SMM WEB SERVICE

REST API

The information in this document is subject to change without notice.

This document contains information protected by copyright. No part of this document may be photocopied or reproduced in any form without prior written consent from San Marino Mail Italia Srl.

© Copyright 2020 San Marino Mail Italia.

All rights reserved.

Version	Date	Author	Description
1.0	01-02-2012	SMM IT Staff	Initial version
1.1	10-04-2015	SMM IT Staff	Version update
1.2	24-06-2015	SMM IT Staff	Version update
2.3	12-05-2019	SMM IT Staff	Version update

Index

1. **Document purpose.. 4**
2. **REST definition 4**
3. **Invoking REST Service.. 5**
4. **SMM REST API ENDPOINT 9**
5. **SMM REST API - Authentication..10**
6. **SMM REST API - Label 11**
7. **SMM REST API - Manifest 18**
8. **TEST PHASE & MIGRATION 19**
9. **Reference 19**

Document purpose

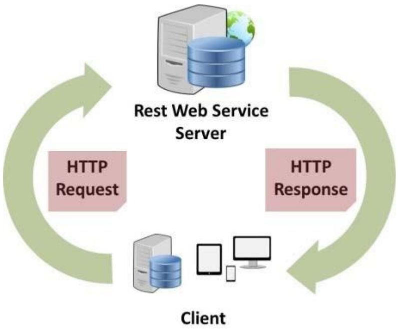
This document briefly describes how to interact with SMM REST Service. Also are described the operations (i.e. methods) it exposes and response message. This is accomplished by a set of API which interacts with SMM REST Service.

REST definition

REST (REpresentational State Transfer) is an architectural style, and an approach to communications, that is often used in the development of Web services.

The use of REST is often preferred over the more heavyweight SOAP (Simple Object Access Protocol) style because REST does not leverage as much bandwidth, which makes it a better fit for use over the Internet. The SOAP approach requires writing or using a provided server program (to serve data) and a client program (to request data).

REST'S decoupled architecture, and lighter weight communications between producer and consumer, make REST a popular building style for cloud-based APIs. When Web services use REST architecture, they are called RESTful APIs (Application Programming Interfaces) or REST APIs.REST architecture involves reading a designated Web page that contains an XML file. The XML file describes and includes the desired content. Once dynamically defined, consumers may access the interface.



REST, which typically runs over HTTP / HTTPS (Hypertext Transfer Protocol), has several architectural constraints:

1. Decouples consumers from producers;
2. Stateless existence;
3. Able to leverage a cache;
4. Leverages a layered system;
5. Leverages a uniform interface;

Invoking REST Service

Before using the web service, the following is required:

1. A valid customer ID
2. A valid Login Name and Password
3. Knowledge of the countries where you been authorized to ship

Request and response works basically as follow:

- Request (Order) a shipments by:

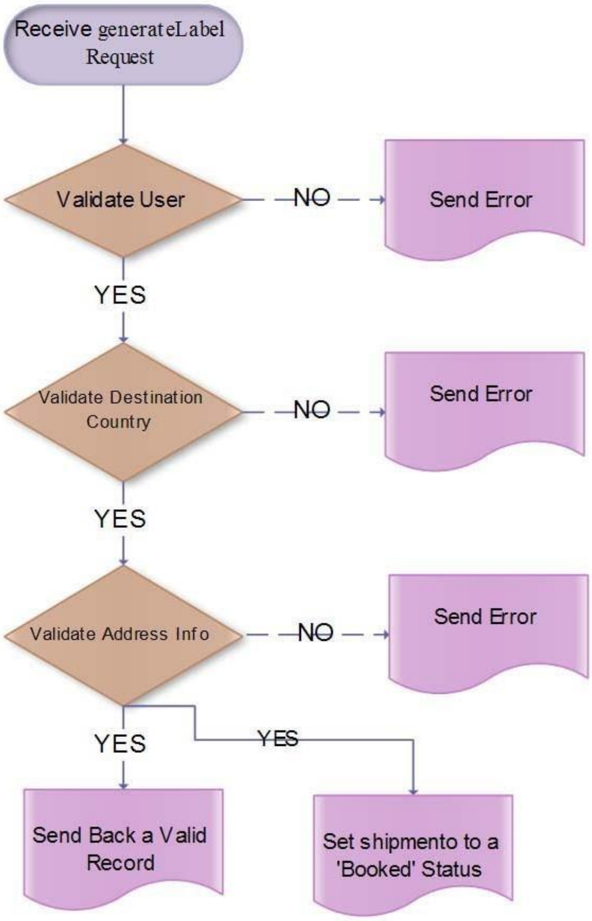
  a) Login into the system and send shipping data

- Respond message:

  a) If the User credentials and shipping data is validated then a record with data is returned and the shipment is set to a 'Booked' Status in our system

  b) If the either the User credentials or shipping data are not validate then an error message is returned with empty data.

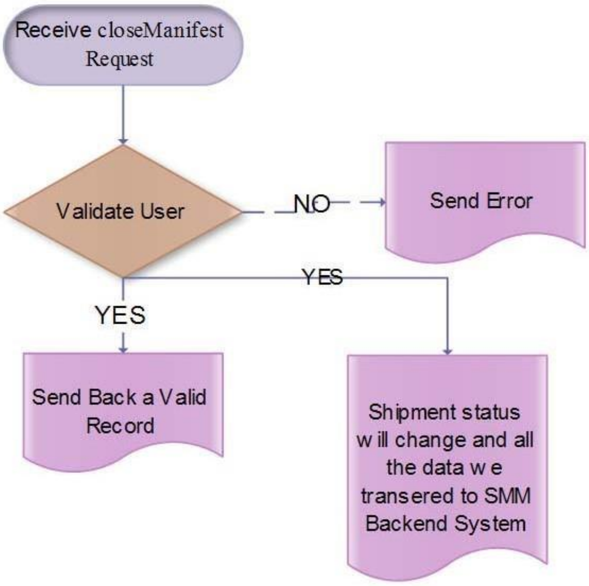
If the request returns a valid record, the shipment is set in our system as 'booked' meaning that an order to ship has been made.



We will not make further action to that record until we are advised that goods have actually left the warehouse, this is done by sending us a manifest.

The manifesting process for us begins when the goods leave the warehouse, shipments created will be buffered for the time being, until they are being manifested. Manifesting can be triggered by the warehouse operator or by company that is handling the line-haul.

In our case the manifest ID is send in the ‘Shipping Data’ and it will group a set of shipments. When a *closeManifest (/api/manifest)* request is send it will change the shipment(s) status and the data will be moved to SMM Backend systems

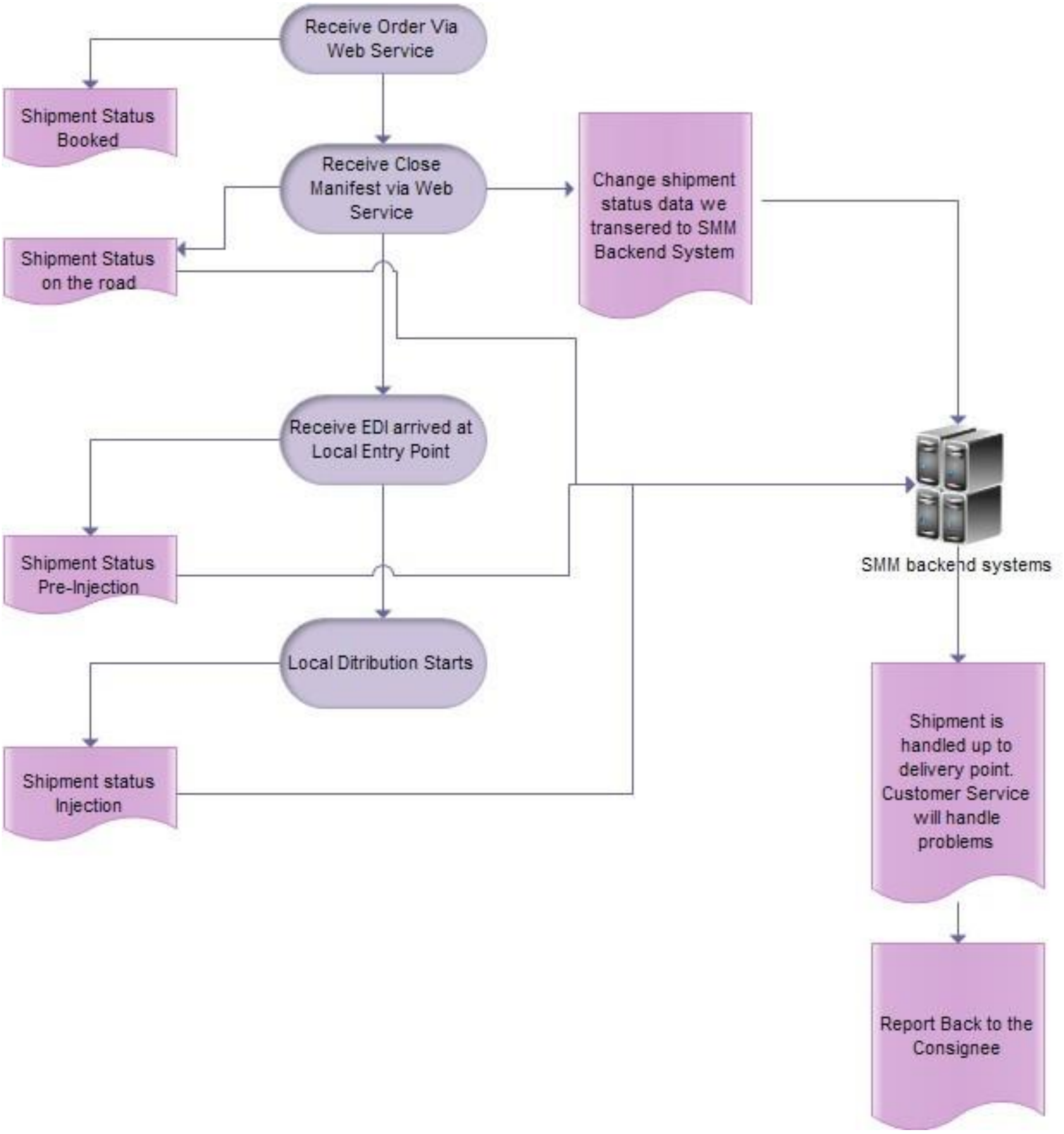

Receive closeManifest Request
Validate User
NO
Send Error
YES
YES
Send Back a Valid Record
Shipment status will change and all the data we transered to SMM Backend System

At this point the goods have left the warehouse via line haul company, system will record goods is on the road to the injection point. We call ‘Injection Point’ where the goods are unloaded and prepared to be delivered to the final destination (B to C). The company that handles the line haul will be sending us a message (to be decided) via a Manifest ID. The shipments will change to a Pre-Injection Status.

At the Injection Point, the single package will be weighted, measured and shipped. The local Distribution Company will notify us via EDI that the parcel has left their warehouse.

From now on all the package will be monitored via our system that notifies our internal Customer Service on the action to take when a problem arises.

Overview diagram:


Receive Order Via Web Service
Shipment Status Booked
Receive Close Manifest via Web Service
Change shipment status data we transered to SMM Backend System
Shipment Status on the road
Receive EDI arrived at Local Entry Point
SMM backend systems
Shipment Status Pre-Injection
Local Ditribution Starts
Shipment is handled up to delivery point. Customer Service will handle problems
Shipment status Injection
Report Back to the Consignee

SMM REST API ENDPOINT

Here are some resources that will help you understand the basics of all SMM APIs.

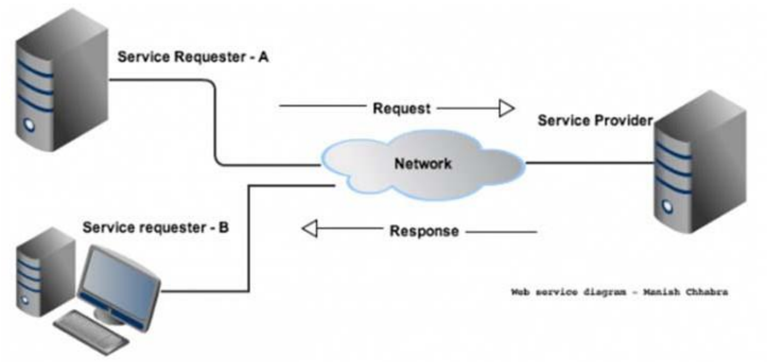
The *San Marino Mail REST APIs* provide a standard interface for interacting with the SMM Label System and our other applications. REST APIs provide access to resources (data entities) via URI paths. To use a REST API, your application will make an HTTP request and parse the response. Your methods will be the standard HTTP methods like GET, PUT, POST and DELETE. REST APIs operate over HTTP(s) making it easy to use with any programming language or framework.

The following endpoint address our environment:

SMM Label System
(Production): http://labels-api.sanmarinomail.it/SanMarinoMailWeb2

A complete REST operation is formed by combining an HTTP method (or "verb") with the full URI to the resource you're addressing.

The output format for the san Marino Mail REST APIs is XML



Here below you can find a detailed description about all available methods.

- **user:** for authentication purpose

  GET /api/user

- **label:** to create, delete, look up label

  GET POST DELETE /api/label

- **manifest:** to close manifest session

  PUT /api/manifest

SMM REST API - Authentication

There are two different way to perform authentication inside SMM label platform system.

The first one is based on the api **/api/user** and traditional username / password**.**

GET : Perform authentication based on session cookie

INPUT PARAMETERS :

USERNAME: account username (release by SMM)

PASSWORD : account password (release by SMM)

OUTPUT :

If the authentication success, the system will release an authentication code used to automatically identify all next REST requests. The response is like below:

```
<?xml version=\"1.0\" encoding=\"UTF-8\" ?>
<response>
     <haserror>0</haserror>
     <token>token_value_identification</token>
</response>
```

If the authentication fail, the system will send back an HTTP status 400 and the error message below:

```
<?xml version=\"1.0\" encoding=\"UTF-8\" ?>
<response>
     <haserror>1</haserror>
     <errorCode>errorCode from system</errorCode>
     <errorMessage>error message</errorMessage>
</response>
```

The session will be automatically invalidated after 60 minutes.

The second one is based on api key, a long series of random characters. The api key is released by SMM and must be sent on all performed REST requests, specified inside a variable called ***Tcode*** (see next paragraphs for more details)

SMM REST API - Label

The label api is the core of the system and is useful to create, search and delete labels.

GET : Search a label previously generated

INPUT PARAMETERS :

SMM_CUST_CODE: company unique identifier (Mandatory field)

CLIENT_SHIP_REF: customer shipment unique reference;

USN_PARCEL_NUMBER: usn number;

TCODE: apiKey (Not mandatory if previously logged in) released by SMM

OUTPUT :

If the authentication success and the label exist, the system will send the response below:

```
        <?xml version="1.0" encoding="UTF-8" ?>
        <response>
              <haserror>0</haserror>
              <label>
                    <clientShipRef>clientShipRef</clientShipRef>
                    <clientShipRef2>clientShipRef2</clientShipRef2>
                    <usn>usn</usn>
                    <usnM>usnM</usnM>
                    <data>data zpl code/pdf label/img label in base
64</zpl>
                    <output>output type</output>
                    <generationDate>generation Date</generationDate>
              </label>
        </response>
```

If the label doesn't exist, the system will send the response with the field above empty.

If the authentication fail the system will send an http status 400 and the response below:

```
<?xml version=\"1.0\" encoding=\"UTF-8\" ?>
 <response>
      <haserror>1</haserror>
      <errorCode>errorCode from system</errorCode>
      <errorMessage>error message</errorMessage>
 </response>
```

POST: Request for a new label generation

INPUT PARAMETERS :

TCODE: apiKey (Not mandatory if previously logged in) released by SMM

ACTION: action to be perform

- **LABEL** for label generation.
- **CONSISTENCY** for consistency check(Mandatory field)
- **LABEL_ONLY** for direct label download.

	NAME	DESCRIPTION	TYPE	MAX LENGTH	MANDA TORY
1	SMM_CUST_CODE	Company Id	T	20	Y
2	SMM_PRODUCT_CODE	Product code	T	50	N
3	PRE_ALLERT_DATE	Pre allert Date	T (yyyy-MM-dd)	10	N
4	CLIENT_SHIP_REF	Client Shipment Reference	T	35	Y
5	USN_PARCEL_NUMBER	USN parcel number	T	50	N
6	MASTER_USN_PARCEL_NR	Master USN	T	30	N
7	DELIVERY_NAME	Delivery Name	T	35	Y
8	DELIVERY_STR_TYPE	Delivery Street type	T	30	N
9	DELIVERY_ADDR	Delivery address 1	T	35	Y
10	DELIVERY_ADDR_NO	Delivery address No	T	5	N
11	DELIVERY_ZIP_CODE	Delivery Zip code	T	12	Y
12	DELIVERY_CITY	Delivery City	T	50	Y
13	DELIVERY_DISTRICT	Delivery district (Mandatory for italy)	T	50	Y/N
14	DEL_ISO_COUNTRY_CODE	Delivery iso country	T	3	Y
15	NR_OF_PCS	Number of items	N	10	Y
16	WEIGHT	Weight in KG	N	10	Y
17	PACKAGING_TYPE	Packaging type	T	40	N
18	LDV_PRINTING	Ldv printing (1 , 0) (default 0)	N	1	N
19	ADDRESS_CLEANING	Address cleaning (1,0)	N	1	N
20	CUST_SERVICE_ACTIVITY	Customer service (1,0)	N	1	N
21	WAREHOUSE_TRANSIT	Warehouse Transit (1,0)	N	1	N
22	PARCEL	Parcel Number	N	10	Y
23	SMS_PREADVISE_REQUEST	SMS preadvise	N	1	N
24	EMAIL_PREADVISE_REQUEST	Email preadvise	N	1	N
25	CLIENT_SHIP_REF_2	Client Shipment reference 2	T	35	N
26	DELIVERY_NAME_2	Delivery name 2	T	35	N
27	DELIVERY_ADDR_2	Delivery address 2	T	35	N
28	DELIVERY_ADDR_3	Delivery address 3	T	35	N
29	DELIVERY_ADDR_4	Delivery address 4	T	35	N
30	DELIVERY_PHONE_NR	Delivery phone number	T	50	N
31	DELIVERY_FAX_NR	Delivery fax number	T	50	N
32	DELIVERY_MOBILE_PH	Delivery mobile phone	T	50	N
33	DELIVERY_EMAIL	Delivery email	T	50	N
34	DELIVERY_INSTRUCTION	Delivery Instruction	T	150	N
35	DELIVERY_ADDRESS_ID	Delivery address Id	T	12	N
36	DELIVERY_FISCAL_CODE	Delivery fiscal code	T	50	N
37	DELIVERY_SATURDAY	Delivery saturday (1,0)	N	1	N
38	DIMEN_PCS_LENGTH	Length	N	10	N
39	DIMEN_PCS_WIDTH	Width	N	10	N
40	DIMEN_PCS_HEIGHT	Height	N	10	N
41	TOT_VOLUME	Total volume	N	10	N
42	DECLARED_GOODS_VALUE	Declared goods value	N	10	N
43	CUST_VALUE_CURRENCY	Value currency	T	10	N
44	SENDER_VAT_NUMBER	VAT number	T	50	N
45	SENDER_DOCUMENT_NR	Document Nr	T	50	N
46	INVOICE_NR_FOLLOWING	Invoice Nr	T	50	N
47	COD_AMOUNT	COD amount (otherwise 0)	N	10	N
48	COD_PAYM_METHOD	COD Payment method (CON)	T	50	N
49	COD_CURRENCY	COD currency	T	10	N
50	INSURANCE_VALUE	Insurance value	N	10	N
51	INSURANCE_CURRENCY	Insurance currency	T	10	N
52	PICKUP_NAME	Pickup name	T	50	N

53	PICKUP_NAME2	Pickup name 2	T	50	N
54	PICKUP_ADDR	Pickup address 1	T	100	N
55	PICKUP_ADDR_NO	Pickup address No	T	5	N
56	PICKUP_ADDR_2	Pickup address 2	T	100	N
57	PICKUP_ADDR_3	Pickup address 3	T	100	N
58	PICKUP_ADDR_4	Pickup address 4	T	100	N
59	PICKUP_ZIP_CODE	Pickup zipcode	T	10	N
60	PICKUP_CITY	Pickup city	T	50	N
61	PICKUP_PHONE_NR	Pickup phone nr	T	50	N
62	PICKUP_FAX_NR	Pickup fax nr	T	50	N
63	PICKUP_MOBILE_PH	Pickup mobile nr	T	50	N
64	PICKUP_EMAIL	Pickup email	T	50	N
65	PICKUP_ISO_COUNTRY_CODE	Pickup ISO country	T	3	N
66	PICKUP_ADDR_ID	Pickup address Id	T	12	N
67	DELIVERY_AT_FLOOR	Delivery at floor (1,0)	N	1	N
68	DELIVERY_APPOINTMENT	Delivery appointment (1,0)	N	1	N
69	DELIVERY_EVENING	Delivery evening (1,0)	N	1	N
70	PICKUP_USED_RAEE	Used RAEE (1,0)	N	10	N
71	PICKUP_DATE	Pickup date	T (yyyy-MM-dd)	10	N
72	OPENING_HOUR_MORNING	Opening hour morning	N	1	N
73	CLOSING_HOUR_MORNING	Closing hour morning	N	1	N
74	OPENING HOUR_AFTERNOON	Opening afternoon	N	1	N
75	CLOSING HOUR_AFTERNOON	Closing afternoon	N	1	N
76	PICKUP_WINDOW	Pickup window	T	2	N
77	PICKUP_DISTRICT	Pickup district	T	2	N
78	CONTENT	Content type	T	50	N
79	GOODS_VALUE	Commercial value	N	10	N
80	GOODS_CURRENCY	Commercial currency	T	3	N
81	CUSTOM_CODE	Customs code	T	12	N
82	OUTPUT_TYPE	OutputType (0=ZPL,1=PDF)	N	1	Y

OUTPUT :

If the authentication success, all the data are valid and action is equals to LABEL, the system will send for label the response below:

```
<?xml version="1.0" encoding="UTF-8" ?>
<response>
      <haserror>0</haserror>
      <label>
            <clientShipRef>clientShipRef</clientShipRef>
            <clientShipRef2>clientShipRef2</clientShipRef2>
            <usn>usn</usn>
            <usnM>usnM</usnM>
            <data>zpl code (in base 64 if requested)</data>
            <output>output type</output>
            <generationDate>generation Date</generationDate>
      </label>
</response>
```

If action is equals to LABEL_ONLY the system will send back the label with the correct content type

If action is equals to CONSISTENCY the system will send back the feedback about consistency check process

```
<?xml version="1.0" encoding="UTF-8" ?>
<response>
      <haserror>0</haserror>
      <label>
            <clientShipRef>clientShipRef</clientShipRef>
            <clientShipRef2>clientShipRef2</clientShipRef2>
            <result>true</result>
            <product>sent product</product>
      </label>
</response>
```

If for any reason, the consistency fail, the system will send back a response like:

```
<?xml version="1.0" encoding="UTF-8" ?>
<response>
      <haserror>1</haserror>
       <errorCode>errorCode from system</errorCode>
       <errorMessage>error message</errorMessage>
</response>
```

If the authentication fail, the system will send the response below:

```
<?xml version=\"1.0\" encoding=\"UTF-8\" ?>
 <response>
      <haserror>1</haserror>
      <errorCode>errorCode from system</errorCode>
      <errorMessage>error message</errorMessage>
 </response>
```

DELETE :

INPUT PARAMETERS :

CLIENT_SHIP_REF: customer shipment unique reference;

USN_PARCEL_NUMBER: usn value;

SMM_CUST_CODE: usn value;

TCODE: apiKey (Not mandatory if previously logged in) released by SMM

If the authentication success and the label exist:

```
<?xml version=\"1.0\" encoding=\"UTF-8\" ?>" +
<response>
      <haserror>0</haserror>
      <result>true</result>
</response>
```

If the authentication success and the label doesn't exist:

```
<?xml version=\"1.0\" encoding=\"UTF-8\" ?>" +
<response>
      <haserror>0</haserror>
      <result>false</result>
</response>
```

If the authentication fail, the system will send the response below:

```
<?xml version=\"1.0\" encoding=\"UTF-8\" ?>
 <response>
      <haserror>1</haserror>
      <errorCode>errorCode from system</errorCode>
      <errorMessage>error message</errorMessage>
 </response>
```

SMM REST API - Manifest

The Manifest api is useful to manage manifest operations on previously generated data.

PUT :Request for manifest close

INPUT PARAMETERS :

SMM_CUST_CODE: company unique identifier (Mandatory field);

MANIFEST_ID: manifest Identifier;

MANIFEST_DATETIME: manifest date-time. Format: dd/mm/yyyy hh:mm:ss;

CLIENT_SHIP_REF_LIST: list of clientShipref splitted by ;

TCODE: apiKey (Not mandatory if previously logged in) released by SMM

OUTPUT :

If the authentication success and all the data are valid, the system will send the response below:

```
      <?xml version=\"1.0\" encoding=\"UTF-8\" ?>" +
      <response>
            <haserror>0</haserror>
            <rows>
            <row> CompanyId|ClientShipRef|ClientShipRef2|Usn|Status</row>
                  <row>
CompanyId|ClientShipRef|ClientShipRef2|UsnA|StatusA</row>
            <row> CompanyId|ClientShipRef|ClientShipRef2|UsnB|StatusB</row>
            ...
            </rows>
      </response>
```

If the authentication fail or an error occurred, the system will send the response below:

```
<?xml version=\"1.0\" encoding=\"UTF-8\" ?>
 <response>
      <haserror>1</haserror>
      <errorCode>errorCode from system</errorCode>
      <errorMessage>error message</errorMessage>
 </response>
```

TEST PHASE & MIGRATION

The endpoint above, can be used also for testing purposes. The condition useful to identify a call as a test, is to add at the beginning of the CLIENT_SHIP_REF field the word **TEST**; Any call sent to the system without the word **TEST** will be considered as a real shipment and will be invoiced.

During test phase both systems (the old one and the new one) will be available, but with an important limitation: the first call performed to the new system without the word "TEST" at the beginning of the CLIENT_SHIP_REF field, will cause the deactivation the account on the old system.

Reference:

http://searchsoa.techtarget.com/definition/REST
https://stormpath.com/blog/secure-your-rest-api-right-way/