DOCUMENTATION – API WEBSHOP ORESTO V1.29

DOCUMENTATION – API WEBSHOP ORESTO V1.29 1

1. SUMMARY 9

1.1 IN THIS DOCUMENT 10
1.2 API 11
1.2.1 Authentication *11*
1.2.2 Code and examples *11*
1.2.3 Use of the date/time fields *11*
1.2.4 Use of special characters in XML *12*
1.2.5 Order in XML *12*

2. COURSE AND STATUS OF THE SYNCHRONIZATION 13

3. ROADMAP 14

3.1 CREATE CLIENT AND USER 14
3.2 RETRIEVE ORDERS 14
3.3 CREATE CLIENTLOCATIONS 14
3.4 DELIVERY DATES OF A CLIENT 14
3.5 LINK A CENTRAL PRODUCT 15
3.6 CREATE (OWN) PRODUCT 15
3.7 IN BETWEEN CHECK OWN <-> CENTRAL PRODUCTS 16
3.8 CREATE CLIENTPRODUCTS 16
3.9 ORDER MULTIPLES (WITH DECIMALS) 16
3.10 PRICE PER KILOGRAM OR LITER, BUT SOLD PER PIECE 17

4. BOUNDARY TABLES 18

4.1 CATEGORIES 19
4.1.1 GET *19*
4.1.1.1 URL 19
4.1.1.2 Fields 19
4.1.2 GET (concatenated) *19*
4.1.2.1 URL 19
4.1.2.2 Parameters 19
4.1.2.3 Fields 19
4.2 BRANDS 20
4.2.1 GET *20*
4.2.1.1 URL 20
4.2.1.2 Fields 20
4.3 STORAGE CONDITIONS 21
4.3.1 GET *21*
4.3.1.1 URL 21
4.3.1.2 Fields 21
4.4 CONTENT TYPES 22
4.4.1 GET *22*
4.4.1.1 URL 22
4.4.1.2 Fields 22
4.5 UNIT SALES PRICE 23
4.5.1 GET *23*
4.5.1.1 URL 23
4.5.1.2 Fields 23
4.6 MULTIPLIER TYPES 24
4.6.1 GET *24*
4.6.1.1 URL 24
4.6.1.2 Fields 24
4.7 PRODUCERS 25
4.7.1 GET *25*
4.7.1.1 URL 25
4.7.1.2 Fields 25
4.8 SUPPLIERS 26

4.8.1 GET ... *26*
4.8.1.1 Url ... 26
4.8.1.2 Fields ... 26
4.8.2 GET (vat-number) ... *26*
4.8.2.1 Url ... 26
4.8.2.2 Parameters ... 26
4.8.2.3 Fields ... 26

5. CENTRAL PRODUCTS ... 27

5.1 PRODUCTS ... 28
5.1.1 GET ... *28*
5.1.1.1 Url ... 28
5.1.1.2 Parameters ... 28
5.1.1.3 Fields ... 28
5.2 EAN-LIST ... 29
5.2.1 GET ... *29*
5.2.1.1 Url ... 29
5.2.1.2 Parameters ... 29
5.2.1.3 Fields ... 29
5.3 PRODUCT BY EAN ... 30
5.3.1 GET ... *30*
5.3.1.1 Url ... 30
5.3.1.2 Parameters ... 30
5.3.1.3 Fields ... 30
5.4 PRODUCT BY SUPPLIER ... 31
5.4.1 GET ... *31*
5.4.1.1 Url ... 31
5.4.1.2 Parameters ... 31
5.4.1.3 Fields ... 31
1.1 ARTICLECODES BY SUPPLIER ... 32
1.1.1 GET ... *32*
1.1.1.1 Url ... 32
1.1.1.2 Parameters ... 32
1.1.1.3 Field ... 32

6. PRODUCTS (WHOLESALER) ... 33

6.1 PRODUCTS ... 34
6.1.1 GET ... *34*
6.1.1.1 Url ... 34
6.1.1.2 Parameters ... 34
6.1.1.3 Fields ... 34
6.1.2 POST ... *35*
6.1.2.1 Url ... 35
6.1.2.2 Dependencies ... 35
6.1.2.3 Fields ... 35
6.1.3 PUT ... *36*
6.1.3.1 Url ... 36
6.1.3.2 Dependencies ... 36
6.1.3.3 Fields ... 36
6.1.4 PUT (Sales) ... *36*
6.1.4.1 Url ... 36
6.1.4.2 Fields ... 36
6.1.5 DELETE ... *36*
6.1.5.1 Url ... 36
6.1.5.2 Parameters ... 36
6.1.5.3 Fields ... 36
6.2 EAN-LIST ... 37
6.2.1 GET ... *37*
6.2.1.1 Url ... 37
6.2.1.2 Parameters ... 37
6.2.1.3 Fields ... 37

6.3 WHOLESALER ARTICLE NUMBER LIST 38
6.3.1 GET *38*
6.3.1.1 Url 38
6.3.1.2 Parameters 38
6.3.1.3 Fields 38
6.4 PRODUCT BY EAN 39
6.4.1 GET *39*
6.4.1.1 Url 39
6.4.1.2 Parameters 39
6.4.1.3 Fields 39
6.5 PRODUCT BY SUPPLIER 40
6.5.1 GET *40*
6.5.1.1 Url 40
6.5.1.2 Parameters 40
6.5.1.3 Fields 40
6.6 PRODUCT BY WHOLESALER ARTICLE NUMBER 41
6.6.1 GET *41*
6.6.1.1 Url 41
6.6.1.2 Parameters 41
6.6.1.3 Fields 41
6.7 PRODUCT PHOTOS 42
6.7.1 GET *42*
6.7.1.1 Url 42
6.7.1.2 Parameters 42
6.7.1.3 Fields 42
6.7.2 POST *42*
6.7.2.1 Url 42
6.7.2.2 Fields 42
6.7.3 DELETE *42*
6.7.3.1 Url 42
6.7.3.2 Parameters 42
6.7.3.3 Fields 42

7. LINKING OWN PRODUCTS WITH CENTRAL PRODUCTS 43

7.1 PRODUCTLIST 43
7.1.1 GET *43*
7.1.1.1 Url 43
7.1.1.2 Parameters 43
7.1.1.3 Fields 43
7.1.2 POST *43*
7.1.2.1 Url 43
7.1.2.2 Fields 43
7.1.3 PUT *44*
7.1.3.1 Url 44
7.1.3.2 Fields 44
7.1.4 PUT (Sales) *44*
7.1.4.1 Url 44
7.1.4.2 Fields 44
7.1.5 DELETE *44*
7.1.5.1 Url 44
7.1.5.2 Parameters 44
7.1.5.3 Fields 44
7.2 EAN-LIST 45
7.2.1 GET *45*
7.2.1.1 Url 45
7.2.1.2 Parameters 45
7.2.1.3 Fields 45
7.3 WHOLESALER ARTICLE NUMBER LIST 46
7.3.1 GET *46*
7.3.1.1 Url 46
7.3.1.2 Parameters 46

7.3.1.3 Fields 46
7.4 PRODUCT BY EAN 47
7.4.1 GET *47*
7.4.1.1 Url 47
7.4.1.2 Parameters 47
7.4.1.3 Fields 47
7.5 PRODUCT BY SUPPLIER 48
7.5.1 GET *48*
7.5.1.1 Url 48
7.5.1.2 Parameters 48
7.5.1.3 Fields 48
7.6 PRODUCT BY ARTICLE NUMBER 49
7.6.1 GET *49*
7.6.1.1 Url 49
7.6.1.2 Parameters 49
7.6.1.3 Fields 49

8. CREATE CLIENTS FOR THE WEBSHOP 50

8.1 CLIENT SEGMENTS 50
8.1.1 GET *50*
8.1.1.1 Url 50
8.1.1.2 Fields 50
8.2 CLIENTS 51
8.2.1 GET *51*
8.2.1.1 Url 51
8.2.1.2 Fields 51
8.2.2 POST *51*
8.2.2.1 Url 51
8.2.2.2 Dependencies 51
8.2.2.3 Fields 51
8.2.3 PUT *52*
8.2.3.1 Url 52
8.2.3.2 Dependencies 52
8.2.3.3 Fields 52
8.2.4 DELETE *52*
8.2.4.1 Url 52
8.2.4.2 Parameters 52
8.2.4.3 Fields 52
8.3 CLIENT LOCATION 53
8.3.1 GET *53*
8.3.1.1 Url 53
8.3.1.2 Dependencies 53
8.3.1.3 Fields 53
8.3.2 POST *53*
8.3.2.1 Url 53
8.3.2.2 Fields 53
8.3.3 PUT *54*
8.3.3.1 Url 54
8.3.3.2 Fields 54
8.3.4 DELETE (by location) *54*
8.3.4.1 Url 54
8.3.4.2 Parameters 54
8.3.5 DELETE (by client) *54*
8.3.5.1 Url 54
8.3.5.2 Parameters 54
8.4 CLIENT LOCATIONS DELIVERY DATES 55
8.4.1 GET *55*
8.4.1.1 Url 55
8.4.1.2 Dependencies 55
8.4.1.3 Fields 55
8.4.2 PUT *55*

8.4.2.1 Url 55
8.4.2.2 Fields 55
8.5 LINK CLIENTS TO CLIENT-USERS 56
8.5.1 POST *56*
8.5.1.1 Url 56
8.5.1.2 Fields 56
8.6 REMOVE CLIENT-USERS FROM CLIENTS 57
8.6.1 POST *57*
8.6.1.1 Url 57
8.6.1.2 Fields 57
8.7 LINK A CLIENT TO A REPRESENTATIVE 58
8.7.1 POST *58*
8.7.1.1 Url 58
8.7.1.2 Fields 58
8.8 REMOVE REPRESENTATIVE FROM CLIENT 59
8.8.1 POST *59*
8.8.1.1 Url 59
8.8.1.2 Fields 59
8.9 REPRESENTATIVES 60
8.9.1 GET *60*
8.9.1.1 Url 60
8.9.1.2 Fields 60
8.9.2 POST *60*
8.9.2.1 Url 60
8.9.2.2 Dependencies 60
8.9.2.3 Fields 60
8.9.3 PUT *61*
8.9.3.1 Url 61
8.9.3.2 Dependencies 61
8.9.3.3 Fields 61
8.9.4 DELETE *61*
8.9.4.1 Url 61
8.9.4.2 Parameters 61
8.9.4.3 Fields 61
8.10 LINK REPRESENTATIVES TO A CLIENT 62
8.10.1 POST *62*
8.10.1.1 Url 62
8.10.1.2 Fields 62
8.11 REMOVE REPRESENTATIVES FROM A CLIENT 63
8.11.1 POST *63*
8.11.1.1 Url 63
8.11.1.2 Fields 63
8.12 CLIENT USERS 64
8.12.1 GET *64*
8.12.1.1 Url 64
8.12.1.2 Fields 64
8.12.2 POST *64*
8.12.2.1 Url 64
8.12.2.2 Dependencies 64
8.12.2.3 Fields 64
8.12.3 PUT *65*
8.12.3.1 Url 65
8.12.3.2 Dependencies 65
8.12.3.3 Fields 65
8.12.4 DELETE *65*
8.12.4.1 Url 65
8.12.4.2 Parameters 65
8.12.4.3 Fields 65
8.13 LINK USERS TO A CLIENT 66
8.13.1 POST *66*
8.13.1.1 Url 66

8.13.1.2 Fields 66
8.14 REMOVE USERS FROM A CLIENT 67
8.14.1 POST *67*
8.14.1.1 Url 67
8.14.1.2 Fields 67
8.15 ORDERLIST 68
8.15.1 GET *68*
8.15.1.1 Url 68
8.15.1.2 Fields 68
8.15.2 POST *68*
8.15.2.1 Url 68
8.15.2.2 Dependencies 68
8.15.2.3 Fields 68
8.15.3 PUT *69*
8.15.3.1 Url 69
8.15.3.2 Dependencies 69
8.15.3.3 Fields 69
8.15.4 DELETE *69*
8.15.4.1 Url 69
8.15.4.2 Parameters 69
8.15.4.3 Fields 69

9. CLIENT PRODUCTS 70

9.1 CLIENT PRODUCTS 71
9.1.1 GET *71*
9.1.1.1 Url 71
9.1.1.2 Fields 71
9.1.2 POST *72*
9.1.2.1 Url 72
9.1.2.2 Dependencies 72
9.1.2.3 Fields 72
9.1.3 POST (Multiple) *72*
9.1.3.1 Url 72
9.1.3.2 Dependencies 72
9.1.3.3 Fields 72
9.1.4 POST (Multiple AddOrUpdate) *73*
9.1.4.1 Url 73
9.1.4.2 Dependencies 73
9.1.4.3 Fields 73
9.1.5 PUT *73*
9.1.5.1 Url 73
9.1.5.2 Dependencies 73
9.1.5.3 Fields 73
9.1.6 PUT (Multiple) *73*
9.1.6.1 Url 74
9.1.6.2 Dependencies 74
9.1.6.3 Fields 74
9.1.7 DELETE *75*
9.1.7.1 Url 75
9.1.7.2 Parameters 75
9.1.8 DELETE (All) *75*
9.1.8.1 Url 75
9.1.8.2 Parameters 75

10. CLIENT ORDERS 76

10.1 RETRIEVE CLIENT ORDERS 76
10.1.1 GET *76*
10.1.1.1 Url 76
10.1.1.2 Parameters 76
10.1.1.3 Fields 76
10.2 PROCESS CLIENT ORDERS 77

10.2.1 PUT *77*
10.2.1.1 URL 77
10.2.1.2 Dependencies 77
10.2.1.3 Fields 77

11. ERRORS 78

11.1 BY DAY 78
11.1.1 GET *78*
11.1.1.1 Url 78
11.1.1.2 Parameters 78
11.1.1.3 Fields 78
11.2 WITH PAGING 79
11.2.1 GET *79*
11.2.1.1 Url 79
11.2.1.2 Parameters 79
11.2.1.3 Fields 79

1. Summary

This document gives in depth information about the functionality of the API.

1.1 In this document

Per function/endpoint you can find the following:

- The url of the endpoint
- The parameters (when applicable)
- The key fields
- The help page
- The help page of the model
    - GET: data returned by the API to the client
    - POST: data expected by the API from the client
    - PUT: data expected by the API from the client
    - DELETE: data returned by the API to the client
- Response object
    - This is a wrapper with useful information about the request
    - For each endpoint the following is returned:
        - RequestDuration: duration of the request in ms
        - WholeSalerId: the wholesaler id in the Oresto DB
        - UserId: the user id in the Oresto DB
    - GET methods
        - Errors: server-side errors
    - POST/PUT/DELETE methods
        - Errors: server-side errors
        - ModelStateErrors: errors created when validating the objects, for instance max length validation.

The documentation is based on the following version:

- **API**: 4.21.0
- **Models**: 3.16.0

1.2 API

The API runs in two environments: one test and one production.

All changes will be pushed to the TEST environment first. Every user will be notified about the changes and when they will be available in production.

You can find the production and test urls in the following table.

	Test	**Productie**
API url	https://api-test.orestofoodpartners.be/api	Wholesaler specific URL
Help url	https://api-test.orestofoodpartners.be/help	Wholesaler specific URL

1.2.1 Authentication

Authentication with the API is done by use of basic authentication. Login and password must be provided in a base64 encoded string.
The format of this string should be: "login:password".
For instance, the login demo with password test should give the following result: "ZGVtbzp0ZXN0".

This login information is provided in the HTTP-header: "Authorization: Basic ZGVtbzp0ZXN0"

1.2.2 Code and examples

The API is writting in C# (.NET).

There is a class library available with every model used in the API.
This library is on-demand. (contact support@analyz-it.be).

Code examples are not available at the moment.

1.2.3 Use of the date/time fields

For the use of date/time fields we use the ISO-8601 standard (https://en.wikipedia.org/wiki/ISO_8601). Because some GET functions are provided by older code, they can also accept "dd/MM/yyyy" but we strongly discourage using this format as this may be deleted in the near future.

In case there is no time zone defined we will save every date of Belgium. If you want to explicitly provide a time zone, do so by adding the correct information, for example: 2019-01-01+00:00 for UTC or 2019-01-01+01:00 for Belgium. In case of GET functions make sure your date/time fields are url-encoded.

1.2.4 Use of special characters in XML

Special characters should be encoded if you are using XML to submit your data. Please refer to the following table:

Name	Character	Encoded
Ampersand	&	&amp;
Double quote	"	&quot;
Single quote	'	&apos;
Less than	<	&lt;
Greater than	>	&gt;

1.2.5 Order in XML

The order of the elements provided in XML **must be** ordered alphabetically for each endpoint.
When not provided this way, the API will cancel the request with the message "request body cannot be null".

With each update please check if elements are renamed or added in order keep functionality. JSON/URL form encoded

2. Course and status of the synchronization

Each implementation should handle a few basic steps.
These are optional but should provide some basic knowledge on how to start developing with the API:

- Test of the API GET endpoints
- Fetch of the central products
- Compare the central product to your own database and enrich your own CRM with the data provided.
- Fetch of the brands, producers, categories, … store locally or use to enrich own data.
- Edit own CRM in order to interact with the API.
- Test of the API POST/PUT endpoints
    - Add a client, representative, user
    - Link or add a product
    - Login to the webshop
- …

3. Roadmap

3.1 Create client and user

When creating a client you can add a segment to it, to do this you need to GET the clientsegments. You only need to do this once with each sync.

	Type		Endpoint	Page
☐	GET	Get clientsegments	api/ClientSegments	50
☐	POST	Create client	api/Clients - Link clientsegment (if needed)	51
☐	POST	Create login	api/Users	64
OPTIONAL				
☐	POST	Create representative	api/Representative	60

3.2 Retrieve orders

In order to retrieve the orders you can do this by client or all of the orders. After the retrieval you can flag an order as "processed". Once this has been done, the order will not be returned in the GET function.

	Type		Endpoint	Page
☐	GET	Get orders	api/ClientOrders	76
☐	PUT	Process orders	api/ClientOrders	77

3.3 Create clientlocations

These locations are considered delivery addresses.

	Type		Endpoint	Page
☐	PUT	Create location	api/ClientLocations	55

3.4 Delivery dates of a client

If needed you can define delivery dates for each client location (settings are configured in the back-end). In order to define delivery dates you need to fetch the ID of the address first. Keep in mind: these endpoints do not append but rather overwrite current data.
If needed these days can be configered in the back-end on client level.

	Type		Endpoint	Page
☐	GET	Get address	api/ClientLocations	53
☐	PUT	Send possible dates	api/ClientLocationDeliveryDates - ID of the address is needed here	55

3.5 Link a central product

You can link your own products with the central database. This is the most efficient way of working with the webshop.

Before you can link a product it is best to check the EAN of the own product against the central database, if you want to link multiple products at once, you can use the endpoint api/CentralProducts/GetEan to retrieve a list of available EANs.

If the EAN code is not known, you can query the endpoint api/CentralProducts/GetBySupplier to check whether the item exists centrally at Oresto based on the supplier information and the supplier's product code. The supplier information can be retrieved via the endpoint api/Suppliers/GetByVat using the VAT number in combination with the country.

Should there be no match, it is best to create the product.

	Type		Endpoint	Page
If EAN is known				
☐	GET	Check product	api/CentralProducts/GetByEan - EAN	30
If EAN is not known				
☐	GET	Get supplier	api/Suppliers/GetByVat - VAT number - Landcode	26
☐	GET	Check product	api/CentralProducts/GetBySupplier - Supplier – ID search - Product code	31
If product found				
☐	GET	Get Unit sales prices	api/UnitSalesPrices	23
☐	POST	Create product(s)	api/LinkProducts - Unit sales price – ID	43

3.6 Create (own) product

Own products are products that are not found in the central database. It is best to check if a product you want to add is not already defined in the system. If it is not found you may create it as product. Keep in mind creating a product requires alot of references to the boundary tables.

	Type		Endpoint	Page
If EAN is known				
☐	GET	Check product	api/CentralProducts/GetByEan - EAN	30
If EAN is not known				
☐	GET	Get supplier	api/Suppliers/GetByVat - VAT number - Landcode	26
☐	GET	Check product	api/CentralProducts/GetBySupplier - Supplier – ID search - Product code	31
If product not found				
☐	GET	Get brand	api/Brands	20
☐	GET	Get categories	api/Categories/GetConcatenated?reverse=true	19
☐	GET	Get content type	api/ContentTypes	22
☐	GET	Get unit sales price type	api/UnitSalesPrices	23
☐	GET	Get producer	api/Producers	25

☐	GET	Get storage condition	api/StorageConditions	21
☐	GET	Get supplier	api/Suppliers	26
☐	POST	Create product(s)	api/Products - Category – ID search - Brand – ID search - Content type – ID search - Storage – ID search - Unit sales price type – ID search - Producer – ID search - Supplier – ID search (=OPTIONAL)	35
OPTIONAL				
☐	POST	Add product photos	api/ProductPhotos	42

3.7 In between check own <-> central products

In order to keep your system in sync, it is best you check every X days/time if your own created products are not available as a central product. This will not be done by the webshop, and should be done by the wholesaler. Benefits of linking to the central database are automatic data population (content, brands, allergenes, …), photos, packaging.

	Type		**Endpoint**	**Page**
☐	GET	Fetch EAN list	api/CentralProducts/GetEan	29
☐	/	Check own product	Check own <-> central products	
If EAN existent				
☐	DELETE	Delete own product	api/Products	36
☐	GET	Get unit sales price type	api/UnitSalesPrices	23
☐	POST	Create product(s)	api/LinkProducts - Unit sales price – ID	43

3.8 Create clientproducts

When needed you can define extra data for each client and product. These data can contain a specific price, promo or exclusivity.

	Type		**Endpoint**	**Page**
☐	POST	Create client product	api/ClientProducts	72

3.9 Order Multiples (with Decimals)

You may want to sell a product in multiples while displaying the price per unit. For example, a product consists of 6 bottles of wine, but the price is displayed per bottle.

You can achieve this by specifying the multiple, e.g. 6, in the 'OrderQuantity' field in either the api/LinkProducts endpoint (central product) or the api/Products endpoint (own product) when creating a product. The default value of this field is 1.

When products are not sold per piece but by weight or volume, you may want to allow decimal values as order quantities. For example, the price of meat is per kg, but you sell it in multiples of 1.5 kg or 0.1 kg.

In that case, you specify a decimal value as the multiple, e.g. 1.5 or 0.1, in the 'OrderQuantity' field in either the api/LinkProducts endpoint (central product) or the api/Products endpoint (own product) when creating a product. The maximum number of allowed decimal places is 4.

If you want to allow a freely entered value with decimals, you should set the 'OrderQuantity' field to 0.

	Type		Endpoint	Pagina
☐	POST	Create product	api/LinkProducts - OrderQuantity	43

	Type		Endpoint	Pagina
☐	POST	Create product	api/Products - OrderQuantity	35

3.10 Price per kilogram or liter, but sold per piece

If you want to display the price of an item per kilogram/liter, but sell it per piece, you can use the multiplier to indicate how many kilograms or liters one piece weighs or contains. The webshop will then automatically calculate the correct price per piece.

Example: the price of cheese is €10 per kilogram, but it is sold per piece, and one piece weighs 5 kilograms. In this case, the multiplier is 5 and the multiplier unit is KG. One piece will therefore cost 5 × €10 = €50.

Oresto manages the multiplier values for centrally managed products. To use the multipliers and have the webshop automatically perform the recalculation, it is sufficient to set the sales unit to "per piece" when linking the product to the central product via the api/LinkProducts endpoint.

To determine which central products are multiplier products, you can retrieve their EAN codes via the api/CentralProducts/GetEan endpoint and set the MultiplicatorProducts filter to true.

	Type		Endpoint	Pagina
☐	GET	Get multiplier products	api/CentralProducts/GetEan?MultiplicatorProducts=true	29
☐	POST	Create product	api/LinkProducts - UnitSalesPriceAbbreviation: pc - SalesPriceList: sales price per MultiplierAbbreviation (e.g. 10)	43

If you want to use multipliers for your own products, you must add them yourself via the api/Products endpoint. First retrieve which multiplier units are supported in the webshop via the api/MultiplierTypes endpoint.

	Type		Endpoint	Pagina
☐	GET	MultiplierTypes	api/MultiplierTypes	24
☐	POST	Create product	api/Products - UnitSalesPriceAbbreviation: pc - SalesPriceList: sales price per MultiplierAbbreviation (e.g. 10) - MultiplierAbbreviation – search ID (e.g. kg) - MultiplierAmount: how many MultiplierAbbreviation in 1 UnitSalesPriceAbbreviation (e.g. 5)	35

4. Boundary tables

Before you can upload your own products to the database, you need to enrich these products with basic table data provided by the API. For example, brand ID, producer ID, category ID, …

You can fetch this data by employing the GET endpoints. Either store these locally or make a lookup query each time you need information. Once the data is enough you can submit your own products.

Synchronizing once a day should be enough for all implementations. Real-time implementations are possible but can provide negative impact on performance.

4.1 Categories

4.1.1 GET

Products are categorized. Each category consists of a unique key and can contain multiple subcategories which can contain their own subcategories.
Each subcategory has a reference to the parent category.

Once a category contains products it cannot contain subcategories. Products can therefor only be added to categories with no subcategories.

Remarks:
- Products can be added to multiple categories

4.1.1.1 URL

api/Categories

4.1.1.2 Fields

Field	Key	Type	Remarks
Id	PK	Int	KeyField

Help page - Model

4.1.2 GET (concatenated)

This will return a list with each of the category and subcategories concatenated.
The list will have the main category at the start and end with the latest subcategory.
For instance: Food – Desserten & Zoet – **Schepijs & Sorbet**

When needed the list can be reversed so that the subcategory is shown first. For instance: **Schepijs & Sorbet** | Desserten & Zoet | Food

Each category is divided by a pipe-sign ("|").

4.1.2.1 URL

api/Categories/GetConcatenated

4.1.2.2 Parameters

Name	Type	Remarks
Reverse	Boolean	

4.1.2.3 Fields

Field	Key	Type	Remarks
Id	PK	Int	KeyField

Help page - Model – Response object

4.2 Brands

4.2.1 GET

Returns all the brands in the database. Brands are linked with a product by use of the Brand Id.

4.2.1.1 URL

api/Brands

4.2.1.2 Fields

Field	Key	Type	Remarks
Id	PK	Int	KeyField

Help page - Model – Response object

4.3 Storage conditions

4.3.1 GET

The storage condition of a product. For instance, “dry”, “cool”, “frozen”, …

Remarks:

- A product can only be linked to one storage condition. The condition “All” is used for products which cannot be accommodated with a storage condition.

4.3.1.1 URL

api/StorageConditions

4.3.1.2 Fields

Field	Key	Type	Remarks
Id	PK	Int	KeyField

Help page - Model – Response object

4.4 Content Types

4.4.1 GET

In the webshop a product is defined by content and its content type.
For instance, 100 **grams** | 2,5 **KG** | 1 **piece** | 2 **L**
You can find this information in the webshop under "total content"

Remarks:
- A product can have only one content type.
- This field may not be confused by unit sales price type. This field is provided to determine if a product should be sold as KG, L or piece.
- Abbreviations are provided by ISO-format.

4.4.1.1 URL

api/ContentTypes

4.4.1.2 Fields

Field	Key	Type	Remarks
Abbreviation		String	Abbreviation of the content type (ISO-format)

Help page - Model – Response object

4.5 Unit sales price

4.5.1 GET

Products have a sales price and a **unit sales price**.
You can provide a product per kg, liter or piece.

Remarks:
- Do not confuse this field with content type.
- Abbreviations are provided by ISO-format.

4.5.1.1 URL

api/UnitSalesPrices

4.5.1.2 Fields

Field	Key	Type	Remarks
Abbreviation		String	Abbreviation of the content type (ISO-format)

Help page - Model – Response object

4.6 Multiplier types

4.6.1 GET

A product can have an multiplier and a type.
For instance: a product can be sold by piece but each piece may contain 5 kg. The webshop will show the product pricing by price / kg. In this example the multiplier would be 5 and the type would be kg.

Remarks:
- Abbreviations are provided by ISO-format.

4.6.1.1 URL

api/MultiplierTypes

4.6.1.2 Fields

Field	Key	Type	Remarks
Abbreviation		String	Abbreviation of the content type (ISO-format)

Help page - Model – Response object

4.7 Producers

4.7.1 GET

The producers of the products.
A producer can be provided for your own products but is not mandatory.

4.7.1.1 URL

api/Producers

4.7.1.2 Fields

Field	Key	Type	Remarks
Id	PK	Int	KeyField

Help page - Model – Response object

4.8 Suppliers

4.8.1 GET

The suppliers of the products.

A product can be supplied by multiple suppliers. Each supplier can have a different supplier code for a product.

For own products you can use a supplier (optional). Only one supplier can be provided.

Remarks:
- Products can have multiple suppliers. Each product can have one or more supplier keys and codes

4.8.1.1 Url

api/Suppliers

4.8.1.2 Fields

Field	Key	Type	Remarks
Id	PK	Int	KeyField

Help page - Model – Response object

4.8.2 GET (vat-number)

Request a supplier by use of the VAT-number and the country code.
Returns the same information as the GET-endpoint of the suppliers.

4.8.2.1 Url

api/Supplier/GetByVat

4.8.2.2 Parameters

Name	Type	Remarks
VatNumber	String	Mandatory
VatCountryCode	String	Mandatory

4.8.2.3 Fields

Field	Key	Type	Remarks
Id	PK	Int	KeyField

Help page - Model – Response object

5. Central Products

The central products are the products provided by Oresto Food Partners (Hasselt).
Wholesaler products are not contained by this catalogue but can be later transformed to a central product (when needed).

5.1 Products

5.1.1 GET

This returns a list of all the central products.

Following options are provided to filter your results:
- fetch products changed after a certain date/time
- fetch products by use of paging
- fetch products which are deleted (can return multiple objects of the same product)
- fetch only products with a multiplicator defined

For all dates we use the time zone of Belgium. You can however request the data with a different time zone (See explanation of date/time fields).

5.1.1.1 Url

api/CentralProducts

5.1.1.2 Parameters

Name	Type	Remarks
modifiedDate	Date	Optional
modifiedEndDate	Date	Optional
PageSize	Integer	Optional
Page	Integer	Optional
DeletedProducts	Boolean	Optional
`MultiplicatorProducts`	Boolean	Optional

5.1.1.3 Fields

Field	Key	Type	Remarks
EanCode		String	Unique EAN-code used in the entire database

Help page - Model - Response object

5.2 EAN-list

5.2.1 GET

Returns a list of available EAN-codes (unit as package products).

Following options are provided to filter your results:
- fetch EAN-codes changed after a certain date/time
- fetch only products with a multiplicator defined
-

For all dates we use the time zone of Belgium. You can however request the data with a different time zone (See explanation of date/time fields).

5.2.1.1 Url

api/CentralProducts/GetEan

5.2.1.2 Parameters

Name	Type	Remarks
modifiedDate	Date	Optional
modifiedEndDate	Date	Optional
ExcludeCentralProducts	Boolean	Optional
`MultiplicatorProducts`	Boolean	Optional

5.2.1.3 Fields

Help page – Response object

5.3 Product by EAN

5.3.1 GET

Returns a product by use of EAN-code.

5.3.1.1 Url

api/CentralProducts/GetByEan

5.3.1.2 Parameters

Name	Type	Remarks
EanCode	String	Mandatory

5.3.1.3 Fields

Help page – Response object

5.4 Product by Supplier

5.4.1 GET

Returns a product by use of the supplier ID and the article number of the supplier.
A supplier code can be the same for the unit as the packaging product, please use IsPackageProduct when requesting package products.

5.4.1.1 Url

api/CentralProducts/GetBySupplier

5.4.1.2 Parameters

Name	Type	Remarks
Id	Int	Mandatory
ArticleCode	String	Mandatory
IsPackageProduct	Boolean	Mandatory

5.4.1.3 Fields

Help page – Response object

1.1 Articlecodes by Supplier

1.1.1 GET

Return a list of all active article codes, grouped by supplier. Using the optional parameter OnlyEmptyEan allows for filtering on only articles with no EAN code.

1.1.1.1 Url

api/CentralProducts/GetArticlecodesBySupplier

1.1.1.2 Parameters

Name	Type	Remarks
`OnlyEmptyEan`	Boolean	Optional

1.1.1.3 Field

Help page – Response object

6. Products (wholesaler)

These endpoints are provided for your own product data.
You can request, submit or delete own product data. If a product is already known in the webshop it is better to link your own product to the central product rather than submitting a new one.

6.1 Products

6.1.1 GET

This endpoint returns your own products. These products are not managed by Oresto Food Partners (Hasselt).

Following options are provided to filter your results:
- fetch products changed after a certain date/time
- fetch products by use of paging
- fetch products which are deleted (can return multiple objects of the same product)

For all dates we use the time zone of Belgium. You can however request the data with a different time zone (See explanation of date/time fields).

6.1.1.1 Url

api/Products

6.1.1.2 Parameters

Name	Type	Remarks
ModifiedDate	Date	Optional
ModifiedEndDate	Date	Optional
ExcludeCentralProducts	Boolean	Optional
PageSize	Integer	Optional
Page	Integer	Optional
MultiplicatorProducts	Boolean	Optional

6.1.1.3 Fields

Field	Key	Type	Remarks
WholesalerArticleNumber		String	Unique code for the wholesaler

Help page - Model – Response object

6.1.2 POST

Add products not provided by the central database.

Article code is used to make a product unique. EAN-code and supplier information is optional.

Warning, characters from the set "\+<>?#%@&*:./" will be blocked.

6.1.2.1 Url

api/Products

6.1.2.2 Dependencies

- Brand
- Producer
- Content type
- Unit sales price
- Supplier

6.1.2.3 Fields

Field	Key	Type	Remarks
WholesalerArticleNumber		String	Unieke code for the wholesaler

Help page – Model - Response object

6.1.3 PUT

Update own products. Use the article number to find the product.

6.1.3.1 Url

api/Products

6.1.3.2 Dependencies

- Brand
- Producer
- Content type
- Unit sales price
- Supplier

6.1.3.3 Fields

Field	Key	Type	Remarks
WholesalerArticleNumber		String	KeyField Unique code for the wholesaler

Help page – Model – Response object

6.1.4 PUT (Sales)

Update the sales information of own or linked products. Use article number to find the product.

6.1.4.1 Url

api/Products/Sales

6.1.4.2 Fields

Field	Key	Type	Remarks
WholesalerArticleNumber		String	KeyField Unique code for the wholesaler

Help page – Model – Response object

6.1.5 DELETE

Remove own products from the catalogue. Use article number to find the product.

6.1.5.1 Url

api/Products

6.1.5.2 Parameters

Name	Type	Remarks
WholesalerArticleNumber	String	Mandatory

6.1.5.3 Fields

Help page – Response object

6.2 EAN-list

6.2.1 GET

Returns a list of available EAN-codes (unit as package products).

Following options are provided to filter your results:
- fetch EAN-codes changed after a certain date/time

For all dates we use the time zone of Belgium. You can however request the data with a different time zone (See explanation of date/time fields).

6.2.1.1 Url

api/Products/GetEan

6.2.1.2 Parameters

Name	Type	Remarks
modifiedDate	Date	Optional
modifiedEndDate	Date	Optional
ExcludeCentralProducts	Boolean	Optional
`MultiplicatorProducts`	Boolean	Optional

6.2.1.3 Fields

Help page – Response object

6.3 Wholesaler article number list

6.3.1 GET

Returns a list of available wholesaler article numbers (unit as package products).

Following options are provided to filter your results:
- fetch codes changed after a certain date/time

For all dates we use the time zone of Belgium. You can however request the data with a different time zone (See explanation of date/time fields).

6.3.1.1 Url

api/Products/GetWholesalerArticleNumbers

6.3.1.2 Parameters

Name	Type	Remarks
modifiedDate	Date	Optional
modifiedEndDate	Date	Optional
ExcludeCentralProducts	Boolean	Optional
MultiplicatorProducts	Boolean	Optional

6.3.1.3 Fields

Help page – Response object

6.4 Product by EAN

6.4.1 GET

Returns a product by use of EAN-code.

6.4.1.1 Url

api/Products/GetByEan

6.4.1.2 Parameters

Name	Type	Remarks
EanCode	String	Mandatory

6.4.1.3 Fields

Help page – Response object

6.5 Product by Supplier

6.5.1 GET

Returns a product by use of the supplier ID and the article number of the supplier.
A supplier code can be the same for the unit as the packaging product, please use IsPackageProduct when requesting package products.

6.5.1.1 Url

api/Products/GetbySupplier

6.5.1.2 Parameters

Name	Type	Remarks
Id	Int	Mandatory
ArticleCode	String	Mandatory
IsPackageProduct	Boolean	Mandatory

6.5.1.3 Fields

Help page – Response object

6.6 Product by wholesaler article number

6.6.1 GET

Returns a product by use of the wholesaler article number.

6.6.1.1 Url

api/Products/GetByWholesalerArticleNumber

6.6.1.2 Parameters

Name	Type	Remarks
WholesalerArticleNumber	String	Mandatory

6.6.1.3 Fields

Help page – Response object

6.7 Product photos

6.7.1 GET

This function returns product photos of own products (wholesaler).
You can request a list of products with no photos.

6.7.1.1 Url

api/ProductPhotos

6.7.1.2 Parameters

Name	Type	Remarks
HasPhotos	Date	Optional

6.7.1.3 Fields

Field	Key	Type	Remarks
PhotoId	Ja	Integer	Unique id per photo

Help page - Response object

6.7.2 POST

Submit a product photo for own products. This product can be specified for a specific language. If no language is provided the photo will be added every language.

6.7.2.1 Url

api/ProductPhotos

6.7.2.2 Fields

Field	Key	Type	Remarks
WholesalerArticleNumber		String	Unique code for the wholesaler
Photo		Byte-array	File
PhotoName		String	Photo name

Help page – Model - Response object

6.7.3 DELETE

Delete a product photo. You need to provide the ID of the photo (see GET endpoint to determine the ID).

6.7.3.1 Url

api/ProductPhotos

6.7.3.2 Parameters

Name	Type	Remarks
WholesalerArticleNumber	String	Mandatory
PhotoId	Integer	Mandatory

6.7.3.3 Fields

Help page – Response object

7. Linking own products with central products

7.1 Productlist

7.1.1 GET

This returns a list of all the linked products (own products with the central products).

Following options are provided to filter your results:
- fetch products changed after a certain date/time
- fetch products by use of paging
- fetch products which are deleted (can return multiple objects of the same product)

For all dates we use the time zone of Belgium. You can however request the data with a different time zone (See explanation of date/time fields).

7.1.1.1 Url

api/LinkProducts

7.1.1.2 Parameters

Name	Type	Remarks
ModifiedDate	Date	Optional
ModifiedEndDate	Date	Optional
PageSize	Integer	Optional
Page	Integer	Optional

7.1.1.3 Fields

Field	Key	Type	Remarks
WholesalerArticleNumber		String	Unique code for the wholesaler

Help page - Model – Response object

7.1.2 POST

This endpoint is provided to link your own products with a central product managed by Oresto Food Partners. You can provide sales information for the linked product. Linking a product is done by use of the EAN-code or supplier information.

7.1.2.1 Url

api/LinkProducts

7.1.2.2 Fields

Field	Key	Type	Remarks
EanCode		String	
Supplier		Supplier	
ArticleNumberWholesaler		String	KeyField Unique code for the wholesaler

Help page – Model – Response object

7.1.3 PUT

Update the linked products information. Use the article number of the wholesaler to find the product.

7.1.3.1 Url

api/LinkProducts

7.1.3.2 Fields

Field	Key	Type	Remarks
WholesalerArticleNumber		String	KeyField Unique code for the wholesaler

Help page – Model – Response object

7.1.4 PUT (Sales)

Update the sales information of the linked product. Use the article number of the wholesaler to find the product.

7.1.4.1 Url

api/LinkProducts/Sales

7.1.4.2 Fields

Field	Key	Type	Remarks
WholesalerArticleNumber		String	KeyField Unique code for the wholesaler

Help page – Model – Response object

7.1.5 DELETE

Delete a linked product. Use the article number of the wholesaler to find the product.

7.1.5.1 Url

api/Products

7.1.5.2 Parameters

Name	Type	Remarks
WholesalerArticleNumber	String	Mandatory

7.1.5.3 Fields

Help page – Response object

7.2 EAN-list

7.2.1 GET

Returns a list of available EAN-codes (unit as package products).

Following options are provided to filter your results:
- fetch EAN-codes changed after a certain date/time

For all dates we use the time zone of Belgium. You can however request the data with a different time zone (See explanation of date/time fields).

7.2.1.1 Url

api/LinkProducts/GetEan

7.2.1.2 Parameters

Name	Type	Remarks
modifiedDate	Date	Optional
modifiedEndDate	Date	Optional

7.2.1.3 Fields

Help page – Response object

7.3 Wholesaler article number list

7.3.1 GET

Returns a list of available wholesaler article numbers (unit as package products).

Following options are provided to filter your results:
- fetch codes changed after a certain date/time

For all dates we use the time zone of Belgium. You can however request the data with a different time zone (See explanation of date/time fields).

7.3.1.1 Url

api/LinkProducts/GetWholesalerArticleNumbers

7.3.1.2 Parameters

Name	Type	Remarks
modifiedDate	Date	Optional
modifiedEndDate	Date	Optional

7.3.1.3 Fields

Help page – Response object

7.4 Product by EAN

7.4.1 GET

Returns a product by use of EAN-code.

7.4.1.1 Url

api/LinkProducts/GetByEan

7.4.1.2 Parameters

Name	Type	Remarks
EanCode	String	Mandatory

7.4.1.3 Fields

Help page – Response object

7.5 Product by Supplier

7.5.1 GET

Returns a product by use of the supplier ID and the article number of the supplier.
A supplier code can be the same for the unit as the packaging product, please use IsPackageProduct when requesting package products.
A product is unique but can contain multiple records, for instance if a product is sold per kg and per piece.

7.5.1.1 Url

api/LinkProducts/GetBySupplier

7.5.1.2 Parameters

Name	Type	Remarks
Id	Int	Mandatory
ArticleCode	String	Mandatory
IsPackageProduct	Boolean	Mandatory

7.5.1.3 Fields

Help page – Response object

7.6 Product by article number

7.6.1 GET

Returns a product by use of the wholesaler article number.

7.6.1.1 Url

api/LinkProducts/GetByWholesalerArticleNumber

7.6.1.2 Parameters

Name	Type	Remarks
WholesalerArticleNumber	String	Mandatory

7.6.1.3 Fields

Help page – Response object

8. Create clients for the webshop

8.1 Client segments

When adding clients, you can add it to a segment. In a later phase segments will be used to further filter the product segment in the webshop and/or to alter the look and feel of the shop. For instance: hotel/resto, sandwich bar, …

8.1.1 GET

Retrieve all the client segments available.

8.1.1.1 Url

api/ClientSegments

8.1.1.2 Fields

Field	Key	Type	Remarks
Id	PK	Int	KeyField

Help page – Model – Response object

8.2 Clients

These are the clients of the wholesaler and are defined by use of the client number provided by the wholesaler.

For instance: assume client is "Restaurant X". A client must have at least one client-user to be able to login to the webshop. An example of a client-user could be "Chef Y of restaurant X".

A client can also have one or more representatives.
A client can have one or more locations.

8.2.1 GET

Retrieve all the clients for the wholesaler.

8.2.1.1 Url

api/Clients

8.2.1.2 Fields

Field	Key	Type	Remarks
ClientNumber		String	Unique client number

Help page – Model – Response object

8.2.2 POST

Add own clients to the webshop.

8.2.2.1 Url

api/Clients

8.2.2.2 Dependencies

- Client segment

8.2.2.3 Fields

Field	Key	Type	Remarks
ClientNumber		String	Unique client number

Help page – Model – Response object

8.2.3 PUT

Update the client. Client number is used to lookup the client.

8.2.3.1 Url

api/Clients

8.2.3.2 Dependencies

- Client segment

8.2.3.3 Fields

Field	Key	Type	Remarks
ClientNumber		String	Unique client number

Help page – Model – Response object

8.2.4 DELETE

Erase a client. Client number is used to lookup the client.

8.2.4.1 Url

api/Clients

8.2.4.2 Parameters

Name	Type	Remarks
ClientNumber	String	Mandatory

8.2.4.3 Fields

Help page – Response object

8.3 Client location

8.3.1 GET

Retrieve all known locations of a client.

8.3.1.1 Url

api/ClientLocations

8.3.1.2 Dependencies

- Client

8.3.1.3 Fields

Field	Key	Type	Remarks
Id		integer	KeyField Unique code for the location

Help page - Response object

8.3.2 POST

Add a new client location for a specific client. ID is retrieved by use of the GET endpoint.

8.3.2.1 Url

api/ClientLocations

8.3.2.2 Fields

Field	Key	Type	Remarks
Id		integer	KeyField Unique code for the location

Help page - Model - Response object

8.3.3 PUT

Update a known client location. ID is retrieved by use of the GET endpoint.

8.3.3.1 Url

api/ClientLocations

8.3.3.2 Fields

Field	Key	Type	Remarks
Id		integer	KeyField Unique code for the location

Help page - Model - Response object

8.3.4 DELETE (by location)

Delete a known client location. ID is retrieved by use of the GET endpoint. Client number is provided by the wholesaler.

8.3.4.1 Url

api/ClientLocations

8.3.4.2 Parameters

Name	Type	Remarks
Id	integer	Required
ClientNumber	string	Required

Help page - Response object

8.3.5 DELETE (by client)

Delete all known client locations of one client. Locations are looked up by use of the client number provided by the wholesaler.

8.3.5.1 Url

api/ClientLocations/All

8.3.5.2 Parameters

Name	Type	Remarks
ClientNumber	string	Required

Help page - Response object

8.4 Client locations delivery dates

8.4.1 GET

Get the delivery dates available for all known client locations.

8.4.1.1 Url

api/ClientLocationDeliveryDates

8.4.1.2 Dependencies

- Client
- ClientLocation

8.4.1.3 Fields

Field	Key	Type	Remarks
ClientNumber		String	Unique client number
ID		Integer	ID of the location

Help page – Response object

8.4.2 PUT

Overwrite all known delivery dates of a known location. You must submit a list of possible delivery dates. Please do not use time zones in this endpoint. All known dates for the location will be erased and the new ones will be saved.

8.4.2.1 Url

api/ClientLocationDeliveryDates

8.4.2.2 Fields

Field	Key	Type	Remarks
ClientLocationId		integer	ID of the location

Help page - Model - Response object

8.5 Link clients to client-users

8.5.1 POST

Link a client to one or more users (users are able to login to the webshop).

8.5.1.1 Url

api/Clients/LinkClientToUsers

8.5.1.2 Fields

Field	Key	Type	Remarks
ClientNumber		String	
EmailUsers		User	

Help page – Model – Response object

8.6 Remove client-users from clients

8.6.1 POST

Remove client-users from a client.

8.6.1.1 Url

api/Clients/UnlinkClientFromUsers

8.6.1.2 Fields

Field	Key	Type	Remarks
ClientNumber		String	
EmailUsers		User	

Help page – Model – Response object

8.7 Link a client to a representative

8.7.1 POST

Link a known client to a representative

8.7.1.1 Url

api/Clients/LinkClientToRepresentative

8.7.1.2 Fields

Field	Key	Type	Remarks
ClientNumber		String	
EmailRepresentative		String	

Help page – Model – Response object

8.8 Remove representative from client

8.8.1 POST

Remove a known representative from a known client.

8.8.1.1 Url

api/Clients/UnlinkClientFromRepresentative

8.8.1.2 Fields

Field	Key	Type	Remarks
ClientNumber		String	
EmailRepresentative		String	

Help page – Model – Response object

8.9 Representatives

These are the representatives of a client (for instance Representative Z of Wholesaler X). These users can also login to the webshop.

A representative is unique by use of e-mail address.

8.9.1 GET

Retrieve all the representatives.

8.9.1.1 Url

api/Representatives

8.9.1.2 Fields

Field	Key	Type	Remarks
Email		String	Unique username, used to login

Help page – Model – Response object

8.9.2 POST

Endpoint to create a new representative. The e-mail address is used to identify the user. The e-mail address is unique and can only be used once.

When created a representative, one or more client numbers can be given to link the user.

8.9.2.1 Url

api/Representatives

8.9.2.2 Dependencies

- Client

8.9.2.3 Fields

Field	Key	Type	Remarks
Email		String	Unique username, used to login

Help page – Model – Response object

8.9.3 PUT

Endpoint to update a known representative. The e-mail address is used to identify the user. The e-mail address is unique and can only be used once.
When needed to change the e-mail address you can do so using the field "NewEmail".

8.9.3.1 Url

api/Representatives

8.9.3.2 Dependencies

- Client

8.9.3.3 Fields

Field	Key	Type	Remarks
Email		String	Unique username, used to login

Help page – Model – Response object

8.9.4 DELETE

Delete a representative. The e-mail address is used to identify the user.

8.9.4.1 Url

api/Representatives

8.9.4.2 Parameters

Name	Type	Remarks
Email	String	Mandatory

8.9.4.3 Fields

Help page – Response object

8.10 Link representatives to a client

8.10.1 POST

Link a representative to one or more clients by use of the e-mail address and client number.

8.10.1.1 Url

api/Representatives/LinkRepresentativeToClients

8.10.1.2 Fields

Field	Key	Type	Remarks
EmailRepresentative		String	
ClientNumbers		User	

Help page – Model – Response object

8.11 Remove representatives from a client

8.11.1 POST

Delete a representative from one or more clients by use of the e-mail address and client number.

8.11.1.1 Url

api/Representatives/UnlinkRepresentativeFromClients

8.11.1.2 Fields

Field	Key	Type	Remarks
EmailRepresentative		String	
ClientNumbers		User	

Help page – Model – Response object

8.12 Client users

These are the end-users that login for a client to the webshop. For instance, Chef Y of Restaurant X. There must exist a client before you can create a client user. Each client should have at least one user.

A user is unique by his e-mail address. You can use the client number to link the user to the client.

You can create multiple users for a client (optional).

8.12.1 GET

Get all the users of a client.

8.12.1.1 Url

api/Users

8.12.1.2 Fields

Field	Key	Type	Remarks
Email		String	Unique username, used to login

Help page – Model – Response object

8.12.2 POST

Endpoint to create a new user. The e-mail address is used to identify the user. The e-mail address is unique and can only be used once.

8.12.2.1 Url

api/Users

8.12.2.2 Dependencies

- Client

8.12.2.3 Fields

Field	Key	Type	Remarks
Email		String	Unique username, used to login

Help Page – Model – Response object

8.12.3 PUT

Endpoint to update a known user. The e-mail address is used to identify the user. The e-mail address is unique and can only be used once.
When needed to change the e-mail address you can do so using the field "NewEmail".

8.12.3.1 Url

api/Users

8.12.3.2 Dependencies

- Client

8.12.3.3 Fields

Field	Key	Type	Remarks
Email		String	Unique username, used to login

Help Page – Model – Response object

8.12.4 DELETE

Delete a user. The e-mail address is used to identify the user.

8.12.4.1 Url

api/Users

8.12.4.2 Parameters

Name	Type	Remarks
Email	String	Mandatory

8.12.4.3 Fields

Help page – Response object

8.13 Link users to a client

8.13.1 POST

Link a client user to a client.

8.13.1.1 Url

api/Users/LinkUserToClient

8.13.1.2 Fields

Field	Key	Type	Remarks
EmailUser		String	
ClientNumber		String	

Help page – Model – Response object

8.14 Remove users from a client

8.14.1 POST

Remove a client user from a client.

8.14.1.1 Url

api/Users/UnlinkUserFromClient

8.14.1.2 Fields

Field	Key	Type	Remarks
EmailUser		String	
ClientNumber		String	

Help page – Model – Response object

8.15 Orderlist

This is a wholesaler-curated list of products. Used for those cases where clients have subset of products they are encouraged to order from, or which the wholesalers wants to make more easily accessible.

8.15.1 GET

Get all the articlecodes of a client's orderlist.

8.15.1.1 Url

api/Clients/OrderListItems

8.15.1.2 Fields

Field	Key	Type	Remarks
Clientnumber		String	Required

Help page – Response object

8.15.2 POST

Endpoint to add one or more products to an order list

8.15.2.1 Url

Api/Clients/OrderListItems

8.15.2.2 Dependencies

- Client

8.15.2.3 Fields

Field	Key	Type	Remarks
ClientNumber		String	Required
WholesalerArticleNumbers		List of String	Required

Help Page – Model – Response object

8.15.3 PUT

Endpoint to replace an order list

8.15.3.1 Url

Api/Clients/OrderListItems

8.15.3.2 Dependencies

- Client

8.15.3.3 Fields

Field	Key	Type	Remarks
ClientNumber		String	Required
WholesalerArticleNumbers		List of String	Required

Help Page – Model – Response object

8.15.4 DELETE

Erase an item from the orderlist. Client number is used to lookup the client.

8.15.4.1 Url

api/Clients/OrderListItems

8.15.4.2 Parameters

Name	Type	Remarks
ClientNumber	String	Required
WholesalerArticleNumber	String	Required

8.15.4.3 Fields

Help page – Response object

9. Client products

Client products can be used to specify custom prices for certain products depending on the client. Products can also be made exclusive so that only one client has access to them.

9.1 Client products

9.1.1 GET

Retrieve all the client products.

9.1.1.1 Url

api/ClientProducts

9.1.1.2 Fields

Field	Key	Type	Remarks
WholesalerArticleNumber		String	KeyField Unique code for the wholesaler
EanCode		String	EAN code
ClientNumber		String	Client number

Help page - Model - Response object

9.1.2 POST

Add client products to the database.

9.1.2.1 Url

api/ClientProducts

9.1.2.2 Dependencies

- Wholesaler product
- Client
- Product

9.1.2.3 Fields

Field	Key	Type	Remarks
WholesalerArticleNumber		String	KeyField Unique code for the wholesaler
ClientNumber		String	Client number

Help page - Model - Response object

9.1.3 POST (Multiple)

Add multiple client products at once to the database.

The result object contains only information about the errors.
Validate the errors to see if products were added successfully. If a product key is not found in the Errors or ModelStateErrors object, it is safe to assume the product was saved successfully.
Each error has a key which contains: Key1 = WholesalerArticlenumber and Key2 = ClientNumber.

9.1.3.1 Url

api/ClientProducts/Multiple

9.1.3.2 Dependencies

- Wholesaler product
- Client
- Product

9.1.3.3 Fields

Field	Key	Type	Remarks
WholesalerArticleNumber		String	KeyField Unique code for the wholesaler
ClientNumber		String	Client number

Help page - Model - Response object

9.1.4 POST (Multiple AddOrUpdate)

Add or update multiple client products at once to the database.

The result object contains only information about the errors.
Validate the errors to see if products were added successfully. If a product key is not found in the Errors or ModelStateErrors object, it is safe to assume the product was saved successfully.
Each error has a key which contains: Key1 = WholesalerArticlenumber and Key2 = ClientNumber.

9.1.4.1 Url

api/ClientProducts/Multiple/AddOrUpdate

9.1.4.2 Dependencies

- Wholesaler product
- Client
- Product

9.1.4.3 Fields

Field	Key	Type	Remarks
WholesalerArticleNumber		String	KeyField Unique code for the wholesaler
ClientNumber		String	Client number

Help page - Model - Response object

9.1.5 PUT

Update a client product.

9.1.5.1 Url

api/ClientProducts

9.1.5.2 Dependencies

- Wholesaler product
- Client
- Product

9.1.5.3 Fields

Field	Key	Type	Remarks
WholesalerArticleNumber		String	KeyField Unique code for the wholesaler
ClientNumber		String	Client number

Help page - Model - Response object

9.1.6 PUT (Multiple)

Update multiple client products at once to the database.

The result object contains only information about the errors.
Validate the errors to see if products were added successfully. If a product key is not found in the Errors or ModelStateErrors object, it is safe to assume the product was saved successfully.
Each error has a key which contains: Key1 = WholesalerArticlenumber and Key2 = ClientNumber.

9.1.6.1 Url

api/ClientProducts/Multiple

9.1.6.2 Dependencies

- Wholesaler product
- Client
- Product

9.1.6.3 Fields

Field	Key	Type	Remarks
WholesalerArticleNumber		String	KeyField Unique code for the wholesaler
ClientNumber		String	Client number

Help page - Model - Response object

9.1.7 DELETE

Delete a client product.

9.1.7.1 Url

api/ClientProducts

9.1.7.2 Parameters

Name	Type	Remarks
WholesalerArticleNumber	string	Article number
ClientNumber	string	Client number

Help page - Response object

9.1.8 DELETE (All)

Delete all client products of one client.

9.1.8.1 Url

api/ClientProducts

9.1.8.2 Parameters

Name	Type	Remarks
ClientNumber	string	Client number

Help page - Response object

10. Client orders

10.1 Retrieve client orders

After a client has placed an order, this order can be retrieved by use of the API.
You can determine if you want all client orders or just the orders of one specific client. Another option is to retrieve processed orders or not in combination with previous filters. By default all non processed orders will be retrieved. By default the most recently added orders are listed first, this can be overridden by setting the 'OldestFirst' parameter to 'true'.

In addition, if the client has requested any of the available free samples, these will also be returned as part of the order, in the 'FreeSample' response field.

Following options are provided to filter your results:
- fetch products by use of paging

10.1.1 GET

10.1.1.1 Url

api/ Clients/GetClientOrders

10.1.1.2 Parameters

Name	Type	Remarks
ClientNumber	String	Optional
OldestFirst	Boolean	Optional
Page	Integer	Optional
PageSize	Integer	Optional
Processed	Boolean	Mandatory

10.1.1.3 Fields

Field	Key	Type	Remarks
OrderNumber		String	Unique code for the order
Client		Client	Client information

Help page - Model - Response object

10.2 Process client orders

10.2.1 PUT

Tell the API the client order has been received and processed so that the GET endpoint will no longer return this.

10.2.1.1 URL

api/ClientOrders/ProcessOrders

10.2.1.2 Dependencies

- Client order

10.2.1.3 Fields

Name	Type	Remarks
ClientNumber	String	Mandatory

Help page - Model - Response object

11. Errors

11.1 By day

11.1.1 GET

Retrieve the latest known errors of API calls for a specific day.

11.1.1.1 Url

api/ErrorLogs

11.1.1.2 Parameters

Name	Type	Remarks
Date	Date	Mandatory

11.1.1.3 Fields

Help page – Model – Response object

11.2 With paging

11.2.1 GET

Retrieve the latest known errors of API calls.
This endpoint uses paging to retrieve records.

11.2.1.1 Url

api/ErrorLogs

11.2.1.2 Parameters

Name	Type	Remarks
Page	Integer	Mandatory

11.2.1.3 Fields

Help page – Model – Response object