Functional Specification

1. Basic information

eFurniture is an web application for people who are looking for stylish and affordable furniture. Currently it operates only in Poland but there are plans to expand for other european markets. Store offers furniture and accesories for kichen, bedroom and other rooms. User can search through countless inspirations and collections to better fit products to their style. Additionally, the user can buy furniture boards in case he wants to build the furniture himself. Store offers cutting and trimming services for all kind of products to suit the user's needs.

1.2. Project goals

- Designing and implementing an online store based on the latest technologies
- Intuitive and transparent online store that will efficiently enable the purchase of furniture
- Advanced search engine that will allow you to search for products by names, numbers and categories

1.3. Measures of success

The following are the measures of success:

- an operating online store that allows users to place orders
- automatic handling of returns and complaints
- friendly interface built on the basis of the best practices

2. Fundamental concepts

Concepts	Descriptions
CMS	System that have content management functions.
ERP System	system ERP, zainstalowany w siedzibie firmy Drogeria Edito. System wspiera realizację najważniejszych procesów w przedsiębiorstwie jak ofertowanie, obsługę zamówień, gospodarkę magazynową, fakturowanie.
Frontend	the public, client part of the system that does not contain administrative functions
Backend	the non-public, administrative part of the system, allowing the management of its settings and content, including the management of the store's order handling process.
CMS Administrator	A person who has full access to the resources of the CMS administration panel.
User	The person who visits the store to purchase a product or service.
Module	It's part of online store, corresponding to view. Examples of modules: product card, product list or basket.
Block	otherwise Widget, it's part of module/view. Examples of blocks: header, footer or filters.
Inspirations	It's part of online store, that have galleries with different usages of furniture.
Custom-made furniture	It's furniture that can be customised by user. Things that can be customised depends on furniture type.

3. Actors and roles

3.1. CMS Administrator

Role with all possible permissions

3.2. Rating administrator

Role that has access to ratings and comments for publication on the site

3.3. Promotion administrator

Role with access to configuration of discounts and promotions

3.4. Complaints administrator

Role with access only to handling returns and complaints

3.5. Product administrator

A role that has access to products, is authorized to create product sets and recommendations.

4. System architecture

4.1. Current architecture of system

4.2. Target architecture after implementation of new system

5. Business processes

5.1. Order process

5.2. Registration process

5.3. Return / complaint process

5.4. Notification process

6. Functional requirements

Functional requirements (FR) are things (mechanisms, funcionalities) that need to be implemented as a part of this project.

6.1. Products

6.1.1. [FR01] Add to favourites

Objective:

As user I want to add products to favourites so that I can come back to them and buy them later.

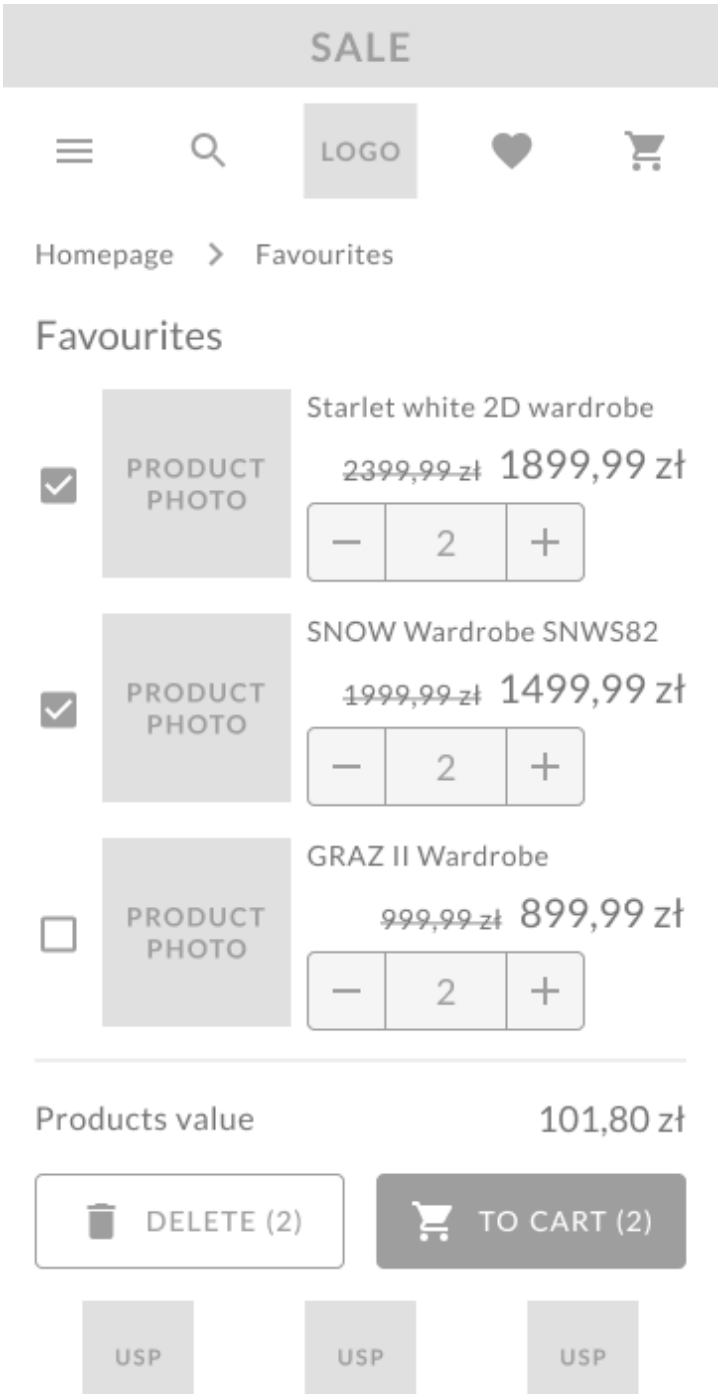
Scope:

Mechanism "add to favourites" need to work with every product from store. Users should have possibility to delete and add to basket selected products from their favourite list. Not logged users should have possibility to add products to favourites and this information should be stored in cookies mechanism for one week. Logged users should have favourites linked to their profile so everything is stored as long as user want. Everytime product is seen, user need to have possibility to add that product to favourites.

Admin panel:

No changes to the panel

Wireframe:



Wireframe "Favourites"

6.1.2. [FR02] Prices and Stock

6.1.3. [FR03] Promotions

6.1.4. [FR04]Product variants

6.2. Categories

6.3. User account

6.3.1. [FR05] Addresses

6.3.2. [FR06] Orders

6.3.3. [FR07] Password change

6.4. Basket

6.4.1. [FR08] Delivery

6.4.2. [FR09] Payment

6.5. Modules

6.5.4. [FR10] Product List

Objective:

As user I want to view product list so that I can add product to basket and buy them.

Scope:

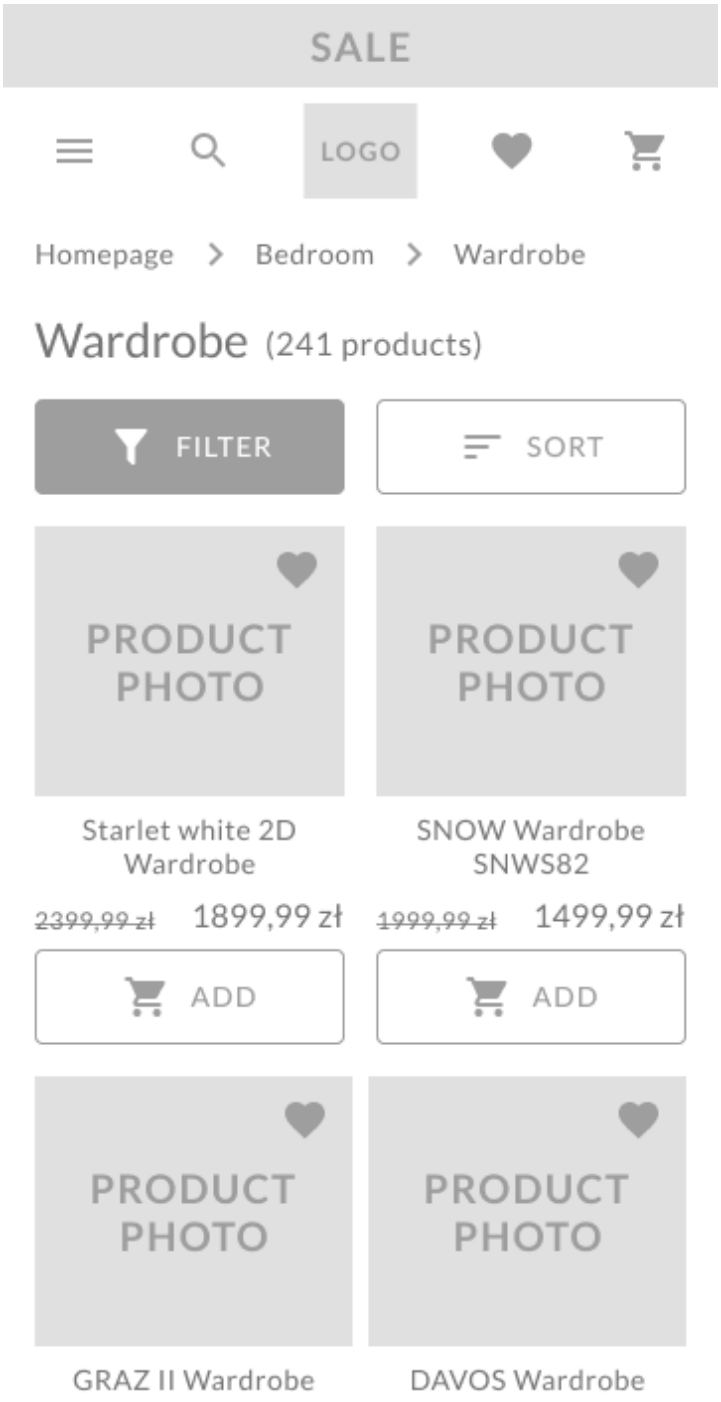
Actions that can be performed by the user on the product list:

Name	Description
Filter	User should have possibility to user filter on product list.
Sort	User should have possibility to user sort on product list.
Pagination	User should have possibility to user pagination on product list.
Add to basket	User should have possibility to add products to basket on product list.
Ask for product availability	User should have possibility to ask for product availability on product list.

Administrator panel:

- Requests for product availability should be stored in CMS database and presented in CMS panel for reading purposes.

Wireframe:



Wireframe "Product List"

6.5.1. [FR11] Contact form

6.5.3. [FR12] Blog

6.6. Blocks

6.6.2. [FR13] Product ratings

6.6.3. [FR14] Shop ratings

6.6.4. [FR15] Search engine

6.6.5. [FR16] Promotion banner

6.6.8. [FR17] Header

6.6.9. [FR18] Footer

6.6.11. [FR19] Newsletter

7. Integrations

As part of the implementation of CMS, integration with external system will be needed.

7.1. ERP System

Integration with the ERP system will be based on exchange of information about orders and products (in terms of purchase prices and inventory). All communication will be initiated by the CMS. All methods should be available through the API of the ERP System.

List of methods that should be used in two-way communication is presented below.

7.1.1. Method "sendOrder"

This method should be triggered immediately when the customer places an order in the store.

Input parameters:

Name	Type	Description
date	date	Format: DD-MM-YYYY HH:MM:SS
currency	char(3)	currency in ISO 4217 standard (np. EUR, PLN).
payment_method_id	int	payment method id: 0 - cash on delivery, 1 - bank transfer
delivery_method_id	int	delivery method id: 0 - pickup point, 1 - delivery
user_comments	varchar(510)	
user_email	varchar(150)	

Name	Type	Description
user_phone	varchar(100)	Format: XX XXXXXXXXX
user_firstname	varchar(100)	
user_lastname	varchar(100)	
user_address	varchar(100)	
user_zip_code	varchar(100)	Format: XX-XXX
user_city	varchar(100)	
invoice_name	varchar(100)	
invoice_taxnumber	varchar(100)	
invoice_address	varchar(100)	
invoice_zip_code	varchar(100)	Format: XX-XXX
invoice_city	varchar(100)	
invoice	bool	1 - user want invoice, 0 - user dont want invoice
products	array	product_id (varchar) gross_price (float) vat (int) quantity (int)

Output parameters:

Nazwa	Typ	Opis
status	varchar(30)	SUCCESS or ERROR
order_id	int	order_id will be sended only when status = SUCCESS

Example of input JSON:

```
{
  "date": "11.03.2021 15:56:53",
  "currency": "PLN",
  "payment_method_id": 2,
  "delivery_method_id": 2,
  "user_comments": "",
  "user_email": "jkowalski@email.com",
  "user_phone": "48 123123123",
  "user_firstname": "Jan",
  "user_lastname": "Kowalski",
  "user_address": "Wiejska",
  "user_zip_code": "35-897",
  "user_city": "Warszawa",
  "invoice_name": "",
  "invoice_taxnumber": "",
  "invoice_address": "",
  "invoice_zip_code": "",
  "invoice_city": "",
  "invoice": true
  "produkty":
```

```
    [
      {
        "product_id": "15632A",
        "gross_price": 22.22,
        "vat": 23,
        "quantity": 3
      },
      {
        "product_id": "1563B",
        "gross_price": 56.22,
        "vat": 23,
        "quantity": 1
      }
    ]
}
```

Example of output JSON:

```
{
  "status": "SUCCESS",
  "order_id": "16331079"
}
```

7.1.2. Method "getProduct"

7.1.3. Method "getPricesAndStock"

7.1.4. Method "getOrders"

7.1.5. Method "getOrder"

7.2. Payment

7.3. Delivery

7.3.1. InPost parcels

Integration with InPost parcels should be made with two steps:

- adding an geowidget to site
- sending information about parcel id to ERP

Geowidget

Parameters

Name	Description	Default values
instance	Override default config by config prepared for instance.Default: plAvailable: pl,uk,it	
apiEndpoint	URL to API Points which will be connected to Geowidget	pl: https://api-pl-points.easypack24.net/v1uk: https://api-uk-points.easypack24.net/v1it: https://api-it-points.easypack24.net/v1
defaultLocale	Geowidget language that will be uses in front-end.Available: pl,uk	pl: pluk: uk
locales	Languages that will be display as list in front layer. They can be dynamically changed.	pl: [‘pl’]uk: [‘uk’]
mobileSize	Screen width below which mobile version is triggered.	768
langSelection	Determines if language selection bar should be displayed.	false
filters	Determines if filters (functions) bar should be displayed.	false
addressFormat	Visible format of address	‘{street} {building_number} {post_code} {city}’
listItemFormat	Visible format of address on list	Expand source
display	Enable/disable visible type filters and search bar	Expand source
mapType	Setup map type, available to use Open Street Map or Google Maps.Available: osm, googleFor ‘google’ option map.googleKey is required.	‘osm’
searchType	Setup search type, available to use Open Street Map Nominatim or Google Maps Autocomplete.Available: osm, googleFor ‘google’ option map.googleKey is required.	‘osm’
map	Setup map details.	Expand source

Name	Description	Default values
points	Setup points detailsAvailable functions:parcel - points with functions send and collect parcel_send - points with functions send onlyparcel_collect - points with functions collect only	Expand source
customDetailsCallback	Allows to set custom callback for details actionAvailable: function(point){…},	false
customMapAndListInRow	Allow to change layout where list of point is below map. Points on list were paginated, number of point per page can be configured.	Expand source
listItemFormat	Allows to change default layout of point information on points list.	Expand source
mobileFiltersAsCheckbox	Option set to false make filter option as radio in mobile, option true as checkobx.	true
paymentFilter		False

7.3.2. InPost Courier

7.3.3. DPD

7.4. Newsletter

7.5. Baselinker

7.6. Marketing automation

7.7. Google tools

7.7.1. Google Tag Manager

7.7.2. Google Analytics

7.7.3. Google Search Console

7.7.4. Google Ads

7.7.5. Google Merchant Center

7.8. Other integrations

8. Non-functional requirements

8.1. Attributes and tags

8.1.1. [NR01] <h1> ... <h5> Tag

Scope:

Every page (including homepage) should use h1 tag as main title header. Tags from h1 to h5 should be represented in nested structure within page.

Admin panel:

There should be possiblity:

- to manually override generated h1 tag

8.1.2. [NR02] <title> Tag

8.1.3. [NR03] <description> Tag

8.1.4. [NR04] <head> Tag

8.1.5. [NR05] Canonic version

8.1.6. [NR06] Links

8.1.7. [NR07] Images

8.1.8. [NR08] PDF Files

8.2. Other elements

8.2.1. [NR09] URL Addresses

8.2.2. [NR10] 404 Error Page