List Comparison API v1

Document Revision 1.1
Date of Issue: 08 July 2022
Date of revision: 21 April 2023

Barnabas Mullan
Business Analyst

Table of Contents

1. Purpose

To provide the API end point information and examples of the web services available for the List Comparison API

2. Glossary of Terms

Term	Meaning
LWIN	LWIN - the Liv-ex Wine Identification Number – serves as a universal wine identifier for the wine trade. A unique seven to eighteen-digit numerical code used to quickly and accurately identify a product. LWIN allows wine companies to keep their preferred naming system, while introducing a new universal code.
Wine	The word wine below is referring to a specific wine (the producer and brand, grape or vineyard), vintage and unit size combination.

3. Technical Standards

- Permitted users will be issued with a unique token (CLIENT_KEY) and password (CLIENT_SECRET) combination to control the access for all the web services covered under Exchange Integration.
- The web services will consume and produce both XML and JSON. The user can provide the contents type in the request header. If the user does not provide any information, then the default content type will be JSON.
- The project will support ISO 8601.
- The project will only support HTTPS protocol for client and server communications.
- The API will support the following methods:
    - POST for create operation
- Pretty printing for output readability only is supported if required
- Compression for bandwidth savings are used
- Authentication mechanism will be custom based on CLIENT_KEY and CLIENT_SECRET
- The APIs will be accessible at https://api.liv-ex.com/ followed by their specific base URIs

4. Request Header

This information will be used to authenticate valid access to the REST API. Each user will have to provide the following information in the request header. Please note that the API expects the 4 headers as listed within this documentation and submitting a request with additional headers may lead to errors and/or failed responses.

Param

Name	Mandatory	Description
CLIENT_KEY	Y	A valid merchant GUID which will be unique for each merchant.
CLIENT_SECRET	Y	Password/Secret for the merchants CLIENT_KEY.
ACCEPT	Y	Accept header is a way for a client to specify the media type of the response content it is expecting.

		The values for the content type will be application/json or application/xml. If no/invalid content type is found in the request, then JSON format will be used by default.
CONTENT-TYPE	Y for POST requests	Content-type is a way to specify the media type of request being sent from the client to the server. The values for the content type will be application/json or application/xml. If no/invalid content type is found in the request, then JSON format will be used by default.

e.g.
CLIENT_KEY: 94B5CC70-BC3D-49C3-B636-C3C7552E543D
CLIENT_SECRET: merchantpasswd
ACCEPT: application/json
CONTENT-TYPE: application/json

Invalid header JSON response

```
{
    "status": "Unauthorized",
    "httpCode": "401",
    "message": "Request was unsuccessful",
    "livexCode": "R000"
    "apiInfo": {
        "version": "1.0",
        "timestamp": 1518524979121,
        "provider": "Liv-ex"
    }
}
```

Invalid header XML response

```
<Response>
    <Status>Unauthorized</Status>
    <HttpCode>401</Code>
    <Message>Request was unsuccessful.</Message>
    <LivexCode>R001</LivexCode>
     <ApiInfo>
        <Version>1.0</Version>
        <Timestamp>2022-07-01T11:12:30</Timestamp>
        <Provider>Liv-ex</Provider>
    </ApiInfo>
<Response>
```

5. API Listing

5.1 List Comparison service (POST method)

Description

This service will be used to call prices from the stock lists of specified peers.

Base URI

data/v1/listComparison

URI Pagination Parameters

Name	Mandatory	Description
?limit	N default = all results	Values: Any value **Type:** integer
?offset	N default = 1	**Type:** alphanumeric

If the limit is not specified in the request then the API returns all results on the offset 1.
Example base URI including pagination: /data/v1/listComparison?limit=500&offset=1

When using the listName request parameter for large stock lists, it is recommended to use a pagination limit of no more than 1000, and increment the offset value in subsequent API calls to consume all the data.

Request parameters

Name	Mandatory	Description
lwin	Y	A list of valid lwin18, lwin16 or lwin11. LWINs submitted must be of the same length. Type: alphanumeric array
currency	Y	Any possible trading currency, including the btt option. Type: alphanumeric
listID	Y (unless LWIN is supplied)	When providing with the value “Stock List” this parameter will retrieve List Comparison data for every LWIN saved to your current stock list. This parameter is ignored if you specify values with the “lwin” request parameter. Type: alphanumeric
lwinDepth	N (default value LWIN11)	Determines the length of LWIN considered when using the listID request parameter. Possible values are “lwin11”, “lwin16” and “lwin18”. LWINs longer than the specified lwinDepth will be truncated to the specified lwinDepth. Please see the Operational Notes section below to see how this API utilises LWINs of different lengths. Type: alphanumeric
merchantName	N	A list of valid merchant names from your peer list. Valid inputs can be obtained by calling the Merchant Names API. If no values are provided for this parameter, the API will call prices against all merchants on your peer list. Type: alphanumeric

Operational notes:

1. A request must only contain LWINs of one length (i.e. all LWIN16s or all LWIN18s) or it will be rejected. Please use multiple requests if querying at a variety of LWIN lengths.
2. Price data values are returned differently depending on the LWIN length submitted.

LWIN Length	Data Response
LWIN11	Considers all pack and bottle format combinations for the vintage specified by the LWIN11, and returns the best 9L equivalent price.
LWIN16	Considers all prices corresponding to the bottle size format specified by the LWIN16, and returns the value of a single bottle.
LWIN18	Considers prices that correspond to the pack and bottle format specified by the LWIN18, and returns the best unit price.

Sample Request Body

JSON Request

```
{
    "listComparison":{
        "currency":"gbp",
        "lwin":[
10024382004],
        "merchantName":["Example Merchant 1","Example Merchant 2","Example Merchant 3"]
    }
}
```

XML Request

```
<listComparisonRequest>
    <listComparison>
        <currency>gbp</currency>
        <lwin>10024382004</lwin>
        <merchantName>Example Merchant 1</merchantName>
        <merchantName>Example Merchant 2</merchantName>
        <merchantName>Example Merchant 3</merchantName>
    </listComparison>
</listComparisonRequest>
```

Response parameters

Name	Description
lwin	LWIN11/16/18 Type: alphanumeric
iwp	Weblink to the Liv-ex exchange for this wine. Type: alphanumeric
merchantName	Name of the merchant who owns the list price Type: alphanumeric
listPrice	List price Type: integer
listPriceDate	Date that the list price was recorded Type: date (Epoch if in JSON)
vintage	Vintage of the wine. Type: alphanumeric
packSize	Number of bottles in the pack

	Type: alphanumeric
bottleSize	Volume of the bottles Type: alphanumeric
lwinName	The LWIN display name of the wine. Type: alphanumeric
currency	The currency provided in the input Type: alphanumeric

Sample Response Body

JSON Response

The response is sent per request. The example below shows the response for the JSON Request example.


```
{
    "status": "OK",
    "httpCode": "200",
    "message": "Request completed successfully",
    "internalErrorCode": "R001",
    "apiInfo": {
        "version": "1.0",
        "timestamp": 1657206467248,
        "provider": "Liv-ex"
    },
    "pageInfo": {
        "totalResults": 1,
        "limit": 1,
        "offset": 1
    },
    "listComparison": {
        "currency": "GBP",
        "list": [
            {
                "lwin": "10024382004",
                "lwinName": "Glaetzer, Shiraz, Barossa Valley",
                "vintage": "2004",
                "packSize": "12",
                "bottleSize": "00750",
                "iwp": "https://app.liv-ex.com/#/wine-page?lwin11=10024382004",
                "values": [
                    {
                        "merchantName": "Example Merchant 1",
                        "listPrice": 999.9,
                        "listPriceDate": 1648771200000
                    },
                    {
                        "merchantName": "Example Merchant 2",
                        "listPrice": null,
                        "listPriceDate": null
                    },
                    {
                        "merchantName": "Example Merchant 3",
                        "listPrice": null,
                        "listPriceDate": null
                    }
                ]
            }
        ]
    },
    "errors": null
}
```

Invalid JSON response


```
{
    "status": "Bad Request",
    "httpCode": "400",
    "message": "Request was unsuccessful",
    "internalErrorCode": "R000",
    "apiInfo": {
        "version": "1.0",
        "timestamp": 1658224589161,
        "provider": "Liv-ex"
    },
    "pageInfo": null,
    "listComparison": null,
    "errors": {
        "error": [
            {
                "code": "V125",
                "message": "Requests with multiple LWINs must contain LWINs of the same
length."
            }
        ]
    }
}
```


XML Response

The response is sent per request. The example below shows the response for the XML Request example.


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<listComparisonResponse>
    <Status>OK</Status>
    <HttpCode>200</HttpCode>
    <Message>Request completed successfully</Message>
    <InternalErrorCode>R001</InternalErrorCode>
    <ApiInfo>
        <Version>1.0</Version>
        <Timestamp>2022-07-11T15:37:41.402Z</Timestamp>
        <Provider>Liv-ex</Provider>
    </ApiInfo>
    <pageInfo>
        <totalResults>1</totalResults>
        <limit>1</limit>
        <offset>1</offset>
    </pageInfo>
    <listComparison>
        <currency>GBP</currency>
        <list>
            <lwin>10024382004</lwin>
            <lwinName>Glaetzer, Shiraz, Barossa Valley</lwinName>
            <vintage>2004</vintage>
            <packSize>12</packSize>
            <bottleSize>00750</bottleSize>
            <iwp>https://app.liv-ex.com/#/wine-page?lwin11=10024382004</iwp>
            <values>
                <merchantName>Example Merchant 1</merchantName>
                <listPrice>999.9</listPrice>
                <listPriceDate>2022-07-04T00:00:00Z</listPriceDate>            </values>
            <values>
                <merchantName>Example Merchant 2</merchantName>
                <listPrice xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:nil="true"/>
                <listPriceDate xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:nil="true"/>
            </values>
```

```
            <values>
                <merchantName>Example Merchant 3</merchantName>
                <listPrice xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:nil="true"/>
                <listPriceDate xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:nil="true"/>
            </values>
        </list>
    </listComparison>
</listComparisonResponse>
```

Invalid XML Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<listComparisonResponse>
    <Status>Bad Request</Status>
    <HttpCode>400</HttpCode>
    <Message>Request was unsuccessful</Message>
    <InternalErrorCode>R000</InternalErrorCode>
    <ApiInfo>
        <Version>1.0</Version>
        <Timestamp>2022-07-19T09:58:19.483Z</Timestamp>
        <Provider>Liv-ex</Provider>
    </ApiInfo>
    <errors>
        <error>
            <code>V125</code>
            <message>Requests with multiple LWINs must contain LWINs of the same
length.</message>
        </error>
    </errors>
</listComparisonResponse>
```

6. Response Codes

This section describes the response codes that will be returned by the Exchange Integration services.

Code	Message
R000	Request was unsuccessful
R001	Request completed successfully
R002	Request partially completed

6.1 Request validation error codes

Code	Message
V018	Mandatory field missing(Please provide either a valid LWIN or ListID value)
V058	Invalid / incorrect LWIN: [${v}]. Please provide a valid LWIN11, LWIN16 or LWIN18 code.
V061	Invalid / incorrect currency: [%s]. Possible values are 'gbp', 'eur', 'chf', 'usd', 'hkd', 'jpy', 'sgd', 'gbpbtt', 'eurbtt', 'chfbtt', 'gbp/btt', 'eur/btt', 'chf/btt'.
V125	Requests with multiple LWINs must contain LWINs of the same length.

V173	Invalid Merchant Name [%s]. Please contact your account manager.
V174	Invalid / incorrect listID: [%]. Please provide a valid listID value.
V175	The listID provided does not contain any LWINs of the specified lwinDepth [${v}].

6.2 HTTP Status codes

HTTP defines a bunch of meaningful status codes that can be returned from our API. These can be leveraged to help our API Merchants/consumers route their responses accordingly:

Code	Message
200 OK	Response to a successful GET, POST, PUT, DELETE. Can also be used for a POST that doesn't result in a creation.
201 Created	Response to a POST that results in a creation.
202 Accepted	The request has been accepted and will be processed later. It is a classic answer to asynchronous calls (for better UX or performances).
204 No Content	Response to a successful request that won't be returning a body (like a DELETE request)
400 Bad Request	The request is malformed, such as if the body does not parse
401 Unauthorized	When no and/or invalid authentication details are provided. Can also be used to trigger an auth popup if API is used from a browser
403 Forbidden	When authentication succeeded but authenticated user doesn't have access to the resource
404 Not Found	When a non-existent resource is requested
405 Method Not Allowed	When an HTTP method is being requested that isn't allowed for the authenticated user
406 Not Acceptable	Nothing matches the Accept-* Header of the request. As an example, you ask for an XML formatted resource, but it is only available as JSON.
409 Conflict	Indicates one or more supplied parameters are triggering a validation error. A relevant TR code should be returned in the response.
410 Gone	Indicates that the resource at this end point is no longer available. Useful as a blanket response for old API versions
415 Unsupported Media Type	If incorrect content type was provided as part of the request

422 Unprocessable Entity	Used for validation errors. Should be used if the server cannot process the entity, e.g. if an image cannot be formatted or mandatory fields are missing in the payload.
429 Too Many Requests	When a request is rejected due to rate limiting
500 Internal Server Error	The general catch-all error when the server-side throws an exception. The request may be correct, but an execution problem has been encountered at our end.