Verification Router Service
Technical Specifications for:

1. Responder Connectivity Information (CI) upload to Look-Up Directory (LD)
2. LD Synchronization

Version:	1.10
Date:	12-September-2018

Table of Content

Revision History

Version	Date	Revision Description
0.1	12-APR-2018	Initial draft assembled from sub-work group output
0.2	13-APR-2018	GUID clarification
0.3	20-APR-2018	updates from sub-work stream team meeting
0.4	14-MAY-2018	updates from tech work stream meeting 11-May-2018
1.0	24-MAY-2018	Format and release as v1.0
1.1	25-JUNE-2018	Suggestions from TraceLink
1.2	29-JUNE-2018	Suggestions from ADENTS
1.2	03-JULY-2018	Additional feedback and comments from TraceLink
1.3	25-JULY-2018	VRS Workshop changes: • removed ownerGln • Renamed GCP to recordOwner • Changed responderCi to Ci and decided to only track Ci • Renamed isActive to status and changed from Boolean to string • Renamed newResponderGln to newRecordOwner • Added sourceVrsId • Consolidated the LD attribute list table into 1 to support both interactions 1 & 2.
1.4	01-AUG-2018	• Cleaned-up LD field definitions • Physically removed content marked for deletion based on workshop discussions • Added specification for GET API in support of PULL synchronization method • Added specification for PUSH synchronization method • Added draft list of HTTP status codes in the Error Handling/Exceptions section • Added message protocol and message security sections based on July 26 workshop decision log
1.5	03-AUG-2018	Applied changes to address workgroup feedback during 03-Aug-2018 VRS Technical workgroup meeting.

		• Changed the length of recordOwner and nextRecordOwner to 6. This accommodates the existing 4 or 5 digit FDA labeler code and the anticipated 6 digit FDA labeler code. Made corresponding description changes in other sections. • Added field validation statement for checking that labeler code imbedded in the GTIN matches the recordOwner field value for the initial LD record entry for a given GTIN. • Removed recordOwner and sourceVrsId from the fields for logging since these are redundant • Specified UTC for lastModifiedDateTime and removed redundant explanation on format • Included the minimum / maximum frequency for LD synch / refresh in the Pull sync section. • Since guid has been renamed to recordGuid and guid in the log has been renamed to logGuid, updated examples to refer to recordGuid instead of guid. • For readability, reformatted example Push message to reorder the LD attributes.
1.6	24-AUG-2018	• Changed occurrences of property names that were not formatted as camel cased. Changed recordGUID to recordGuid, logGUID to logGuid and sourceVRSId to sourceVrsId. • Cleaned up the property names listed in the Pull and Push Sync Method description and message examples in section 1.4 to ensure camel case formatting and remove extraneous property names. • Initiated removal of comments
1.7	29-AUG-2018	• Changed lingering instance of logGUID to logGuid for came casel formatting. • Changed PushSynchronization to pushsynchronization in example of Push method.
1.8	06-SEPT-2018	• Added JSON Schema • Addressed comment to spell out PI by changing the section title to Product Identifier (PI) in 1.2.2

		• Removed other comments since the workgroup determined they were either addressed or being addressed. • Updated lastModifiedDateTime format to always specify 3 digits to represent decimal fraction of a second. The date format is now YYYY-MM-DDThh:mm:ss.***sss***TZD (e.g. 2018-06-16T19:20:30.450Z)and expressed in UTC).
1.9	07-SEPT-2018	• Changed the retrieval condition for the get method to the records whose lastModifiedDateTime is greater than the lastModifiedDateTime parameter value. Previously, the lastModifiedDateTime filter was set to greater than or equal. • Updated the recordGuid example in the Push method to use the pattern defined in the JSON schema.
1.10	12-SEPT-2018	• Pertaining to the change in the retrieval condition for the get method, add commentary to restore the lastModifiedDateTime to “greater than or equal” to instead of just “greater” for the purpose of ensuring the capture of multiple records having the same time stamp and guarding against skipping them in case of server boundary record size constraints.

1. Verification Router Service: Request and Response Messaging Standard

1.1. Background on VRS

Healthcare Distribution Alliance (HDA) formed the Traceability Pilots Work Group in 2016 to focus on a pilot study of nine (9) potential methods or solutions to meet the 2019 Saleable Returns DSCSA Requirements. Through the process of evaluating nine scenarios, the Work Group acknowledged no single solution for the supply chain existed, and put forward two preferred options, keeping in mind solution cost, implementation effort, process execution, exception handling, and other advantages and disadvantages.

One of the options studied in the pilot was a Verification Router Service (VRS). A proof of concept was successfully built and utilized during a live pilot, but it was a temporary system designed solely for the purposes of the pilot. (For the full pilots report, see https://healthcaredistribution.org/resources/hda-saleable-returns-pilots-report.) At the conclusion of the pilot study, the Work Group concluded that the Verification Router Service was a verification method that should be pursued.

HDA subsequently formed a task force consisting of industry members and later expanded the team to include solution providers in order to mobilize efforts for defining the business requirements, high-level functionality, solution components, component interactions, minimal required data and recommended communications protocols. With assistance from KPMG LLP, HDA and the task force members conducted virtual meetings and in-person workshops throughout 2017. The resulting output is consolidated into two documents:

1. VRS Business Requirements Document (BRD)
2. Solution Architecture Reference Document (SARD)

Additional technical specifications are required to support the framework provided in the BRD and SARD documents.

One of these specifications is to define the format and content for the following processes:

1. Connectivity Information (CI) upload to a Look-Up Directory (LD) in support of Interaction 1 from the SARD.
2. LD synchronization across VRS Solution Providers managing/using LD information

It is anticipated that this document will be used by solution providers to support their build and testing activities in 2018. Once approved, frequent updates to this document are not anticipated.

1.2. Use cases for publishing/broadcasting updates

Push method will be used to broadcast LD changes (delta). Use cases triggering the push are described below.

Push method will also be used to populate LD of the new VRS provider (full update). This process will be described in the relevant governance document, describing the onboarding of the new VRS provider.

Product discontinuation for any reason (recall, withdrawal, voluntary exit of the product from the market, etc.) will not result in LD update and will be handled by the Responder.

1.2.1. New GTIN

New GTIN needs to be added to the LD for a brand-new product or for existing product when sourcing of the product changes or when packaging changes.

Example: Manufacturer X was packing a blister of 10 pills into the serialized carton, but made the decision to pack two blisters of 10 pills into the serialized carton – new GTIN would need to be obtained by the manufacturer and added to LD.

1.2.2. Product Identifier (PI) repository change

PI repository change occurs due to product ownership change such as product divestiture or company merger and acquisition (M&A), or due to solution change to replace PI repository or to replace VRS provider.

Example 1: Manufacturer A sells the product identified by GTIN 123 to Manufacturer B. Manufacturer B intends to start manufacturing the product using the same GTIN 123, while storing the PI in their own repository.

LD need an update to set the “endExpDate” on existing GTIN 123 LD record, and to create new LD record for this GTIN with new “ResponderCi” and “startExpDate” after the “endExpDate” of the original GTIN 123 record.

Example 2: Manufacturer decides to migrate to a new PI repository solution and migrate all PI data from repository X to repository Y.

LD needs to update ResponderCi for all products whose PI information was moved from X to Y.

1.2.3 VRS Provider change

Responder chooses to change its VRS provider

Example 3: Manufacturer decides to start using a new VRS provider solution. LD update would need to be made to update sourceVrsId.

1.3. Look-up Directory (LD) Field Definitions for SARD Interaction 1 and 2

The LD fields defined in the following table specify the LD content for uploading CI for SARD interaction 1 and for synchronizing across VRS Providers for interaction 2.

1.3.1 **Field contents for Interactions 1 and 2**

#	Field Name	Req'd	Length	Type	Description; Sample Value
1.1	recordGuid	Y	8-4-4-4-12	String	Globally unique identifier for the LD record assigned by source VRS system; Based on Version4 UUID GUID will be shared between VRS Example: 3ab5d7b6-3fcb-4a25-86f5-02fe5f5761bd
1.2	recordOwner	Y	6	string	4, 5 or 6 digit FDA labeler code identifying the LD record owner
1.3	gtin	Y*	14	string	GS1 Global Trade Item Number Example: 00300015555015
1.4	ci	Y	255	string	The connectivity information (i.e. URL) of either designated VRS Provider or the GTIN owner if they are responding directly to verification requests. Example: https://example.connectivityinformation.responder
1.5	sourceVrsId	Y	13	string	Identifies the VRS that sourced the LD entry
1.6	startExpDate	Y	DB-Acceptable Date Format	Date	Lower bound of expiration date value. Scanned expiration date value must fall in between lower/upper bound when populated. Date format is YYMMDD
1.7	endExpDate	N	DB-Acceptable Date Format	Date	Upper bound of expiration date value. If populated, scanned expiration date value must fall in between lower/upper bound when populated. Required when nextRecordOwner is populated Date format is YYMMDD
1.8	status	Y	15	string	By default: active Possible values: active, inactive, deleted
1.9	nextRecordOwner	N	6	string	4, 5 or 6 labeler code identifying the owner of the next LD entry for this GTIN
1.10	lastModifiedDateTime	Y		dateTime	Date & Timestamp of when the LD record is modified. Format: YYYY-MM-DDThh:mm:ss.sssTZD Express in UTC

Field validations and other notes:

1. Field values should be validated for conformance to field length and type definitions.
2. startExpDate is required, must be a valid date value and startExpDate must be <= endExpDate.
3. If a value exists for nextRecordOwner, the endExpDate must not be null
4. If '00' value is encoded for 'DD' and scanned as PI data,it needs to be assumed that DD is last day of the month (per USP) in order to determine if EXP falls between start/end date when populated
5. The LD record entries cannot have an overlapping startExpDate and endExpDate for the same GTIN
6. To get the LD entries valid for a given expiration date from scanned PI data, the condition check should be startExpDate <= scannedExpirationDate <= endExpDate and status = active
7. Duplicate and cross record checks (i.e. date overlap) should only be applied to records with status = active.
8. The labeler code imbedded in the GS1 company prefix segment of the GTIN should match the recordOwner value of the initial LD record entry for a given GTIN.

1.3.2 **Field contents for logging / audit trail upon processing in the VRS for interaction 1 and 2**

For audit purpose every alteration of the LD should be logged. The content of this log should include in addition of the data specified above the following data:

#	Field Name	Req'd?	Length	Type	Description; Sample Value
LD1.1	logGuid	Y	8-4-4-4-12	String	globally unique identifier for the LD log record; Based on Version4 UUID Example: 998CDC77-6860-4351-9277-6F3E6F870AC6
LD1.2	dateTimeProcessed	Y		dateTime YYYY-MM-DDThh:mm:ss	the date/time that the record was processed in the LD; Example: 2018-04-16:T14:30:20
LD1.3	interactionType	Y	15	String	The VRS Solution Architecture interaction type to distinguish the nature of the LD entry. Possible values: interaction1, interaction2

- GTIN Authorization required for interaction 1; refer to governance business requirements; no technical specification required
- GTIN authorization required for interaction 1& 2 follow rules of record ownership:
    - Any LD entry (aka "record") can only be modified by the record owner originally associated by that entry
    - The record owner of an entry cannot be changed

  - New entry for a GTIN can only be made by either the record Owner of the last entry for this GTIN or by the record owner identified as “nextRecordOwner” in that entry.
- Data integrity checks and validation will be recommended for incoming data during synchronization.

1.4. Synchronization Methodology

1.4.1 PULL method using the GET method

The GET method is required for receiving LD systems 1) to catch-up and resume operations after experiencing down time 2) to perform health-checks to make sure LD master is up 3) to bootstrap and initialize the content of new receiving LD systems joining the network and 4) to retrieve and synchronize the LD changes from LD source systems.

LD receiving systems will refresh LD content from the source system at a minimum once per day and no more than once per hour.

By implementing the GET method, LD systems enable the various usages and requirements.

"Pull Sync" Message Format

GET /v1/ld?lastModifiedDateTime={*lastModifiedDateTime*}

- **lastModifiedDateTime** –qualifies the LD records to be retrieved from the source system. Limits the result set returned by the GET API to the records whose lastModifiedDateTime is greater than or equal to the lastModifiedDateTime parameter value. It is assumed that all the records in the source system with the lastModifiedDateTime greater than or equal to the lastModifiedDateTime parameter are returned in one batch.
- **lastModifiedDateTime** is formatted as YYYY-MM-DDThh:mm:ss.sssTZD (eg 2018-06-16T19:20:30.450Z) and expressed in UTC where
  YYYY = four-digit year
  MM = two-digit month (01=January, etc.)
  DD = two-digit day of month (01 through 31)
  hh = two digits of hour (00 through 23) (am/pm NOT allowed)
  mm = two digits of minute (00 through 59)
  ss = two digits of second (00 through 59)
  sss = 3 digits representing a decimal fraction of a second
  TZD = time zone designator (should specify Z for UTC)

Example of “Pull Sync” request to initialize the subscribing LD system
This example uses data populated from one of the source LD system i.e. VRS108. For initialization, the date of 1970-01-01T00:00:00.000Z is given as the lastModifiedDateTime:

```
GET https://example.com/v1/ld?lastModifiedDateTime=1970-01-01T00:00:00.000Z
HTTP/1.1
Host: 3and9.org
Accept: application/json
```

“Pull Sync” Response format

- **sourceVrsId -** Identifies the VRS that sourced the LD entry
- **ldEntries –** list of LD records which has been created or modified as of *lastModifiedDateTime* by the source LD system
- **recordGuid –** globally unique identifier for the LD record assigned by source VRS system; Based on Version4 UUID; GUID will be shared between VRS; Example guid value: 3ab5d7b6-3fcb-4a25-86f5-02fe5f5761bd
- **recordOwner –** 4, 5 or 6 digit FDA labeler code identifying the LD record owner
- **gtin** –GS1 global trade item number
- **ci** – The connectivity Information (i.e. URL) of either designated VRS Provider or the GTIN owner if they are responding directly to verification requests.
- **startExpDate –** lower bound of expiration date value. Scanned expiration date value must fall between lower/upper bound when populated. Date format is YYMMDD.
- **endExpDate –** upper bound of expiration date value. Scanned expiration date value must fall in between lower/upper bound when populated. Date format is YYMMDD.
- **status** – LD record status; by default, status is active. Possible values: active, inactive, deleted.
- **nextRecordOwner** – 4, 5 or 6 digit labeler code identifying the owner of the next LD entry for this GTIN
- **lastModifiedDateTime** – date & timestamp of when the LD record is modified. This will be formatted as YYYY-MM-DDThh:mm:ss.sssTZD and expressed in UTC. (eg 2018-06-16T19:20:30.450Z)

In this example, VRS108 retrieves and returns all of the LD records sourced by VRS108 whose lastModifiedDateTime is >= 1970-01-01T00:00:00.000Z. The records in the result set are ordered by ascending lastModifiedDateTime.

Example “Pull Sync” Response Message

```
HTTP 1.1 200 OK
Cache-Control: private, no-cache
Content-Type: application/json
{
        "sourceVrsId": "VRS108",
        "ldEntries" :
```

```
                    [{      "recordGuid": "3ab5d7b6-3fcb-4a25-86f5-02fe5f5761bd",
                            "recordOwner": "12345",
                            "gtin": "00312345555016",
                            "ci": "https://ex.connectivityinfo.responder",
                            "startExpDate": "170728",
                            "endExpDate": null,
                            "status": "active",
                            "nextRecordOwner": null,
                            "lastModifiedDateTime": "2018-05-20T21:15:45.250Z"},
                     {      "recordGuid": "3ab5d7b6-3fcb-4a25-86f5-02fe5f5761bd",
                            "recordOwner": "12345",
                            "gtin": "00312345555016",
                            "ci": "https://ex.connectivityinfo.responder",
                            "startExpDate": "170728",
                            "endExpDate": 201031,
                            "status": "active",
                            "nextRecordOwner": "24680",
                            "lastModifiedDateTime": "2018-07-15T21:16:45.450Z"},
                    {       "recordGuid": "bc2c5ba7-eecd-48bc-af35-f384193edb0f",
                            "recordOwner": "24680",
                             "gtin": "00312345555016",
                             "ci": "https://ex.connectivityinfo.responder",
                             "startExpDate": "201130",
                             "endExpDate": null,
                             "status": "active",
                             "nextRecordOwner": null,
                             "lastModifiedDateTime": "2018-07-16T21:15:45.450Z"},
                     {      "recordGuid": "09040f86-24d9-49f3-8eda-2ca2aa056e9a",
                            "recordOwner": "24680",
                             "gtin": "00324680555026",
                             "ci": "https://ex.connectivityinfo.responder",
                             "startExpDate": "210630",
                             "endExpDate": null,
                             "status": "active",
                             "nextRecordOwner": null,
                             "lastModifiedDateTime": "2018-07-22T21:15:45.450Z"}]
    }
```

Error Handling / Exceptions

When processing a GET request, the LD source system may respond with any of the HTTP codes listed in the following table:

Code	Description
200	A response code of 200 means the request was successful and details about the response can be found in the body of the response. Only a 200 response will issue a JSON payload.

400	The request was not formatted properly. Please verify the request conforms to this specification, and re-issue the request in the correct format.
401	The request was not allowed because the request did not pass authentication.
404	The requested resource does not exist.
408	The server timed out waiting for the request.
500	System failed to process the request because of an error inside the system.
503	System is undergoing maintenance or is otherwise temporarily unavailable for API queries.

1.4.2 **PUSH method**

Upon update in the LD, The VRS responsible for the change will notify other VRS as per interacion2 using the POST method.

The "push synchronization" enable up to date LD record across all VRS in real time.

The push message is posted to each VRS on record, every time a responder makes a modification to the LD.
The push method assumes that each VRS has an up-to-date list of other VRS and that each VRS has an API method to receive this notification.

"Push synchronization" message format

- **recordGuid –** globally unique identifier created by source VRS system. Based on Version4 UUID; GUID will be shared between VRS; Example guid value: 3ab5d7b6-3fcb-4a25-86f5-02fe5f5761bd
- **recordOwner –** 4, 5 or 6 digit FDA labeler code identifying the LD record owner
- **gtin** – GS1 global trade item number
- **ci** – The connectivity Information (i.e. URL) of either designated VRS Provider or the GTIN owner if they are responding directly to verification requests.
- **sourceVrsId -** Identifies the VRS that sourced the LD entry
- **startExpDate –** lower bound of expiration date value. Scanned expiration date value must fall between lower/upper bound when populated. Date format is YYMMDD.
- **endExpDate –** upper bound of expiration date value. Scanned expiration date value must fall in between lower/upper bound when populated. Date format is YYMMDD.
- **status** – LD record status; by default, status is active. Possible values: active, inactive, deleted.
- **nextRecordOwner** – labeler code identifying the owner of the next LD entry for this GTIN
- **lastModifiedDateTime** – date & timestamp of when the LD record is modified. This will be formatted as YYYY-MM-DDThh:mm:ss.sssTZD and expressed in UTC. (eg 2018-06-16T19:20:30.450Z)

“Push synchronization”sample message:

```
POST https://example.com/v1/ld/pushSynchronization/
HTTP/1.1
Content-Type: application/json

{
"recordGuid": "a1096c5c-d7b5-452e-bd2b-e494af47ac10",
"recordOwner": "0300",
"gtin": "00300015555015",
"ci": "https://ex.connectivityinfo.vrs",
"sourceVrsId": "VRS107",
"startExpDate": "190729",
"endExpDate": null,
"status": "active",
"nextRecordOwner": null,
"lastModifiedDateTime": "2018-06-20T21:15:45.450Z"
}
```

“Push synchronization” response format

When processing a push request, the receiver should respond with any of the HTTP codes listed in the following table:

Code	Description
200	A response code of 200 means the request was successful and details about the response can be found in the body of the response. Only a 200 response will issue a JSON payload.
400	The request was not formatted properly. Please verify the request conforms to this specification, and re-issue the request in the correct format.
401	The request was not allowed because the request did not pass authentication.
404	The requested resource does not exist.
408	The server timed out waiting for the request.
500	System failed to process the request because of an error inside the system.
503	System is undergoing maintenance or is otherwise temporarily unavailable for API queries.

1.5. Message Protocol

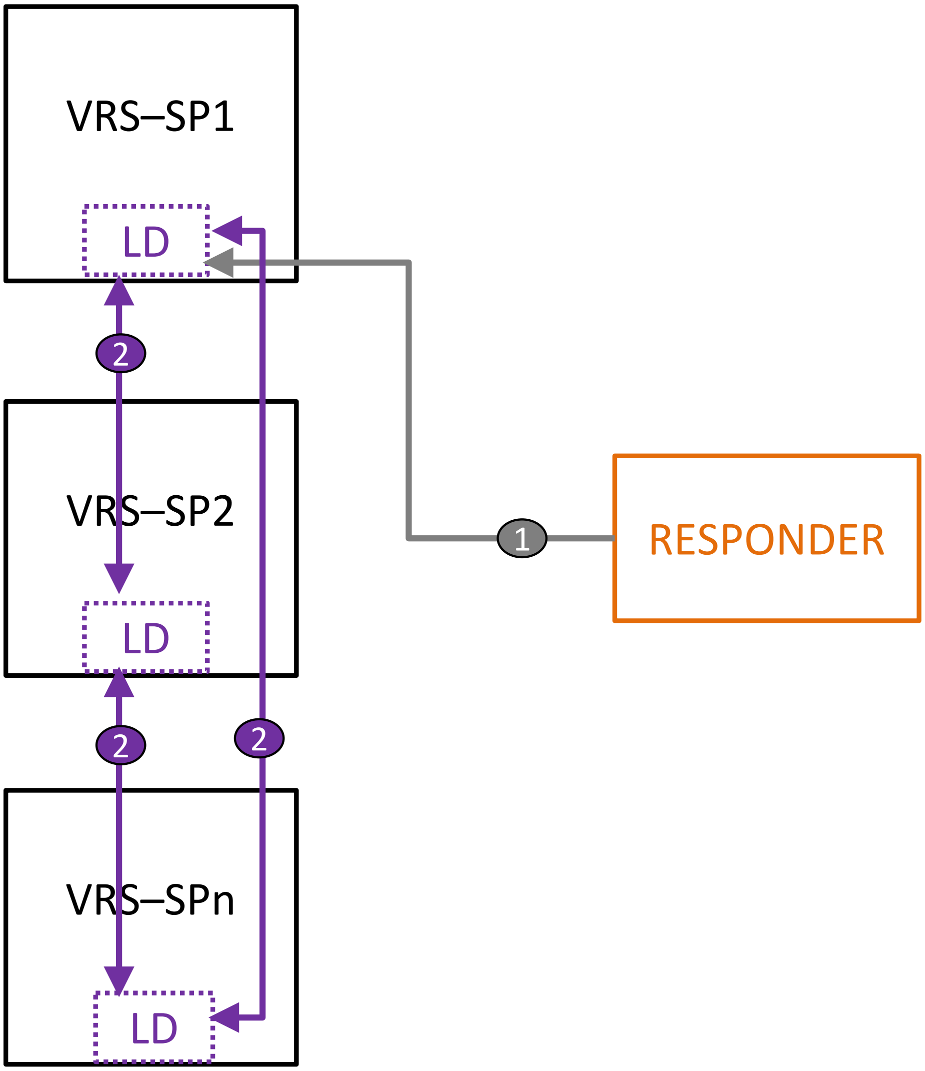
REST API using HTTPS. Message format will be JSON

1.6. Message Security

X.509 certificate self-signed or issued by a certificate authority. Authentication is mutual. Certificate expiration / validity will be determined and communicated between solution providers.

2. Appendix – Solution Architecture Illustrations

Interaction #	Process Category	Interaction Description
1	Enabling / Other	Connectivity Information (CI) provided by the Responder to multiple VRS Providers [Scenario 1] or one VRS Provider [Scenario 2] .
2	Enabling / Other	Exchange of VRS Provider ID used by Responder at GCP/GTIN level amongst VRS Providers for routing of VRs. This applies to Scenario 2 only.



3. Appendix – JSON Schema

```
{
 "openapi": "3.0.0",
 "info": {
  "version": "1.0.0",
  "title": "VRS LD Synchronization Standard",
  "contact": {
   "email": "apiteam@verificationrouterservice.com"
  },
  "description": "This the API specification for peer-to-peer communication between Verification Router Services or VRS for synchronizing directories"
 },
 "servers": [
  {
   "url": "https://apis.company.com/vrs"
  },
  {
   "url": "https://anothervrs.com/gateway/verificationrouterservice"
  }
 ],
 "paths": {
  "/v1/ld": {
   "get": {
    "description": "VRS Pull Synchronization",
    "parameters": [
     {
      "name": "lastModifiedDateTime",
      "in": "query",
      "description": "Last modified date time in UTC format",
      "required": true,
      "schema": {
       "type": "string"
      }
     }
    ],
    "responses": {
     "200": {
      "description": "A response code of 200 means the request was successful and details about the response can be found in the body of the response. Only a 200 response will issue a JSON payload.",
      "content": {
       "application/json": {
        "schema": {
         "properties": {
          "sourceVrsId": {
           "type": "string",
           "description": "Identifies the VRS that sourced the LD entry",
           "example": "VRS108"
          },
          "ldEntries": {
           "type": "array",
           "items": {
            "type": "object",
            "properties": {
             "recordGuid": {
```

```
            "type": "string",
            "description": "Globally unique identifier for the LD record assigned by source VRS system; Based on Version4 UUID; GUID will be shared between VRS",
            "example": "3ab5d7b6-3fcb-4a25-86f5-02fe5f5761bd",
            "pattern": "^[a-fA-F\\d]{8}-[a-fA-F\\d]{4}-4[a-fA-F\\d]{3}-[89abAB][a-fA-F\\d]{3}-[a-fA-F\\d]{12}$"
          },
          "recordOwner": {
            "type": "string",
            "description": "4, 5 or 6 digit FDA labeler code identifying the LD record owner",
            "example": "12345",
            "pattern": "^[\\d{4}|\\d{5}|\\d{6}]$"
          },
          "gtin": {
            "type": "string",
            "description": "GS1 global trade item number ",
            "minLength": 8,
            "maxLength": 14,
            "example": "00312345555016",
            "pattern": "^[\\d{14}|\\d{13}|\\d{12}|\\d{8}]$"
          },
          "ci": {
            "type": "string",
            "description": "The connectivity information (i.e. URL) of either designated VRS Provider or the GTIN owner if they are responding directly to verification requests.",
            "example": "https://ex.connectivityinfo.responder"
          },
          "startExpDate": {
            "type": "string",
            "description": "Lower bound of expiration date value.  Scanned expiration date value must fall between lower/upper bound when populated.  Date format is YYMMDD",
            "minLength": 6,
            "maxLength": 6,
            "example": "201130",
            "pattern": "^\\d{6}$"
          },
          "endExpDate": {
            "type": "string",
            "description": "Upper bound of expiration date value.  Scanned expiration date value must fall in between lower/upper bound when populated. Date format is YYMMDD",
            "minLength": 6,
            "maxLength": 6,
            "example": "221130",
            "pattern": "^\\d{6}$"
          },
          "status": {
            "type": "string",
            "description": "LD record status; by default, status is active.  Possible values: active, inactive, deleted.",
            "enum": [
              "active",
              "inactive",
              "deleted"
            ],
            "example": "active"
          },
          "nextRecordOwner": {
```

```
                            "type": "string",
                            "description": "4, 5 or 6 digit labeler code identifying the owner of the next LD entry for this GTIN",
                            "example": "12345",
                            "pattern": "^[\\d{4}|\\d{5}|\\d{6}]$"
                          },
                          "lastModifiedDateTime": {
                            "type": "string",
                            "description": "Date & timestamp of when the LD record is modified. This will be formatted as
YYYY-MM-DDThh:mm:ss.SSSZ and expressed in UTC. ",
                            "example": "2018-06-16T19:20:30.450Z",
                            "pattern": "^[\\d]{4}-[\\d]{2}-[\\d]{2}T[\\d]{2}:[\\d]{2}:[\\d]{2}.[\\d]{3}Z$"
                          }
                        },
                        "required": [
                          "recordGuid",
                          "recordOwner",
                          "gtin",
                          "ci",
                          "startExpDate",
                          "status",
                          "lastModifiedDateTime"
                        ]
                      }
                    }
                  },
                  "required": [
                    "sourceVrsId",
                    "ldEntries"
                  ]
                }
              }
            }
          },
          "400": {
            "description": "The request was not formatted properly. Please verify the request conforms to this
specification, and re-issue the request in the correct format. "
          },
          "401": {
            "description": "The request was not allowed because the request did not pass authentication."
          },
          "404": {
            "description": "The requested resource does not exist."
          },
          "408": {
            "description": "The server timed out waiting for the request."
          },
          "500": {
            "description": "System failed to process the request because of an error inside the system."
          },
          "503": {
            "description": "System is undergoing maintenance or is otherwise temporarily unavailable for API queries."
          }
        }
      }
    },
    "/v1/ld/pushsynchronization": {
```

```
   "post": {
    "description": "VRS Push Synchronization",
    "requestBody": {
     "required": true,
     "content": {
      "application/json": {
       "schema": {
        "properties": {
         "recordGuid": {
          "type": "string",
          "description": "Globally unique identifier for the LD record assigned by source VRS system; Based on Version4 UUID; GUID will be shared between VRS",
          "example": "3ab5d7b6-3fcb-4a25-86f5-02fe5f5761bd",
          "pattern": "^[a-fA-F\\d]{8}-[a-fA-F\\d]{4}-4[a-fA-F\\d]{3}-[89abAB][a-fA-F\\d]{3}-[a-fA-F\\d]{12}$"
         },
         "recordOwner": {
          "type": "string",
          "description": "4, 5 or 6 digit FDA labeler code identifying the LD record owner",
          "example": "12345",
          "pattern": "^[\\d{4}|\\d{5}|\\d{6}]$"
         },
         "gtin": {
          "type": "string",
          "description": "GS1 global trade item number ",
          "minLength": 8,
          "maxLength": 14,
          "example": "00312345555016",
          "pattern": "^[\\d{14}|\\d{13}|\\d{12}|\\d{8}]$"
         },
         "ci": {
          "type": "string",
          "description": "The connectivity information (i.e. URL) of either designated VRS Provider or the GTIN owner if they are responding directly to verification requests.",
          "example": "https://ex.connectivityinfo.responder"
         },
         "startExpDate": {
          "type": "string",
          "description": "Lower bound of expiration date value.  Scanned expiration date value must fall between lower/upper bound when populated.  Date format is YYMMDD",
          "minLength": 6,
          "maxLength": 6,
          "example": "201130",
          "pattern": "^\\d{6}$"
         },
         "endExpDate": {
          "type": "string",
          "description": "Upper bound of expiration date value.  Scanned expiration date value must fall in between lower/upper bound when populated. Date format is YYMMDD",
          "minLength": 6,
          "maxLength": 6,
          "example": "221130",
          "pattern": "^\\d{6}$"
         },
         "status": {
          "type": "string",
          "description": "LD record status; by default, status is active.  Possible values: active, inactive, deleted.",
          "enum": [
           "active",
           "inactive",
           "deleted"
          ],
          "example": "active"
         },
         "nextRecordOwner": {
          "type": "string",
          "description": "4, 5 or 6 digit labeler code identifying the owner of the next LD entry for this GTIN",
          "example": "12345",
          "pattern": "^[\\d{4}|\\d{5}|\\d{6}]$"
         },
         "lastModifiedDateTime": {
          "type": "string",
          "description": "Date & timestamp of when the LD record is modified. This will be formatted as YYYY-
MM-DDThh:mm:ss.SSSZ and expressed in UTC. ",
          "example": "2018-06-16T19:20:30.450Z",
          "pattern": "^[\\d]{4}-[\\d]{2}-[\\d]{2}T[\\d]{2}:[\\d]{2}:[\\d]{2}.[\\d]{3}Z$"
         }
        },
        "required": [
         "recordGuid",
         "recordOwner",
         "gtin",
         "ci",
         "startExpDate",
         "status",
         "lastModifiedDateTime"
        ]
       }
      }
     }
    },
    "responses": {
     "200": {
      "description": "A response code of 200 means the request was successful and details about the response
can be found in the body of the response. Only a 200 response will issue a JSON payload."
     },
     "400": {
      "description": "The request was not formatted properly. Please verify the request conforms to this
specification, and re-issue the request in the correct format. "
     },
     "401": {
      "description": "The request was not allowed because the request did not pass authentication."
     },
     "404": {
      "description": "The requested resource does not exist."
     },
     "408": {
      "description": "The server timed out waiting for the request."
     },
     "500": {
      "description": "System failed to process the request because of an error inside the system."
     },
     "503": {
            "description": "System is undergoing maintenance or is otherwise temporarily unavailable for API queries."
          }
        }
      }
    }
  }
}
```