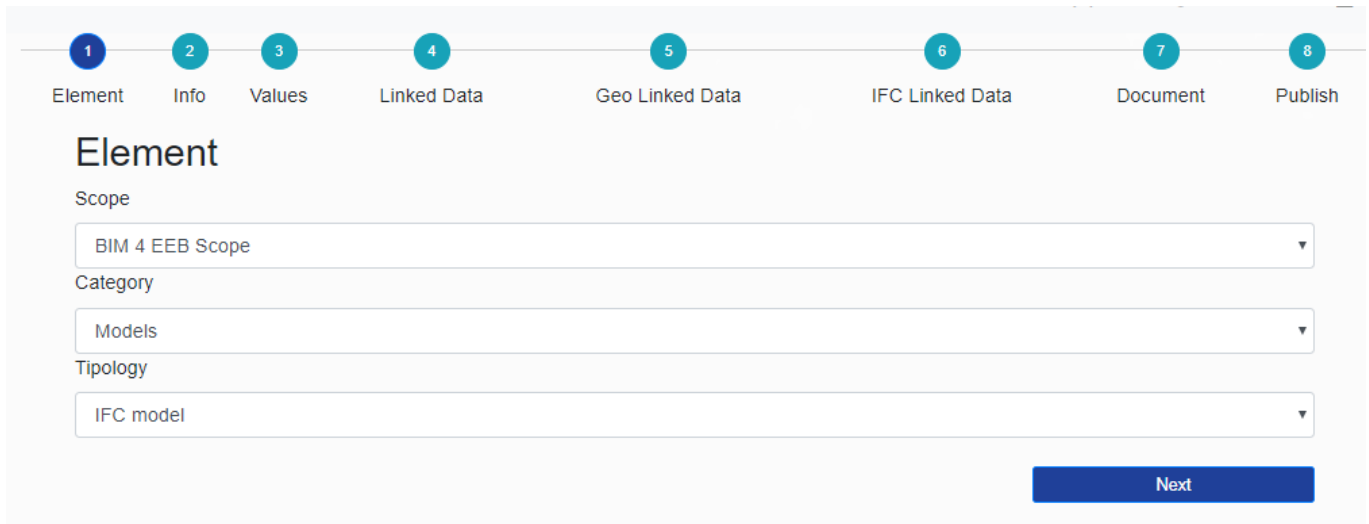
### New Revision

New Revision button allow to create a new revision. The access to this feature depending on the users' role and permissions.

Users can create a new revision of the resource following the steps presented in a wizard. The wizard is the same presented in 4.6.3.1 and it is composed in different steps that ask the user to complete a list of items that define the new resource revision.

For the new revision, all fields filled in the previous revision are shown. The users can change the data provided in the previous revision, by add, delete, or edit the information shown in the wizard.

You could refer to 4.6.3.1 to details about the steps of the wizard.



1 Element 2 Info 3 Values 4 Linked Data 5 Geo Linked Data 6 IFC Linked Data 7 Document 8 Publish

## Element

Scope  
BIM 4 EEB Scope

Category  
Models

Tipology  
IFC model

Next

**Figure 28 – Resource new revision step 1**



### 4.6.3.3 Geo Linked Data

Geo Linked Data button allow to view all geo linked data available in the project. The access to this feature depending on the users role and permissions.

The page shows an OpenStreetMap where the resources of the project are located. The location of the resource is reported by a point or a cluster of points if there are more than 1 resource in the location or in the same area depending on zoom settings. Users can navigate through the map with the zoom and the pan

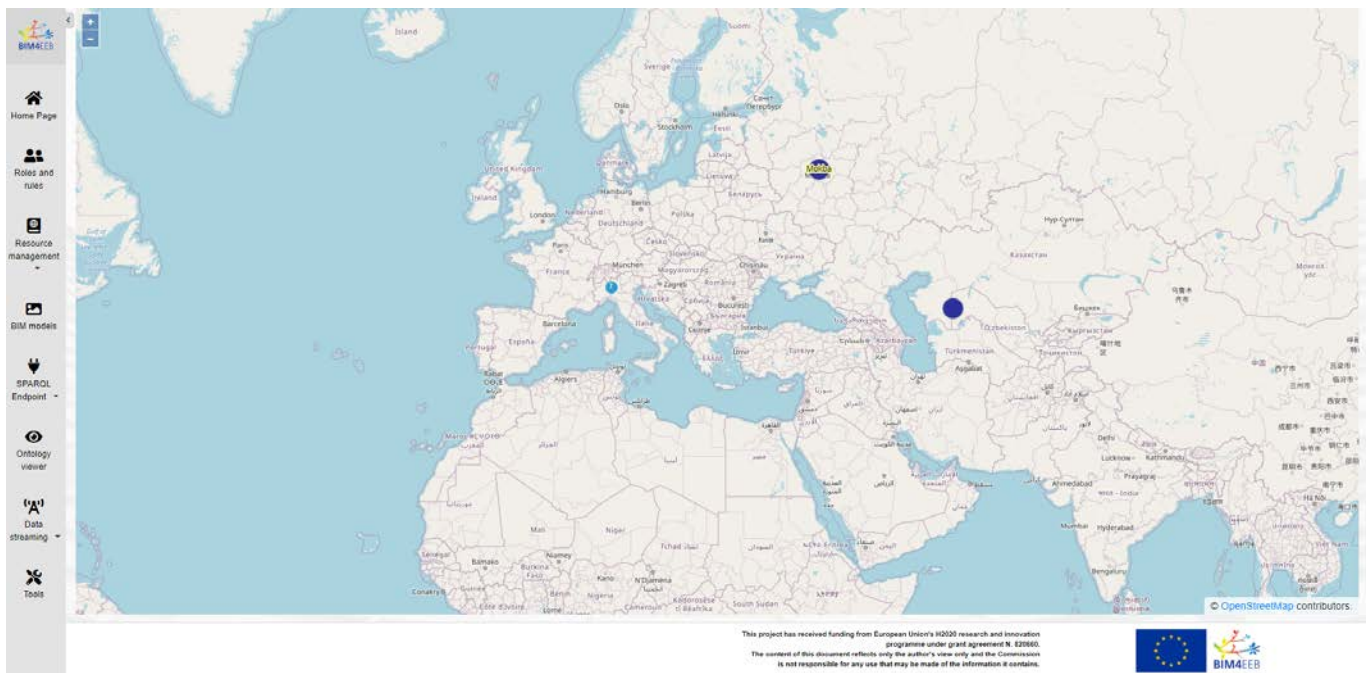


Figure 29 – GIS map of georeferenced Project data

#### 4.6.3.4 New Linked Data Object

The New Linked Data Object button allows to create a new Linked Data Object. The access to this feature depending on the users' role and permissions.

A linked Data Object is a resource defined only by ontology.

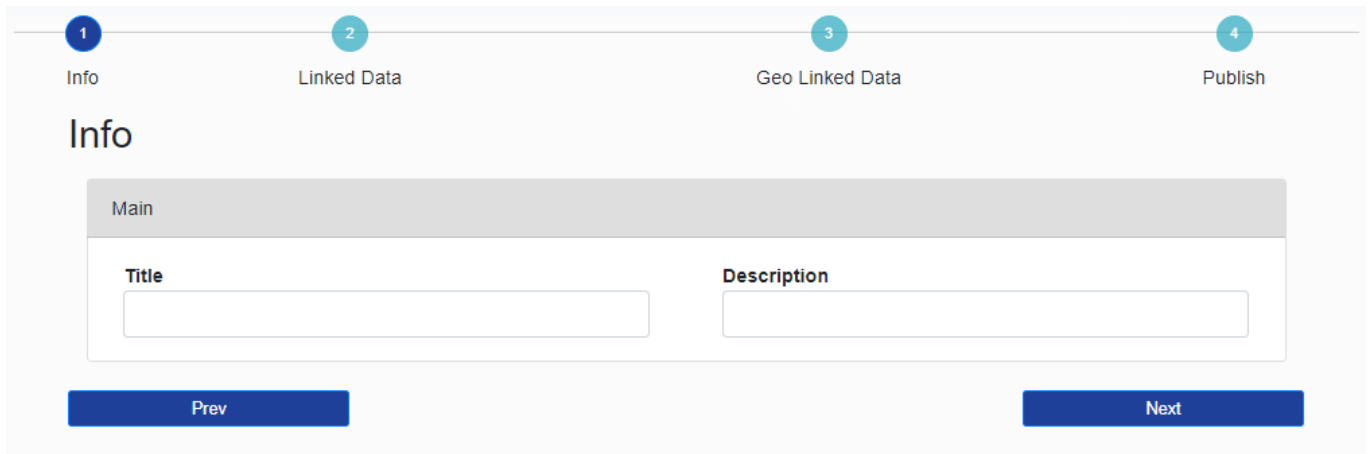


Figure 30 - New linked data object wizard step 1

A new Linked Data Object shall be created following the actions steps presented in a wizard. The wizard is composed in different steps that ask the user to complete a list of items that define the new resource.

The Linked Data Object wizard has 4 steps: Info, Linked Data, Geo Linked Data and Publish.

The first step of the wizard is called Info. In this step the user can add the Title and the Description to the new Resource. Some categories of resources might have additional fields that could be compiled. This step is not mandatory but is recommended to be completed. This information could be useful to later recognize the resource in the BIM Management and to query and find easily the data uploaded.

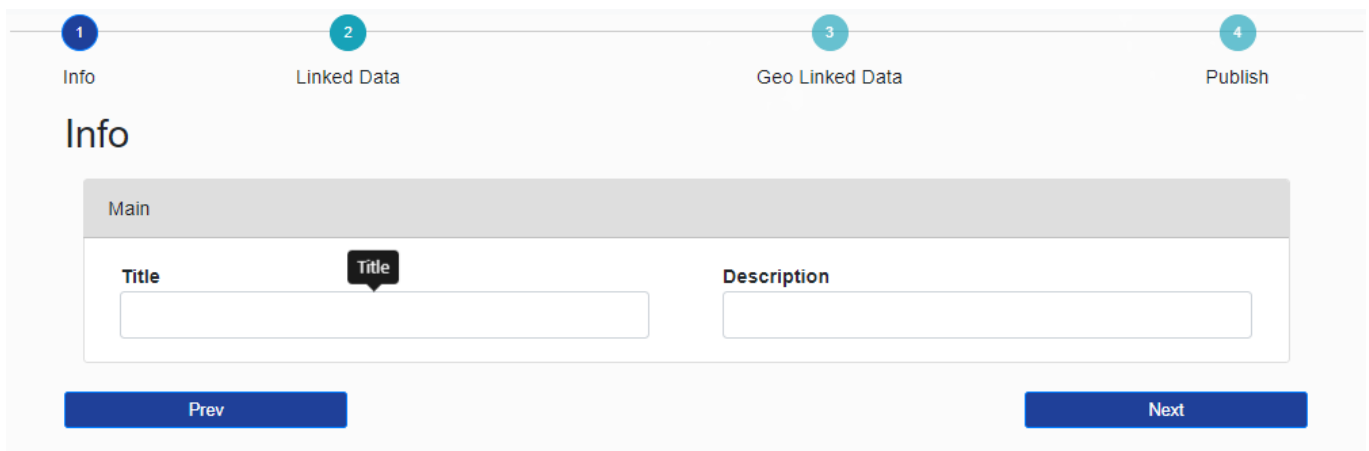
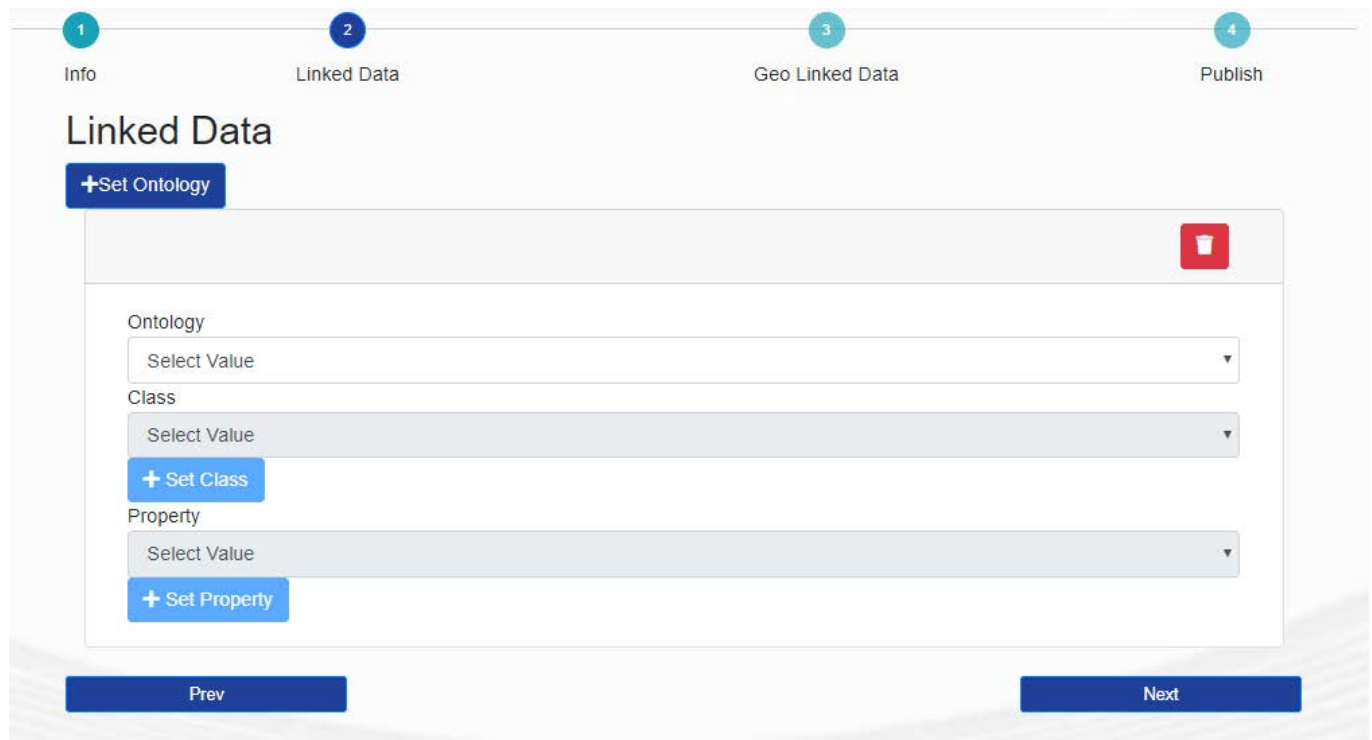


Figure 31 - New linked data object wizard step 2

## Linked Data

The second step of the wizard is called Linked Data. This step is already described for general resource creation.



The screenshot shows a wizard interface with four steps: 1. Info, 2. Linked Data, 3. Geo Linked Data, and 4. Publish. The 'Linked Data' step is active. It features a '+Set Ontology' button, a trash icon, and three dropdown menus for 'Ontology', 'Class', and 'Property', each with a '+ Set' button below it. 'Prev' and 'Next' buttons are at the bottom.

**Figure 32 - New linked data object wizard step 3**

## Geo Linked Data

The third step of the wizard is called Geo Linked Data. This step is already described for general resource creation.

## Publish

The fourth and the last step of the wizard is called Publish. This step allows the user to publish the new resource to the BIMMS. This step is required.

#### 4.6.4 BIM models

BIM Models button shows the page where all BIM Models available in the project that user can have permission to access are listed. The users have access to this route and to the BIM Models list depending on user role and permissions assigned.

The BIM Models are listed in a table that shows:

File name: the name of the file uploaded in the BIMMS;

Title: assigned by the user during the uploading of the BIM Model as new resource;

Description: assigned by the user during the uploading of the BIM Model as new resource;

Keywords;

Synonyms;

Creator User;

Creation Date;

Note;

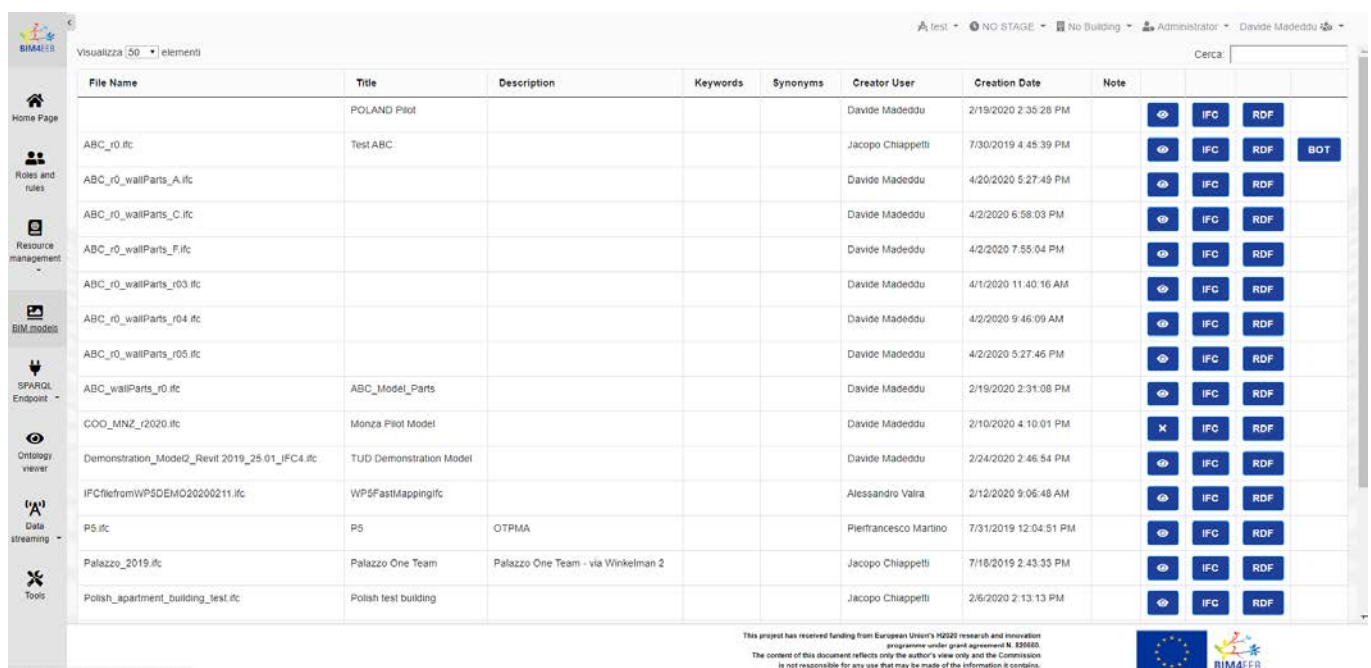
A button to view the BIM Model in the 3D Viewer;

A button to download the IFC file of the 3D Model;

A button to download the RDF IfcOWL ontology file of the 3D Model;

A button to download the RDF BOT ontology file of the 3D Model;

All columns could be ordered in ascending or descending order clicking on the header name on top of the table.



File Name	Title	Description	Keywords	Synonyms	Creator User	Creation Date	Note				
	POLAND Pilot				Davide Madeddu	2/19/2020 2:35:28 PM					
ABC_r0.ifc	Test ABC				Jacopo Chiappetti	7/30/2019 4:45:39 PM					
ABC_r0_wallParts_A.ifc					Davide Madeddu	4/20/2020 5:27:49 PM					
ABC_r0_wallParts_C.ifc					Davide Madeddu	4/2/2020 6:58:03 PM					
ABC_r0_wallParts_F.ifc					Davide Madeddu	4/2/2020 7:55:04 PM					
ABC_r0_wallParts_r03.ifc					Davide Madeddu	4/1/2020 11:40:16 AM					
ABC_r0_wallParts_r04.ifc					Davide Madeddu	4/2/2020 9:46:09 AM					
ABC_r0_wallParts_r05.ifc					Davide Madeddu	4/2/2020 5:27:46 PM					
ABC_wallParts_r0.ifc	ABC_Model_Parts				Davide Madeddu	2/19/2020 2:31:08 PM					
COO_MNZ_r2020.ifc	Monza Pilot Model				Davide Madeddu	2/10/2020 4:10:01 PM					
Demonstration_Model2_Revit_2019_25_01_IFC4.ifc	TUD Demonstration Model				Davide Madeddu	2/24/2020 2:46:54 PM					
IFCfilefromWPSDEMO20200211.ifc	WPSFastMappingIfc				Alessandro Valira	2/12/2020 9:06:48 AM					
P5.ifc	P5	OTPIA			Pierfrancesco Martino	7/31/2019 12:04:51 PM					
Palazzo_2019.ifc	Palazzo One Team	Palazzo One Team - via Winkelman 2			Jacopo Chiappetti	7/18/2019 2:43:33 PM					
Polish_apartment_building_test.ifc	Polish test building				Jacopo Chiappetti	2/6/2020 2:13:13 PM					

Figure 33 – BIM Models dashboard

Above the table header, on the left side there is a selector to page the list of the resources by 10, 25, 50 (set by default), or 100 items.

Show  entries

Above the table header, on the right side there is a field to search items on the list of the resources.

Search:

### 4.6.5 BIMMS 3D IFC BIM Viewer

The View Model button opens the selected 3D Model in the IFC BIM Viewer. Users can have access to this route and to the IFC BIM Viewer depending on user role and permissions assigned.

The BIM viewer page layout is composed by two areas:

- On the left is shown the 3D viewer with a toolbar on the bottom with buttons to change visualization or to active specific functionalities.
- On the right is shown a panel where the user can see the IFC data grouped in data tables or in a hierarchical tree viewer.

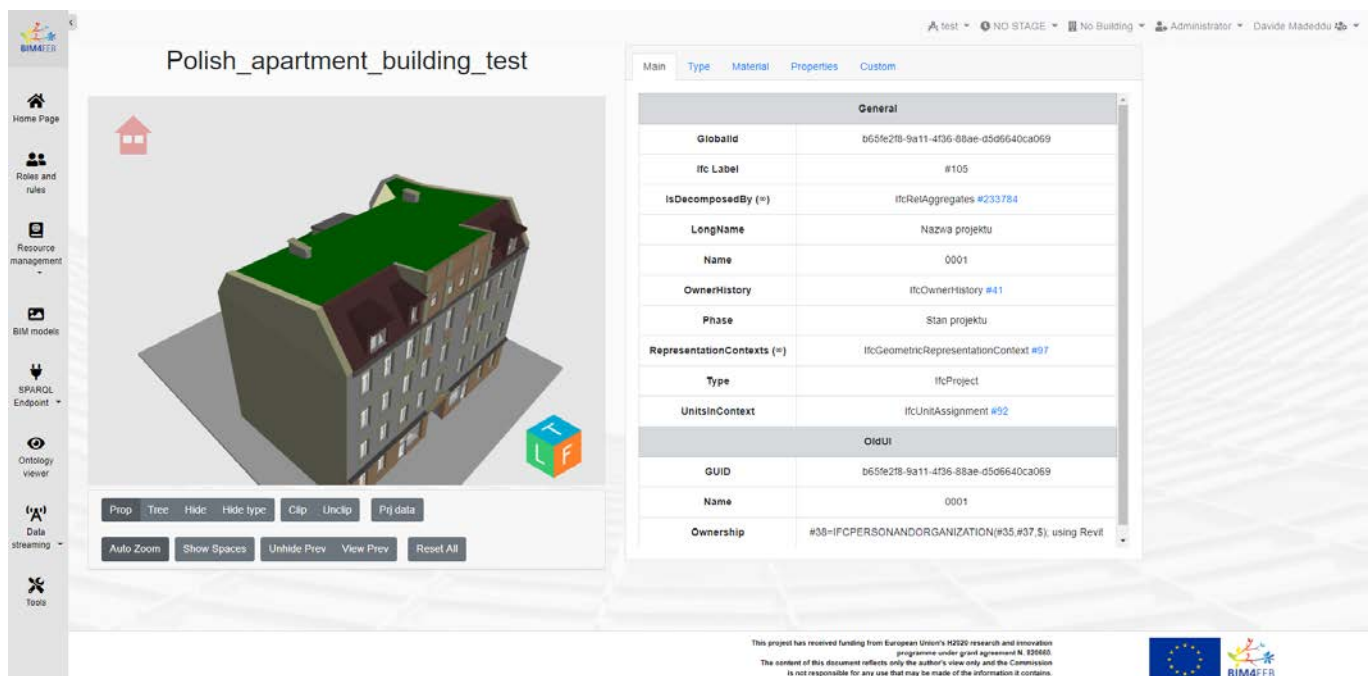


Figure 34 – BIM Models BIMMS viewer

#### 4.6.5.1 Left Panel: the 3D viewer

The 3D Viewer is able to show 3D IFC files.

The user can orbit, zoom, pan, the models in the 3D canvas using the mouse and the keyboard interaction.

**Orbit:** User can orbit the model holding down the left mouse button and drag in the direction of orbit. During orbit the model reorients around a pivot point, by default the pivot point is on the centre of the model or in the centre of the selected item.

**Zoom:** user can zoom in and out the model with the roller wheel on the mouse in the view.

**Pan:** user can move the model parallel to the screen holding down the right mouse button and drag in the direction of pan.

On the bottom right, there is a View Cube that can help to orient the 3D view. User can reorient the current view to a predefined orientation (Top, Bottom, Left, Right, Front and Rear side) by clicking in one of the faces, edges or corners of the view cube.

On the top left, there is a Home icon that orient the 3D view on the default orientation set on perspective from top left corner.



Figure 35 – BIM Models viewer

The view can be clipped to slice the 3D model on arbitrary plane. Items can be hide or unhide using a single selection or by type. Users can select items and see the information about selected elements on the right-side panel.



**Figure 36 – BIM Models viewer – hide objects**

A toolbar on the bottom side of the 3D canvas allow to change the 3D visualization and to activate the data visualization on the right-side panel from properties tables or hierarchical tree view:

Prop: shown on the table at the right side of the Model properties panel

Tree: shown on the table at the right side of the Model hierarchical tree view

Hide: Allow to hide the items on the 3D view by clicking on it. The command must be active/selected before selection.

Hide Type: Allow to hide all items of the same type on the 3D view by clicking on it. The command must be active/selected before selection.

Clip: Allow to clip the view and shown a 3D model section by a clip plane. The plane could be set defining a slice plane clicking on two points on the view

Unclip: Disable clipping view. Section plane will be deleted

**Proj Data:** Shown the Project data in the right side of the layout. The Project data are the properties defined in the upper hierarchy of the IFC files, like IfcBuilding and IfcSite class properties.

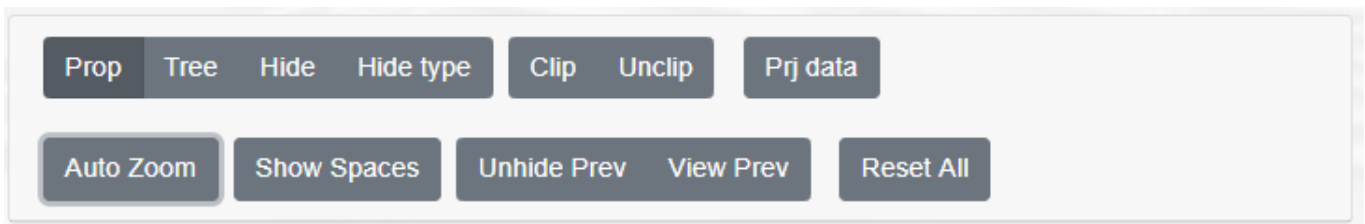
**Auto Zoom:** Enable or Disable the zoom on items when selected

**Show Spaces:** Enable or Disable the visibility of IfcSpaces and IfcZones on the 3D Model

**Unhide Prev:** Unhide the previous item hidden

**View Prev:** View and show the data properties of the previous item selected

**Reset All:** Reset all previous settings and set defaults



**Figure 37 – BIM Models BIMMS viewer – commands and options**

User can select an item on the model view by clicking the left mouse button. If Prop button on the toolbar is active, the selected item properties will be shown on the right side of the page layout. User can also select an item on the hierarchy tree view clicking with the left mouse button and then the selection will be shown on the 3D viewer on the left side panel of the page. By default, the selected item will be zoomed and shown fitted in the view to help user to find the object in 3D view. This behaviour could be disabled with Auto Zoon button in the toolbar.



#### 4.6.5.2 Right Panel: The property and hierarchy view

The right panel host the property view and the tree hierarchy view. The two views can be enabled by clicking on the buttons on the toolbar, below the viewer on the left side panel. The property view can be enabled clicking on Prop button. The hierarchy tree view can be enabled clicking on Tree button.

#### 4.6.5.3 The Property View

This view shows the IFC properties of the selected item in the model. If nothing is selected, by default shows the IFC properties related to IfcProject class

Properties are grouped in tabs:

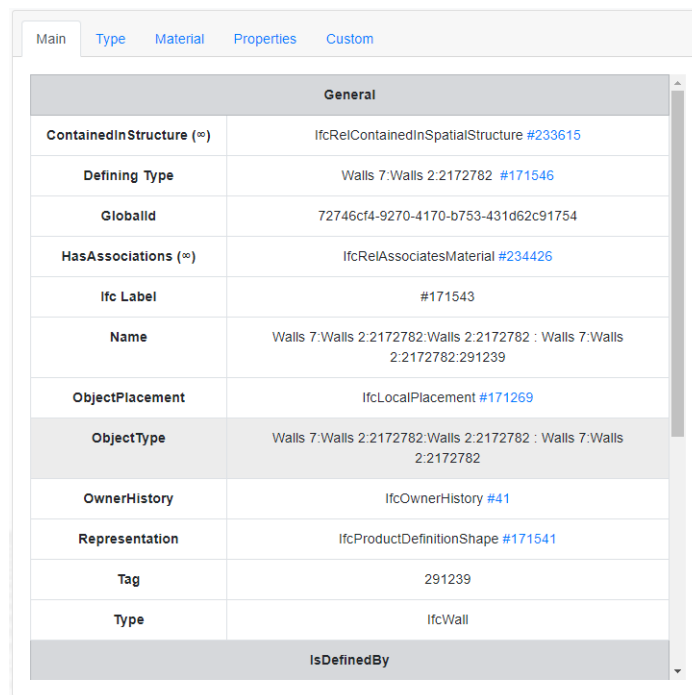
Main: shows instance attributes of the item selected;

Type: shows type attribute, if the item has a type;

Material: shows material attributes of the item selected;

Properties: shows IFC Properties related to the IfcClass of the item selected;

Custom: shows custom properties linked to the item selected



General	
ContainedInStructure (∞)	IfcRelContainedInSpatialStructure #233615
Defining Type	Walls 7:Walls 2:2172782 #171546
GlobalId	72746cf4-9270-4170-b753-431d62c91754
HasAssociations (∞)	IfcRelAssociatesMaterial #234426
Ifc Label	#171543
Name	Walls 7:Walls 2:2172782:Walls 2:2172782 : Walls 7:Walls 2:2172782:291239
ObjectPlacement	IfcLocalPlacement #171269
ObjectType	Walls 7:Walls 2:2172782:Walls 2:2172782 : Walls 7:Walls 2:2172782
OwnerHistory	IfcOwnerHistory #41
Representation	IfcProductDefinitionShape #171541
Tag	291239
Type	IfcWall
IsDefinedBy	

Figure 38 – BIM Models viewer – Data access

The IFC Model could save and define relationships between items according to their IFC Schema definitions. If is defined a relationship between another IFC node, this could be shown as hyperlink.

IsTypedBy (∞)	IfcRelDefinesByType #32515
---------------	----------------------------

Clicking on the link, user can see the properties of the related node in the property view. This feature could be used to navigate through the IfcClasses and relationships between nodes, without leaving the interface or selecting items on the 3D view.

If user wants to come back to the previous data, they can use the View Prev button.

#### 4.6.5.4 The hierarchy view

The hierarchy view mode can be activated by clicking on Tree button on the toolbar.

#### 4.6.5.5 Hierarchy mode

The hierarchy view shows a hierarchical tree of items that describe the 3D Model. The hierarchy is defined in the IFC Schema and starts from the IfcProject as root. By default, the IFC file has a relationship that starts from IfcProject and then proceed in IfcSite, IfcBuilding.

IfcBuilding has IfcBuildingStorey as child, and then the rest of elements hosted in the BuildingStoreys.

This kind of visualization help the user to navigate through IFC Hierarchy to select items. After the selection, the item is shown on the BIM Viewer and zoomed in (if Auto Zoom option on the toolbar is enabled).



Figure 39 – BIM Models viewer – Hierarchy & Zones Access

#### 4.6.5.6 Zones and Spaces Mode

If the IFC files had Zones and Spaces defined, they could be shown in the Zones tab on top of the panel. Zones mode shows a hierarchy view that start from the root IfcProject, then IfcZone and IfcSpace. If no zones or spaces are defined, then the tree view show only the root IfcProject.

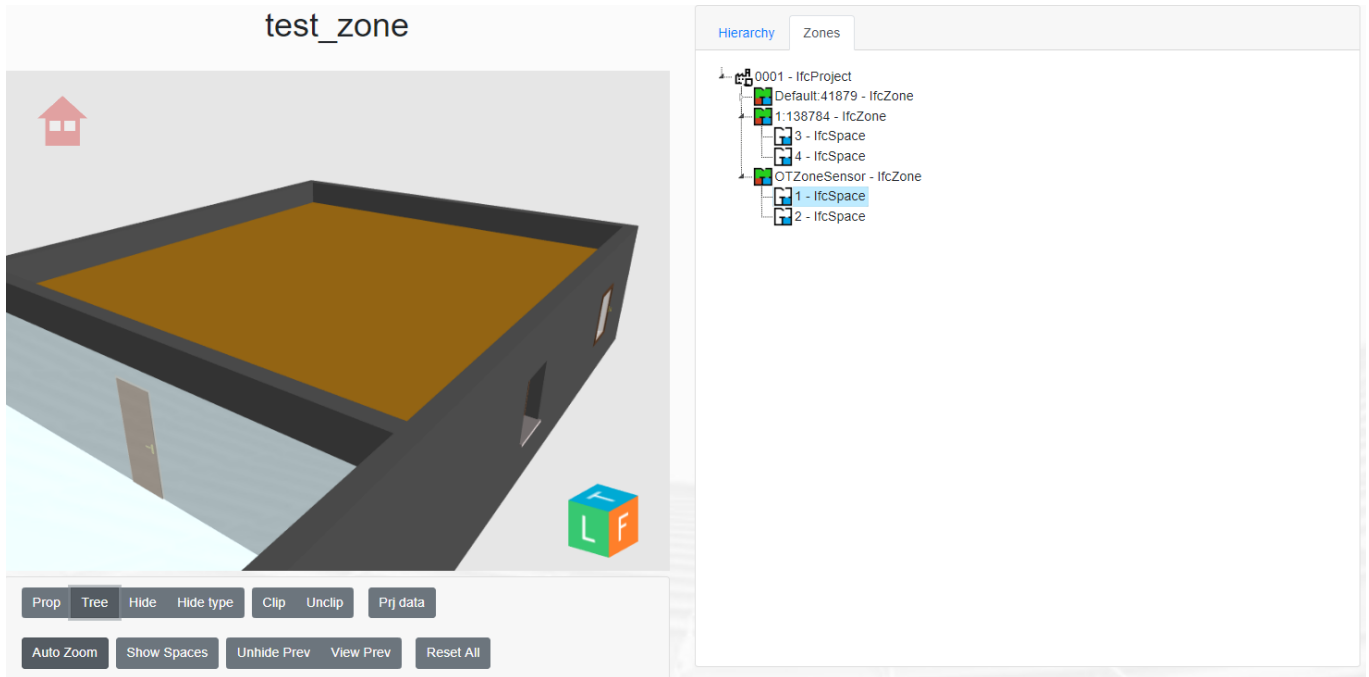


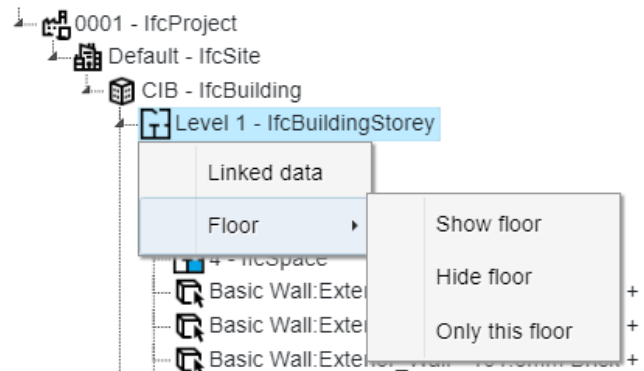
Figure 40 – BIM Models viewer – Zones

#### 4.6.5.7 Shortcuts available in the Hierarchy Mode

After item selection, the user can right click on the item to show a contextual menu with some shortcuts. All items can be mapped to other linked data objects and resources and if they are a particular kind of IfcClasses could be associated other features.

#### 4.6.5.8 Floor Visibility shortcut

If the selected item is an IfcBuildingStorey, selecting the Floor menu, appears three options to control the visibility of the storey selected.



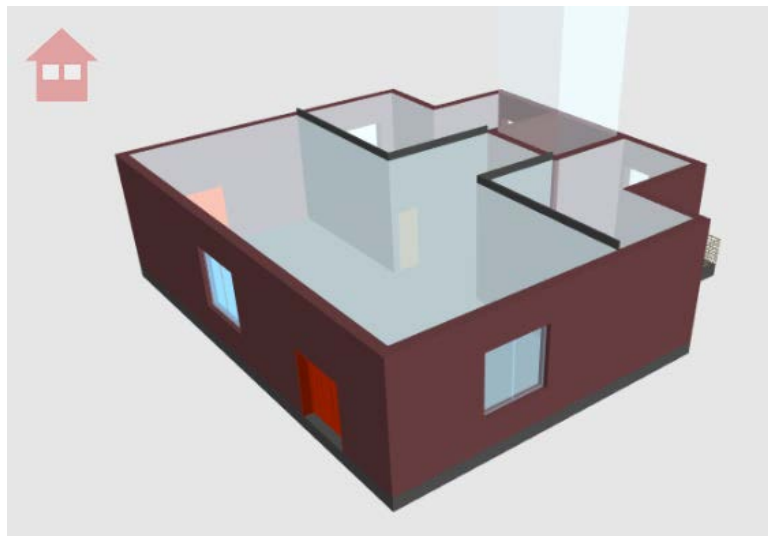
**Figure 41 – BIM Models viewer – Floors management**

Show floor, show all the items related to this IfcBuildingStorey

Hide floor, hide all the items related to this IfcBuildingStorey

Only this floor, show in 3D Viewer only the items related to this IfcBuildingStorey. Other items will be hidden.

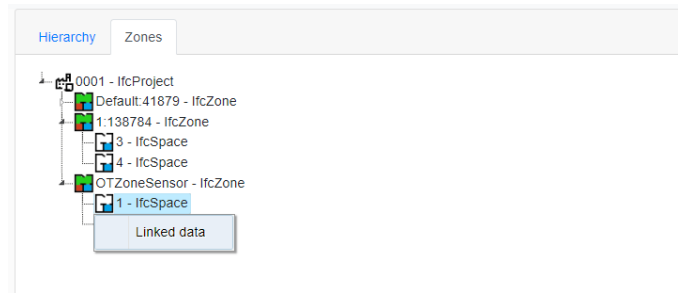
Only this floor option:



**Figure 42 – BIM Models viewer – Floors management result**

**4.6.5.9 Linked Data shortcut**

Items can be mapped with linked data resources directly from the BIM viewer using a shortcut command. After item selection, the user can right click and select the Linked Data feature on the menu.



**Figure 43 – BIM Models viewer – object Linked data access**

This shortcut opens a new window panel that shows all Linked Data resources associated to the item. The Linked Data is shown as triples of [ Subject (column S) , Predicate (column P) and Object (column O) ] retrieved by a SPARQL query performed on the fly.



**Figure 44 – BIM Models viewer – object Linked data access result**

User can add a new resource (Add resource button) or add a new linked data object (Extend Data button), using the buttons on the top.

Add Resource create a new Resource using the wizard described in 4.6.3.1.

Extend Data create a new Linked Data object using the wizard described in 4.6.3.4

If the item selected is an IfcSpace or an IfcZone the new window panel shows all Linked Data resources grouped in:

Occupants: list all occupants associated to this Space or Zone;

Sensors: list all sensors inside this Space or Zone;

Equipments: list all equipments inside this Space or Zone;

3Dscan survey: list all 3D Scan Surveys done in this Space or Zone;

Others: list all other kind of data associated in this Space or Zone.

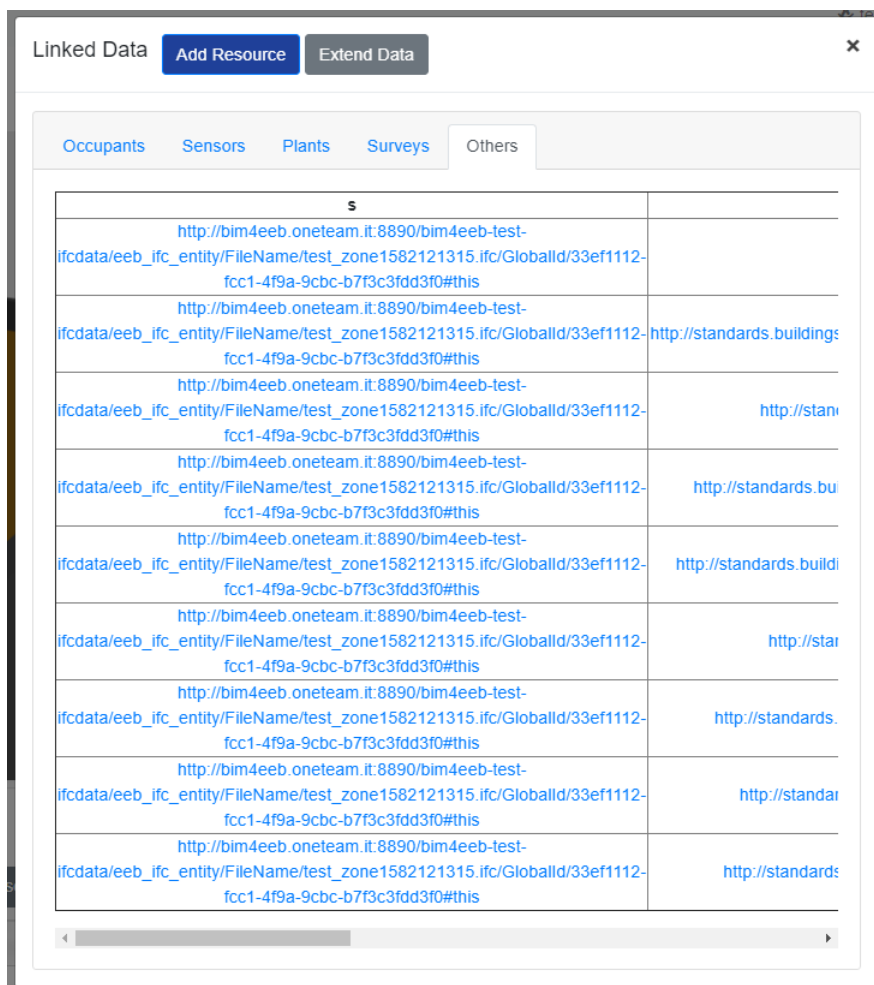


Figure 45 – BIM Models viewer – object Linked data

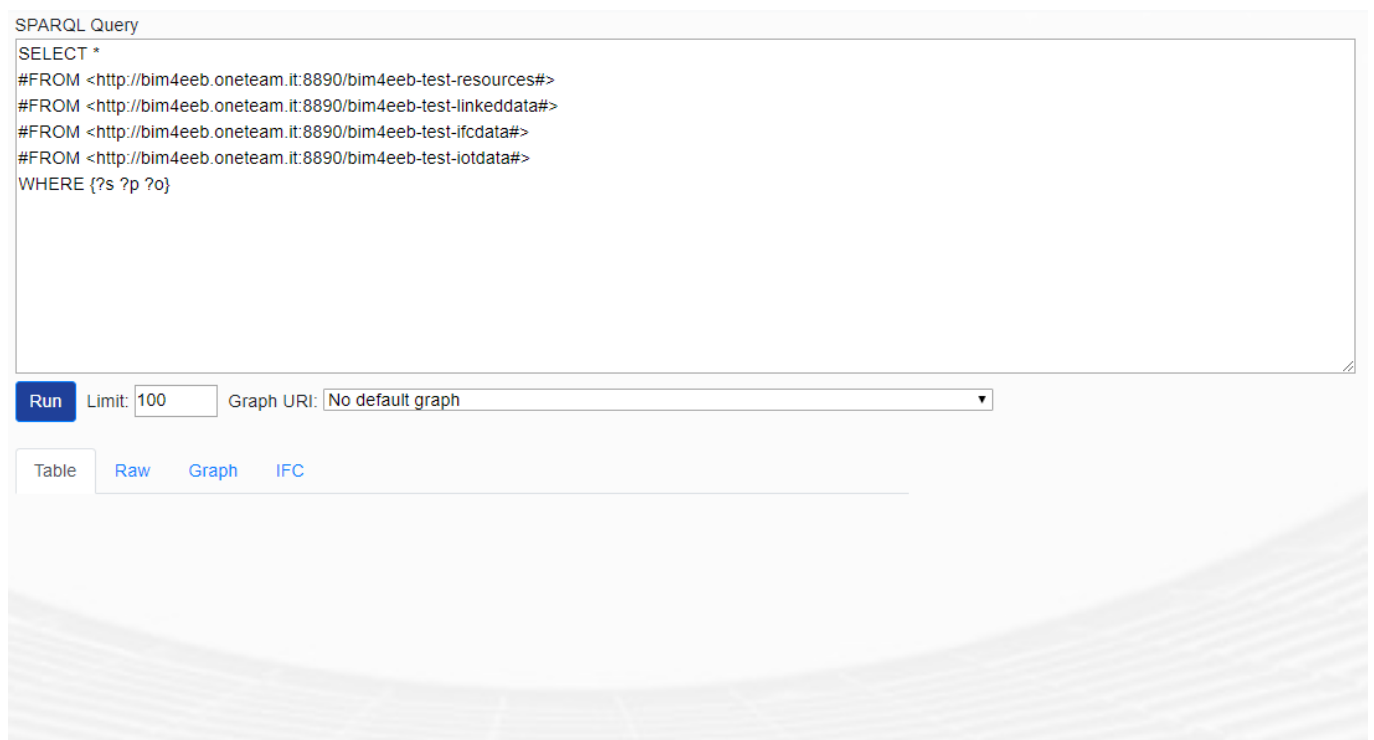
## 4.6.6 SPARQL Endpoint

The SPARQL Endpoint button give access to the BIM Management System Virtuoso SPARQL Endpoint and to the functionalities to import RDF (Resource Description Framework) files into the BIM Management System Virtuoso Database. Only specific user roles can access to this route.

### 4.6.6.1 SPARQL Endpoint

The SPARQL Endpoint page contains a text area where can be written the SPARQL Query, and an area where will be shown the query results.

The text area allow copy and paste of text and follow the SPARQL Language Syntax. By default, the text area could contain an example SPARQL Query that could be used to test the endpoint.



SPARQL Query

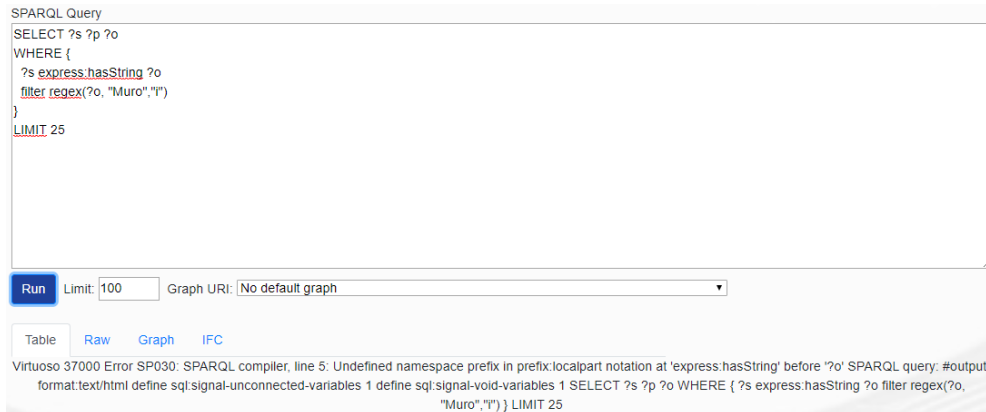
```
SELECT *  
#FROM <http://bim4eeb.oneteam.it:8890/bim4eeb-test-resources#>  
#FROM <http://bim4eeb.oneteam.it:8890/bim4eeb-test-linkeddata#>  
#FROM <http://bim4eeb.oneteam.it:8890/bim4eeb-test-ifcdata#>  
#FROM <http://bim4eeb.oneteam.it:8890/bim4eeb-test-iotdata#>  
WHERE {?s ?p ?o}
```

Run Limit: 100 Graph URI: No default graph

Table Raw Graph IFC

Figure 46 – SPARQL Interface

If a query is malformed or contains errors, they will be shown in the results area.



**Figure 47 – SPARQL sample query**

At the bottom, on the left side of the text area, there are a Run button to run the SPARQL Query, Limit field, and a Graph URI selector:

**Run button:** click to run the SPARQL Query inputted in the text area; some queries could take time to give results depending on their complexity;

**Limit field:** insert the maximum number of query results to show; by default, the queries are limited to 100 results. More results can take time depending on the complexity of the queries;

**Graph URI selector:** can be used to select from which graph data will be retrieved



**Figure 48 – SPARQL sample query**

The result area shown the query results. Four tabs can be selected to change how results are shown:



**Table view:** Results are shown in a table view depending on the number of variables declared in the SPARQL queries. By default, the results are shown in tabular view format.



s	p	o
<a href="http://www.openlinksw.com/virtrdf-data-formats#default-iid">http://www.openlinksw.com/virtrdf-data-formats#default-iid</a>	<a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">http://www.w3.org/1999/02/22-rdf-syntax-ns#type</a>	<a href="http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat">http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat</a>
<a href="http://www.openlinksw.com/virtrdf-data-formats#default-iid-nullable">http://www.openlinksw.com/virtrdf-data-formats#default-iid-nullable</a>	<a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">http://www.w3.org/1999/02/22-rdf-syntax-ns#type</a>	<a href="http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat">http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat</a>
<a href="http://www.openlinksw.com/virtrdf-data-formats#default-iid-nonblank">http://www.openlinksw.com/virtrdf-data-formats#default-iid-nonblank</a>	<a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">http://www.w3.org/1999/02/22-rdf-syntax-ns#type</a>	<a href="http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat">http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat</a>
<a href="http://www.openlinksw.com/virtrdf-data-formats#default-iid-nonblank-nullable">http://www.openlinksw.com/virtrdf-data-formats#default-iid-nonblank-nullable</a>	<a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">http://www.w3.org/1999/02/22-rdf-syntax-ns#type</a>	<a href="http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat">http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat</a>
<a href="http://www.openlinksw.com/virtrdf-data-formats#default">http://www.openlinksw.com/virtrdf-data-formats#default</a>	<a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">http://www.w3.org/1999/02/22-rdf-syntax-ns#type</a>	<a href="http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat">http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat</a>
<a href="http://www.openlinksw.com/virtrdf-data-formats#default-nullable">http://www.openlinksw.com/virtrdf-data-formats#default-nullable</a>	<a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">http://www.w3.org/1999/02/22-rdf-syntax-ns#type</a>	<a href="http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat">http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat</a>
<a href="http://www.openlinksw.com/virtrdf-data-formats#sql-varchar">http://www.openlinksw.com/virtrdf-data-formats#sql-varchar</a>	<a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">http://www.w3.org/1999/02/22-rdf-syntax-ns#type</a>	<a href="http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat">http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat</a>
<a href="http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-nullable">http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-nullable</a>	<a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">http://www.w3.org/1999/02/22-rdf-syntax-ns#type</a>	<a href="http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat">http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat</a>
<a href="http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-dt">http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-dt</a>	<a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">http://www.w3.org/1999/02/22-rdf-syntax-ns#type</a>	<a href="http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat">http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat</a>
<a href="http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-dt-nullable">http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-dt-nullable</a>	<a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">http://www.w3.org/1999/02/22-rdf-syntax-ns#type</a>	<a href="http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat">http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat</a>
<a href="http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-fixeddt-x-stub">http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-fixeddt-x-stub</a>	<a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">http://www.w3.org/1999/02/22-rdf-syntax-ns#type</a>	<a href="http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat">http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat</a>
<a href="http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-fixeddt-x-stub-nullable">http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-fixeddt-x-stub-nullable</a>	<a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">http://www.w3.org/1999/02/22-rdf-syntax-ns#type</a>	<a href="http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat">http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat</a>

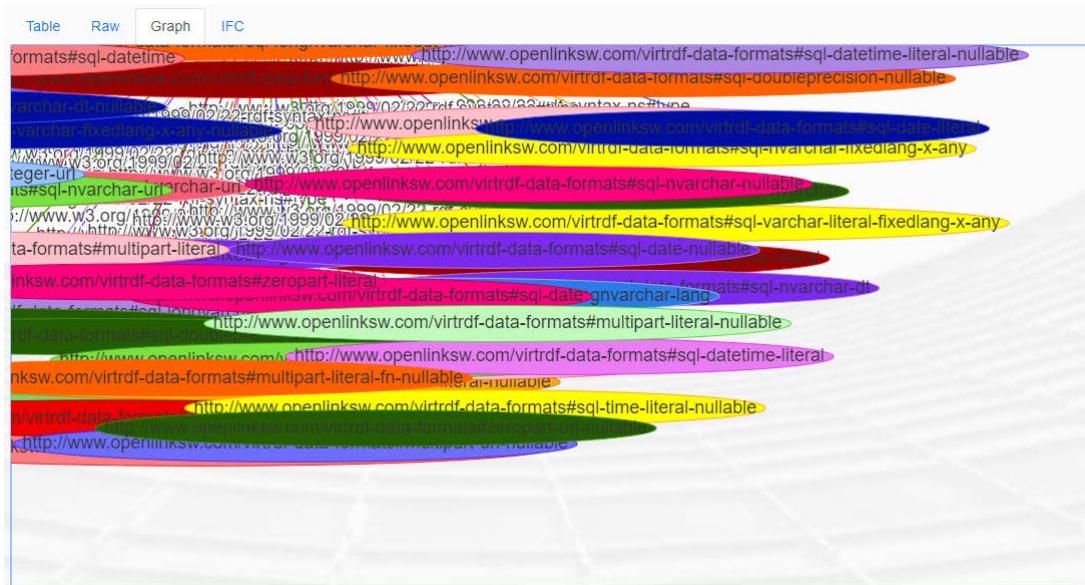
Figure 49 – SPARQL sample query result HTML

Raw view: Results are shown as raw data in JSON format. User can copy and paste raw data from this text area.

Table	Raw	Graph	IFC
<pre>[{"s":{"type":"uri","value":"http://www.openlinksw.com/virtrdf-data-formats#default-iid"},"p":{"type":"uri","value":"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},"o":{"type":"uri","value":"http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat"}}, {"s":{"type":"uri","value":"http://www.openlinksw.com/virtrdf-data-formats#default-iid-nullable"},"p":{"type":"uri","value":"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},"o":{"type":"uri","value":"http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat"}}, {"s":{"type":"uri","value":"http://www.openlinksw.com/virtrdf-data-formats#default-iid-nonblank"},"p":{"type":"uri","value":"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},"o":{"type":"uri","value":"http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat"}}, {"s":{"type":"uri","value":"http://www.openlinksw.com/virtrdf-data-formats#default-iid-nonblank-nullable"},"p":{"type":"uri","value":"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},"o":{"type":"uri","value":"http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat"}}, {"s":{"type":"uri","value":"http://www.openlinksw.com/virtrdf-data-formats#default"},"p":{"type":"uri","value":"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},"o":{"type":"uri","value":"http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat"}}, {"s":{"type":"uri","value":"http://www.openlinksw.com/virtrdf-data-formats#default-nullable"},"p":{"type":"uri","value":"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},"o":{"type":"uri","value":"http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat"}}, {"s":{"type":"uri","value":"http://www.openlinksw.com/virtrdf-data-formats#sql-varchar"},"p":{"type":"uri","value":"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},"o":{"type":"uri","value":"http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat"}}, {"s":{"type":"uri","value":"http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-nullable"},"p":{"type":"uri","value":"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},"o":{"type":"uri","value":"http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat"}}, {"s":{"type":"uri","value":"http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-dt"},"p":{"type":"uri","value":"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},"o":{"type":"uri","value":"http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat"}}, {"s":{"type":"uri","value":"http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-dt-nullable"},"p":{"type":"uri","value":"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},"o":{"type":"uri","value":"http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat"}}, {"s":{"type":"uri","value":"http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-fixeddt-x-stub"},"p":{"type":"uri","value":"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},"o":{"type":"uri","value":"http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat"}}, {"s":{"type":"uri","value":"http://www.openlinksw.com/virtrdf-data-formats#sql-varchar-fixeddt-x-stub-nullable"},"p":{"type":"uri","value":"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},"o":{"type":"uri","value":"http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat"}}]</pre>			

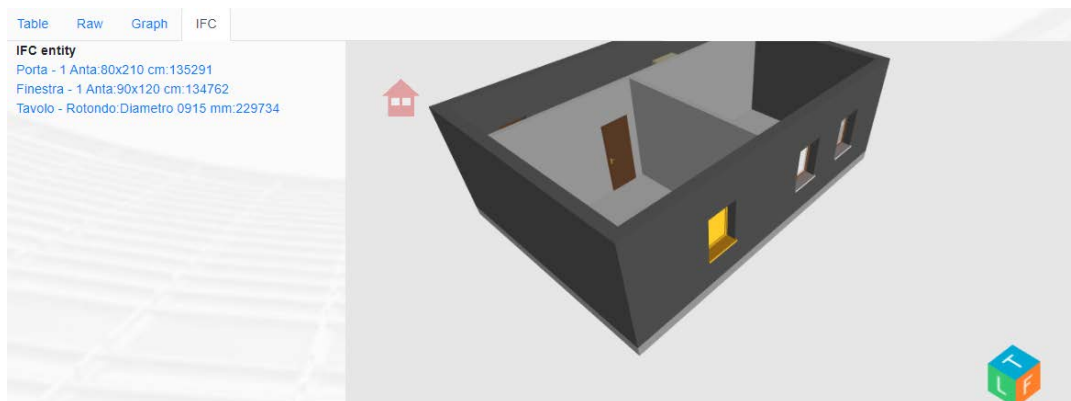
Figure 50 – SPARQL sample query result Raw

Graph view: Results are shown graphically in a graph view with coloured nodes and relationships. User can zoom, pan and move the nodes to view relationships and details. A right click on the graph view allow to save a snapshot image of the current graph.



**Figure 51 – SPARQL sample query result Graph**

IFC view: If results contain direct relationships to IFC files, in this area will be shown the IFC Items and it could be selected to see them in a 3D BIM Viewer. The BIM Viewer interface with the selected IFC model will be shown in the area.



**Figure 52 – SPARQL sample query result – ifc linked objects**

#### 4.6.6.2 Import RDF

The submenu Import RDF button allow to directly import an RDF (Resource Description Framework) file to the BIM Management System Virtuoso Database. Only specific user roles can access to this route.

User can select an RDF file from his\her computer clicking on “Choose file” button, and then click on Import button. User can also copy and paste RDF data in the main text area and then click the Import button.

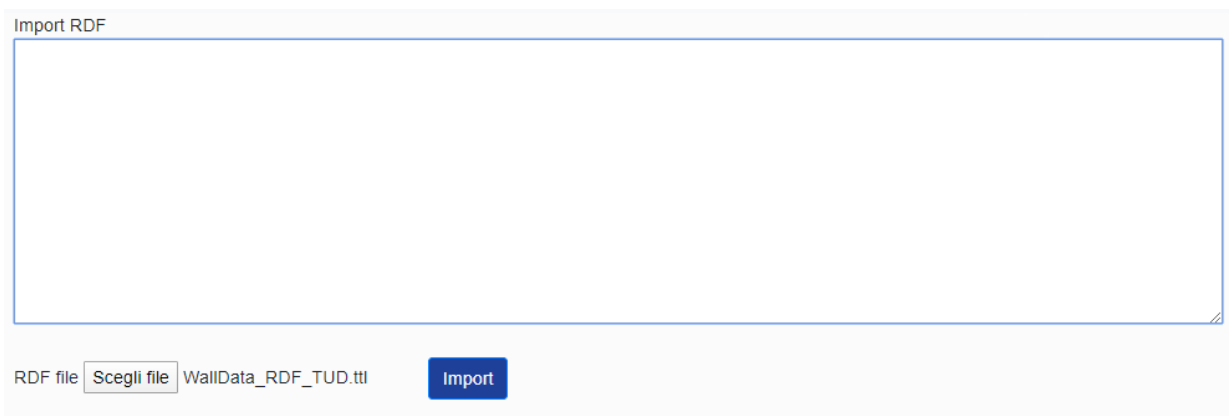


Figure 53 – RDF Import interface

#### 4.6.7 Ontology viewer

The Ontology viewer button opens a WebOWL instance in a blank page. Only specific user roles can access to this route.

The Ontology Viewer allow to graphically navigate through the ontologies loaded in the BIMMS using the VOWL framework.

The WebOWL is implementation of VOWL work licensed under CC4.0 developed by Lohmann, S., Negru, S., Haag F., Ertl, T.

The Visual notation for OWL ontologies defines a visual language for the user-oriented representation of ontologies. It provides graphical depictions for elements of the Web Ontology Language (OWL) that are combined to a force-directed graph layout visualizing the ontology.

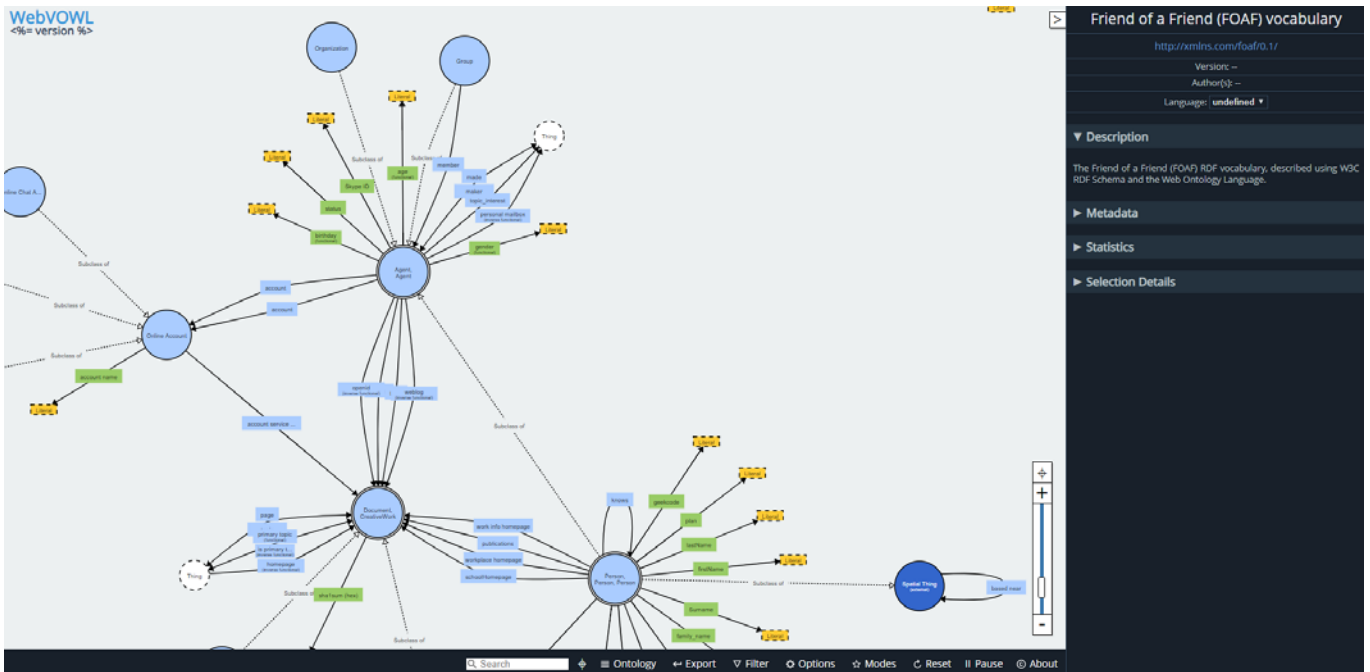


Figure 54 – Ontology viewer interface

### 4.6.8 Data streaming

The Data Streaming button give access to the functionalities related to the data streaming used in the projects. The button opens a sub menu with functionalities for Sensors, Measurements and Endpoints

Only specific user roles can access to this route.

#### 4.6.8.1 Sensors

The Sensors Button enable access to sensors list page. In this table user can access all sensors associated with active Project. This functionality will be developed to enable profiled access to data in future months, depending on Pilots progress.

Code	Type
402c1384	SmartSense Multi Sensor
402c153a	SmartSense Multi Sensor
4030c9cf	SmartSense Multi Sensor
4033d01f	SmartSense Multi Sensor
404b6dea	SmartSense Multi Sensor
404b6f7b	SmartSense Multi Sensor
404c0450	SmartSense Multi Sensor
404c0459	SmartSense Multi Sensor
4052daa7	SmartSense Multi Sensor
4054aa24	SmartSense Multi Sensor
405d7f6d	SmartSense Multi Sensor

Figure 55 – Project Sensor List

### 4.6.8.2 Measurements

The Measurements Button enable access to measurements list page. In this table user can access all measurements associated with active Project. This functionality will be developed to enable profiled access to data in future months, depending on Pilots progress.

Code	Type
402c1384	SmartSense Multi Sensor
402c153a	SmartSense Multi Sensor
4030c9cf	SmartSense Multi Sensor
4033d01f	SmartSense Multi Sensor
404b6dea	SmartSense Multi Sensor
404b9f7b	SmartSense Multi Sensor
404c0450	SmartSense Multi Sensor
404c0459	SmartSense Multi Sensor
4052daa7	SmartSense Multi Sensor
4054ae24	SmartSense Multi Sensor
405d7fdd	SmartSense Multi Sensor

Figure 56 – Project Measurements List

### 4.6.8.3 Sensors Import

The Import Sensor button allow to directly import Sensors saved in CSV file format into BIMMS. CSV values can be provided using comma (,) or semicolon (;) as separator between values. All entries must be in single rows with new line (\n) at the end of the row.

The CSV file must contain the information necessary to identify uniquely the device (deviceID), where is located the device (locationID or locationName), and the type of device (deviceType) that could identify what measure could be retrieved. The CSV file could contain other information than the minimum required in order to describe the device, but must not contain the measurements. To import measurements there is a specific functionality described in 4.6.8.4

The sensors page has a text area where, by default, is shown the minimum information necessary to import a device into BIMMS, and a description example.

Import SENSORS

SAMPLE CSV FILE

Code - sensor code  
 Location - location\space name  
 Type - sensor type

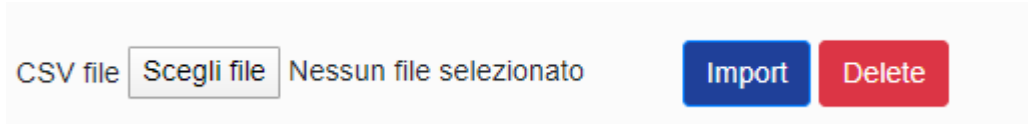
ex.

4086a0a1;Data Sensing Lab;Temperature  
 4086a336;Registration desk;Temperature  
 4090299d;Back corridor;Temperature

CSV file  Nessun file selezionato

Figure 57 – Sensors Import interface

Below the text area, there is a file selector that accept CSV file format. User can choose and select a CSV file from the computer and then upload in the BIMMS clicking on Import button. A delete button can delete the data uploaded in the BIMMS



**Figure 58 – Sensors Import interface**

In the CSV file a sample device can be defined with a row of data formatted as:

sensorCode, sensorLocation, sensorType

where

sensorCode is the deviceID

sensorLocation is the locationID\SpaceID\ZoneID name where is the device

sensorType is the deviceType that identify the type of sensor and the measurement provided.

A CSV sample can be written as follow:

```
4086a0a1;Data Sensing Lab;Temperature
```

```
4086a336;Registration desk;Temperature
```

```
4090299d;Back corridor;Temperature
```

This sample describe three devices located in three different locations, and all of them measure the temperature in these locations.

#### 4.6.8.4 Measurements Import

The Import Measurement button allow to directly import Measurement saved in CSV file format into BIMMS. CSV values can be provided using comma (,) or semicolon (;) as separator between values. All entries must be in single rows with new line (\n) at the end of the row.

The CSV file must contain the information necessary to identify uniquely the device (deviceID) that provide the measure, when the measure has been done (date), the value of measurement (Value), and the type of measure (measureType) that could identify what measure could be retrieved. The CSV file could contain other information than the minimum required in order to describe the measure. The deviceID that identify the device and the location where measure took place must be defined in Sensor functionality described in 4.6.8.4

The measurement page has a text area where, by default, is shown the minimum information necessary to import a list of measures into BIMMS, and a description example.



Import MEASUREMENTS

SAMPLE CSV FILE

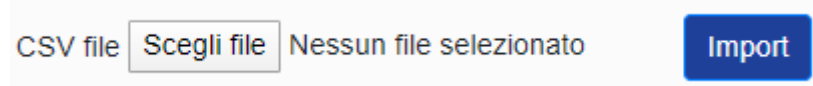
Code - sensor code  
Date - date of measurement  
Value - value of measurement  
Type - measurement type

ex.  
402c1384;23/10/2012 19:56;20.8;Temperature  
402c1384;23/10/2012 20:18;20.9;Temperature  
402c1384;23/10/2012 20:24;20.6;Temperature

CSV file  Nessun file selezionato

Figure 59 – Measurements Import interface

Below the text area, there is a file selector that accept CSV file format. User can choose and select a CSV file from the computer and then upload in the BIMMS clicking on Import button.



**Figure 60 – Measurements Import interface**

In the CSV file a sample measurement list can be defined with a row of data formatted as:

sensorCode, date, value, type

Where:

sensorCode is the deviceID

Date - date of measurement in (example, 2020-05-07T08:50:14+00:00)

Value - value of measurement

Type - measurement type

A CSV sample can be written as follow:

```
402c1384;23/10/2012 19:56;20.8;Temperature
```

```
402c1384;23/10/2012 20:18;20.9;Temperature
```

```
402c1384;23/10/2012 20:24;20.6;Temperature
```

This sample describe a measure done three times by the same device. All measures are listed and are temperatures.



### 4.6.8.5 Endpoints

The endpoints button shows the BIM4EEB Web API Help page.

The help page document the API methods used to exchange data between the BIMMS and the other tools developed in the BIM4EEB Project and will be the reference documentation page for Developers that would access to the data and functionalities.

The BIM4EEB Web API supports REST, resources are protected with authentication tokens.

The BIM4EEB Web API Help page contains the list of REST methods developed and the documentation page that describe the request and response information provided by the method. For detailed description of API methods please refer also to the Deliverable D 4.7 “API, Master End User Front End”.

#### BIM4EEB Web API Help Page

##### Introduction

REST EndPoints for BIM4EEB Project

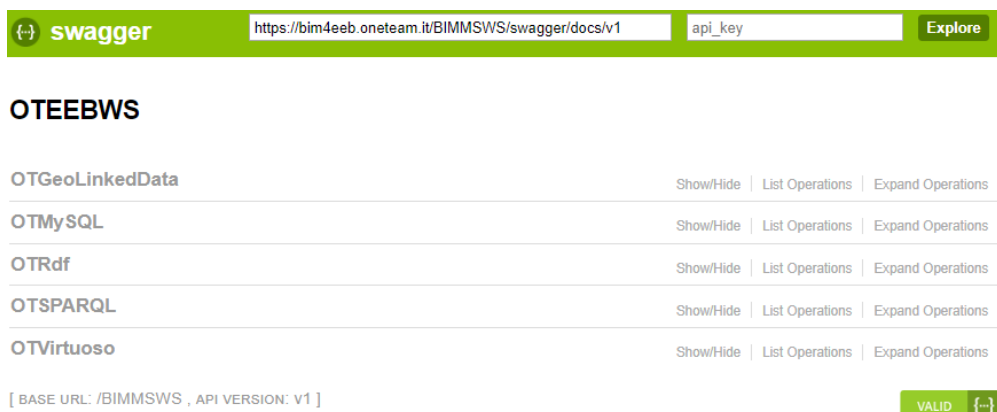
##### OTMySQL

API	Description
GET /user/getUserToken?username={username}&password={password}	Get the access token for the given user
GET /user/getUserProjects?tokenUser={tokenUser}	Get the projects for the given user token
GET /user/getUserRoles?tokenUser={tokenUser}&tokenProject={tokenProject}	Get the roles for the given project and user token
GET /user/getUserStages?tokenUser={tokenUser}&tokenProject={tokenProject}&role={role}	Get the projects stages for the given project, user token and role
GET /user/getUserApplications?tokenUser={tokenUser}	Get the applications for the given user token (for adding end user token to applications)
GET /project/getProjectIFC?tokenUser={tokenUser}&tokenProject={tokenProject}	Get the IFC files for the given project and user token
GET /project/getProjectBuildings?tokenUser={tokenUser}&tokenProject={tokenProject}	Get the buildings for the given project and user token
GET /project/getProjectGraphs?tokenUser={tokenUser}&tokenProject={tokenProject}	Get the graphs for the given project and user token
GET /ifc/getIFCFile?tokenUser={tokenUser}&tokenProject={tokenProject}&fileName={fileName}	Get the IFC file for the given file name, project and user token

**Figure 61 – BIMMS endpoints Help Page**

In the Endpoints page is also available a link to Swagger UI tool. The Swagger UI tool is an open source tool sponsored by SmartBear Software with strong support of open source software community.

This tool list, document and consume the REST Web services and was included to allow the developers to directly test the API calls for BIMMS.



**Figure 62 – BIMMS endpoints Swagger test page - 1**

## OTEEBWS

OTGeoLinkedData		Show/Hide	List Operations	Expand Operations
<b>OTMySQL</b>		Show/Hide	List Operations	Expand Operations
GET	/user/getUserToken	Get the access token for the given user		
GET	/user/getUserProjects	Get the projects for the given user token		
GET	/user/getUserRoles	Get the roles for the given project and user token		
GET	/user/getUserStages	Get the projects stages for the given project, user token and role		
GET	/user/getUserApplications	Get the applications for the given user token (for adding end user token to applications)		
GET	/project/getProjectIFC	Get the IFC files for the given project and user token		
GET	/project/getProjectBuildings	Get the buildings for the given project and user token		
GET	/project/getProjectGraphs	Get the graphs for the given project and user token		
GET	/ifc/getIFCFile	Get the IFC file for the given file name, project and user token		
GET	/ifc/getIFCHierarchy	Get the IFC hierarchy for the given file name, project and user token		
GET	/ifc/getIFCSpaceObj	Get the IFC objects in the locationID space for the given file name, project and user token		
GET	/ifc/getIFCZones	Get the IFC zones for the given file name, project and user token		
POST	/ifc/setIFCZone	Create new IFC zone for the given file name, project and user token		
PUT	/ifc/modIFCZone	Modify IFC zone for the given file name, project and user token, zone name		
DELETE	/ifc/dellFCZone	Delete IFC zone for the given file name, project and user token, zone name		

Figure 63 – BIMMS endpoints Swagger test page - 2

All REST methods are listed with their HTTP verbs, descriptions and status. User can select API group, the API call needed, and then fill the requests parameters to test the BIMMS response. The Swagger UI can be used to test specific responses for the developers needs

## OTEEBWS

OTGeoLinkedData Show/Hide List Operations Expand Operations

OTMySQL Show/Hide List Operations Expand Operations

**GET** /user/getUserToken Get the access token for the given user

Response Class (Status 200)  
OK

Model | Example Value

```
{}
```

Response Content Type application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
username	<input type="text" value="(required)"/>	Username in BIMMS	query	string
password	<input type="text" value="(required)"/>		query	string

Figure 64 – BIMMS endpoints Swagger test page - 3

### 4.6.9 API Enabled users

The API Enabled Users button is used by BIMM Administrators to enable account to use BIMMS Rest services API interface.

In this section Users are enabled and associated to a specific External Tool. Only enabled account can use BIMMS API to interact with the system

Show 50 entries Search:

User	Email	Application
	davide.madeddu@gmail.com	- Selected -
	alessandro.valra@gmail.com	- Selected -
aleksander.bartoszewski	abartoszewski@prochem.com.pl	- Selected -
anders.bratt	anders.bratt@ogi.com	- Selected -
BeatriceLanzuolo	beatrice.lanzuolo@mail.polimi.it	- Selected -
brian.oregan	brian.oregan@jerc.ie	- Selected -
Bruno Daniotti	bruno.daniotti@polimi.it	- Selected -
campagnolad	davide.campagnola@oneteam.it	- Selected -
cassanom	manuele.cassano@oneteam.it	- Selected -
chiappetti	giacopo.chiappetti@oneteam.it	BIM4Occupants
CiuSimone	simone.ciufrada@polimi.it	- Selected -
dbik	ad@iaa.com	- Selected -

Figure 65 – API Enabled user management interface

#### 4.6.10 Tools

The Tools button give access to the entire toolset developed in the BIM4EEB project by the project partners. Only specific user roles can access to this route.

The tools are listed in a table with a tool icon and a summary description.

Tools are available with a link that redirect the user to the login page of the tool, or by a download link that allow the user to install the tool as stand-alone application in his organization. All details about how to use and install the tools are available by respective developer partners.

The tools developed in the BIM4EEB Project are (the list is updated to M18):

- BIM4EEB Fast Mapping Building Toolkit developed RISE
- BIM4EEB BIMeaser tool developed by VTT
- BIM4EEB BIM4Occupants tool developed by Suite5
- BIM4EEB BIMcpd tool developed by UCC
- BIM4EEB Auteras tool developed by TUD
- BIM4EEB BIMPlanner tool developed by VTT

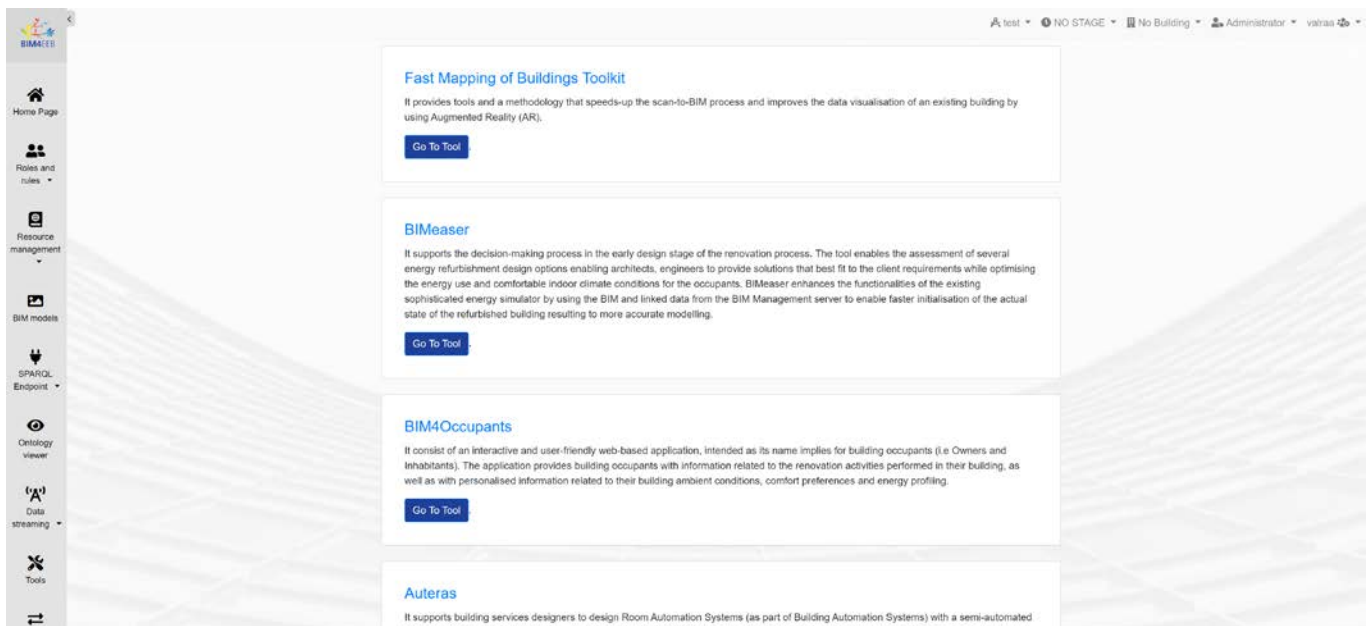


Figure 66 – Tools page

## 5 Conclusions

---

This deliverable illustrates the status of the APIs and the BIMMS front end available until M18 (June2020). Further developments are expected, and the final documentation will be updated based on the new features that will be implemented in the final toolset. The functionalities expected in the reporting period have been developed and set to allow, if necessary, the further development in the second phase.

The current BIMMS version meets all the requirements that have emerged so far; in the next months, new implementations are expected on the entire architecture of the system:

- Database and CDE: Database will be enhanced depending on new external tools and BIMMS development requirements, CDE will be completed with enhancements on users' roles, privileges mechanisms and resources workflow process.
- WEB interface: web interface will be upgraded with new functionalities development and suggested improvements on UI.
- API Rest Services: new services and methods will be developed according new functional requirements from other WPs that are developing tools.

The new implementation will be documented in the next deliverables.