FURNIT-SAVER

Smart Augmented and Virtual Reality Marketplace for Furniture Customisation

D2.2In-depth design of the main modules

Grant Agreement Number	645067
Call identifier	ICT-18-2014
Project Acronym	FURNIT-SAVER
Project Title	Smart Augmented and Virtual Reality Marketplace for Furniture Customisation
Funding Scheme	Innovation Action
Project Starting date	1st February 2015
Project Duration	14 months
Deliverable Number	D2.2
Deliverable Title	In-depth design of the main modules
Nature of Deliverable	**R**
Dissemination Level	**PU**
Due date of deliverable	M5
Actual Date of deliverable	29/07/2015
Produced by	ACS –Alessandra Giustiniani
Validated by	ASCAMM –Jesús Pablo González

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement no. 645067

Document change record

Issue Date	Version	Author	Sections affected / Change
05/05/2015	V0.1	ASCAMM	Working document for D2.1 and D2.2 circulated to ACS
29/05/2015	V0.2	ACS	Contributions provided for system architecture and modules definition.
22/05/2015	V0.3	ACS	New proposal of document structure with more detail in the modules definition. Potential content to D2.2 included.
29/05/2015	V0.4	ASCAMM	Architecture schematics provided for discussion.
2/06/2015	V0.5	ACS	Modifications added to modules descriptions based on discussions.
6/06/2015	V0.6	ASCAMM	New description of the Back-office
7/06/2015	V0.7	ACS	Modifications added to modules descriptions based on discussions.
26/06/2015	V0.8	ACS	Modification in modules descriptions. All sections affected.
07/07/2015	V1.0	ASCAMM	First complete draft
17/07/2015	V2.1	ASCAMM	Modification of all sections based on discussions.
24/07/2015	V2.2	ACS	Small modifications of all sections based on further discussions.
29/07/2015	V2.3	ASCAMM	Document revised and approved for submission.

Table of Contents

Table of Figures

1 FurnIT-SAVER project introduction

The traditional nature of the furniture industry and the limited incorporation of ICT tools have reduced the ability of SMEs in the sector to innovate and respond to the competition coming from larger companies. These specialised furniture shops and small furniture manufacturers have been unable to compete with the economies of scale advantages that larger furniture retailers can offer.

On the other hand, smaller furniture companies can offer higher levels of personalization and quality of customized goods that truly meet customers' preferences and needs which represents a potential competitive advantage over larger furniture providers. Nevertheless, as it is impossible to envisage how the furniture will look and fit into the customers home, customised furniture also bears an expensive risk if the final piece of furniture does not meet the customer's needs or does not complement other furniture. Furthermore, these customised services are predominantly provided on a face-to-face basis in local and fragmented markets which prevents small manufacturers to benefit from ecommerce growth and limit their international reach.

The FURNIT-SAVER project makes use of innovative ICT solutions based on a combination of Virtual and Augmented Reality (VR/AR) technologies, recommendation engines and ecommerce solutions, to produce a smart marketplace for furniture customisation. Customers will be able to select among an extensive furniture catalogue and properties and virtually try the selected pieces in their rooms with three very simple steps: (1) Creating an accurate 3D virtual representation of their place, (2) Trying furniture of different manufacturers in this virtual scenario and get recommendations according to their preferences of a wide range of properties and pieces, and (3) Visualizing the fit of the chosen products in their place using augmented reality.

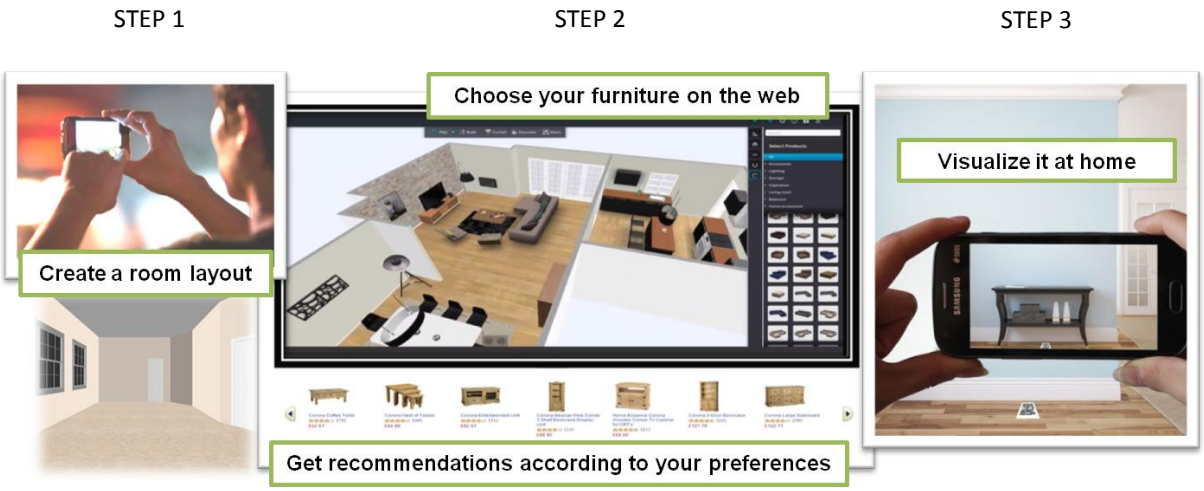

Figure 1 FurnIT-SAVER phases representation

2 Scope of the document

This is the document collecting the in-depth description of the technical modules identified in the FurnIT-SAVER system architecture; mainly, it is focused on describing the AR / VR frameworks and the recommendations engine.

It is the result of the FurnIT-SAVER consortium discussion, iterations and feedback collection about the communications channels and exchange data formats between the modules composing the platform. The leading documents used to generate the current one are, mainly:

FurnIT-SAVER working document containing the basements of the technical discussion.

- D1.1 User Requirement Document (URD)
- D1.2Application Scenarios Deliverable (APD)
- D2.1 System Architecture

3 Technological context

The FurnIT-SAVER platform makes use different technological tools such as portable devices as a room mapping tool, Virtual and Augmented Reality (VR/AR) technologies and recommendation engines to produce a smart marketplace for furniture customisation. The following diagram summarises the process in 3 steps taking into account the different stakeholders involved (buyers and retailers as users and furniture manufacturers):

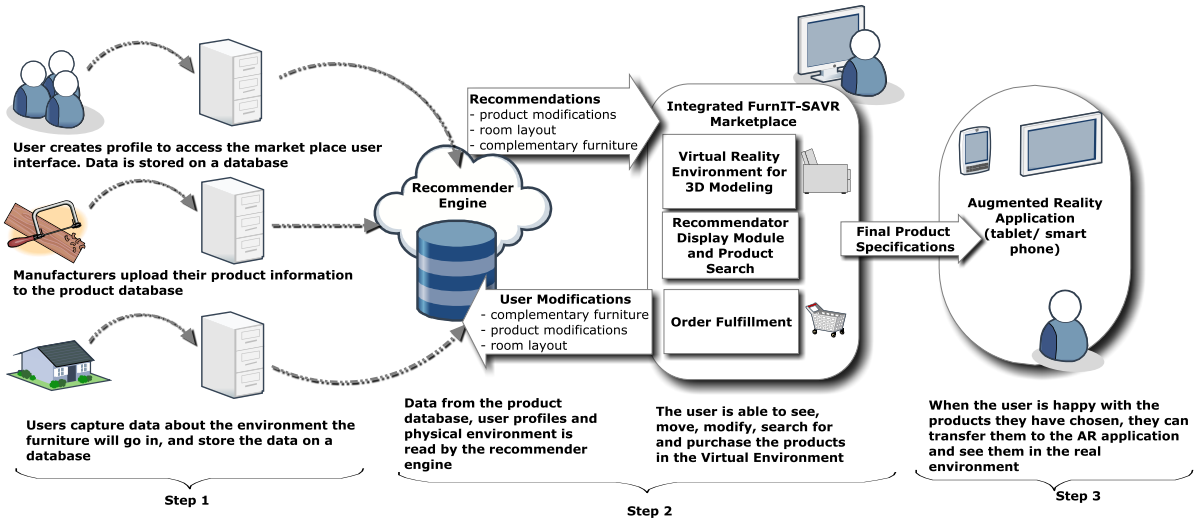

Figure 2 FurnIT-SAVER process

4 FurnIT-SAVER main modules in-depth description

As described in D2.1 System Architecture the main modules for FurnIT-SAVER platform are.

- User registration and profiling
- Layout Generation
- Furniture Catalogue management module
- Configuration of furniture and recommendation module
- Augmented Reality application

4.1 User registration and profiling

The user registration and profiling module offers the necessary interfaces to gather and store the user identification information and their preferences when required by the type of user (domestic and professional user) as described in D2.1. This module is a JavaScript component that will be used on the web portal and on the mobile application to manage user authentication. This module offers the following services and RESTFUL interfaces:

4.1.1 Creation and modification of user profile

This RESTFUL interface must create a new user representation in the user database. Several information, like user preference, will be registered using the interface with json data format. The user database will only be accessible through this interface. Data will be stored for later retrieval with no interpretation at backend level.

The furnit.backend.eurecat.org[1]/user will offer a RESTFULinterface for that purpose with the following methods.

end point	arguments	used in methods
user	name	ALL
	surname	POST/PUT
	gender	POST/PUT
	age	POST/PUT
	password	POST
	email	ALL
	notifications	POST
	binary_data	POST

	Available methods	Returns
PUT	Add new user	user identifier if success
GET	List of users	names and emails
GET<ID>	User information	user data stored
DELETE	Delete user	status success or fail
POST	Modify user data	status success or fail

[1] For illustrative purposes only. The url may change.

Examples (JSON)

Method	Send	Response

PUT	{
	"name":"John",
	"surname":"Smith",
	"email":"johns@mit.edu"
	"password":"l0'v3cr4f7"
	}
		status: OK
		
		{
		"userID"="7237812638"
		}

GET	{
	"name":"John",
	}
		status: OK
		
		{
		{
		"name":"John",
		"surname":"Smith",
		"email":"johns@mit.edu"
		},
		{
		"name":"John",
		"surname":"Badía",
		"email":"jbdia@google.org"
		},
		}

DELETE	{
	"name":"John",
	"email":"johns@mit.edu"
	"userID"="7237812638"
	}
		status OK

The user identifier is a unique identifier assigned to a single user. This identifier is not useful to the user, it is for the web server use only.

For registering initial customer questionnaire data, the endpoint furnit.backend.eurecat/<user identifier>/questionnaire

end point	arguments	used in methods
<user id>/questionnaire	label	ALL
	type	POST/PUT
	value	POST/PUT

	Available methods	Returns
PUT	Add/modify new value	status success or fail
GET	Get data	user data
DELETE	Delete data	status success or fail
POST	Modify user data	status success or fail

The field label must be a unique identifier of the value associated with the user. The type field must be descriptive categories whose logic must determine both, the kind of value and his domain. For example, we have the following type/value[2]. The list list may be extended for project convenience.

type	value
colour	three dimensional floating vector
colourname	string text from set
like	floating value between 0 and 1
child	integer
single	boolean

4.1.2 User Login

This RESTFUL interface will check the existence of a specific user and return the user session identifier (User_ID) or login errors. After a valid login, the user session can be used for retrieval or modification of user profile information in the Database.

The user login RESTFUL interface will provide the login features and return a unique user session identifier used to store, delete, modify or query about user preferences, characteristics or recommendations.

In case of success, the user login service will return both the user identifier (it is a unique identifier to be used all over the system) and one session identifier that is a temporary value used only in the time frame of the current login session.

end point	arguments	used in methods
user_login	email	POST
	password	POST

Methods		Returns
PUT	Not available	---
GET	Not available	---
DELETE	Not available	---

POST	User login	status success or fail
		
		{
		"internal user id":<value>,
		"session id":<value>
		}

[2] Example for illustrative purposes. These values may change.

The value returned as "session id" will be used when required by a user identifier argument in the RESTFUL interface, while the "internal user id" will be used as part of the url of private endpoints.

4.1.3 Manufacturer registration

The manufacturer registration RESTFUL interface provides both the capacity to store initial manufacturer information such as name, and identification tokens and catalogues. The manufacturer registration provides mechanisms to handle a session user identifier and associate it with new manufacturer information.

The manufacturer registration interface can store some binary data related to the manufacturer (like a photo). This kind of information will be stored in the database and it could be retrieved for any possible usage but not further analyzed by the recommender module. For instance, it may be interesting to gather some additional business information such as export furniture per year or per country for provider control purposes.

This RESTFUL interface will allow creating, modifying or deleting manufacturer data using the following methods:

end point	arguments	used in methods
manufacturer	name	ALL
	surname	POST/PUT
	gender	POST/PUT
	age	POST/PUT
	password	POST
	email	ALL
	size	POST/PUT
	notifications	POST
	binary_data	POST

	Available methods	Returns
PUT	Add new manufacturer	manufacturer identifier if success
GET	List of manufacturers	names and emails
GET<ID>	Manufacturer information	manufacturer data stored
DELETE	Delete manufacturer	status success or fail
POST	Modify manufacturer data	status success or fail

4.1.4 Manufacturer login

The manufacturer login interface is similar to the user login interface and uses similar methods. The manufacturer login RESTFUL interface, providing email and password returns the session-only user identifier. Similarly to the user session id, the internal identifier is for private use of the web servers and it is not sent to the user.

end point	arguments	used in methods
manufacturer	email	POST
	password	POST

	Methods	Returns
PUT	Not available	---
GET	Not available	---
DELETE	Not available	---

POST	Manufacturer login	status success or fail
		{
		"internal manufacturer id":<value>,
		"session id":<value>
		}

4.2 Layout Generation

This module provides the functionality for the user to create the room layout. The output of this module will be a JSON file (or compatible format) that describes the room properties. The module is logically composed by a mobile application and a web-based application. Existing applications will be integrated in FurnIT-SAVER platform so that users can map the room layout and export a JSON file or a compatible file to get the 3D room layout.

As part of this module, users will be able to use a JavaScript component to refine the exported file from the mobile application, edit a new one from scratch on the web or edit pre-defined default layouts on the web. This component will run on the main web interface and will allow adding information to the layout such as doors, windows or existing furniture.

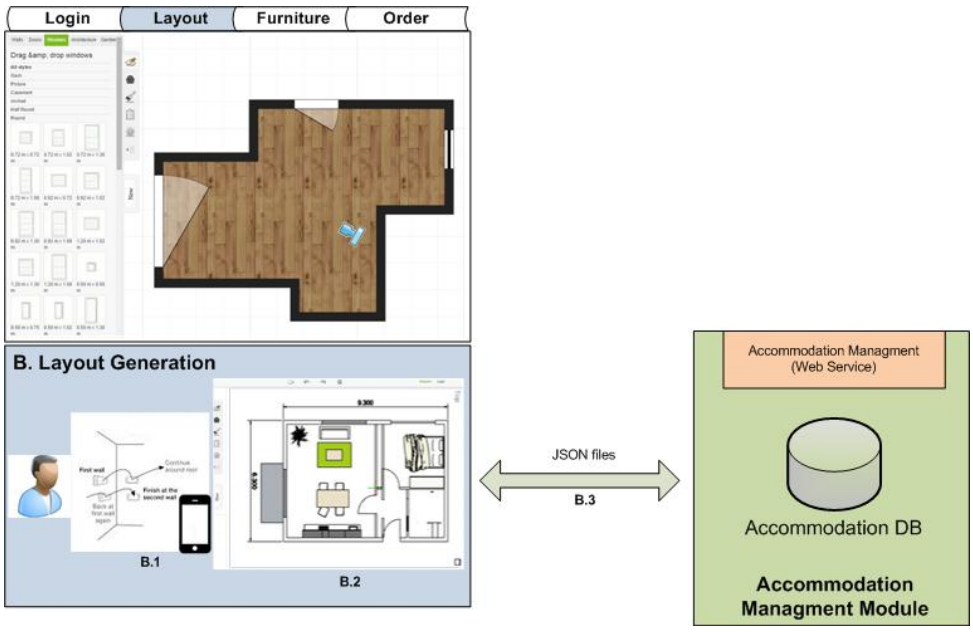

Figure 3 Layout Generation

This module will manage the creation, registration, modification and sharing of new and existing room layouts into the Accommodation Database through the following RESTFUL interfaces:

4.2.1 Creation and modification of layouts

This RESTFUL interface provides the ability to store layouts associated with a logged user (using his session identifier it should be linked to the user-id that is unique the session-Id is temporary). The layout internal structure may be binary and will be stored and will be accessed without any kind of interpretation from the backoffice. The layouts can be created (stored) or deleted in the Database, through the register layout interface. The layout application should compute parameter relevant for the recommendation, they are: total area of the room, current location of the user, etc. according to possible future needs of the recommender. Those parameters are stored using this interface. It will use the following end point and methods.

end point	arguments	used in methods
<user id>/layout	name	PUT/POST/DELETE/GET
	description	PUT/POST/GET
	status	PUT/POST/GET
	area	PUT/POST/GET
	binary_data	PUT/POST/GET
	layoutid	POST/GET
	is_owner	GET
	gps_position	PUT/POST/GET

The layout identifier will be generated automatically when the layout is registered the first time. The user becomes by default the owner of the layout.

4.2.2 Sharing layouts

Layouts will be owned by one user but can be accessed by other users. A single user may have associated layouts as owner or granted access, on the other way, layouts will only have one owner but can have many granted users (for example a salesperson that shares the created layout with clients).

The access for a layout is implicitly granted for the owner and the backoffice provides an interface to the owner for granting access to other users through the following end point, where the email refers to the user that receives granted access:

end point	arguments	used in methods
<user id>/sharelayout	layoutid	PUT/POST/DELETE/GET
	email	PUT/POST/GET
	permissions	PUT/POST/GET

4.3 Furniture Catalogue management module

This module is a JavaScript component that will be used on the web portal to fill the furniture metadata and populate the furniture catalogue database. This module includes a set of guidelines and tools to obtain a useful 3D model ready to be used in the Virtual and Augmented Reality modules.

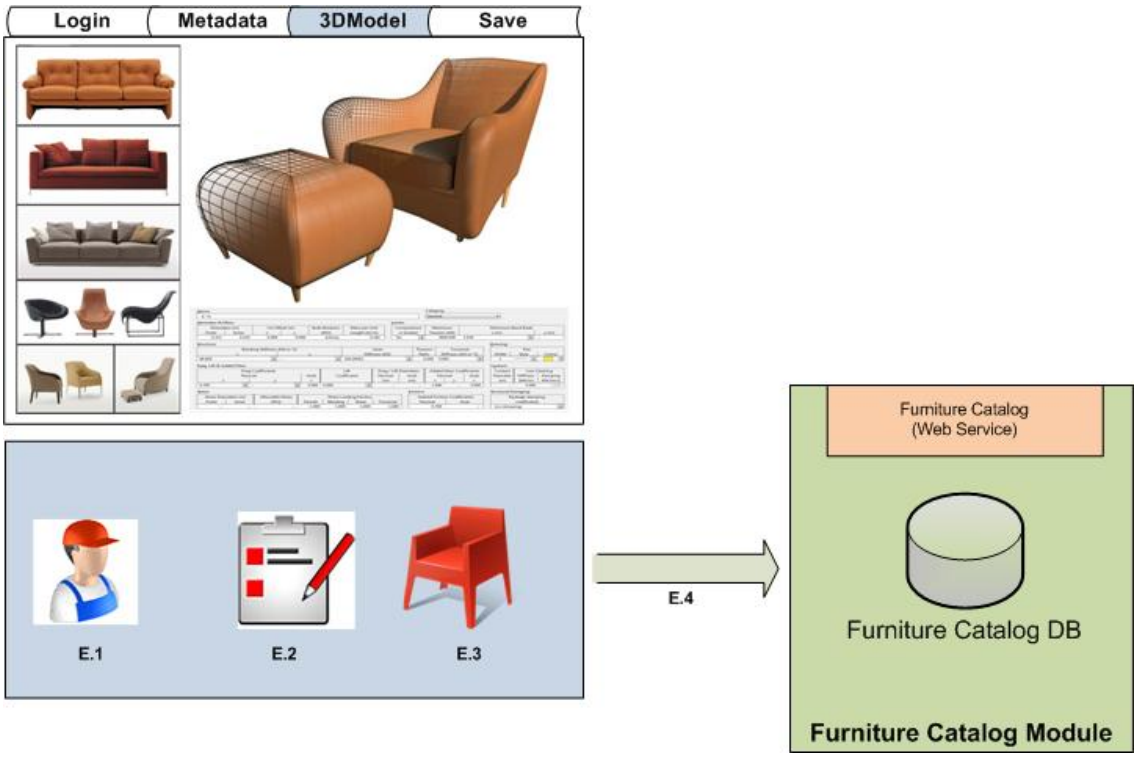

Figure 4 Furniture Catalogue

The metadata information, stored with the furniture, has to be useful for the recommended engine so the definition of fields will be related with the queries and the algorithms implemented by the recommended engine. The furniture metadata will include at least the following parameters:

- Name
- Collection name
- Thumbnail image reference
- Style
- Colour/s
- Material/s
- Dimensions: LxHxW
- Price

How to obtain 3D furniture models

The registration of furniture models is of paramount importance for the success of FurnIT-SAVER platform, therefore this module has to guarantee in an easy way that manufacturers, and furniture providers in general, are able to introduce new furniture. There are different alternative how this can happen, defined hereunder, but all will have to follow these guidelines:

- The provided furniture data file will have to be compatible with 3D formats conversion and as a general rule compatible with Open Collada format.
- In case the provided file is already a 3D models, it will have to be composed only of polygons (no nurbs or parametric geometry can be accepted).
- The file will have to contain texture information for all parts of the furniture. The supported texture file formats are png, jpeg or similar.
- The textures will have to be applied without parametric functions.

The following alternatives are considered for manufacturers and furniture providers to be able to provide 3D furniture models:

- Direct provision of a 3D format file: Some providers already work with 3D format files or have in-house capabilities to produce them. Others may decide to subcontract a service to render their 2D files in a 3D format. This will shorten the process to convert the provided 3D files to real-time ready format useful for the Virtual Reality environment. The conversion procedure would take into account the best possible graphical quality for the real-time environment. One of the main aspects will be to support high quality resolution of texture and a realistic lighting.
- By 3D scanner: This alternative requires that manufacturers have access to 3D scanning-enabled equipment, which will not be the case in most of cases. Alternatively there are some low cost applications such as mobile applications or open source software that can obtain 3D models base on camera images. Nevertheless, the current quality of the final result of these methods is not sufficient for FurnIT-SAVER. However, the platform will be compatible with this type of registration method to extend the possible ways of using the FurnIT-SAVER platform.

This module will implement the following RESTFUL interface. In general, the interface includes PUT/POST/DELETE/GET methods to store the metadata and 3D model file reference of the furniture on the *Furniture Database*. The metadata will be used for query and by the recommender engine and the 3D model will be used by the Virtual and Augmented Reality modules.

4.3.1 Registration of Furniture Catalogue

This module will offer manufacturers the functionality to register furniture collections, *catalogues*, and associate to them individual furniture through a RESTFUL interface. The furniture collections and individual furniture pieces or accessories will be defined by a set of parameters or *attributes*, and design information through which a 3D model for the Virtual Reality environment should be obtained. The attributes will be either character, integer of binary data (using python syntax and replacing spaces by underscores in attribute names. Binary attributes will be stored and recovered from the database but not interpreted by the backoffice system. The RESTFUL interface will provide methods to query, modify and delete catalogues, furniture and their attributes.

The registration of furniture catalogues comprises two stages: The creation/modification of catalogues and the maintenance of individual furniture pieces and accessories.

Creation and modification of catalogues

This interface provides methods to create, list, modify and delete catalogues. The modification of catalogues in this interface refers only to catalogue attributes, for individual furniture, refer to the next interface.

end point	arguments	used in methods
<internal manufacturer id>/catalog	name	PUT/POST/DELETE/GET
	description	PUT/POST/GET
	dateBegin	PUT/POST/GET
	dateEnd	PUT/POST/GET
	status	PUT/POST/GET
	catalogueid	POST/GET

	Methods	Returns

PUT	Creation of catalogue	status : success or fail
		if success, returns the catalogueiduuid
		for example,
		
		{
		"catalogueid":"62562"
		}

GET	Query catalogues	status: success of fail If success, returns the list of catalogues matching specified fields. example: { { "name":"WOODUP",

Methods		Returns

		"description":"Well tasted, high quality, crafted european wood furniture",
		"dateBegin":"10/10/2015",
		"dateEnd":"01/01/2016",
		"status":"active reservation",
		"catalogueid":"62562"
		},
		{
		"name":"WORLDUP",
		"description":"Well tasted, high quality, historical furniture restored by artisan",
		"dateBegin":"10/10/2015",
		"dateEnd":"01/01/2016",
		"status":"active reservation",
		"catalogueid":"16736"
		}
		}

DELETE	Delete catalogue	status: success of fail
POST	Modify catalogue	Status: success or fail

Creation and modification of individual furniture: Catalogue maintenance

Furniture pieces or accessories may be registered, modified or deleted from a catalogue using its identifier and the following methods.

end point	arguments	used in methods
<catalogueid>/furniture	name	PUT/POST/DELETE/GET
	description	PUT/POST/GET
	dateBegin	PUT/POST/GET
	dateEnd	PUT/POST/GET
	status	PUT/POST/GET
	price	PUT/POST/GET
	color	PUT/POST/GET
	material	PUT/POST/GET
	furnitureid	POST/GET

4.4 Configuration of furniture and recommendation module

This module refers to the main web interface of FurnIT-SAVER platform composed mainly for three subframes: The Virtual Reality environment where the users' 3D layout will be available and where the 3D furniture models can be placed; the search module providing access to the whole available catalogue; and the Recommender module which will provide furniture recommendations based on the user profile and a functional interface so that the user can introduce new parameters about his/her profile such as marital status, number of children or other relevant data. Additionally the interface will offer a set of menus to configure the interface.

Figure 5 Graphical mock up of FurnIT-SAVER main user interface

This module is a JavaScript component with the support of WebGL technology for the real-time graphical representation. This module will use the ThreeJS software library (http://threejs.org/). The module will offer functionalities such as:

- Rotating the scene;
- Zooming in/out the scene;
- Changing the light shading;
- First person view, also full screen, and navigation capabilities;
- Changing floor and walls colours and texture;
- Drag and drop furniture from the search module and the recommender module;
- Moving the location and rotation the furniture within the 3D room layout;
- Removing a placed furniture;

This module will use the following RESTFUL interfaces:

4.4.1 Selecting furniture from the catalogue

This submodule is a JavaScript component that will be used on the web portal to retrieve the useful furniture from the catalogue. Individual furniture pieces will be queried by any of their attributes according to the filtered selection of the user (excluding binary attributes) and available to be uploaded in the Virtual Reality environment.

4.4.2 Recommendation of furniture

This is a JavaScript component that will be integrated in the main web interface to show the recommended furniture. The recommendation engine will be accessible through a RESTFUL interface providing the following functionalities:

Tracking user activity for recommendation

The user activity tracking RESTFUL interface will provide methods for registering activity associated with the usage of the platform along his user session. This kind of information may include user choices, accepted or rejected suggestions, searchers and other relevant

traceable data. The interface will provide methods for registering and querying this type of data. Not all data may be used in the recommendation engine at the same time.

end point	arguments	used in methods
<user id>**/tracks**	date	PUT/POST/DELETE/GET
	key	PUT/POST/GET
	value	PUT/POST/GET

Query recommender

The interface will provide access to the recommender engine for querying recommendations for logged users through his session identifiers. The recommender engine will evaluate the user data experience and will find a set of furniture for recommendation. The recommendation engine will be queried with filters based on attributes indicated by the user. The filters will be passed through the RESTFUL interface as arguments containing logical expressions.

end point	arguments	used in methods
<user id>**/query**	distributionid	PUT/POST/GET
	filter	GET

In general associated binary attributes like texture or colours will be not interpreted by the recommender although it can be stored and edited. The furniture association will be interpreted by the recommender as a binary choice, with not interpretation about position or orientation.

4.4.3 Creating, modifying and sharing accommodations

An accommodation is defined by the combination of a room layout plus the distribution of a set of furniture and their attributes in the layout. An accommodation is stored in the Accomodation Database as a set of metadata and a JSON file that actually contains the 3D room representation.

Creating and saving accommodations

This type of objects is stored in the *Accommodation Database*by means of an available RESTFUL interface defined as a JSON file with the following attributes:

- User ID
- Room layout ID
- AR marker location
- For each furniture piece or accessory:
  - Furniture ID
  - Selected values for the attributes (colour, texture,etc.)
  - Position of the item in the layout (translation and rotation)

This RESTFUL interface will allow registering a new Accommodation for a User_ID.

end point	arguments	used in methods
Accommodation	date	PUT/POST/DELETE/GET
	userID	PUT/POST/GET
	layoutID	PUT/POST/GET
	roomTypeID	PUT/POST/GET
	StyleID	PUT/POST/GET
	ARMarkerRepresentationID	PUT/POST/GET

Modifying accommodations

This RESTFUL interface will provide methods to edit accommodations that mainly refer to changing the list of individual furniture items associated to a specific layout and their attributes. Since it will be possible to handle several furniture distributions for a specific layout, each distribution will be handled as a separate argument for the accommodation:

end point	arguments	used in methods
<acommodationtid>/distribution	name	PUT/POST/DELETE/GET
	comment	PUT/POST/GET
	distributionid	DELETE/GET

Then, oncethe distribution is created (PUT) in the accommodation with a unique name, it can be commented (POST) or selected by name (GET) using the *distributionid* in order to modify their individual furniture items.

end point	arguments	used in methods
<distributionid>**/individuals**	furnitureid	PUT/POST/DELETE/GET
	position	PUT/POST/GET
	orientation	PUT/POST/GET
	comments	PUT/POST/GET

Sharing accommodations

Accommodation information, as layouts, can be shared by the users. The owner will have the possibility to create public links to the accommodation so that another user can see and navigate in an accommodation, taking advantage that the only need to visualize it is a web browser (with WebGL support). This enables to disseminate very easily accommodations createdwith the platform.The access for an accommodation is implicitly granted for the owner and the backoffice provides an interface to the owner for granting access to others through the following end point, where the email refers to the user that receives granted access:

end point	arguments	used in methods
<user id>/shareacommodation	accommodationid	PUT/POST/DELETE/GET
	email	PUT/POST/GET
	public_url	PUT/POST/GET

4.5 Augmented Reality Visualization

This module is part of a mobile application and provides the Augmented Reality functionality that allows displaying the real environment overlapping the 3D models of the user selected furniture configuration. The AR application will try to make use of existing AR engines such Layar, Metaio or Wikitude to increase market uptake.

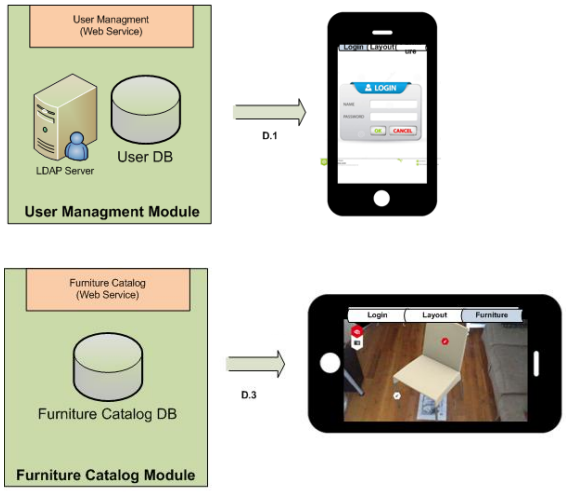

Figure 6 Augmented Reality

The user logs on the application which will check whether the user exists in the User Database, requesting for registering in case the user is not in the database. Then the application will retrieve for this specific user the list of created accommodations and then the created furniture distributions for the selected accommodation. The user will then be able to see the 3D furniture models overlapped in the real environment (prior placement of the AR markers as seen in D2.1). The access to the accommodation information by the application is supposed to be “read-only”, therefore no conflict could raise in case of simultaneous accesses. In case the possibility to edit accommodation from the application is considered simultaneous accesses have to be better managed.