List Manager API v1

Document Revision 1.0
Date of Issue: 18 July 2023

Fred Haselton
Business Analyst

Table of Contents

1. Purpose

To provide the API end point information and examples of the List Manager API v1. The web service handles POST, PATCH and DELETE methods to facilitate the creation of new lists, editing of existing lists, or deleting of existing lists.

2. Glossary of Terms

Term	Meaning
listID	Identifier for each individual product list uploaded to the Liv-ex system
lineID	Identifier for each individual line within the list
listType	The type of list to help differentiate the listID between “Custom List”, “Saved From Search”, “Wishlist”, “Watch List”.
LWIN	LWIN - the Liv-ex Wine Identification Number – serves as a universal wine identifier for the wine trade. LWIN is a unique seven to eighteen-digit numerical code that can be used to quickly and accurately identify a product. LWIN allows wine companies to keep their preferred naming system, while introducing a new universal code.

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
    - PATCH for edit operation
    - DELETE for delete operation
- Pretty printing for output readability only is supported if required.
- Compression for bandwidth savings are used.
- For HTTP users who can only work on GET & POST methods, we provide a Header ‘X-HTTP-Method-Override’ for DELETE
- Authentication mechanism will be custom based on CLIENT_KEY and CLIENT_SECRET.
- For any PUSH services we require a direct POST URL which should be backed by

as service capable of accepting and process XML payload as POST request.

- 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 user GUID which will be unique for each merchant
CLIENT_SECRET	Y	Password/Secret for the user CLIENT_KEY
ACCEPT	Y	Accept header is a way for the user to specify the media type of the response content it is expecting. Th 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.

Example Header (JSON)

```
CLIENT_KEY: 12A34BC56-DE7F-89G0-H1J2345K678L
CLIENT_SECRET: dummy_password
ACCEPT: application/json
CONTENT-TYPE: application/json
```

Invalid header (JSON response)

```
{
    "status": "Unauthorized",
    "httpCode": "401",
    "message": "Unauthorized",
    "internalErrorCode": null,
```

```
    "apiInfo": {
        "version": "1.0",
        "timestamp": 1550676412005,
        "provider": "Liv-ex"
    }
}
```

Invalid header (XML response)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="https://aby-uat-
api.liv-ex.com/v1 https://aby-uat-api.liv-ex.com/schema/v1/services.xsd">
    <Status>Unauthorized</Status>
    <HttpCode>401</HttpCode>
    <Message>Unauthorized</Message>
    <InternalErrorCode xsi:nil="true"/>
    <ApiInfo>
        <Version>1.0</Version>
        <Timestamp>2019-02-20T15:28:48.623Z</Timestamp>
        <Provider>Liv-ex</Provider>
    </ApiInfo>
</Response>
```

5. API Listing

5.1 Add Custom List Service – POST method

Description

This service will be used to create new Custom List. A successful POST request will be responded with a listID value that should be recorded. This listID can then be used alongside Line Manager POST to products to the custom list. This listID can also be used in edit (PATCH) and delete (DELETE) requests to manipulate the Custom List accordingly.

Base URI

listAnalysis/v1/listManager

Request parameters

Name	Mandatory	Description
listName	Y	The name of the list. If the provided listName parameter value exceeds the character limit of this field, then it will be truncated down to the max character limit. If the provided listName parameter value is identical to an existing listName, then the new listName will be appended with a number. Example: My List (1)

		Type: alphanumeric (50 character limit)
note	N	A note associated with the list. If the provided note parameter value exceeds the character limit of this field, then it will be truncated down to the max character limit. Type: alphanumeric (250 character limit)
listType	N Default = custom list	The list type associated with the list. The possible list types include: “Custom List”, “Saved From Search”, “Wishlist”, “Watch List”. Type: alphanumeric

Response Parameter:

Name	Description
listID	Unique identifying key attached to each list, generated upon a successful call. Type: 128-bit hexadecimal
listName	Name of the list, as supplied in the request, with any truncations applied. Type: alphanumeric
linesMatched	Number of lines in the list matched to LWIN. Type: integer Always 0 at the moment of creation
linesUnmatched	Number of lines in the list not matched to LWIN. Type: integer Always 0 at the moment of creation
linesTotal	Total number of lines in the list that are non-deleted. I.e. linesMatched + linesUnmatched. Type: integer Always 0 at the moment of creation
createdDate	The date that the list was created. Type: datetime (ISO8601) / EPOCH for JSON
createdBy	The name of the user that created the list Type: alphanumeric
note	Any user submitted note applied at a list level Type: alphanumeric

listType	The type of list that has been uploaded. I.e “Custom List”, “Saved From Search”, “Wishlist”, “Watch List”. Type: alphanumeric

Sample Request Body

JSON Request

```
{
   "listManager": {
      "listName": "Name of list",
      "note": "line manager POST notes",
      "listType": "custom list"
   }
}
```

XML Request

```
<root>
<listManager>
<listName>Name of list</listName>
<note>line manager POST notes</note>
<listType>custom list</listType>
</listManager>
</root>
```

Sample Response Body

JSON response

```
{
   "status": "OK",
   "httpCode": "200",
   "message": "Request completed successfully",
   "internalErrorCode": "R001",
   "apiInfo": {
      "version": "1.0",
      "timestamp": 1689594529330,
```

```
    "provider": "Liv-ex"
  },
  "listManagerResponse": {
    "listID": "0bbcf74f-4eeb-4f9d-bea0-366d16c12348",
    "listName": "Name of list",
    "linesMatched": 0,
    "linesUnmatched": 0,
    "linesTotal": 0,
    "createdDate": 1689594529324,
    "createdBy": "Fred Haselton",
    "note": "line manager POST notes",
    "listType": "Custom List"
  },
  "errors": null
}
```


XML Response


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<root>
  <Status>OK</Status>
  <HttpCode>200</HttpCode>
  <Message>Request completed successfully</Message>
  <InternalErrorCode>R001</InternalErrorCode>
  <ApiInfo>
    <Version>1.0</Version>
    <Timestamp>2023-07-17T14:56:20.987Z</Timestamp>
    <Provider>Liv-ex</Provider>
  </ApiInfo>
  <listManagerResponse>
    <listID>ae0a234c-5b4a-4d46-80c2-d3279d123456</listID>
    <listName>Name of list (1)</listName>
    <linesMatched>0</linesMatched>
    <linesUnmatched>0</linesUnmatched>
    <linesTotal>0</linesTotal>
    <createdDate>2023-07-17T14:56:20.982Z</createdDate>
    <createdBy>Fred Haselton</createdBy>
    <note>line manager POST notes</note>
```

```
    <listType>Custom List</listType>
  </listManagerResponse>
</root>
```

5.2 Edit Custom List Service – PATCH method

Description

This service will be used to amend the editable metadata of a list (listName, note, listType). It will require an input of the listID. The API response will return the updated metadata. Any live lists that belong to your company can be edited. It is only possible to edit lists that belong to the account calling the API.

Base URI

listAnalysis/v1/listManager

Request parameters

Name	Mandatory	Description
listID	Y	Unique identifying key attached to each list. Type: 128-bit hexadecimal
listName	N	Name of the list. If not supplied in the request as a parameter, we will maintain the previous value. If supplied blank, a validation will be triggered. Type: alphanumeric
listType	N	Accepted values: “Custom List”, “Saved From Search”, “Wishlist”, “Watch List”. Type: alphanumeric
note	N	Any user submitted note applied at a list level. If not supplied in the request as a parameter, we will maintain the previous value. Type: alphanumeric

Response Parameters

Name	Description
listID	Unique identifying key attached to each list Type: 128-bit hexadecimal
listName	Name of the list Type: alphanumeric

	Character limit is 50
linesMatched	Number of lines in the list matched to LWIN Type: integer
linesUnmatched	Number of lines in the list not matched to LWIN Type: integer
linesTotal	Total number of non-deleted lines in the list. Total = unmatched + matched. Type: integer
lastAccessedDate	The date that the list was last accessed by a user Type: datetime / EPOCH for JSON
lastModifiedDate	The date that the list was last modified by a user Type: datetime / EPOCH for JSON
createdBy	The name of the user that created the list Type: alphanumeric
note	Any user submitted note applied at a list level Type: alphanumeric
listType	Type: alphanumeric
listStatus	Type: alphanumeric
lastModifiedBy	The name of the user that last modified the list. Type: alphanumeric

Sample Request Body

JSON request

```
{
   "listManager": {
      "listID": "0bbcf74f-4eeb-4f9d-bea0-366d16c12348",
      "listName": "Name of list",
      "listType": "Custom List",
      "note": "Change the text of this note"
   }
}
```

XML request

```
<root>
<listManager>
<listID>0bbcf74f-4eeb-4f9d-bea0-366d16c12348</listID>
<listName>Name of list</listName>
<listType>Custom list</listType>
<note>Change the text of this note.</note>
</listManager>
</root>
```

Sample Response Body

JSON response


```
{
   "status": "OK",
   "httpCode": "200",
   "message": "Request completed successfully",
   "internalErrorCode": "R001",
   "apiInfo": {
      "version": "1.0",
      "timestamp": 1689608226486,
      "provider": "Liv-ex"
   },
   "listManagerResponse": {
      "listID": "0bbcf74f-4eeb-4f9d-bea0-366d16c12348",
      "listName": "Name of list",
      "linesMatched": 0,
      "linesUnmatched": 0,
      "linesTotal": 0,
      "lastAccessedDate": 1689605679465,
      "lastModifiedDate": 1689608226476,
      "createdBy": "Fred Haselton",
      "lastModifiedBy": "Liv-ex",
      "note": "Change the text of this note",
      "listType": "Custom List",
      "listStatus": "live"
   },
   "errors": null
}
```


XML response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<root>
  <Status>OK</Status>
  <HttpCode>200</HttpCode>
  <Message>Request completed successfully</Message>
  <InternalErrorCode>R001</InternalErrorCode>
  <ApiInfo>
    <Version>1.0</Version>
    <Timestamp>2023-07-17T15:40:52.881Z</Timestamp>
    <Provider>Liv-ex</Provider>
  </ApiInfo>
  <listManagerResponse>
    <listID>0bbcf74f-4eeb-4f9d-bea0-366d16c12348</listID>
    <listName>Name of list</listName>
    <linesMatched>0</linesMatched>
    <linesUnmatched>0</linesUnmatched>
    <linesTotal>0</linesTotal>
    <lastAccessedDate>2023-07-17T14:54:39.465Z</lastAccessedDate>
    <lastModifiedDate>2023-07-17T15:40:52.870Z</lastModifiedDate>
    <createdBy>Fred Haselton</createdBy>
    <lastModifiedBy>Liv-ex</lastModifiedBy>
    <note>Change the text of this note.</note>
    <listType>Custom List</listType>
    <listStatus>live</listStatus>
  </listManagerResponse>
</root>
```


5.3 Delete Custom List Service – DELETE method

Description

This service will be used to delete a listID. In the response, the API should return a confirmation of both the input listID and confirmation that it has been deleted successfully. It is only possible to delete lists that belong to the account calling the API.

Base URI

listAnalysis/v1/listManager

Request parameters

Name	Mandatory	Description
listID	Y	List ID of the list to be deleted.

		128-bit hexadecimal

Response parameters

Name	Description
listID	List ID of the deleted list. Type: 128-bit hexadecimal

Sample Request Body

JSON request

```
{
  "listManager":{
    "listID":"0bbcf74f-4eeb-4f9d-bea0-366d16c12348"
  }
}
```

XML request

```
<root>
  <listManager>
    <listID>0bbcf74f-4eeb-4f9d-bea0-366d16c12348</listID>
  </listManager>
</root>
```

Sample Response Body

JSON response

```
{
  "status": "OK",
  "httpCode": "200",
  "message": "Request completed successfully",
  "internalErrorCode": "R001",
  "apiInfo": {
    "version": "1.0",
    "timestamp": 1689608952355,
    "provider": "Liv-ex"
  },
  "listManagerResponse": {
    "listID": "0bbcf74f-4eeb-4f9d-bea0-366d16c12348
```

```
  },
  "errors": null
}
```


XML response


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<root>
  <Status>OK</Status>
  <HttpCode>200</HttpCode>
  <Message>Request completed successfully</Message>
  <InternalErrorCode>R001</InternalErrorCode>
  <ApiInfo>
    <Version>1.0</Version>
    <Timestamp>2023-07-17T15:47:13.896Z</Timestamp>
    <Provider>Liv-ex</Provider>
  </ApiInfo>
  <listManagerResponse>
    <listID>0bbcf74f-4eeb-4f9d-bea0-366d16c12348listID>
  </listManagerResponse>
</root>
```


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	Trigger
V018	Mandatory field missing [%s].	Validation trigger when attempting to pass a request without the mandatory field(s) being supplied.

V174	Invalid/incorrect listID: [%]. Please provide a valid listID value.	When a listID is supplied while using the PATCH or DELETE service that does not exist, does not match the listIDs associated with the calling account or has a status other than live.
V178	Invalid list name [%s]. List names must not be blank.	When the user supplies a totally blank listName.
V179	Invalid listType [%s]. Accepted value are “Custom List”, “Saved From Search”, “Wishlist”, “Watch List”.	When the user attempts to submit a PATCH request with the invalid listType value.

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.