Volume 1

BIO-TECH MEDICAL SOFTWARE, INC.

BioTrackTHC XML API

BioTrackTHC API

BIO-TECH MEDICAL SOFTWARE, INC.

BioTrackTHC XML API

Fort Lauderdale, FL
Phone 800.797.4711

Document Version 1.01

Table of Contents

Prefix: Getting Started

Welcome to BioTrackTHC XML platform. This manual serves as a comprehensive guide that details the various functions and data points that are relevant for the BioTrackTHC traceability system.

Please note: There may be additional enhancements, functions, etc. in the future to this specification.

Although this document is public and may be read by anyone; much of it assumes that the reader has a basic understanding of web technologies and programming interfaces. It is geared towards individuals looking to interface directly to the state traceability system without utilizing the official state web interface. The official state web interface will be available at no cost for individuals who wish to upload their data without a commercial application. However, the official web interface is intended to only collect the minimum amount of information for the state compliance and does not collect information related to e.g. sales; every licensee is responsible for keeping their own business records.

Inventory Types

5	Kief
6	Flower
7	Clone
9	Other Material (stems, leaves, etc to be processed)
10	Seed
11	Plant Tissue
12	Mature Plant
13	Flower Lot
14	Other Material Lot
15	Bubble Hash
16	Hash
17	Hydrocarbon Wax
18	CO2 Hash Oil
19	Food Grade Solvent Extract
20	Infused Dairy Butter or Fat in Solid Form
21	Infused Cooking Oil
22	Solid Cannabis Infused Edible
23	Liquid Cannabis Infused Edible
24	Cannabis Extract for Inhalation
25	Cannabis Infused Topicals
26	Sample Jar
27	Waste

28	Usable Cannabis
29	Wet Flower
30	Cannabis Mix
31	Cannabis Mix Packaged (mixed packaged flower)
32	Cannabis Mix Infused (mixed concentrates)
34	Capsule
35	Tincture
36	Transdermal Patch
37	Suppository
38	Non Smokeable Infused Extract

Unique Identifiers

The system will generate unique identifiers for all plants and inventory. Inventory (eg. Lots, batches, etc.) and plants items will be provided 18 digit identifiers. The breakdown is listed below:

Characters	Description
1	Single character which indicates whether the line item is medical (M) or recreational (R)
2-7	Six-digit license number of the location creating the item
8-9	Two-digit county id
10-11	Two-digit inventory type (Please reference the inventory type table listed above)
12-18	Seven digit plant or inventory specific ID

Default_Unit Values

id	name	ml	mg	Weight Based
1	Drop(s)	0.05		
2	Spray(s)	0.07		
3	Cup(s)	236.588237		
4	Pint(s)	473.176		
5	Quart(s)	946.353		
6	Gallons	3785.41		
7	Liter(s)	1000		
8	Milliliter(s)	1		
9	Fluid Ounce(s)	29.5735		
10	Tablespoon(s)	14.7868		
11	Teaspoon(s)	4.92892		
12	Grams		1000	1

Chapter 1: Authentication

In this chapter, you'll learn how to:

- ✓ Communicate with the traceability system
- ✓ Authenticate
- ✓ Create and modify users
- ✓ Elevate privileges, when necessary

Every request begins with <xml> and ends with </xml>. The current iteration of our API is now at 4.0. It is **strongly** recommended that every application specify this with every request. We do anticipate future changes and specifying the API will ensure your application does not receive errors when features are added or deprecated, but not entirely removed. Otherwise, the system will assume you are referencing the latest version. Every API request has an action associated with it. Any request that does not specify an action will automatically be rejected. Also, per XML specifications, any improperly formatted XML request will also be rejected. When in doubt, see: http://www.w3schools.com/xml/xml_validator.asp. So, at bare minimum, a request should appear as follows:

```
<xml>
  <API>4.0</API>
  <action>foo</action>
</xml>
```

The request should be sent as a raw POST request of the type text/xml. The result will also be of text/xml type.

The URL is: https:// mminventory.health.nd.gov /serverxml.asp

login

When registering with the NDDOH, an account administrator will receive a password in their email that will grant full access. This email address and password can then be shared, stored or utilized by a commercial application to initially authenticate with the traceability system.

Parameters:

action	variable length text field
username	variable length text field
password	variable length text field

license_number	ubi number. variable length text field

```
<xml>
  <API>4.0</API>
  <action>login</action>
  <password>foobar</password>
  <license_number>000000010</license_number>
  <username>username@domain.com</username>
</xml>
```

A client should login with their username, password and the UBI number of their account. A successful authentication will result in the following:

```
<xml>
  <admin>1</admin>
<sessionid>2f58596cad6db73d6cdd599b11cd169263a54cd37dc75ae0bfefe0cd9c9c571c10
7059f23fe8cf7d4572f4878b9e1d9821e097e9348aa7b59a31180ab8c9e6c8</sessionid>
  <time>1384323370</time>
  <success>1</success>
</xml>
```

Returned Parameters:

admin	Boolean value
sessionid	sha512 hex encoded string
time	Unix 32-bit integer timestamp
success	Boolean value

The admin parameter will indicate that the authenticated user is an administrator capable of creating other users, setting permissions, etc. The sessionid parameter can be used for future requests under the user who originally authenticated for quicker requests.

If an application is not interested in maintaining sessions, they may also choose to simply include the aforementioned values with the nosession parameter. For example:

```
<xml>
  <API>4.0</API>
  <action>test</action>
```

```
  <password>foobar</password>
  <license_number>000000009</license_number>
  <username>username@domain.com</username>
  <nosession>1</nosession>
</xml>
```

By setting the nosession parameter to 1, requests can be made without creating a stateful session, if necessary.

During the course of a normal session, a session's credentials can also be temporarily elevated for the duration of the action by passing the super_user and super_password parameters.

```
<xml>
  <API>4.0</API>
  <action>admin_action_example</action>
<sessionid>2f58596cad6db73d6cdd599b11cd169263a54cd37dc75ae0bfefe0cd9c9c571c10
7059f23fe8cf7d4572f4878b9e1d9821e097e9348aa7b59a31180ab8c9e</sessionid>
  <super_password>foobar</super_password>
  <super_user>username@domain.com</super_user>
  <param>foo</param>
</xml>
```

If a function call returns 0 value for success, it will also set an <error>explanation</error> for easier error handling. For brevity, all code examples hereafter will omit the sessionid parameter; but it is assumed that either that or the proper nosession credentials are provided for **every** request.

The application interface also supports a testing interface. If a licensee wishes to practice or a commercial application wishes to test their integration capabilities a request may include the <training>1</training> node within a request. Users cannot be created, modified or removed in training mode. They are automatically transposed from the production environment. Every user automatically has full capabilities in training mode; that is, there are no ACL controls (as the data is not real). If a session is created in training mode, and an attempt is made to perform an action in production mode (or vice versa) an invalid session will be triggered as they operate completely separate from one another. It will be up to the application to save state as to which mode the connection was initiated with. As can be seen below, training mode is easy to trigger:

```
<xml>
  <API>4.0</API>
  <training>1</training>
  <action>login</action>
```

```
  <password>foobar</password>
  <license_number>123456789</license_number>
  <username>username@domain.com</username>
</xml>
```

user_add

Users with administrative privileges can add other users via the user_add function. As demonstrated below, each function is discrete and robust ACLs can be utilized by an integrating party.

Parameters:

action	variable length text field
new_username	variable length text field
new_password	variable length text field
new_permissions	nested field that includes boolean values for each permission

```
<xml>
  <API>4.0</API>
  <action>user_add</action>
  <new_admin>1</new_admin>
  <new_password>foobar</new_password>
  <new_username>user1@domain.com</new_username>
  <new_permissions>
    <inventory_convert>1</inventory_convert>
    <sale_dispense>1</sale_dispense>
    <sale_modify>1</sale_modify>
    <sale_void>1</sale_void>
    <sale_refund>1</sale_refund>
    <justauthenticate>1</justauthenticate>
    <employee_add>1</employee_add>
    <employee_modify>1</employee_modify>
    <employee_remove>1</employee_remove>
    <vehicle_add>1</vehicle_add>
    <vehicle_modify>1</vehicle_modify>
    <vehicle_remove>1</vehicle_remove>
    <plant_room_add>1</plant_room_add>
    <plant_room_modify>1</plant_room_modify>
    <plant_room_remove>1</plant_room_remove>
```

<inventory_room_add>1</inventory_room_add>
<inventory_room_modify>1</inventory_room_modify>
<inventory_room_remove>1</inventory_room_remove>
<plant_destroy_schedule>1</plant_destroy_schedule>
<plant_destroy_schedule_undo>1</plant_destroy_schedule_undo>
<plant_destroy>1</plant_destroy>
<plant_destroy_undo>1</plant_destroy_undo>
<plant_harvest_schedule>1</plant_harvest_schedule>
<plant_harvest_schedule_undo>1</plant_harvest_schedule_undo>
<plant_harvest>1</plant_harvest>
<plant_harvest_undo>1</plant_harvest_undo>
<plant_convert_to_inventory>1</plant_convert_to_inventory>
<plant_cure>1</plant_cure>
<plant_cure_undo>1</plant_cure_undo>
<plant_yield_modify>1</plant_yield_modify>
<plant_waste_weigh>1</plant_waste_weigh>
<plant_additive_remove>1</plant_additive_remove>
<plant_additive_add>1</plant_additive_add>
<inventory_new>1</inventory_new>
<plant_new_undo>1</plant_new_undo>
<inventory_manifest_lookup>1</inventory_manifest_lookup>
<inventory_manifest_order>1</inventory_manifest_order>
<inventory_transfer_inbound>1</inventory_transfer_inbound>
<inventory_transfer_inbound_modify>1</inventory_transfer_inbound_modify>
<inventory_transfer_lookup>1</inventory_transfer_lookup>
<inventory_transfer_outbound>1</inventory_transfer_outbound>
<inventory_transfer_outbound_modify>1</inventory_transfer_outbound_modify>
<inventory_transfer_outbound_void>1</inventory_transfer_outbound_void>
<plant_move>1</plant_move>
<plant_modify>1</plant_modify>
<inventory_adjust>1</inventory_adjust>
<inventory_adjust_usable>1</inventory_adjust_usable>
<inventory_check>1</inventory_check>
<inventory_convert>1</inventory_convert>
<inventory_sample>1</inventory_sample>
<inventory_qa_check>1</inventory_qa_check>
<inventory_qa_check_all>1</inventory_qa_check_all>
<inventory_qa_sample>1</inventory_qa_sample>

<inventory_qa_sample_void>1</inventory_qa_sample_void>
<inventory_qa_sample_results>1</inventory_qa_sample_results>
<inventory_manifest_pickup>1</inventory_manifest_pickup>
<inventory_manifest_modify>1</inventory_manifest_modify>
<inventory_manifest>1</inventory_manifest>
<inventory_manifest_void>1</inventory_manifest_void>
<inventory_manifest_void_stop>1</inventory_manifest_void_stop>
<inventory_manifest_void_items>1</inventory_manifest_void_items>
<inventory_create_lot>1</inventory_create_lot>
<inventory_split>1</inventory_split>
<user_add>1</user_add>
<user_modify>1</user_modify>
<user_remove>1</user_remove>
<inventory_move>1</inventory_move>
<inventory_modify>1</inventory_modify>
<inventory_destroy_schedule>1</inventory_destroy_schedule>
<inventory_destroy_schedule_undo>1</inventory_destroy_schedule_undo>
<inventory_destroy>1</inventory_destroy>
<nonce_replay>1</nonce_replay>
<sync_vehicle>1</sync_vehicle>
<sync_plant_additive>1</sync_plant_additive>
<sync_additive>1</sync_additive>
<sync_additive_type>1</sync_additive_type>
<additive_remove>1</additive_remove>
<additive_modify>1</additive_modify>
<additive_add>1</additive_add>
<additive_type_remove>1</additive_type_remove>
<additive_type_modify>1</additive_type_modify>
<additive_type_add>1</additive_type_add>
<sync_employee>1</sync_employee>
<sync_plant_room>1</sync_plant_room>
<sync_inventory_room>1</sync_inventory_room>
<sync_inventory>1</sync_inventory>
<sync_plant>1</sync_plant>
<sync_plant_derivative>1</sync_plant_derivative>
<sync_manifest>1</sync_manifest>
<sync_inventory_sample>1</sync_inventory_sample>
<sync_inventory_transfer>1</sync_inventory_transfer>

```
    <sync_inventory_transfer_inbound>1</sync_inventory_transfer_inbound>
    <sync_sale>1</sync_sale>
    <sync_vendor>1</sync_vendor>
    <sync_qa_lab>1</sync_qa_lab>
    <sync_check>1</sync_check>
    <sync_inventory_adjust>1</sync_inventory_adjust>
    <sync_inventory_qa_sample>1</sync_inventory_qa_sample>
    <inventory_manifest_void_stop>1</inventory_manifest_void_stop>
    <inventory_manifest_void_items>1</inventory_manifest_void_items>
<inventory_transfer_outbound_return_lookup>1</inventory_transfer_outbound_return
_lookup>
    <inventory_transfer_outbound_return>1</inventory_transfer_outbound_return>
    <inventory_convert_undo>1</inventory_convert_undo>
    <card_lookup>1</card_lookup>
</new_permissions>
</xml>
```

Each permission should either be 1 for true, 0 for false. Any nested parameter for the new_permissions parameter that are not included shall be assumed to be 0.

Returned Parameters:

success	Boolean value

user_modify

Users with administrative privileges can modify other users via the user_modify function.

Parameters:

action	variable length text field
new_username	variable length text field
new_password	variable length text field
new_permissions	nested field that includes boolean values for each permission

```
<xml>
 <API>4.0</API>
 <action>user_modify</action>
 <new_admin>1</new_admin>
```

```
  <new_password>foobar</new_password>
<new_username>user1@domain.com</new_username>
<new_permissions>
...
</new_permissions>
</xml>
```

Returned Parameters:

success	Boolean value

user_remove

Users with administrative privileges can remove other users via the user_remove function. Please note: The initial user that was created with the license cannot be removed.

Parameters:

action	variable length text field
new_username	variable length text field

```
<xml>
  <API>4.0</API>
  <action>user_remove</action>
<new_username>user1@domain.com</new_username>
</xml>
```

Returned Parameters:

success	Boolean value

Chapter 2: Employees & Vehicles

In this chapter, you'll learn how to:

- ✓ Add, modify and remove employees
- ✓ Add, modify and remove vehicles

employee_add

Every organization will need to input basic information on their employees when providing samples or submitting transport manifests. Organizations will not be required to provide comprehensive employee lists, but, rather, on an as-needed basis for actions requiring an employee identification.

Parameters:

action	variable length text field
employee_name	variable length text field
employee_id	unique variable length text field
birth_month	two character integer
birth_day	two character integer
birth_year	four character integer
hire_month	two character integer
hire_day	two character integer
hire_year	four character integer

```
<xml>
  <API>4.0</API>
  <action>employee_add</action>
  <employee_name>Joe Employee</employee_name>
  <employee_id>12345</employee_id>
  <birth_month>01</birth_month>
  <birth_day>01</birth_day>
  <birth_year>1980</birth_year>
  <hire_month>01</hire_month >
  <hire_day>01</hire_day>
  <hire_year>2014</hire_year>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

employee_modify

This function should be used to update an existing employee.

Parameters:

action	variable length text field
employee_name	variable length text field
employee_id	unique variable length text field
birth_month	two character integer
birth_day	two character integer
birth_year	four character integer
hire_month	two character integer
hire_day	two character integer
hire_year	four character integer
transactionid_original	Optional, integer, this is the first transactionid value received from creation of this employee. This can also be used to identify and update an existing record.

```
<xml>
  <API>4.0</API>
  <action>employee_modify</action>
  <employee_name> Joe Employee</employee_name>
  <employee_id>12345</employee_id>
  <birth_month>01</birth_month>
  <birth_day>01</birth_day>
  <birth_year>1980</birth_year>
  <hire_month>01</hire_month >
  <hire_day>01</hire_day>
  <hire_year>2014</hire_year>
</xml>
```

Returned Parameters:

success	Boolean value

transactionid	integer value

employee_remove

This function should be used to remove an employee.

Parameters:

action	variable length text field
employee_id	unique variable length text field
transactionid_original	Optional, integer, this is the first transactionid value received from creation of this employee. This can also be used to identify and update an existing record.

```
<xml>
  <API>4.0</API>
  <action>employee_remove</action>
  <employee_id>12345</employee_id >
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

vehicle_add

Every organization will need to input basic information on their vehicles when submitting transport manifests. This includes an integer id number that should be associated with the vehicle and the associated information for that vehicle, including: Color, make, model, plate and VIN.

Parameters:

action	variable length text field
vehicle_id	unique integer
color	variable length text field
make	variable length text field
model	variable length text field
plate	variable length text field
vin	variable length text field
year	integer

```
<xml>
  <API>4.0</API>
  <action>vehicle_add</action>
  <vehicle_id>2</vehicle_id >
  <color>Red</color >
  <make>Ford</make >
  <model>Mustang</model >
  <plate>ABC124</plate >
  <year>2010</year >
  <vin>123242365566</vin >
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

vehicle_modify

This function should be used to update an existing vehicle.

Parameters:

action	variable length text field
vehicle_id	unique integer
color	variable length text field
make	variable length text field
model	variable length text field
plate	variable length text field
vin	variable length text field
year	integer

```
<xml>
  <API>4.0</API>
  <action>vehicle_modify</action>
  <vehicle_id>2</vehicle_id >
  <color>Blue</color >
  <make>Ford</make >
```

```
  <model>Mustang</model >
  <plate>ABC124</plate >
  <year>2010</year >
  <vin>123242365566</vin >
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

vehicle_remove

This function should be used to remove an employee.

Parameters:

action	variable length text field
vehicle_id	unique integer

```
<xml>
  <API>4.0</API>
  <action>vehicle_remove</action>
  <vehicle_id>2</vehicle_id >
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

Chapter 3: Rooms

In this chapter, you'll learn how to:

- ✓ Add, modify and remove plant rooms
- ✓ Add, modify and remove inventory rooms

plant_room_add

Plant rooms represent a way to logically segregate plants in a specific location. These can include actual rooms inside of indoor facility or fields in an outdoor facility.

Parameters:

action	variable length text field
name	variable length text field
location	license number of location value
id	integer value

```
<xml>
  <API>4.0</API>
  <action>plant_room_add</action>
  <name>Veg 1</name>
  <id>1</id>
  <location>12345</location>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

plant_room_modify

Plant rooms can be renamed or re-activated with this function.

Parameters:

action	variable length text field
name	variable length text field
location	license number of location value
id	integer value

```
<xml>
 <API>4.0</API>
 <action>plant_room_modify</action>
 <name>Veg 2</name>
 <id>1</id>
 <location>12345</location>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

plant_room_remove

Plant rooms can be removed with this function.

Parameters:

action	variable length text field
location	license number of location value
id	integer value

```
<xml>
 <API>4.0</API>
 <action>plant_room_remove</action>
 <id>1</id>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

inventory_room_add

Inventory rooms represent a way to logically segregate inventory in a specific location. This can offer a real-time representation not only of the overall on-hand amount of a specific item but also the amount in a specific area of a facility. A room can be designated as a quarantine room with this function, as well. At least one quarantine

room is required for segregating inventory before transportation. A room identifier must always be greater than zero. The room 0 is reserved as a general identifier for inventory that has not been assigned to a room.

Parameters:

action	variable length text field
name	variable length text field
location	license number of location value
id	integer value
quarantine	Boolean value

```
<xml>
 <API>4.0</API>
 <action>inventory_room_add</action>
 <name>Veg 1</name>
 <id>1</id>
 <quarantine>0</quarantine>
 <location>12345</location>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

inventory_room_modify

Inventory rooms can be renamed or re-activated with this function.

Parameters:

action	variable length text field
name	variable length text field
location	license number of location value
id	integer value
quarantine	Boolean value

```
<xml>
 <API>4.0</API>
 <action>inventory_room_modify</action>
 <name>Veg 2</name>
 <id>1</id>
```

```
  <quarantine>0</quarantine>
  <location>12345</location>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

inventory_room_remove

Inventory rooms can be removed with this function.

Parameters:

action	variable length text field
location	license number of location value
id	integer value

```
<xml>
  <API>4.0</API>
  <action>inventory_room_remove</action>
  <id>1</id>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

Chapter 4: Plants

In this chapter, you'll learn how to:

- ✓ Add and remove plants
- ✓ Harvest and cure plants
- ✓ …and much, much more!

plant_new

The plant_new function will allow a cultivator to enter new plants into the traceability system. This function will require the strain, quantity, location, new room, whether the plant will be used as a mother plant (this can be toggled later if necessary) and the source identification number. The source identification number can be from one of the following inventory types: Clone, Seed, Mature Plant and Plant Tissue. Clone, Seed and Mature Plant are depletable inventory items in that any plant creation will automatically deduct from the count in inventory (so ensure that the quantity of new plants does not exceed that available from inventory).

Parameters:

action	variable length text field
strain	variable length text field
location	license number of location
room	integer value
sourcetext	field representing unique identifier
quantity	integer value
mother	integer value
birthdate	Optional, 8 character birthdate in the following format: YYYYMMDD. If not provided, the system will default to the current date.

```
<xml>
 <API>4.0</API>
 <action>plant_new</action>
 <location>12345</location>
 <source> M10000100104638848</source>
 <quantity>2</quantity>
 <room>1</room>
 <strain>Blueberry</strain>
 <mother>0</mother>
```

```
</xml>
```

Return example:

```
<xml>
 <barcode_id> M00342Q00003990761</barcode_id>
 <barcode_id> M00342Q00006357962</barcode_id>
 <sessiontime>1384476925</sessiontime>
 <success>1</success>
 <transactionid>3278</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
barcode_id	Array of 1 or more text fields representing the new unique identifiers attached to the plants

Transaction IDs are generated for every action which involves the submission of licensee data. These TIDs are used for audit purposes and should be maintained.

plant_new_undo

The plant_new_undo function will allow a cultivator to correct a mistake. This function can be used when a user accidentally moves items from the inventory to the plant area inadvertently. It can only be used on plants that have not been destroyed or harvested. Also, the parent item the plant was sourced from must also still be in possession of the licensee. Once called on a plant identifier, the system will automatically remove the plant from the system and increment the quantity of the parent source by one.

Parameters:

action	variable length text field
barcodeid	Array of 1 or more text fields representing the plants to undo

```
<xml>
 <API>4.0</API>
 <action>plant_new_undo</action>
 <barcodeid> M00342Q00006357962</barcodeid>
</xml>
```

Return example:

```
<xml>
 <sessiontime>1384476925</sessiontime>
 <success>1</success>
 <transactionid>3278</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

plant_move

The plant_move function will allow a cultivator to move plants from their current room to a new one.

Parameters:

action	variable length text field
room	integer value
barcodeid	Array of 1 or more text fields representing the plants to move

```
<xml>
 <API>4.0</API>
 <action>plant_move</action>
 <barcode_id>M00342Q00003990761</barcode_id>
 <barcode_id>M00342Q00006357962</barcode_id>
 <room>2</room>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

plant_destroy_schedule

The plant_destroy_schedule function will allow a cultivator to schedule for destruction a plant or set of plants. The optional override parameter can be used in instances where a user successfully initiated a scheduled

destruction across one or more plants but, e.g. they failed to commit locally to a user's platform. Essentially, it will suppress the error message that indicates an item has already been scheduled and will handle any additional items within the list. It will NOT suppress any other error messages.

Parameters:

action	variable length text field
reason	variable length text field
barcodeid	Array of 1 or more text fields representing the plants
override	Optional, 0 or 1 Boolean value (defaults to 0 if omitted)
reason_extended	Integer value corresponding to a pre-defined set of values. If set to 0 or not provided, the reason field must be provided. The acceptable values are: 0 (Other), 1 (Waste), 2 (Unhealthy or Died), 3 (Infestation), 4 (Product Return), 5 (Mistake), 6 (Spoilage), 7 (Quality Control).

```
<xml>
 <API>4.0</API>
 <action>plant_destroy_schedule</action>
 <barcode_id>M00342Q00003990761</barcode_id>
 <barcode_id>M00342Q00006357962</barcode_id>
 <reason>Mold</reason>
  <reason_extended>0</reason_extended>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

plant_destroy_schedule_undo

The plant_destroy_schedule_undo function will allow a licensee to correct plants that were accidentally scheduled for destruction; before they've actually been destroyed.

Parameters:

action	variable length text field
reason	variable length text field
barcodeid	Array of 1 or more text fields representing the plants

```
<xml>
 <API>4.0</API>
 <action>plant_destroy_schedule_undo</action>
 <barcode_id>M00342Q00003990761</barcode_id>
 <barcode_id>M00342Q00006357962</barcode_id>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

plant_destroy

The plant_destroy function will allow a licensee to destroy a plant or set of plants. Please see the plant_destroy_schedule function for an explanation on the optional override parameter.

Parameters:

action	variable length text field
barcodeid	Array of 1 or more text fields representing the plants
override	Optional, 0 or 1 Boolean value (defaults to 0 if omitted)

```
<xml>
 <API>4.0</API>
 <action>plant_destroy</action>
 <barcode_id>M00342Q00003990761</barcode_id>
 <barcode_id>M00342Q00006357962</barcode_id>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

plant_destroy_undo

The plant_destroy_undo function will allow a licensee to undo a plant destruction. Once completed the plant or plants will be returned to the "Scheduled For Destruction" state.

Parameters:

action	variable length text field
barcodeid	Array of 1 or more text fields representing the destroyed plants

```
<xml>
 <API>4.0</API>
 <action>plant_destroy_undo</action>
 <barcode_id>M00342Q00003990761</barcode_id>
 <barcode_id>M00342Q00006357962</barcode_id>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

plant_harvest_schedule

The plant_harvest_schedule function will notify the traceability system of intent to begin harvesting a plant or set of plants. This notification must occur before the plant_harvest is called on these plants.

Parameters:

action	variable length text field
barcodeid	Array of 1 or more text fields representing the plants

```
<xml>
 <API>4.0</API>
 <action>plant_harvest_schedule</action>
 <barcode_id>M00342Q00003990761</barcode_id>
 <barcode_id>M00342Q00006357962</barcode_id>
</xml>
```

Returned Parameters:

success	Boolean value

transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

plant_harvest_schedule_undo

The plant_harvest_schedule_undo function will allow a licensee to correct plants that were accidentally scheduled for harvest; before they've actually been harvested.

Parameters:

action	variable length text field
barcodeid	Array of 1 or more text fields representing the plants

```
<xml>
 <API>4.0</API>
 <action>plant_harvest_schedule_undo</action>
 <barcode_id>M00342Q00003990761</barcode_id>
 <barcode_id>M00342Q00006357962</barcode_id>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

plant_harvest

The plant_harvest function will begin the process of harvesting a plant. This will move said plant from the "growing" phase to the "drying" phase. During this process, a cultivator must take, at a minimum, a wet weight of the plant. In addition, a cultivator may also gather two additional derivatives defined by their inventory type. Specifically, the system requires inventory type 6 (Flower) and optionally allows type 9 (Other Plant Material) and type 27 (Waste).

Harvests can be partial, as well. In other words, if part of the plant is harvested and the rest of the plant will be processed later (commonly known as re-flowering), then the collectadditional parameter should be 1. This will inform the traceability system to expect another additional wet weight.

Each harvest event should be on a per-plant basis. Every individual plant will need its own wet weight reported. Both Other Plant Material and Waste collected during this process will receive random unique identifiers. For Other Plant Material, this will facilitate the process of creating a lot. For Waste, this will allow a user to accumulate waste in a traceable manner and schedule a destruction event at a later point.

Parameters:

action	variable length text field
collectiontime	Optional, Unix 32-bit integer timestamp, defaults to current time
barcodeid	Array of one or more unique plant identifiers
weights	Array of 1 or more nodes containing weight information
amount	decimal value
invtype	integer value representing the derivative type
uom	variable length text field. Valid values are: g, mg, kg, oz, lb. These represent: grams, milligrams, kilograms, ounces and pounds.
collectadditional	Keeps the plant in the growing phase and allows the user to take another wet weight of the plant(s) at a later point that will compound to the original wet weight.
new_room	Optional, will move the now drying plant(s) to another plant room.
wet	Optional, will move the plant into inventory for drying at another facility.

Example:

```
<xml>
 <API>4.0</API>
 <action>plant_harvest</action>
 <barcodeid>M00342Q00003990761</barcodeid>
 <collectadditional>0</collectadditional>
 <new_room>3</new_room>
 <weights>
   <amount>250.00</amount>
   <invtype>6</invtype>
   <uom>g</uom>
 </weights>
 <weights>
   <amount>500.00</amount>
   <invtype>9</invtype>
   <uom>g</uom>
 </weights>
 <weights>
   <amount>125.00</amount>
```

```
    <invtype>27</invtype>
    <uom>g</uom>
  </weights>
</xml>
```

Returns:

```
<xml>
  <derivatives>
    <barcode_id>M00342Q00092237875</barcode_id>
    <barcode_type>9</barcode_type>
  </derivatives>
  <derivatives>
    <barcode_id>M00342Q00270897891</barcode_id>
    <barcode_type>27</barcode_type>
  </derivatives>

  <sessiontime>1384487873</sessiontime>
  <success>1</success>
  <transactionid>3284</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
derivatives	Array of 1 or more nodes containing new identifiers with their associated inventory types.
barcode_id	New identifier for the inventory specified by barcode_type.
barcode_type	Specifies the type of derivative.

plant_harvest_undo

The plant_harvest_undo function will allow a licensee to correct plants that were accidentally harvested and need to be placed into the growth phase of cultivation. If derivative items were collected and have been altered; this function will report an error. Or, if the plant is no longer in cultivation this function will also report an error. For example, if an inventory adjustment were made to flower collected from this specific cure process, the system will then disallow the action. It is designed to correct simple mistakes and actively prevents individuals from abusing the function to hide inventory.

Parameters:

action	variable length text field
transactionid	The transaction ID of the original plant_harvest call

```
<xml>
 <API>4.0</API>
 <action>plant_harvest_undo</action>
 <transactionid>123455</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

plant_waste_weigh

The plant_waste_weigh function will allow a cultivator to take a general waste weight for destruction accountability at a later point. General leaf, stem, veg trimming, etc. collection can thus be facilitated in a more generalized fashion without unduly burdening a licensee.

The return inventory will be typed as 27 and must be scheduled for destruction at a later point.

Parameters:

action	variable length text field
collectiontime	Optional, Unix 32-bit integer timestamp, defaults to current time
weight	decimal value
uom	variable length text field. Valid values are: g, mg, kg, oz, lb. These represent: grams, milligrams, kilograms, ounces and pounds.
location	license number of location

Example:

```
<xml>
 <API>4.0</API>
 <action>plant_waste_weigh</action>
 <location>123456</location>
 <weight>250.00</weight>
 <uom>g</uom>
</xml>
```

Returns:

```
<xml>
 <barcode_id>M00342Q00270897891</barcode_id>
 <barcode_type>27</barcode_type>
 <sessiontime>1384487873</sessiontime>
 <success>1</success>
 <transactionid>3286</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
barcode_id	New identifier for the inventory specified by barcode_type.
barcode_type	Specifies the type of derivative, always 27.

plant_cure

The plant_cure function will begin the process of curing a plant. This will move said plant from the drying phase to inventory. During this process, a cultivator must take, at a minimum, a dry weight of the plant. In addition, a cultivator may also gather additional derivatives defined by their inventory type. Specifically, the system requires inventory type 6 (Flower) and optionally allows type 9 (Other Plant Material) and type 27 (Waste).

If the cultivator is doing a partial harvest/cure, the plant can pass through this function again to accumulate an additional dry weight. If the cultivator is re-flowering, ensure the collectadditional field is set to 1.

Parameters:

action	variable length text field
collectiontime	Optional, Unix 32-bit integer timestamp, defaults to current time
barcodeid	Array of one or more unique plant identifiers
weights	Array of 1 or more nodes containing weight information
amount	decimal value
invtype	integer value representing the derivative type
uom	variable length text field. Valid values are: g, mg, kg, oz, lb. These represent: grams, milligrams, kilograms, ounces and pounds.
collectadditional	Keeps the plant in the growing phase and allows the user to take another wet weight of the plant(s) at a later point that will compound to the original wet weight.
room	integer, room the collection occurred in
location	license number of location

Example:

```
<xml>
 <API>4.0</API>
 <action>plant_cure</action>
 <barcodeid>M00342Q00006357962</barcodeid>
 <collectadditional>0</collectadditional>
 <location>12345</location>
 <room>2</room>
 <weights>
  <amount>250.00</amount>
  <invtype>6</invtype>
  <uom>g</uom>
 </weights>
 <weights>
  <amount>500.00</amount>
  <invtype>9</invtype>
  <uom>g</uom>
 </weights>
 <weights>
  <amount>125.00</amount>
```

```
    <invtype>27</invtype>
    <uom>g</uom>
  </weights>
</xml>
```

Returns:

```
<xml>
  <derivatives>
    <barcode_id>M00342Q00066590176</barcode_id>
    <barcode_type>6</barcode_type>
  </derivatives>
  <derivatives>
    <barcode_id>M00342Q00092237875</barcode_id>
    <barcode_type>9</barcode_type>
  </derivatives>
  <sessiontime>1384487873</sessiontime>
  <success>1</success>
  <transactionid>3290</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
derivatives	Array of 1 or more nodes containing new identifiers with their associated inventory types.
barcode_id	New identifier for the inventory specified by barcode_type.
barcode_type	Specifies the type of derivative.

plant_cure_undo

The plant_cure_undo function will allow a licensee to correct plants that were accidentally cured and need to be placed into cultivation. If any of the derivative items have been altered, this function will report an error. For example, if an inventory adjustment were made to flower collected from this specific cure process, the system will then disallow the action. It is designed to correct simple mistakes and actively prevents individuals from abusing the function to hide inventory.

Parameters:

action	variable length text field
transactionid	The transaction ID of the original plant_cure call

```
<xml>
 <API>4.0</API>
 <action>plant_cure_undo</action>
 <transactionid>123456</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

plant_convert_to_inventory

The plant_convert_to_inventory function will allow a licensee to convert a plant that is growing (but not flowering) into an inventory item that can then be transferred and sold. Once converted, the new item will keep its identifier but will now have an inventory type of 12 (Mature Plant).

Parameters:

action	variable length text field
barcodeid	Array of 1 or more text fields representing the plants to convert

```
<xml>
 <API>4.0</API>
 <action>plant_convert_to_inventory</action>
 <barcodeid>M00342Q00003990761</barcodeid>
 <barcodeid>M00342Q00006357962</barcodeid>
</xml>
```

Returns:

```
<xml>
 <sessiontime>1384487873</sessiontime>
 <success>1</success>
 <transactionid>3290</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

plant_yield_modify

The plant_yield_modify function will allow direct access to modify previously stored values for harvest and cure collections. The user will need to specify one transaction at a time. The integrator is, of course, free to hide this from the end-user with multiple API calls behind the scenes if they display the capability to modify collected values in a unique or innovative way.

The user can, however, specify all values that would have been specifiable at the time of the original transaction. That is, if the transaction relates to the plant_harvest, wet weight and any derivative can be specified. If the original transaction was a plant_cure, dry weight could be specified, instead. Only values that are included will be modified. If a user wishes to zero out a value, it must be declared. Null or absent values will retain their previous values.

Parameters:

action	variable length text field
collectiontime	Optional, Unix 32-bit integer timestamp, defaults to current time
transactionid	integer, the transaction to correct
weights	Array of 1 or more nodes containing weight information
amount	Optional, decimal value
invtype	integer value representing the derivative type
uom	variable length text field. Valid values are: g, mg, kg, oz, lb. These represent: grams, milligrams, kilograms, ounces and pounds.

Example:

```
<xml>
 <API>4.0</API>
 <action>plant_yield_modify</action>
 <transactionid>3290</transactionid>
 <weights>
  <amount>450.00</amount>
  <invtype>6</invtype>
  <uom>g</uom>
```

```
  </weights>
</xml>
```

Returns:

```
<xml>
  <sessiontime>1384487873</sessiontime>
  <success>1</success>
  <transactionid>3309</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
derivatives	Optional, Array of 1 or more nodes containing new identifiers with their associated inventory types. Only returned if the inventory type was previously unaccounted for.
barcode_id	New identifier for the inventory specified by barcode_type.
barcode_type	Specifies the type of derivative.

plant_modify

The plant_modify function will allow direct access to modify previously stored values for a plant. The user will need to specify one plant at a time. The integrator is, of course, free to hide this from the end-user with multiple API calls behind the scenes if they display the capability to modify collected values in a unique or innovative way.

The user will need to specify the barcode id and, optionally the new strain, new mother flag or new room.

Parameters:

action	variable length text field
strain	Optional, variable length text field of the new strain name
room	Optional, integer value that will move the plant to another plant room.
mother	Optional, integer value indicating if the plant is a mother plant
birthdate	Optional, 8 character birthdate in the following format: YYYYMMDD. If not provided, the system will default to the current date.

Example:

```
<xml>
  <API>4.0</API>
  <action>plant_modify</action>
  <barcodeid>M00342Q00003990761</barcodeid>
  <strain>Blueberry</strain>
  <room>6</room>
  <mother>1</mother>
</xml>
```

Returns:

```
<xml>
  <sessiontime>1384487873</sessiontime>
  <success>1</success>
  <transactionid>3309</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

additive_type_add

The additive_type_add function adds additional additive types for use other than the three defaults (listed below).

Id	Name	Default_Unit
1	Nutrient	3
2	Pesticide	7
3	Custom Mix	

Parameters:

action	variable length text field
id	integer, unique identifier
name	variable length text field of name
default_unit	optional, integer value of default unit (reference table for values)
location	license of location

Example:

```
<xml>
  <API>4.0</API>
  <action>additive_type_add</action>
  <id>4</id>
  <name>Herbicide</name>
  <default_unit>2</default_unit>
  <location>000000006</location>
</xml>
```

Returns:

```
<xml>
  <sessiontime>1531179485</sessiontime>
  <success>1</success>
  <transactionid>3290</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

additive_type_modify

The additive_type_modify function modifies a previously created additive type. Please note that the default additives types cannot be modified (reference additive_type_add for table of default types).

Parameters:

action	variable length text field
id	integer value of type identifier

name	variable length text field of name
default_unit	optional, integer value of default unit (reference table for values)
location	license of location

Example:

```
<xml>
 <API>4.0</API>
 <action>additive_type_modify</action>
 <id>4</id>
 <name>Herbicide</name>
 <default_unit></default_unit>
 <location>000000006</location>
</xml>
```

Returns:

```
<xml>
 <sessiontime>1384487873</sessiontime>
 <success>1</success>
 <transactionid>3290</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

additive_type_remove

The additive_type_remove function removes a previously created additive type.

Parameters:

action	variable length text field
id	integer value
location	license of location

Returns:

```
<xml>
```

```
 <API>4.0</API>
 <action>additive_type_remove</action>
 <id>4</id>
 <location>000000006</location>
</xml>
```

Returns:

```
<xml>
 <sessiontime>1384487873</sessiontime>
 <success>1</success>
 <transactionid>3290</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

additive_add

The additive_add function allows the user to store different additives for later use.

Parameters:

action	variable length text field
location	license number of location
additive_type	integer, associated to additive type
id	integer value of additive
lot_number	Optional, variable length text field
name	Variable length text field of the new additive name
weight_based	Optional, 0 or 1 Boolean value (defaults to 0 if omitted)

Example:

```
<xml>
 <API>4.0</API>
 <action>additive_add</action>
 <location>12345</location>
 <additive_type>3</additive_type>
 <id>2</id>
```

```
  <lot_number>Batch 1134-1B</lot_number>
  <name>Hydro Liquid</name>
  <weight_based>0</weight_based>
</xml>
```

Returns:

```
<xml>
  <sessiontime>1531160987</sessiontime>
  <success>1</success>
  <transactionid>2385</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

additive_modify

The additive_modify function allows the user to modify previously recorded additives.

Parameters:

action	variable length text field
location	license number of location
additive_type	integer value of the associated additive type
id	integer, unique identifier
lot_number	optional, variable length text field
name	variable length text field
weight_based	optional, 0 or 1 Boolean value

Example:

```
<xml>
  <API>4.0</API>
  <action>additive_modify</action>
  <location>12345</location>
  <additive_type>12</additive_type>
  <id>2</id>
  <lot_number>Batch 1134-1A</lot_number>
```

```
  <name>Hydro Liquid - Special</name>
  <weight_based>1</weight_based>
</xml>
```

Returns:

```
<xml>
  <sessiontime>1531166264</sessiontime>
  <success>1</success>
  <transactionid>2385</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

additive_remove

The additive_remove function will remove a previously stored additive.

Parameters:

action	variable length text field
location	License number of location
id	integer value of additive

Example:

```
<xml>
  <API>4.0</API>
  <action>additive_remove</action>
  <location>12345</location>
  <id>2</id>
</xml>
```

Returns:

```
<xml>
  <sessiontime>1531166264</sessiontime>
  <success>1</success>
```

```
  <transactionid>2385</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

plant_additive_add

The plant_additive_add function allows the licensee to record when an additive is added to a plant or group of plants. Entry in the quantity_mg or quantity_ml field varies based on whether the additive is weight based.

Parameters:

action	variable length text field
location	license number of location
data	Array of 1 or more nodes containing additive information
added_by	variable length text field of user name
additiveid	integer
default_unit	integer (reference table)
plantid	text field representing the plant id
quantity	numeric value of quantity
quantity_mg	numeric value of quantity represented in mg
quantity_ml	numeric value of quantity represented in ml

Example:

```
<xml>
  <API>4.0</API>
  <action>plant_additive_add</action>
  <location>123456</location>
  <data>
    <added_by>user@biotrackthc.com</added_by>
    <additiveid>11</additiveid>
    <default_unit>9</default_unit>
    <plantid>M00342Q00003990761</plantid>
    <quantity>2</quantity>
    <quantity_mg></quantity_mg>
    <quantity_ml> 59.147</quantity_ml>
```

```
  </data>
</xml>
```

Returns:

```
<xml>
  <sessiontime>1531160987</sessiontime>
  <success>1</success>
  <transactionid>2385</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

plant_additive_remove

The plant_additive_remove function removes an additive applied to a plant. The user will need to specify one plant at a time. The integrator is, of course, free to hide this from the end-user with multiple API calls behind the scenes.

Parameters:

action	variable length text field
id	integer value of additive
plantid	text field
location	license number of location
transactionid	integer value

Example:

```
<xml>
  <API>4.0</API>
  <action>plant_additive_remove</action>
  <id>10</id>
  <plantid>M00342Q00003990761</plantid>
  <location>123456</location>
  <transactionid>2388</transactionid>
</xml>
```

Returns:

```
<xml>
  <success>1</success>
  <transactionid>2400</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

Chapter 5: Inventory

In this chapter, you'll learn how to:

- ✓ Adjust and audit inventory
- ✓ Create new inventory
- ✓ Convert inventory
- ✓ Perform inventory lookups

inventory_adjust

The inventory_adjust function will allow a licensee to adjust the amount or quantity of an inventory item. The type field can represent one of the following: 1 (General Inventory Audit), 2 (Theft), 3, (Seizure by Federal, State, Local or Tribal Law Enforcement), 4 (Correcting a mistake), 5 (Moisture loss, e.g. wet other plant material), 6 (Disposal), 7 (Wastage). For backward compatibility, reason and type can be provided outside of the data array as a fallback default. The integrator can also choose whether to provide the new quantity to adjust to (with the quantity parameter) or can simply provide the remove_quantity parameter. It is recommended to only provide one or the other. The system will look for remove_quantity first and fallback to quantity if not found.

Parameters:

action	variable length text field
data	Array of 1 or more nodes containing inventory information
barcodeid	inventory identifier
quantity	Decimal value, optional if remove_quantity is provided, new quantity to adjust to.
quantity_uom	variable length text field. Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.
remove_quantity	Decimal value, optional if quantity is provided, quantity to remove. Does not need to be remaining quantity (can be a partial removal).
remove_quantity_uom	variable length text field. Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.
reason	variable length text field explaining in greater detail the reason for the removal or addition of inventory
type	Integer value representing the type of adjustment.

```
<xml>
 <API>4.0</API>
 <action>inventory_adjust</action>
 <data>
   <barcode_id>M00342Q00137887615</barcode_id>
   <quantity>690</quantity>
   <reason>Testing</reason>
   <type>1</type>
  </data>
</xml>
```

Return example:

```
<xml>
 <sessiontime>1384476925</sessiontime>
 <success>1</success>
 <transactionid>3311</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

inventory_adjust_usable

The inventory_adjust_usable function will allow a licensee to adjust the usable amount of an eligible inventory item. The current eligible inventory types for this function call are: 24 (Extract for Inhalation), 26 (Sample Jar), 28 (Usable Cannabis), 31 (Cannabis Mix Packaged). This function cannot be used to add inventory to the system via adjustment, but rather it will tie any quantity adjustments directly to the usable amount. That is, if an item has a current quantity of 2 and a usable amount of 2 grams; this function could then be used to change the item quantity to 1 which would cause the system to change the usable amount to 4 grams.

Parameters:

action	variable length text field
barcodeid	inventory identifier
quantity	Integer value, greater than zero, that represents the correct number of units for the identifier.

```
<xml>
 <API>4.0</API>
 <action>inventory_adjust_usable</action>
  <barcode_id>MJ1234507285760184</barcode_id>
  <quantity>1</quantity>
</xml>
```

Return example:

```
<xml>
 <sessiontime>1384476925</sessiontime>
 <success>1</success>
 <transactionid>3311</transactionid>
 <usableweight>4.00</usableweight>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
usableweight	Decimal value that represents new usable weight of item

inventory_destroy_schedule

The inventory_destroy_schedule function will notify the traceability system of intent to destroy an inventory item. Please see the plant_destroy_schedule function for an explanation on the optional override parameter.

Parameters:

action	variable length text field
barcodeid	Array of 1 or more text fields representing the inventory items
reason	reason for the destruction
override	Optional, 0 or 1 Boolean value (defaults to 0 if omitted)
reason_extended	Integer value corresponding to a pre-defined set of values. If set to 0 or not provided, the reason field must be provided. The acceptable values are: 0 (Other), 1 (Waste), 2 (Unhealthy or Died), 3 (Infestation), 4

(Product Return), 5 (Mistake), 6 (Spoilage), 7 (Quality Control).

```
<xml>
 <API>4.0</API>
 <action>inventory_destroy_schedule</action>
 <barcode_id>MJ1234507285760184</barcode_id>
 <barcode_id>M00342Q00245663780</barcode_id>
 <reason>Mold</reason>
  <reason_extended>0</reason_extended>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

inventory_destroy

The inventory_destroy function will allow a licensee to destroy an item that has been previously scheduled for destruction. Please see the plant_destroy_schedule function for an explanation on the optional override parameter.

Parameters:

action	variable length text field
barcodeid	inventory identifier
reason	reason for the removal of inventory
override	Optional, 0 or 1 Boolean value (defaults to 0 if omitted)

```
<xml>
 <API>4.0</API>
 <action>inventory_destroy</action>
 <barcode_id>M00342Q00245663780</barcode_id>
</xml>
```

Return example:

```
<xml>
 <sessiontime>1384476925</sessiontime>
```

```
  <success>1</success>
  <transactionid>3411</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

inventory_move

The inventory_move function will update the current room for the specified inventory items. Essentially, it allows a user to move inventory from one room to another.

Parameters:

action	variable length text field
data	Array of 1 or more nodes containing inventory information
barcodeid	inventory identifier
room	Integer value, represents the identification number of a room

```
<xml>
  <API>4.0</API>
  <action>inventory_move</action>
  <data>
    <barcode_id>M00342Q00245663780</barcode_id>
    <room>1</room>
  </data>
 <data>
    <barcode_id>MJ1234507285760184</barcode_id>
    <room>1</room>
  </data>
</xml>
```

Return example:

```
<xml>
  <sessiontime>1384476925</sessiontime>
```

```
  <success>1</success>
  <transactionid>3626</transactionid>
</xml>
```

inventory_check

The inventory_check function can be used to perform a cursory lookup on an item before an inbound inventory_transfer from an outside licensee. It will pull various pieces of inventory on the inventory identifiers specified in the request. This information can include: strain, quantity available, usable weight (if applicable), product (if applicable) and inventory type.

Parameters:

action	variable length text field
barcodeid	Array of 1 or more text fields representing the inventory to lookup

```
<xml>
  <API>4.0</API>
  <action>inventory_check</action>
  <barcode_id>MJ1234507285760184</barcode_id>
  <barcode_id>MJ1234507307763237</barcode_id>
</xml>
```

Returned Parameters:

success	Boolean value
data	Array of 1 or more nodes containing inventory information
barcode_id	inventory identifier
strain	variable length text field
product	variable length text field
quantity	decimal value
usableweight	decimal value (in grams).
invtype	integer value based on pre-defined inventory types

Return example:

```
<xml>
  <data>
```

```
    <barcode_id> MJ1234507285760184</barcode_id>
    <invtype>28</invtype>
    <quantity>10</quantity>
    <usableweight>3.50</usableweight>
   <strain>Blueberry</strain>
  </data>
  <success>1</success>
</xml>
```

inventory_new

The inventory_new function can be used to create new inventory not previously entered into the system. It may be used for the first 30 days of operation without a source_id. Subsequent calls to this function will require a source_id of a plant in cultivation that has been designated as a mother plant. All types, sans types which require a usable value (Cannabis Extract for Inhalation, Sample Jar, Usable Cannabis, Cannabis Mix Packaged and Cannabis Mix Infused) may be used during the initial window. After the 30 day period, only three types may be provided: Seed, Clone and Plant Tissue.

Parameters:

action	variable length text field
location	license number of location
data	Array of 1 or more nodes containing new inventory information
strain	variable length text field
quantity	integer value
invtype	integer, corresponds to the inventory type system
source_id	text field, optional when within the 30 day period

```
<xml>
  <API>4.0</API>
  <action>inventory_new</action>
  <data>
    <invtype>12</invtype>
    <quantity>50</quantity>
    <strain>Blueberry</strain>
  </data>
  <location>12345</location>
</xml>
```

Return example:

```
<xml>
 <barcode_id>M1119150012288l694</barcode_id>
 <sessiontime>1384476925</sessiontime>
 <success>1</success>
 <transactionid>3278</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
barcode_id	Array of 1 or more text fields representing the new unique identifiers attached to the inventory items

inventory_manifest

The inventory_manifest function will notify the traceability system of intent to transfer an inventory item. This function will need to be called in instances of transfers from one licensee to another. It will also need to be called for licensees which possess multiples licenses (e.g. Cultivator + Processors) that possess different license numbers.

Parameters:

action	variable length text field
employee_id	variable length text field
vehicle_id	integer value
location	license number of origin location
stop_overview	Array of 1 or more nodes containing stop information
approximate_departure	Unix 32-bit integer timestamp, approximate departure time
approximate_arrival	Unix 32-bit integer timestamp, approximate arrival time
approximate_route	variable length text field, route that will be used
vendor_license	license number of vendor the item(s) are being transferred to
stop_number	stop number of the overview, integer greater than or equal to 1
barcodeid	Array of 1 or more text fields representing the items to be transferred on the specific stop

new_room	Optional, can specify the item(s) have been placed into e.g. a quarantine room.

Example:

```
<xml>
 <API>4.0</API>
 <action>inventory_manifest</action>
 <location>12345</location>
  <stop_overview>
<approximate_departure>1384476925</approximate_departure >
<approximate_arrival>1384486925</approximate_arrival>
<approximate_route>Turn left on Main St.</approximate_route>
<vendor_license>25678787644</vendor_license >
<stop_number>1</stop_number>
 <barcode_id>MJ1234507285760184</barcode_id>
 <barcode_id>MJ1234507307763237</barcode_id>
</stop_overview>
 <employee_id>23468</employee_id>
 <vehicle_id>2</vehicle_id>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
barcode_id	Unique identifier attached to the manifest

inventory_manifest_pickup

The inventory_manifest_pickup function will notify the traceability system of intent to transfer an inventory item that will be picked up by the vendor rather than transferred by the licensee. This function will need to be called in instances of transfers from one licensee to another. For internal transfers (e.g. from one part of a facility to another), the inventory_manifest should be used. This manifest type can only have one stop.

Parameters:

action	variable length text field
employee_name	variable length text field
employee_id	unique variable length text field

employee_dob	variable length text field in the format MM/DD/YYYY
vehicle_color	variable length text field
vehicle_make	variable length text field
vehicle_model	variable length text field
vehicle_plate	variable length text field
vehicle_vin	variable length text field
vehicle_year	integer
location	license number of origin location
stop_overview	Array of 1 or more nodes containing stop information
approximate_departure	Unix 32-bit integer timestamp, approximate departure time
approximate_arrival	Unix 32-bit integer timestamp, approximate arrival time
approximate_route	variable length text field, route that will be used
vendor_license	license number of vendor the item(s) are being transferred to
stop_number	stop number of the overview, integer greater than or equal to 1
barcodeid	Array of 1 or more text fields representing the items to be transferred on the specific stop
new_room	Optional, can specify the item(s) have been placed into e.g. a quarantine room.

Example:

```
<xml>
 <API>4.0</API>
 <action>inventory_manifest_pickup</action>
 <employee_dob>01/01/1980</employee_dob>
 <employee_id>124</employee_id>
 <employee_name>Joe Everyman</employee_name>
 <vehicle_color>Black</vehicle_color>
 <vehicle_make>Ford</vehicle_make>
 <vehicle_model>Focus</vehicle_model>
 <vehicle_plate>111</vehicle_plate>
 <vehicle_vin>123</vehicle_vin>
 <vehicle_year>1990</vehicle_year>
 <location>12345</location>
  <stop_overview>
```

```
<approximate_departure>1384476925</approximate_departure >
<approximate_arrival>1384486925</approximate_arrival>
<approximate_route>Turn left on Main St.</approximate_route>
<vendor_license>25678787644</vendor_license >
<stop_number>1</stop_number>
  <barcode_id>MJ1234507285760184</barcode_id>
  <barcode_id>MJ1234507307763237</barcode_id>
</stop_overview>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
barcode_id	Unique identifier attached to the manifest

inventory_manifest_lookup

The inventory_manifest_lookup function can be used to offer a heads up of shipments that have been both manifested and transferred out of one licensee and are ready to be transferred into the receiver's inventory.

Parameters:

action	variable length text field
location	license number of location

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_manifest_lookup</action>
  <location>12345</location>
</xml>
```

Return example:

```
<xml>
```

```
  <data>
  <item_count>1</item_count>
    <license_number>18750</license_number>
    <manifest_id>7949844847294004</manifest_id>
    <return_indicated>0</return_indicated>
    <trade_name>Trade 24</trade_name>
    <transfer_date>01/21/2014</transfer_date>
  </data>
  <success>1</success>
  <sessiontime>1390548537</sessiontime>
</xml>
```

Returned Parameters:

success	Boolean value
sessiontime	Unix 32-bit integer timestamp
data	Array of 1 or more nodes containing transportation information
item_count	Integer, number of separate items
license_number	variable length text field, license number of shipping entity
manifest_id	variable length text field, unique manifest identifier
return_indicated	Boolean value identifying if transfer is a return
trade_name	variable length text field, name of the shipping entity
transfer_date	Date of actual shipment

inventory_manifest_modify

The inventory_manifest_modify function will modify an existing manifest that has not been shipped yet. Currently, it can be used to modify or add an employee/driver on a manifest. The employee ID can be provided for a regular manifest whereas the full driver information will need to be provided for a pick-up manifest.

Parameters:

action	variable length text field
manifest_id	manifest identifier
employee_name	variable length text field, optional for regular manifests
employee_id	unique variable length text field
employee_dob	variable length text field in the format MM/DD/YYYY, optional for regular manifests

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_manifest_modify</action>
  <manifest_id>1234567812345678</manifest_id>
  <employee_id>23468</employee_id>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

inventory_manifest_void

The inventory_manifest_void function will cancel a manifest that has been previously filed.

Parameters:

action	variable length text field
manifest_id	manifest identifier

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_manifest_void</action>
  <manifest_id>1234567812345678</manifest_id>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

inventory_manifest_void_stop

The inventory_manifest_void_stop function will void a specific stop on a manifest that has been previously filed. If there are no remaining active stops, the manifest itself will be automatically voided.

Parameters:

action	variable length text field
manifest_id	manifest identifier
stop_number	integer value

Example:

```
<xml>
 <API>4.0</API>
 <action>inventory_manifest_void</action>
 <manifest_id>1234567812345678</manifest_id>
 <stop_number>1</stop_number>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

inventory_manifest_void_items

The inventory_manifest_void_items function will void one or more items from a manifest. If there are no remaining active items in a specific stop, that stop will be automatically voided. If there are no remaining active items in the entire manifest itself, the manifest will be automatically voided.

Parameters:

action	variable length text field
manifest_id	manifest identifier
barcodeid	Array of 1 or more inventory identifiers

Example:

```
<xml>
 <API>4.0</API>
```

```
  <action>inventory_manifest_void_items</action>
  <manifest_id>1234567812345678</manifest_id>
  <barcode_id>MJ1234507285760184</barcode_id>
  <barcode_id>MJ1234507307763237</barcode_id>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

inventory_manifest_order

The inventory_manifest_order function will notify the traceability system of intent to transfer an inventory item. This function will need to be called in instances of transfers from one licensee to another. This is for inventory that is not medicated or was grown outside of the BioTrackTHC network.

Parameters:

action	variable length text field
location	license number of location
data	array of 1 or more nodes containing inventory information
invtype	integer, corresponds to the inventory type system
item_number	integer, that uniquely identifies each data element. If no item_number was provided upon submission, the system will provide this starting at 0
productname	variable length text field
quantity	decimal value
uom	variable length text field. Valid values are: g, mg, kg, oz, lb. These represent: grams, milligrams, kilograms, ounces and pounds.
strain	variable length text field
usableweight	decimal value (in grams)
vendor_license	license number of vendor the item(s) are being transferred to

Example:

```
<xml>
```

```
 <API>4.0</API>
 <action>inventory_manifest_order</action>
 <data>
    <invtype>13</invtype>
    <item_number>0</item_number>
    <productname>Og Green Kush 1G</productname>
    <quantity>1.50</quantity>
    <strain>PR Green Kush</strain>
    <usableweight></usableweight>
 </data>
 <location>999917</location>
 <vendor_license>000016</vendor_license>
</xml>
```

Return example:

```
<xml>
 <data>
 <barcode_id>6710706103781440</barcode_id>
 <manifest_id>1234567812345678</manifest_id>
 <sessiontime>1473586311</sessiontime>
 <success>1</success
 <transactionid>4820</transactionid>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer
barcode_id	Unique identifier
manifest_id	variable length text field, unique manifest identifier

inventory_transfer_lookup

The inventory_transfer_lookup function can be after the inventory_manifest_lookup function, or, alternatively, after having the manifest identifier in hand to retrieve specific details on the receiving items.

Parameters:

action	variable length text field
location	license number of location

Example:

```
<xml>
 <API>4.0</API>
 <action>inventory_transfer_lookup</action>
 <location>12345</location>
 <manifest_id>1234567812345678</manifest_id>
</xml>
```

Return example:

```
<xml>
 <data>
  <barcode_id>6710706103781440</barcode_id>
  <product>Space Cookie</product>
  <quantity>5</quantity>
  <inventorytype>22</inventorytype>
  <strain></strain>
  <description>Infused Edible</description>
  <is_sample>1</is_sample>
 </data>
 <success>1</success>
 <sessiontime>1390548537</sessiontime>
</xml>
```

Returned Parameters:

success	Boolean value
sessiontime	Unix 32-bit integer timestamp
data	Array of 1 or more nodes containing inventory information
barcode_id	Unique identifier
product	variable length text field, name of product where applicable
quantity	decimal value
inventorytype	integer value based on pre-defined inventory types
strain	variable length text field, name of product where applicable

description	variable length text field, description of item
sample_id	variable length text field, ID of QA sample if directly taken from item
is_return	Boolean value, returned only if item indicates it should be accepted as a return
is_sample	Boolean value, true if the item has been created as a vendor sample
usableweight	Optional, decimal value if the inventory type supports a usable weight

inventory_transfer_outbound

The inventory_transfer_outbound function can be used to transfer inventory that already exists in the system. A manifest must be filed prior to all transfers.

Parameters:

action	variable length text field
manifest_id	manifest identifier obtained from previously filed manifest
data	Array of 1 or more nodes containing inventory information
barcodeid	inventory identifier
price	Optional if inter-UBI transfer, decimal value that indicates how much the item was sold for before any applicable taxes.

```
<xml>
 <API>4.0</API>
 <action>inventory_transfer_outbound</action>
 <manifest_id>1234567812345678</manifest_id>
 <data>
   <barcode_id>MJ1234507285760184</barcode_id>
   <price>100.00</price>
 </data>
</xml>
```

Return example:

```
<xml>
 <sessiontime>1384476925</sessiontime>
```

```
 <success>1</success>
 <transactionid>3778</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

inventory_transfer_outbound_return_lookup

The inventory_transfer_outbound_return_lookup function can be used to perform a lookup of any items that have been sent, but not fully received by the recipient.

Parameters:

action	variable length text field
location	license number of location

Example:

```
<xml>
 <API>4.0</API>
<action>inventory_transfer_outbound_return_lookup</action>
 <location>12345</location>
</xml>
```

Return example:

```
<xml>
 <data>
  <barcode_id>MJ1234507285760184</barcode_id>
  <description>Usable Cannabis</description>
  <inventorytype>28</inventorytype>
  <license_number>12845</license_number>
  <manifest_id>1234567812345678</manifest_id>
  <price>100.00</price>
  <quantity>4</quantity>
  <received>1</received>
  <received_quantity>1</received_quantity>
```

```
    <stop_number>1</stop_number>
    <strain>Blueberry</strain>
    <trade_name>Retail 123</trade_name>
    <transfer_date>04/16/2015</transfer_date>
    <usableweight>2.00</usableweight>
    <return_available>1</return_available>
  </data>
  <sessiontime>1429314044</sessiontime>
  <success>1</success>
</xml>
```

Returned Parameters:

success	Boolean value
sessiontime	Unix 32-bit integer timestamp
data	Array of 1 or more nodes containing transportation information
barcode_id	Unique identifier
description	Variable length text field that describes the item being transported.
inventorytype	integer, corresponds to the inventory type system
license_number	variable length text field, license number of shipping entity
manifest_id	variable length text field, unique manifest identifier
trade_name	variable length text field, name of the shipping entity
transfer_date	Date of actual shipment
price	decimal value that indicates how much the item was sold for, originally.
quantity	decimal value
received	boolean value, indicates if the item was received.
quantity_received	decimal value, indicates the quantity accepted, if any.
stop_number	integer value
strain	variable length text field
usableweight	decimal value that represents usable weight of item, where applicable.
return_available	boolean value, indicates if the item has been specifically rejected by the intended recipient

inventory_transfer_outbound_return

The inventory_transfer_outbound_return function can be used to return items to inventory which were either partially or fully rejected by the recipient.

Parameters:

action	variable length text field
location	license number of location
data	Array of 1 or more nodes containing transportation information
barcodeid	unique identifier
item_number	integer, optional, that uniquely identifies each data element. If no item_number is provided, the system will provide one starting at 0
manifest_id	variable length text field, unique manifest identifier
price	decimal value, optional, if the transfer was partially accepted

Example:

```
<xml>
 <API>4.0</API> <action>inventory_transfer_outbound_return</action>
 <data>
  <barcode_id>MJ1234507285760184</barcode_id>
  <item_number>0</item_number>
  <manifest_id>1234567812345678</manifest_id>
   <price>50.00</price>
 </data>
 <data>
  <barcode_id>MJ1234507307763237</barcode_id>
  <item_number>1</item_number>
  <manifest_id>1234567812345678</manifest_id>
  <price>20.00</price>
 </data>
 <location>12345</location>
</xml>
```

Return example:

```
<xml>
 <data>
  <barcode_id> MJ1234507285760426</barcode_id>
```

```
   <item_number>0</item_number>
    <sub_lot>0</sub_lot>
  </data>
  <data>
   <barcode_id> MJ12345073077630 99</barcode_id>
   <item_number>1</item_number>
    <sub_lot>1</sub_lot>
  </data>
  <sessiontime>1429315773</sessiontime>
  <success>1</success>
  <transactionid>65349</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
sessiontime	Unix 32-bit integer timestamp
transactionid	integer value
data	Array of 1 or more nodes containing transportation information
barcode_id	Unique identifier
item_number	integer, that uniquely identifies each data element. If no item_number was provided upon submission, the system will provide this starting at 0
sub_lot	boolean value, indicates if the item was sub-lotted. An item would be sub-lotted if it were partially accepted. If a sub-lot is generated, the barcode_id will correspond to the new sub-lot. If not, the barcode_id corresponds to the original identifier.

inventory_transfer_outbound_modify

The inventory_transfer_outbound_modify function will allow a user to modify the price recorded for an inventory transfer sale.

Parameters:

action	variable length text field
transactionid	integer value
barcodeid	inventory identifier
price	Decimal value representing the price paid before any applicable taxes.

Example:

```
<xml>
 <API>4.0</API>
 <action>inventory_transfer_outbound_modify</action>
 <transactionid>3590</transactionid>
  <barcodeid>MJ12345072857601 84</barcodeid>
   <price>15.00</price>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

inventory_transfer_outbound_void

The inventory_transfer_outbound_void function will allow a user to void an inventory transfer that has been completed but not yet received by the recipient. This can be used for instances where a sale has been reported complete on the sender end; but was made in error. The transfer can then be made again; or the manifest voided, if necessary.

Parameters:

action	variable length text field
transactionid	integer value

Example:

```
<xml>
 <API>4.0</API>
 <action>inventory_transfer_outbound_void</action>
 <transactionid>5590</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

inventory_transfer_inbound

The inventory_transfer_inbound function can be used to officially receive inventory from another licensee.

Parameters:

action	variable length text field
location	license number of location
data	Array of 1 or more nodes containing inventory information
barcodeid	inventory identifier
quantity	Quantity or amount received
uom	variable length text field. Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.

```
<xml>
 <API>4.0</API>
 <action>inventory_transfer_inbound</action>
 <data>
   <barcodeid>M00342Q00137887615</barcodeid>
   <quantity>100.00</quantity>
   <uom>g</uom>
 </data>
</xml>
```

Return example:

```
<xml>
 <sessiontime>1384476925</sessiontime>
 <success>1</success>
 <transactionid>3778</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

inventory_transfer_inbound_modify

The inventory_transfer_inbound_modify function will allow a user to modify the refund price recorded for an inventory transfer sale that came into a licensed location.

Parameters:

action	variable length text field
transactionid	integer value
barcodeid	inventory identifier
price	Decimal value representing the refund price, if any, paid before any applicable taxes.

Example:

```
<xml>
 <API>4.0</API>
 <action>inventory_transfer_inbound_modify</action>
 <transactionid>3596</transactionid>
  <barcodeid>M00342Q00137887615</barcodeid>
   <price>15.00</price>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

inventory_create_lot

The inventory_create_lot function will allow a user to combine inventory types 6 (Flower) and 9 (Other Plant Material) into batches as mandated by rules. The return types will be 13 (Flower Lot) and 14 (Other Plant Material Lot), respectively. The system will implicitly calculate the new quantity based on what is removed from the original items. Type 30 (Cannabis Mix) can also be created using this function using a combination of flower and other material, as necessary.

Parameters:

action	variable length text field
lot_type	Optional, integer that can be either 13, 14 or 30. If not specified, the system will automatically assign 13 for

	flower, 14 for other material and 30 for submitted barcodes that contain a mix of both.
data	Array of 1 or more nodes containing inventory information
barcodeid	inventory identifier
remove_quantity	Decimal value, quantity to remove. Does not need to be remaining quantity (can be a partial combination).
remove_quantity_uom	variable length text field. Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.

```
<xml>
 <API>4.0</API>
 <action>inventory_create_lot</action>
 <data>
  <barcodeid>M00342Q00137887615</barcodeid>
  <remove_quantity>693.00</remove_quantity>
 </data>
 <data>
  <barcodeid> M00342Q00137887070</barcodeid>
  <remove_quantity>252.00</remove_quantity>
 </data>
</xml>
```

Return example:

```
<xml>
 <sessiontime>1384476925</sessiontime>
 <barcode_id> M00342Q00137887408</barcode_id>
 <barcode_type>13</barcode_type>
 <success>1</success>
 <transactionid>3312</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

sessiontime	Unix 32-bit integer timestamp
barcode_id	text field representing new unique identifier
barcode_type	integer representing new lot type

inventory_split

The inventory_split function will allow a user to split inventory items into sub lots or sub batches. For example, if a user has a lot of Flower and only wishes to sell half of it, they would need to first create a sub lot using this function. Then, with the new lot number, they can sell the desired amount. Multiple lots or batches can be specified at a time, however, keep in mind they will not be combined. Rather, each one will receive a new sub-lot or sub-batch number.

Parameters:

action	variable length text field
data	Array of 1 or more nodes containing inventory information
barcodeid	inventory identifier
remove_quantity	integer value, quantity to remove. Does not need to be remaining quantity (can be a partial combination).
remove_quantity_uom	variable length text field. Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.

```
<xml>
 <API>4.0</API>
 <action>inventory_split</action>
 <data>
  <barcodeid> M00342Q00137887019</barcodeid>
  <remove_quantity>693.00</remove_quantity>
 </data>
 <data>
  <barcodeid> M00342Q00137887615</barcodeid>
  <remove_quantity>252.00</remove_quantity>
 </data>
</xml>
```

Return example:

```
<xml>
```

```
  <sessiontime>1384476925</sessiontime>
  <barcode_id>1234567812345678</barcode_id>
  <success>1</success>
  <transactionid>3312</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
barcode_id	text fields representing new unique identifier, returned in the order of the input identifiers

inventory_convert

The inventory_convert function will allow a user to convert one type of item to another. The system allows for multiple sources. So, for example, a processor may use part of various Other Plant Material Lots in producing a batch of hash oil. Certain derivatives may not be strain specific, so entering a strain is optional under those circumstances. Product name is optional when it is not the end product. If the derivative item will be sold to a consumer (that is, inventory types 22,23,24,25) and is not regular usable Cannabis (type 28), then a product name will be required (e.g. Cookie, Brownie, etc).

Parameters:

action	variable length text field
data	Array of 1 or more nodes containing inventory information
barcodeid	inventory identifier
remove_quantity	decimal value, quantity to remove. Does not need to be remaining quantity (can be a partial combination).
remove_quantity_uom	variable length text field. Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.
waste	decimal value, amount of waste produced by the process, if any
waste_uom	Valid values are: g, mg, kg, oz, lb. These represent: grams, milligrams, kilograms, ounces, pounds.
derivative_type	Inventory type of derivative item
derivative_quantity	decimal value, quantity of new derivative after conversion
derivative_quantity_uom	Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.

derivative_usable	decimal value, quantity of usable Cannabis in new product after conversion
derivative_usable_uom	Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.
derivative_strain	Optional, variable length text field
derivative_product	Optional, variable length text field
net_package	Optional, decimal value that defined the net package weight or volume.
net_package_uom	Optional, defines net_package units. Valid values are: g, mg, kg, oz, lb, ml. These represent: grams, milligrams, kilograms, ounces, pounds, milliliters.
no_modification	Optional, boolean value. If the item being converted is eligible for QA bypass due to no physical change, this should be set to 1.

Example:

```
<xml>
 <API>4.0</API>
 <action>inventory_convert</action>
 <data>
    <barcodeid> M00342Q00137887615</barcodeid>
     <remove_quantity>25.00</remove_quantity>
  </data>
 <waste>15.00</waste>
 <derivative_quantity>10.00</derivative_quantity >
<derivative_inventory_type>18</derivative_inventory_type>
</xml>
```

Return example:

```
<xml>
<derivatives>
   <barcode_id> MJ1234507285760184</barcode_id>
   <barcode_type>28</barcode_type>
 </derivatives>
 <derivatives>
   <barcode_id> M00342Q00270897891</barcode_id>
   <barcode_type>27</barcode_type>
```

```
  </derivatives>
  <success>1</success>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
derivatives	Array of 1 or more nodes containing new identifiers with their associated inventory types.
barcode_id	New identifier for the inventory specified by barcode_type.
barcode_type	Specifies the type of derivative.

inventory_sample

The inventory_sample function will allow a user to provide samples as allowed by law. Specifically, samples can be provided to employees for quality assurance purposes or to vendors for the purposes of negotiating a sale. Either employee_id or vendor_license should be provided; but not both. For a new sample, an inventory ID will be returned for that sample. If this is a vendor sample, the sample must be sent with a manifest and the receiver must then acknowledge the sample with one of their employees.

Parameters:

action	variable length text field
barcodeid	inventory identifier
employee_id	Optional, variable length text field
vendor_license	Optional, variable length text field representing license number of receiving entity
quantity	decimal value, quantity of old product before conversion
quantity_uom	Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.
sample_type	Optional, integer indicated 1 (vendor sample) or 2 (employee sample).

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_sample</action>
  <barcodeid> MJ1234507285760184</barcodeid>
```

```
  <quantity>1.00</quantity>
  <employee_id>12356</employee_id >
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
barcode_id	Optional, new identifier if the call is referencing the creation of a new sample rather than the deduction of an existing one

inventory_qa_sample

The inventory_qa_sample function will allow a user to provide QA samples to qualified testing facilities as allowed by law.

Parameters:

action	variable length text field
barcodeid	inventory identifier
lab_id	variable length text field, license number of the QA facility
quantity	decimal value, quantity of old product before conversion
quantity_uom	Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.
use	Optional. If the inventory type is 13 (flower lot), this field should be 1 to indicate the lot will be used to convert to usable Cannabis (type 28, e.g. pre-packs), or 0 to indicate it will be used for an extract. Converting directly to type 28 will trigger more rigorous QA test requirements.

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_qa_sample</action>
  <barcodeid> M00342Q00137887615</barcodeid>
  <quantity>1.00</quantity>
  <lab_id>12356</lab_id >
```

```
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
sample_id	identifier that required for manifest transportation and other sample functions

inventory_qa_sample_void

The inventory_qa_sample_void function will void a sample that has been sent out (from the traceability system's perspective), but not tested yet.

Parameters:

action	variable length text field
transactionid	integer value

Example:

```
<xml>
 <API>4.0</API>
 <action>inventory_qa_sample_void</action>
 <transactionid>123456</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

inventory_qa_sample_results

The inventory_qa_sample_results function will allow a user or laboratory to provide QA results as allowed by law. As QA facilities will be reporting directly, most licensed facilities will not need to report the results themselves.

Parameters:

action	variable length text field
sample_id	sample identifier

test	Array of 1 or more nodes containing test details The parameters to expect for each test can be found in both the example and tables below.

Example:

```
<xml>
 <API>4.0</API>
 <action>inventory_qa_sample_results</action>
 <sample_id> M00342Q00137887614</sample_id>
<test>
   <moisture>5</moisture>
   <type>1</type>
 </test>
<test>
   <CBD>5</CBD>
   <CBDA>5</CBDA>
   <THC>20</THC>
   <THCA>1</THCA>
   <CBN>25.5</CBN>
   <Total>31</Total>
   <type>2</type>
 </test>
 <test>
   <Other>1</Other>
   <Stems>2</Stems>
   <type>3</type>
 </test>
 <test>
   <aerobic_bacteria>1000</aerobic_bacteria>
   <bile_tolerant>10000</bile_tolerant>
   <coliforms>10000</coliforms>
   <e_coli_and_salmonella>0</e_coli_and_salmonella>
   <type>4</type>
   <yeast_and_mold>2500</yeast_and_mold>
 </test>
```

```
<test>
   <type>5</type>
   <residual_solvent>0</residual_solvent >
  </test>
<test>
   <type>6</type>
   <total_mycotoxins>0</total_mycotoxins>
  </test>
<test>
   <type>7</type>
   <pesticide_residue>0</pesticide_residue>
  </test>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

QA Test Types

1	Moisture Content
2	Potency Analysis
3	Foreign Matter Inspection
4	Microbiological Screening
5	Residual Solvent
6	Mycotoxin Screening
7	Pesticide Residue

Moisture Content Details

Parameter	Details
moisture	Moisture Content, whole number only

Potency Analysis Details

Parameter	Details
THC	THC Content
THCA	THCA Content
CBD	CBD Content
CBDA	CBDA Content
CBN	CBN Content
Total	Total Cannabinoid Profile

Foreign Matter Types

Parameter	Details
Stems	Content of the aforementioned matter, as a percentage
Other	Content of the aforementioned matter, as a percentage

Microbial and Fungal Counts (Colony Forming Units [CFU]/g)

Parameter	Details
aerobic_bacteria	Total viable aerobic bacteria count
yeast_and_mold	Total yeast and mold count
coliforms	Total coliforms count
bile_tolerant	Bile-tolerant gram-negative bacteria
e_coli_and_salmonella	E. coli and Salmonella

Residual Solvent Details

Parameter	Details

residual_solvent	Residual Solvents

Mycotoxin Screening Details

Parameter	Details
total_mycotoxins	Total Mycotoxins

Pesticide Residue Details

Parameter	Details
pesticide_residue	Pesticide residue

inventory_qa_check

The inventory_qa_check function will pull down lab results that have been submitted to the traceability system by a certified QA lab.

Parameters:

action	variable length text field
sample_id	sample identifier

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_qa_check</action>
  <sample_id> M00342Q00137887614</sample_id>
</xml>
```

Returned Parameters:

success	Boolean value
result	integer value, -1 for failure, 1 for success
sessiontime	Unix 32-bit integer timestamp
test	Array of 1 or more nodes containing test details
use	integer value, sample use, 0 for standard test, 1 for test specifically for extract
inventorytype	Inventory type of the item

The parameters to expect for each test can be found in the tables above.

inventory_qa_check_all

The inventory_qa_check_all function will pull down lab results that have been submitted to the traceability system by a certified QA lab given the specific lot or batch numbers.

Parameters:

action	variable length text field
barcodeid	Array of one or more identifiers

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_qa_check_all</action>
  <barcodeid>M00342Q00137887615</barcodeid>
  <barcodeid>M00342Q00245663780</barcodeid>
</xml>
```

Returned Parameters:

success	Boolean value
sessiontime	Unix 32-bit integer timestamp
data	Array of 1 or more nodes containing inventory information
barcode_id	Unique identifier
result	integer value, -1 for failure, 1 for success
test	Array of 1 or more nodes containing test details
use	integer value, sample use, 0 for standard test, 1 for test specifically for extract
inventorytype	Inventory type of the item
parent_id	Unique parent identifier
sample_id	Sample identifier

The parameters to expect for each test can be found in the tables above.

inventory_modify

The inventory_modify function will allow a cultivator to modify the strain on inventory that can be used as a plant source (inventory types 7, 10, 11, 12) or inventory that was incorrectly classified but not yet grouped

(inventory types 6, 9, 27). The function may also be used by any privilege type to modify the product name. Both can be updated simultaneously; provided the cultivator privilege type is possessed by the licensee per the requirement for updating the strain.

Parameters:

action	variable length text field
barcodeid	barcode identifier
strain	Optional variable length text field
productname	Optional variable length text field
net_package	Optional, decimal value that defined the net package weight or volume.
net_package_uom	Optional, defines net_package units. Valid values are: g, mg, kg, oz, lb, ml. These represent: grams, milligrams, kilograms, ounces, pounds, milliliters.

Example:

```
<xml>
 <API>4.0</API>
 <action>inventory_modify</action>
 <barcodeid>M12345601106388383</barcodeid>
 <strain>Raspberry</strain>
</xml>
```

Return example:

```
<xml>
 <sessiontime>1384476925</sessiontime>
 <success>1</success>
 <transactionid>3278</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

inventory_convert_undo

The inventory_convert_undo function will allow a licensee to correct an inventory conversion where a mistake was made. This function can only be used when additional changes (e.g. adjustments) have not been made to the derivative item.

Parameters:

action	variable length text field
barcodeid	Array of 1 or more text fields representing the plants to undo

```
<xml>
 <API>4.0</API>
 <action>inventory_convert_undo</action>
 <barcodeid>MJ1234507285760184</barcodeid>
</xml>
```

Return example:

```
<xml>
 <sessiontime>1384476925</sessiontime>
 <success>1</success>
 <transactionid>3278</transactionid>
 <data>
  <barcodeid>M00342Q00185484887</barcodeid>
  <quantity>693.00</quantity>
 </data>
 <data>
  <barcodeid>M00342Q00137887615</barcodeid>
  <quantity>252.00</quantity>
 </data>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

data	Array of 1 or more nodes containing transportation information
barcode_id	Inventory identifier of parent
quantity	Decimal value, new parent quantity after success

Chapter 6: Sales

In this chapter, you'll learn how to:

- ✓ Perform a card lookup
- ✓ Deduct inventory for a sale
- ✓ Void a sale
- ✓ Refund a sale

card_lookup

The card_lookup function will allow a dispensary to validate a card holder's eligibility in the system. This function will return an ephemeral key that will be valid for 1 hour from issuance of the current time, or sale_time if specified and must be provided to the sale_dispense function for that specific patient.

Parameters:

action	variable length text field
card_id	variable length text field

Example:

```
<xml>
 <API>4.0</API>
 <action>card_lookup</action>
 <card_id>CM-PA-2016-0000</card_id>
</xml>
```

Return example:

```
<xml>
<card_key>dbf06eda1572c8e3c90d951ed9b5b14c32096570a7a902f8120a0b00
6b90c7df131ed2c8105ebbe5f9681f1537739e1969bf3ab684ca20864023a468305
09242</card_key>
 <success>1</success>
</xml>
```

Returned Parameters:

success	Boolean value

card_key	128 character hexadecimal key

sale_dispense

The sale_dispense function will allow a user to deduct items from inventory through the sales process. Since all items sold must be pre-packaged, units will be assumed to be "each". The card_key must be a valid ephemeral key for the associated card_id called within the previous hour.

Parameters:

action	variable length text field
card_key	128 character hexadecimal key as provided by the card_lookup function
card_id	variable length text field
data	Array of 1 or more nodes containing inventory information
barcodeid	inventory identifier
quantity	integer value, quantity to remove
price	Decimal value representing the price paid before any applicable taxes.
item_number	Optional, integer, should be provided if multiple line items of the same barcode were included in one sale. 0 would represent the first item (in the order submitted to the system), 1 the next, etc.
sale_time	Optional, unix 32-bit integer timestamp of when the sale occurred. If not used, will default to current time. Otherwise, the time must not be in the future.
terminal_id	Optional, user-defined text value (max 32 characters) that can be associated with a sale and retrieved at a later point with a synchronization call.

Example:

```
<xml>
 <API>4.0</API>
 <action>sale_dispense</action>
<card_key>dbf06eda1572c8e3c90d951ed9b5b14c32096570a7a902f8120a0b00
6b90c7df131ed2c8105ebbe5f9681f1537739e1969bf3ab684ca20864023a468305
09242</card_key>
 <card_id>CM-PA-2016-0000</card_id>
 <data>
```

```
   <barcodeid>MJ1234507285760184</barcodeid>
   <quantity>1.00</quantity>
   <price>5.00</price>
  </data>
 <data>
   <barcodeid>MJ1234507285760097</barcodeid>
   <quantity>1.00</quantity>
   <price>15.00</price>
  </data>
</xml>
```

Return example:

```
<xml>
 <sessiontime>1384476925</sessiontime>
 <success>1</success>
 <transactionid>3312</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
terminal_counter	Optional, integer value, if terminal_id is provided, that indicates the number of times the terminal ID provided has called the function.

sale_void

The sale_void function will reverse items that have been sold to a customer and return the items to inventory. A refund should be used, instead, when the return is not being used to simply fix a mistake.

Parameters:

action	variable length text field
transactionid	integer value

Example:

```
<xml>
 <API>4.0</API>
 <action>sale_void</action>
 <transactionid>3590</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

sale_modify

The sale_modify function will allow a user to modify the price recorded for a sale.

Parameters:

action	variable length text field
transactionid	integer value
barcodeid	inventory identifier
price	Decimal value representing the price paid before any applicable taxes.
item_number	Optional, integer, should be provided if multiple line items of the same barcode were included in one sale. 0 would represent the first item (in the order submitted to the system), 1 the next, etc.
sale_time	Optional, unix 32-bit integer timestamp of when the sale occurred. If not used, will default to current time. Otherwise, the time must not be in the future

Example:

```
<xml>
 <API>4.0</API>
 <action>sale_modify</action>
 <transactionid>3590</transactionid>
  <barcodeid>MJ12345072857601 84</barcodeid>
   <price>15.00</price>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

sale_refund

The sale_refund function is nearly identical to sale_dispense except that it for items to selectively come back into inventory from a sale. This can take place at any time period after the original sale and will reflect on current sales as opposed to affecting previously reported data. You must specify both a transactionid and one or more identifiers. Dispensarys are not currently allowed by rule to destroy product, so if an open item is received it must be scheduled for transfer back to the processor for destruction.

Parameters:

action	variable length text field
transactionid	integer value
data	Array of 1 or more nodes containing inventory information
barcodeid	inventory identifier
quantity	integer value, quantity to bring in.
price	Negative decimal value representing the price before any applicable taxes.
item_number	Optional, integer, should be provided if multiple line items of the same barcode were included in one sale. 0 would represent the first item (in the order submitted to the system), 1 the next, etc.
sale_time	Optional, unix 32-bit integer timestamp of when the sale occurred. If not used, will default to current time. Otherwise, the time must not be in the future.

Example:

```
<xml>
 <API>4.0</API>
 <action>sale_refund</action>
 <transactionid>3590</transactionid>
  <data>
  <barcodeid>MJ1234507285760184</barcodeid>
  <quantity>1.00</quantity>
  <price>-5.00</price>
 </data>
 <data>
  <barcodeid>MJ1234507285760047</barcodeid>
  <quantity>1.00</quantity>
  <price>-15.00</price>
 </data>
</xml>
```

Return example:

```
<xml>
 <sessiontime>1384476925</sessiontime>
 <success>1</success>
 <transactionid>3312</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp

Chapter 7: Synchronization

In this chapter, you'll learn how to:

- ✓ Replay a transaction's results.
- ✓ Download current plants, inventory, etc. stored in traceability system
- ✓ Receive notifications of inventory seizures, etc.
- ✓ Assist a licensee transition from the state interface to a commercial application

nonce_replay

The system allows for a nonce value to be embedded in any request in which data is being saved. This is a user-defined value that should be unique for every request. It is the integrator's responsibility, should they choose to utilize this functionality, to ensure this. Should the integrator re-use a token, and later request a replay of the results; the system will only return the last result for said token. For simplicity, a user may include a nonce value in non-transactional requests; they will be silently ignored. The system will only store data for which a transaction id is returned. Therefore, if the submitted data was non-transactional or produced an error, replay data would be unavailable and a request for said nonce would simply return a not found error.

To embed a nonce value, simply encode said value into a standard request. For example, one might call the inventory_new function as:

```
<xml>
  <API>4.0</API>
  <action>inventory_new</action>
  <data>
    <invtype>12</invtype>
    <quantity>50</quantity>
    <strain>Blueberry</strain>
  </data>
  <location>12345</location>
<nonce>2ebf8a5981651d7403a40a3a4f710551afab</nonce>
</xml>
```

The results of said (original) request would be returned, as usual. However, the results will now be accessible at any future point with the nonce_replay function.

To execute such a request:

Parameters:

action	variable length text field
nonce	variable length user-defined text field

```
<xml>
  <API>4.0</API>
  <action>nonce_replay</action>
<nonce>2ebf8a5981651d7403a40a3a4f710551afab</nonce>
</xml>
```

Returned Parameters:
Variable

An error will be returned if the nonce value can't be found. Otherwise, the successful results of the original request will be returned. If the specified nonce was not found; it is therefore safe to assume the data was not committed and may be submitted again.

The system will return the data in whatever format the ORIGINAL request was performed in. That is, if the original request was made with XML, and the nonce_replay was performed in JSON, the data from the replay will always be returned in XML (ensuring the replay is always exact and not otherwise re-filtered, processed, etc.).

This functionality is optional, but can be used in cases where a successful request is made but the response not received. For example, if an integrator makes a request, the request is received and acted upon but the end-client disconnects before receiving the response (e.g. loses internet connection); the end-client's system would be considered in an inconsistent state. The end-client would, of course, not want to simply call the function again (e.g. another call to inventory_new might then produce double the inventory). In this way, an integrator can essentially create a psuedo two-phase commit behind the scenes as demonstrated by the following (example) steps:

1. Client submits data.
2. Software integrator stores submitted data locally immediately before submission and toggles the data as incomplete.
3. Connection is established to server, the data is successfully received, but the connection is interrupted and software throws an error stating such before receiving return data.
4. Client attempts to re-submit data.
5. Software recognizes the user is attempting to submit a request that already exists but is toggled as incomplete.
6. Software attempts a replay, instead, with the associated nonce value.

7. Server returns data from original request, the software parses the result, toggles the original data submission as complete, and commits the data locally.
8. Client receives expected results from software (e.g. new barcodes) unaware and unaffected by issues at the lower layers.

sync_check

The sync_check function is the canonical function for synchronization. As indicated throughout this text; the system uses identification numbers for all transactional data received (via the transactionid). This function allows an integrator to determine if the summation of the transactions they have recorded what is currently stored within the traceability system. It can be used to either compare local value to remote values; or it can be used to simultaneously compare and download data that does not match. As these functions are comparing raw data tables the integrator should expect them returned as such.

The data tables can be queried on their own via a specific call directly without doing a summation check or through this function. The direct calls will be detailed later in the chapter.

The consistency check involves, at a minimum, providing a table. An integrator can also provide a start transaction (inclusive), an end transaction (inclusive), a sum value and whether or not only active data points are considered. More on this below.

There are currently 14 tables which can be queried: vehicle, employee, plant_room, inventory_room, inventory, plant, plant_derivative, manifest, inventory_transfer, sale, vendor, qa_lab, inventory_adjust and inventory_qa_sample.

Data Tables

vehicle

This contains vehicle information as previously submitted. It is UBI specific (as opposed to license specific) and can be queried with all records or only active ones.

employee

This contains employee information as previously submitted. It is UBI specific (as opposed to license specific) and can be queried with all records or only active ones.

plant_room

This contains plant room information as previously submitted. It is license specific and can be queried with all records or only active ones.

inventory_room

This contains inventory room information as previously submitted. It is license specific and can be queried with all records or only active ones.

inventory

This contains inventory information as previously submitted. It is license specific and can be queried with all records or only active ones. Active records are considered to be inventory that has not been moved into cultivation, zeroed or destroyed.

plant

This contains plant information as previously submitted. It is license specific and can be queried with all records or only active ones. Active records are considered to be plants that have not been destroyed or moved into inventory.

plant_derivative

This contains plant yield information as previously submitted. It is license specific and can be queried with all records or only active ones.

manifest

This contains manifest information as previously submitted. It is license specific and can be queried with all records or only active ones.

inventory_transfer

This contains inventory transfer information as previously submitted. It is license specific and can be queried with all records or only active ones.

inventory_transfer_inbound

This contains inbound inventory transfer information as previously received and submitted. It is license specific and can be queried with all records or only active ones.

sale

This contains end-customer sale information as previously submitted. It is license specific and can be queried with all records or only active ones.

vendor

This contains all active vendor information (sans phone numbers). This function will only return active entries.

qa_lab

This contains all active quality assurance lab information (sans phone numbers). This function will only return active entries.

inventory_adjust

This contains all inventory adjustment information. It is useful for retrieving historical data and should not be necessary in most scenarios. This function will only return active entries.

inventory_qa_sample

This contains basic quality assurance sample information as previously submitted. It is license specific and can be queried with all records or only active ones. As QA derived samples receive their own identifier; this list can be used to cross-reference said samples currently (or previously) in inventory.

inventory_sample

This contains all inventory samples that have been provided to either employees or to other vendors as samples for negotiation.

additive_type

This contains all additive types information as previously submitted including the three default additive types.

additive

This contains additive information as previously submitted.

plant_additive

This contains plant additive application information as previously submitted.

Parameters:

action	variable length text field
download	integer value of 0 or 1
data	Array of 1 or more nodes containing synchronization information
table	data table to be queried
transaction_start	Optional, minimum transactionid (inclusive) to compare sums with.
transaction_end	Optional, maximum transactionid (inclusive) to compare sums with.
sum	Optional, summation of transactionid values the client side possesses.
active	Optional, indicates only active records should be returned.

Example:

```
<xml>
 <API>4.0</API>
```

```
  <action>sync_check</action>
  <data>
    <table>vehicle</table>
    <transaction_start>0</transaction_start>
    <transaction_end>5</transaction_end>
    <sum>15</sum>
  </data>
</xml>
```

Returned Parameters:

success	Boolean value
summary	Array of 1 or more nodes containing match information.
match	Boolean value that indicates whether the sum was matched by the server.
sum	Integer value indicating the server sum. This value will be the same as provided if match is 1. This value will indicate the correct summation if match is 0.
table	The name of the table that was checked. As this function allows for multiple tables to be checked simultaneously; this will allow an integrator to identify the return values when more than one table is provided.

```
<xml>
<success>1</success>
<summary>
    <match>1</match>
    <sum>15</sum>
    <table>vehicle</table>
  </summary>
</xml>
```

The additional functions outlined in this chapter will provide examples for what an integrator can expect in terms of specific returned data when download is set to 1. The system will query all transactions available (as indicated by the active constraint) when start and end transaction values are not provided. The system will assume an integrator is simply querying for the sum if no sum is provided to check against and download is set to 0. If download is set to 1; the system will return all matching rows for the transaction range specified.

sync_vehicle

The sync_vehicle function will allow a user to synchronize vehicle data as previously submitted.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest
active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
 <API>4.0</API>
 <action>sync_vehicle</action>
</xml>
```

Returned Parameters:

success	Boolean value
vehicle	Array of 1 or more nodes that include all of the relevant data
nickname	Variable length text field that describes the nickname of the vehicle
color	Variable length text field that describes the color of the vehicle
make	Variable length text field that describes the make of the vehicle
model	Variable length text field that describes the model of the vehicle
plate	Variable length text field that describes the plate number of the vehicle
vin	Variable length text field that describes the VIN of the vehicle
vehicle_id	Integer, user provided, that uniquely identifies the vehicle
year	Integer that describes the year of the vehicle
deleted	Boolean (0/1) value that indicates whether or not the vehicle is active
transactionid	Integer, this is the last transactionid value applied to this vehicle. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this vehicle. This will not change with respect to modification, removal, etc.

Return example:

```
<xml>
  <vehicle>
    <color>Red</color>
    <deleted>1</deleted>
    <make>Ford</make>
    <model>Ranger</model>
    <nickname>Red Ford</nickname>
    <plate>23q3432</plate>
    <transactionid>4069</transactionid>
    <transactionid_original>4068</transactionid_original>
    <vehicle_id>28</vehicle_id>
    <vin>234342423</vin>
    <year>1983</year>
  </vehicle>
  <vehicle>
    <color>Black</color>
    <deleted>0</deleted>
    <make>Ford</make>
    <model>Mustang</model>
    <nickname>My Ford</nickname>
    <plate>123501</plate>
    <transactionid>4912</transactionid>
    <transactionid_original>4912</transactionid_original>
    <vehicle_id>28</vehicle_id>
    <vin>18384955</vin>
    <year>2000</year>
  </vehicle>
<success>1</success>
</xml>
```

sync_employee

The sync_employee function will allow a user to synchronize employee data as previously submitted.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest

transaction_end	Optional, integer that indicates the last transactionid of interest
active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
 <API>4.0</API>
 <action>sync_employee</action>
</xml>
```

Returned Parameters:

success	Boolean value
employee	Array of 1 or more nodes that include all of the relevant data
birthday	Integer that describes the birth day of the employee
birthmonth	Integer that describes the birth month of the employee
birthyear	Integer that describes the birth year of the employee
hireday	Integer that describes the hire day of the employee
hiremonth	Integer that describes the hire month of the employee
hireyear	Integer that describes the hire year of the employee
employee_id	Integer, user provided, that uniquely identifies the employee
employee_name	Name of the employee
deleted	Boolean (0/1) value that indicates whether or not the employee is active
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.

Return example:

```
<xml>
  <employee>
    <birthday>01</birthday>
    <birthmonth>02</birthmonth>
    <birthyear>1980</birthyear>
```

```
    <deleted>0</deleted>
    <employee_id>12384</employee_id>
    <employee_name>new Guy</employee_name>
    <hireday>23</hireday>
    <hiremonth>12</hiremonth>
    <hireyear>2013</hireyear>
    <transactionid>3570</transactionid>
    <transactionid_original>3570</transactionid_original>
  </employee>
  <employee>
    <birthday>01</birthday>
    <birthmonth>01</birthmonth>
    <birthyear>1980</birthyear>
    <deleted>0</deleted>
    <employee_id>123467</employee_id>
    <employee_name>Test</employee_name>
    <hireday>03</hireday>
    <hiremonth>03</hiremonth>
    <hireyear>2014</hireyear>
    <transactionid>3946</transactionid>
    <transactionid_original>3946</transactionid_original>
  </employee>
<success>1</success>
</xml>
```

sync_plant_room

The sync_plant_room function will allow a user to synchronize cultivation room data as previously submitted.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest
active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
 <API>4.0</API>
 <action>sync_plant_room</action>
</xml>
```

Returned Parameters:

success	Boolean value
plant_room	Array of 1 or more nodes that include all of the relevant data
roomid	Integer, user provided, that uniquely identifies the cultivation room
name	Variable length text field that identifies the name of the cultivation room
deleted	Boolean (0/1) value that indicates whether or not the cultivation room is active
location	license number of the location the room was created in
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.

Return example:

```
<xml>
 <plant_room>
  <deleted>0</deleted>
  <location>18750</location>
  <name>Default</name>
  <roomid>1</roomid>
  <transactionid>4075</transactionid>
  <transactionid_original>4070</transactionid_original>
 </plant_room>
 <plant_room>
  <deleted>0</deleted>
  <location>18750</location>
  <name>Clone Room 1</name>
  <roomid>7</roomid>
```

```
   <transactionid>4081</transactionid>
   <transactionid_original>4081</transactionid_original>
  </plant_room>
<success>1</success>
</xml>
```

sync_inventory_room

The sync_inventory_room function will allow a user to synchronize inventory room data as previously submitted.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest
active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
 <API>4.0</API>
 <action>sync_inventory_room</action>
</xml>
```

Returned Parameters:

success	Boolean value
inventory_room	Array of 1 or more nodes that include all of the relevant data
roomid	Integer, user provided, that uniquely identifies the inventory room
name	Variable length text field that identifies the name of the inventory room
deleted	Boolean (0/1) value that indicates whether or not the inventory room is active
quarantine	Boolean (0/1) value that indicates whether or not the inventory room has been designated as a quarantine room
location	license number of the location the room was created in

transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.

Return example:

```
<xml>
 <inventory_room>
  <deleted>0</deleted>
  <location>18750</location>
  <name>Quarantine</name>
  <quarantine>1</quarantine>
  <roomid>1</roomid>
  <transactionid>4032</transactionid>
  <transactionid_original>4032</transactionid_original>
 </inventory_room>
 <inventory_room>
  <deleted>0</deleted>
  <location>18750</location>
  <name>New</name>
  <quarantine>0</quarantine>
  <roomid>7</roomid>
  <transactionid>4057</transactionid>
  <transactionid_original>4057</transactionid_original>
 </inventory_room>
<success>1</success>
</xml>
```

sync_inventory

The sync_inventory function will allow a user to synchronize inventory data as previously submitted.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest

active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
 <API>4.0</API>
 <action>sync_inventory</action>
</xml>
```

Returned Parameters:

success	Boolean value
inventory	Array of 1 or more nodes that include all of the relevant data
currentroom	Integer, user provided, that uniquely identifies the inventory room the item is currently in. Can be null to indicate the Bulk Inventory room.
deleted	Boolean (0/1) value that indicates whether or not the inventory item still exists.
id	barcode identifier
inventoryparentid	Array of 1 or more 18 digit identifiers that identify the identifier(s) this item is descended from with respect to QA testing eligibility. That is, if a lot (eligible for testing) is sublotted many times over, this will always be the original lot number.
inventorystatus	Integer, status identifier of the inventory. Can be null, 1 (scheduled for destruction), 2 (scheduled for transport) or 3 (in-transport but not yet received).
inventorystatustime	Unix 32-bit integer timestamp of when the non-null status was added.
inventorytype	Inventory type of the item
location	license number of the location the inventory currently exists in.
parentid	Array of 1 or more 18 digit direct inventory parent identifiers. If an item is sublotted, this would be the inventory id the item was sublotted from.
plantid	Array of 1 or more 18 digit plant identifiers. When an item is harvested and placed into inventory (e.g. inventory type 6), this will indicate the plant(s) the item was harvested from.
productname	Variable length text field
remaining_quantity	Decimal value, quantity currently available
seized	If the item has been seized, this field will indicate 1.

sessiontime	Unix 32-bit integer timestamp of when the item was inserted.
source_id	18 digit identifier of the mother plant, if the item was created directly from one (e.g. clone).
strain	Variable length text field
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.
usable_weight	Decimal value that, for non-weighable inventory types (e.g. usable Cannabis type 28), will indicate the pre-package value. For any weighable types (e.g. type 13 flower lot), this field will indicate the original quantity of the item when created.
wet	If the item was collected during harvest and not cure (e.g. other material type 9), it will be considered wet and can be adjusted for reason type 5.
net_package	Decimal value that defines the net package weight or volume.
is_sample	Optional integer value, 1 if the inventory item is a sample intended for a vendor.

Return example:

```
<xml>
 <inventory>
  <currentroom></currentroom>
  <deleted>0</deleted>
  <id>MJ1234507285760184</id>
  <inventoryparentid></inventoryparentid>
  <inventorystatus></inventorystatus>
  <inventorystatustime></inventorystatustime>
  <inventorytype>6</inventorytype>
  <location>18750</location>
  <parentid></parentid>
  <plantid>MJ1234507285760014</plantid>
  <productname></productname>
  <remaining_quantity>250.00</remaining_quantity>
  <seized></seized>
  <sessiontime>1405844163</sessiontime>
  <source_id></source_id>
```

```
   <strain>Blueberry</strain>
   <transactionid>4861</transactionid>
   <transactionid_original>4861</transactionid_original>
   <usable_weight>250.00</usable_weight>
   <wet>0</wet>
  <net_package>100.00</net_package>
 </inventory>
 <inventory>
   <currentroom></currentroom>
   <deleted>0</deleted>
   <id>M11191500174470167</id>
   <inventoryparentid>M00342Q00137887615</inventoryparentid>
   <inventorystatus></inventorystatus>
   <inventorystatustime></inventorystatustime>
   <inventorytype>13</inventorytype>
   <location>18750</location>
   <parentid>M00342Q00137887615</parentid>
   <plantid></plantid>
   <productname></productname>
   <remaining_quantity>139.00</remaining_quantity>
   <seized></seized>
   <sessiontime>1405844196</sessiontime>
   <source_id></source_id>
   <strain>Blueberry</strain>
   <transactionid>4862</transactionid>
   <transactionid_original>4862</transactionid_original>
   <usable_weight>240</usable_weight>
   <wet>0</wet>
   <net_package>50.00</net_package>
  </inventory>
<success>1</success>
</xml>
```


sync_plant

The sync_plant function will allow a user to synchronize plant data as previously submitted.

Parameters:

Action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest

transaction_end	Optional, integer that indicates the last transactionid of interest
active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
 <API>4.0</API>
 <action>sync_plant</action>
</xml>
```

Returned Parameters:

success	Boolean value
plant	Array of 1 or more nodes that include all of the relevant data
converted	Boolean (0/1) value that indicates whether or not the plant was converted to a sellable clone (and thus removed from cultivation for that reason)
harvestcollect	The number of times the plant has been harvested. Null indicates it has not been harvested yet.
curecollect	The number of times the plant has been cured. Null indicates it has not been cured yet.
deleted	Boolean (0/1) value that indicates whether or not the plant still exists.
id	18 digit barcode identifier
harvestscheduled	Boolean (0/1) value that indicates whether or not the plant has been scheduled for harvest.
harvestscheduletime	If the plant has been scheduled for harvest, this field will indicate the unix 32-bit integer timestamp of when the item was scheduled for harvest (there is currently no waiting period; so it will be the time the notification was sent across).
location	license number of the location the plant is located in.
mother	Boolean (0/1) value that indicates whether or not the plant is tagged as a mother plant.
parentid	18 digit identifier that was the inventory source for the current plant
removereason	Variable length text field that will be non-null if the plant has been scheduled for destruction.

removescheduled	Boolean (0/1) value that indicates whether or not the plant has been scheduled for destruction.
removescheduletime	If the plant has been scheduled for destruction, this field will indicate the unix 32-bit integer timestamp of when the item is eligible, at a minimum, for actual destruction (after a destruction has been scheduled, this will be 7 days from that time).
room	Integer value, user provided, of the room the plant is currently in. If the plant has been destroyed or harvested, it will represent the last room it occupied.
seized	If the item has been seized, this field will indicate 1.
sessiontime	Unix 32-bit integer timestamp of the birth date of the plant.
state	Integer value occupying either a 0 (currently growing), 1 (currently drying) or 2 (fully cured and no longer in the cultivation area). Plants with state 2 will not be displayed in a request only interested in active plants.
strain	Variable length text field
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.

Return example:

```
<xml>
 <plant>
  <converted>0</converted>
  <curecollect></curecollect>
  <deleted>0</deleted>
  <deletetime></deletetime>
  <harvestcollect></harvestcollect>
  <harvestscheduled>0</harvestscheduled>
  <harvestscheduletime></harvestscheduletime>
  <id>M00342Q00003990761</id>
  <location>18750</location>
  <mother>0</mother>
  <parentid>M12345601106388383</parentid>
  <removereason></removereason>
```

```
    <removescheduled>0</removescheduled>
    <removescheduletime></removescheduletime>
    <room>64</room>
    <seized></seized>
    <sessiontime>1405464324</sessiontime>
    <state>0</state>
    <strain>AK-47</strain>
    <transactionid>4815</transactionid>
    <transactionid_original>4815</transactionid_original>
  </plant>
</xml>
```

sync_plant_derivative

The sync_plant_derivative function will allow a user to synchronize plant derivative data (wet and dry weights) as previously submitted.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest
active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
  <API>4.0</API>
  <action>sync_plant_derivative</action>
</xml>
```

Returned Parameters:

success	Boolean value
plant_derivative	Array of 1 or more nodes that include all of the relevant data
harvestcollect	Will be set to 1 if this collection occurred during a harvest (wet) point.

curecollect	Will be set to 1 if this collection occurred during a cure (dry) point.
deleted	Boolean (0/1) value that indicates whether or not the derivative still exists.
plantid	18 digit barcode identifier of the plant
location	license number of the location the derivative was collected in.
inventorytype	Inventory type of the derivative item.
weight	Decimal value of the weight recorded for the specific inventory type.
wholeweight	If a plant was harvested/cured as a group, this would indicate the overall weight of the group being collected (whereas the weight field will indicate the individual weight). If a harvest is done on an individual basis, this will be the same as weight.
room	Integer value, user provided, of the room the plant was in when the action occured.
inventoryid	18 digit identifier of the derivative
sessiontime	Unix 32-bit integer timestamp of the collection time of the plant.
collectadditional	Boolean (0/1) value that indicates whether or not the collection point was requested with additional collection points (re-flowering).
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.

Return example:

```
<xml>
 <plant_derivative>
  <collectadditional>0</collectadditional>
  <curecollect></curecollect>
  <deleted>0</deleted>
  <harvestcollect></harvestcollect>
  <inventoryid>M00342Q00066590176</inventoryid>
  <inventorytype>6</inventorytype>
  <location>18750</location>
  <plantid>M00342Q00006357962</plantid>
```

```
    <room>5</room>
    <sessiontime>1405844163</sessiontime>
    <transactionid>4861</transactionid>
    <transactionid_original>4861</transactionid_original>
    <weight>250</weight>
    <wholeweight>250.00</wholeweight>
  </plant_derivative>
  <success>1</success>
</xml>
```

sync_manifest

The sync_manifest function will allow a user to synchronize manifest data as previously submitted.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest
active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
  <API>4.0</API>
  <action>sync_manifest</action>
</xml>
```

Returned Parameters:

success	Boolean value
manifest	Array of 1 or more nodes that include the high level relevant data of submitted manifests
manifestid	16 digit manifest identifier
sessiontime	Unix 32-bit integer timestamp of the time the manifest was filed.

completion_date	Unix 32-bit integer timestamp of the time the manifest was filed (duplicate of sessiontime kept for backward compatibility).
stopcount	Integer value indicating the number of stops on the manifest.
deleted	Boolean (0/1) value that indicates whether or not the manifest has been voided.
location	license number of the location the manifest was filed from
origination_city	Variable length text field of the city the manifest originates from
origination_license_number	Variable length text field of the license number the manifest originates from
origination_name	Variable length text field of the name of the licensee the manifest originates from
origination_phone	Variable length text field of the phone number of the licensee the manifest originates from
origination_state	Variable length text field of the state the manifest originates from
origination_street	Variable length text field of the street the manifest originates from
origination_zip	Variable length text field of the zip the manifest originates from
total_item_count	Integer value indicating the number of items on the manifest.
transporter_dob	Variable length text field of the birthdate of the employee transporting the product.
transporter_id	Integer, identification number of the employee transporting the product.
transporter_name	Variable length text field of the name of the employee transporting the product.
transporter_vehicle_details	Variable length text field of the vehicle transporting the product.
transporter_vehicle_identification	Variable length text field of the VIN.
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.

manifest_type	Integer, 0 for regular manifests 1 for pick-u manifests.
manifest_fully_completed	Integer, 0 for manifests that have had an employee/driver added, 1 for manifests that are fully ready to ship.
manifest_stop_data	Array of 1 or more nodes that include the stop level data of submitted manifests
sessiontime	Unix 32-bit integer timestamp of the time the manifest was filed.
manifestid	16 digit manifest identifier
arrive_time	Unix 32-bit integer timestamp of the approximate time the items are expected to arrive at their destination
city	Variable length text field that indicates the city of the stop destination
depart_time	Unix 32-bit integer timestamp of the approximate departure time
item_count	Integer value indicating the number of items for the specified stop.
license_number	License number of specific stop destination.
location	license number of the location the manifest was filed from
name	Variable length text field that indicates the name of the stop destination
phone	Variable length text field that indicates the phone of the stop destination
state	Variable length text field that indicates the state of the stop destination
street	Variable length text field that indicates the street of the stop destination
zip	Variable length text field that indicates the zip of the stop destination
travel_route	Variable length text field that indicates the route of travel as filed.
stopnumber	Integer value indicating the stop number on the manifest.
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.
deleted	Boolean (0/1) value that indicates whether or not the stop has been voided

manifest_stop_items	Array of 1 or more nodes that include the item level data of submitted manifests
sessiontime	Unix 32-bit integer timestamp of the time the manifest was filed.
description	Variable length text field that describes the item being transported.
manifestid	16 digit manifest identifier
inventoryid	18 digit barcode identifier of the item being transported.
quantity	Decimal value indicating the number of units of the specified item.
location	license number of the location the manifest was filed from
stopnumber	Integer value indicating the stop number on the manifest.
requiresweighing	(Deprecated) Integer value indicating if the item is a weighable item. This field remains for backward-compatibility. An integrator should rely on the inventory type for determining whether or not an inventory item requires weighing.
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.
deleted	Boolean (0/1) value that indicates whether or not the item has been voided

Return example:

```
<xml>
 <manifest>
  <completion_date>1389796859</completion_date>
  <deleted>0</deleted>
  <fulfilled>1</fulfilled>
  <location>18750</location>
  <manifestid>3692253654269107</manifestid>
  <origination_city>Fargo</origination_city>
```

```
    <origination_license_number>189</origination_license_number>
    <origination_name>Trade 24</origination_name>
    <origination_phone>222-333-4444</origination_phone>
    <origination_state>ND</origination_state>
    <origination_street>2135 Address Way</origination_street>
    <origination_zip>55555</origination_zip>
    <sessiontime>1389192059</sessiontime>
    <stopcount>1</stopcount>
    <total_item_count>1</total_item_count>
    <transactionid>9821</transactionid>
    <transactionid_original>9821</transactionid_original>
    <transporter_dob>01/01/1980</transporter_dob>
    <transporter_id>23486</transporter_id>
    <transporter_name>New Employee</transporter_name>
    <transporter_vehicle_details>Black Chevy Cavalier
23856</transporter_vehicle_details>
<transporter_vehicle_identification>32495954656</transporter_vehicle_identification>
  </manifest>
  <manifest_stop_data>
    <arrive_time>1389886803</arrive_time>
    <city> Fargo </city>
    <depart_time>1389885003</depart_time>
    <item_count>1</item_count>
    <license_number>11111</license_number>
    <location>18750</location>
    <manifestid>3692253654269107</manifestid>
    <name>Some Retail Location</name>
    <phone>444-555-6666</phone>
    <sessiontime>1389796859</sessiontime>
    <state>ND</state>
    <stopnumber>1</stopnumber>
    <street>22993 New Road Way</street>
    <transactionid>9821</transactionid>
    <transactionid_original>9821</transactionid_original>
    <travel_route>Head southwest.</travel_route>
    <zip>98295</zip>
    <deleted>0</deleted>
  </manifest_stop_data>
  <manifest_stop_items>
    <description>Usable Cannabis</description>
```

```
    <inventoryid> MJ1234507285760184 </inventoryid>
    <location>18750</location>
    <manifestid>3692253654269107</manifestid>
    <quantity>15.00</quantity>
    <requiresweighing></requiresweighing>
    <sessiontime>1389796859</sessiontime>
    <stopnumber>1</stopnumber>
    <transactionid>9821</transactionid>
    <transactionid_original>9821</transactionid_original>
    <deleted>0</deleted>
  </manifest_stop_items>
<success>1</success>
</xml>
```

sync_inventory_transfer

The sync_inventory_transfer function will allow a user to synchronize inventory transfer data as previously submitted.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest
active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
  <API>4.0</API>
  <action>sync_inventory_transfer</action>
</xml>
```

Returned Parameters:

success	Boolean value
inventory_transfer	Array of 1 or more nodes that include all of the relevant data

deleted	Boolean (0/1) value that indicates whether or not the transfer has been voided.
inventoryid	18 digit barcode identifier of the item being transported.
inventorytype	Inventory type of the item.
is_refund	Boolean (0/1) value that indicates whether or not the transfer is a refund.
manifestid	16 digit manifest identifier attached to the transfer.
manifest_stop	Stop number on the manifest.
sessiontime	Unix 32-bit integer timestamp of the time the transfer was initiated.
location	license number of the location the transfer was initiated from.
outbound_license	license number of the location the transfer was initiated from.
price	Decimal value indicating the total dollar amount received for the line item.
quantity	Decimal value indicating the total quantity of the item shipped.
strain	Variable length text field
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.

Return example:

```
<xml>
 <inventory_transfer>
  <deleted>1</deleted>
  <inventoryid> MJ1234507285760184 </inventoryid>
  <inventorytype>28</inventorytype>
  <location>18750</location>
  <manifest_stop>1</manifest_stop>
  <manifestid>3387557157087693</manifestid>
  <outbound_license>18750</outbound_license>
  <price>1000.00</price>
  <quantity>50</quantity>
  <sessiontime>1405844437</sessiontime>
  <strain>Blueberry</strain>
```

```
    <transactionid>4918</transactionid>
    <transactionid_original>4918</transactionid_original>
  </inventory_transfer>
  <success>1</success>
</xml>
```

sync_inventory_transfer_inbound

The sync_inventory_transfer_inbound function will allow a user to synchronize inbound inventory transfer data that as previously received and submitted.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest
active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
  <API>4.0</API>
  <action>sync_inventory_transfer_inbound</action>
</xml>
```

Returned Parameters:

success	Boolean value
inventory_transfer_inbound	Array of 1 or more nodes that include all of the relevant data
deleted	Boolean (0/1) value that indicates whether or not the transfer has been voided.
inventoryid	18 digit barcode identifier of the item being transported.
inventorytype	Inventory type of the item.
is_refund	Boolean (0/1) value that indicates whether or not the transfer is a refund.
manifestid	16 digit manifest identifier attached to the transfer.
manifest_stop	Stop number on the manifest.
sessiontime	Unix 32-bit integer timestamp of the time the transfer was received.

location	license number of the location the transfer was received to.
outbound_license	license number of the location the transfer was transferred from.
price	Decimal value indicating the total dollar amount transferred out for the line item.
quantity	Decimal value indicating the total quantity of the item received.
refund_amount	Decimal value indicating the total dollar amount of the refund, if the line item is a refund.
strain	Variable length text field
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from the inbound transfer of this line item. This will not change with respect to modification, removal, etc.

Return example:

```
<xml>
 <inventory_transfer_inbound>
  <deleted>1</deleted>
  <inventoryid> MJ1234507285760184 </inventoryid>
  <inventorytype>28</inventorytype>
  <is_refund>1</is_refund>
  <location>18750</location>
  <manifest_stop>1</manifest_stop>
  <manifestid>3387557157087693</manifestid>
  <outbound_license>18751</outbound_license>
  <price>0.00</price>
  <quantity>50</quantity>
  <refund_amount>50.00</refund_amount>
  <sessiontime>1405844437</sessiontime>
  <strain>Blueberry</strain>
  <transactionid>4919</transactionid>
  <transactionid_original>4918</transactionid_original>
 </inventory_transfer_inbound>
 <success>1</success>
</xml>
```

sync_sale

The sync_sale function will allow a user to synchronize sale data as previously submitted.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest
active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
 <API>4.0</API>
 <action>sync_sale</action>
</xml>
```

Returned Parameters:

success	Boolean value
sale	Array of 1 or more nodes that include all of the relevant data
deleted	Boolean (0/1) value that indicates whether or not the sale has been voided.
inventoryid	18 digit barcode identifier of the item being sold.
itemnumber	Item number, as provided by the integrator, that uniquely identifies the line item for the specific sale.
sessiontime	Unix 32-bit integer timestamp of the time the sale was performed.
location	license number of the location the sale was initiated from.
price	Decimal value indicating the total dollar amount received for the line item.
quantity	Decimal value indicating the total quantity of the item sold.
refunded	Indicates if the item has been refunded. Can be null or set to 1 if it was refunded.
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.

transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.
terminal_id	User-defined text field, when provided.
inventorytype	Inventory type of the sale.

Return example:

```
<xml>
 <sale>
  <deleted>0</deleted>
  <inventoryid> MJ1234507285760184 </inventoryid>
  <itemnumber>0</itemnumber>
  <location>18750</location>
  <price>8.00</price>
  <quantity>1</quantity>
  <refunded></refunded>
  <sessiontime>1405830081</sessiontime>
  <transactionid>4857</transactionid>
  <transactionid_original>4857</transactionid_original>
  <terminal_id>1</terminal_id>
 </sale>
 <success>1</success>
</xml>
```

sync_inventory_adjust

The sync_inventory_adjust function will allow a user to synchronize inventory adjustment report data as previously submitted.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest

Example:

```
<xml>
```

```
  <API>4.0</API>
  <action>sync_inventory_adjust</action>
</xml>
```

Returned Parameters:

success	Boolean value
inventory_adjust	Array of 1 or more nodes that include all of the relevant data
inventoryid	18 digit barcode identifier of the item that was adjusted.
atype	Integer that describes the type of adjustment, as indicated in the inventory_adjust function.
sessiontime	Unix 32-bit integer timestamp of the time the adjustment was performed.
location	license number of the location the adjustment was initiated from.
new_quantity	Decimal value indicating the new quantity of the item.
previous_quantity	Decimal value indicating the previous quantity of the item.
reason	Variable length text field that describes the reason for adjustment as provided by the user.
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.

Return example:

```
<xml>
  <inventory_adjust>
    <atype>4</atype>
    <inventoryid> MJ1234507285760184 </inventoryid>
    <location>18750</location>
    <new_quantity>8.00</new_quantity>
    <previous_quantity>10</previous_quantity>
    <reason>Testing</reason>
    <sessiontime>1405829973</sessiontime>
    <transactionid>4856</transactionid>
    <transactionid_original>4856</transactionid_original>
  </inventory_adjust>
  <success>1</success>
```

</xml>

sync_inventory_qa_sample

The sync_inventory_qa_sample function will allow a user to synchronize inventory quality assurance samples as previously submitted.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest
active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
 <API>4.0</API>
 <action>sync_inventory_qa_sample</action>
</xml>
```

Returned Parameters:

success	Boolean value
inventory_qa_sample	Array of 1 or more nodes that include all of the relevant data
deleted	Boolean (0/1) value that indicates whether or not the sample has been voided.
inventoryid	16 digit barcode identifier of the unique sample.
parentid	16 digit barcode identifier of the batch or lot the sample was taken from.
inventorytype	Inventory type of the item the sample was taken from.
lab_license	Integer, license number of the QA laboratory the sample will be sent to.
sessiontime	Unix 32-bit integer timestamp of the time the sample was taken.
location	license number of the location the sample was initiated from.
quantity	Decimal value indicating the quantity of the sample.
result	Integer value that represents the result of the sample. Valid values can be -1 (fail), 0 (untested), 1 (success).
sample_use	The intended use of the sample, as indicated by the inventory_qa_sample function.

strain	Variable length text field.
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.

Return example:

```
<xml>
 <inventory_qa_sample>
  <deleted>0</deleted>
  <inventoryid> MJ1234507135760184 </inventoryid>
  <inventorytype>13</inventorytype>
  <lab_license>123456</lab_license>
  <location>18750</location>
  <parentid> M00342Q00137887615 </parentid>
  <quantity>1.00</quantity>
  <result>1</result>
  <sample_use>1</sample_use>
  <sessiontime>1405844232</sessiontime>
  <strain>Blueberry</strain>
  <transactionid>4863</transactionid>
  <transactionid_original>4863</transactionid_original>
 </inventory_qa_sample>
 <success>1</success>
</xml>
```

sync_inventory_sample

The sync_inventory_sample function will allow a user to synchronize inventory samples provided to employees or as samples for negotiation, as previously submitted.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest

active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
 <API>4.0</API>
 <action>sync_inventory_sample</action>
</xml>
```

Returned Parameters:

success	Boolean value
inventory_sample	Array of 1 or more nodes that include all of the relevant data
employee_id	Employee license number, if the sample is an employee sample.
inventoryid	18 digit barcode identifier of the unique sample.
sessiontime	Unix 32-bit integer timestamp of the time the sample was taken.
location	license number of the location the sample was initiated from.
quantity	Decimal value indicating the quantity of the sample.
sample_type	Integer value that indicates the type of sample. Valid values can be 1 (external vendor) or 2 (employee).
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.
vendor_license	Vendor license number of the sample, if the sample was provided to a vendor.
sample_inventoryid	Inventory ID of the new sample.

Return example:

```
<xml>
 <inventory_sample>
  <inventoryid> M00342Q00137887615 </inventoryid>
  <vendor_license>123456</vendor_license>
  <employee_id ></employee_id >
  <location>18750</location>
  <quantity>1.00</quantity>
  <sample_type>1</sample_type>
```

```
    <sessiontime>1405844232</sessiontime>
    <transactionid>4863</transactionid>
    <transactionid_original>4863</transactionid_original>
  </inventory_sample>
  <success>1</success>
</xml>
```

sync_vendor

The sync_vendor function will allow a user to synchronize official vendor data.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest

Example:

```
<xml>
  <API>4.0</API>
  <action>sync_vendor</action>
</xml>
```

Returned Parameters:

success	Boolean value
vendor	Array of 1 or more nodes that include all of the relevant data
city	Variable length text field of the vendor's city.
location	Variable length text field of the vendor's license number.
name	Variable length text field of the vendor's name.
state	Variable length text field of the vendor's state.
address1	Variable length text field of the vendor's address.
address2	Variable length text field of the vendor's address continued.
zip	Variable length text field of the vendor's zip.
ubi	Variable length text field of the vendor's UBI.
producer	Boolean (0/1) value that indicates whether or not the vendor possesses the producer license type.

processor	Boolean (0/1) value that indicates whether or not the vendor possesses the processor license type.
retail	Boolean (0/1) value that indicates whether or not the vendor possesses the retail license type.
locationtype	Integer that indicates a combination value that describes the privilege types the vendor possesses as follows: 1 (Cultivator Tier 1), 2 (Cultivator Tier 2), 3 (Cultivator Tier 3), 4 (Cultivator Tier 1 + Processor), 5 (Cultivator Tier 2 + Processor), 6 (Cultivator Tier 3 + Processor), 7 (Processor only), 8 (Dispensary)
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.

Return example:

```
<xml>
 <success>1</success>
 <vendor>
  <address1>1274 Address Way</address1>
  <address2></address2>
  <city>Fargo</city>
  <location>111112</location>
  <locationtype>8</locationtype>
  <name>New Retail Store</name>
  <processor>0</processor>
  <producer>0</producer>
  <retail>1</retail>
  <state>ND</state>
  <transactionid>4898</transactionid>
  <transactionid_original>4898</transactionid_original>
  <ubi>000000009</ubi>
  <zip>555550000</zip>
 </vendor>
</xml>
```

sync_qa_lab

The sync_qa_lab function will allow a user to synchronize official QA labs.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest

Example:

```
<xml>
 <API>4.0</API>
 <action>sync_qa_lab</action>
</xml>
```

Returned Parameters:

success	Boolean value
qa_lab	Array of 1 or more nodes that include all of the relevant data
city	Variable length text field of the QA lab's city.
location	Variable length text field of the QA lab's license number.
name	Variable length text field of the QA lab's name.
state	Variable length text field of the QA lab's state.
address1	Variable length text field of the QA lab's address.
address2	Variable length text field of the QA lab's address continued.
zip	Variable length text field of the QA lab's zip.
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.

Return example:

```
<xml>
 <qa_lab>
  <address1>1234 Address Way</address1>
```

```
   <address2></address2>
   <city>Fargo</city>
   <location>55555</location>
   <name>QTest1</name>
   <state>ND</state>
   <transactionid>4924</transactionid>
   <transactionid_original>4924</transactionid_original>
   <zip>89101</zip>
  </qa_lab>
  <success>1</success>
</xml>
```

sync_additive_type

The sync_additive_type function will allow a user to synchronize official additive type data.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest

Example:

```
<xml>
  <API>4.0</API>
  <action>sync_additive_type</action>
</xml>
```

Returned Parameters:

success	Boolean value
id	Integer, uniquely identifies a type
name	Variable length text field
no_modify	Boolean (0/1) value that indicates whether the type is modifiable.
deleted	Boolean (0/1) value that indicates whether the type is active
default_unit	Integer, reference table for values

transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.
sessiontime	Unix 32-bit integer timestamp of the time the type was created.

Return example:

```
<xml>
 <additive_type>
  <name>Herbicide</name>
  <id>4</id>
  <no_modify>0</no_modify>
  <deleted>0</deleted>
  <defaut_unit>6</defaut_unit>
  <sessiontime>1531179485</sessiontime>
  <transactionid>2410</transactionid>
  <transactionid_original>2410</transactionid_original>
 </additive_type>
 <success>1</success>
</xml>
```

sync_additive

The sync_additive function will allow a user to synchronize official additive data.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest
active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
 <API>4.0</API>
 <action>sync_additive</action>
```

</xml>

Returned Parameters:

success	Boolean value
id	Integer, uniquely identifies additive
name	Variable length text field
additive_type	Integer, id of additive_type
manufacturer	Variable length text field identifying manufacturer of product.
lot_number	Variable length text field
expiration_text	Variable length text field, MM/DD/YYYY format.
expiration_epoch	Unix 32-bit integer timestamp of the expiration date
initial_amount	Numeric quantity representing starting quantity of additive
remaining_amount	Numeric quantity representing ending quantity of additive post applications
details	Variable length text field
unit_of_measurement	Variable length text field. Valid values are: g, mg, kg, oz, lb. These represent: grams, milligrams, kilograms, ounces and pounds.
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.
sessiontime	Unix 32-bit integer timestamp of the time the sample was taken.
location	license number of the location the additive was created
default_unit	Integer, reference default_unit table for values
initial_quantity_default_unit	Integer, reference default_unit table for values
initial_quantity_ml	Numeric, reference default_unit table for values
remaining_quantity_ml	Numeric, reference default_unit table for values
initial_quantity_mg	Numeric, reference default_unit table for values
remaining_quantity_mg	Numeric, reference default_unit table for values
weight_based	0 or 1 Boolean value indicating if the additive is a solid of liquid
custom_mix	Integer

Return example:

```
<xml>
 <additive>
  <id>4</id>
```

```
    <name>Hydro Liquid</name>
    <additive_type>3</additive_type>
    <manufacturer>The Greene Team Inc.</manufacturer>
    <lot_number>11573-2CD</lot_number>
    <expiration_text>08/16/2020</expiration_text>
    <expiration_epoch> 1597536000 </expiration_epoch>
    <initial_amount>500</initial_amount>
    <remaining_amount>500</remaining_amount>
    <details>Use wisely</details>
    <unit_of_measurement>ml</unit_of_measurement>
    <sessiontime>1531179485</sessiontime>
    <transactionid>2410</transactionid>
    <transactionid_original>2410</transactionid_original>
    <location>123456</location>
    <default_unit>8</default_unit>
    <initial_quantity_default_unit>1</initial_quantity_default_unit>
    <initial_quantity_ml>500</initial_quantity_ml>
    <remaining_quantity_ml>500</remaining_quantity_ml>
    <initial_quantity_mg></initial_quantity_mg>
    <remaining_quantity_mg></remaining_quantity_mg>
    <weight_based>0</weight_based>
    <custom_mix></custom_mix>
  </additive>
  <success>1</success>
</xml>
```

sync_plant_additive

The sync_plant_additive function will allow a user to synchronize official plant additive data.

Parameters:

action	variable length text field
transaction_start	Optional, integer that indicates the first transactionid of interest
transaction_end	Optional, integer that indicates the last transactionid of interest
active	Optional, boolean value that indicates whether or not to only return non-deleted records

Example:

```
<xml>
 <API>4.0</API>
 <action>sync_plant_additive</action>
</xml>
```

Returned Parameters:

success	Boolean value
additive	Integer, id of additive applied
plantid	Barcode identifier of plant
location	License number of the location the additive was applied in
transactionid	Integer, this is the last transactionid value applied to this line item. This is updated upon every successful modification.
transactionid_original	Integer, this is the first transactionid value received from creation of this line item. This will not change with respect to modification, removal, etc.
sessiontime	Unix 32-bit integer timestamp of the time the type was created.
default_unit	Integer, reference table of values
quantity	numeric value of quantity
deleted	Boolean (0/1) value that indicates whether the type is active
quantity_mg	numeric value equal to quantity based on default_unit
quantity_ml	numeric value equal to quantity based on default_unit

Return example:

```
<xml>
 <plant_additive>
  <additive>6</additive >
  <plantid> 6710706103781440 </plantid>
  <location>123456</location>
  <deleted>0</deleted>
  <defaut_unit>6</defaut_unit>
  <sessiontime>1531179485</sessiontime>
  <transactionid>2410</transactionid>
  <transactionid_original>2410</transactionid_original>
  <quantity_mg>0.50</quantity_mg>
  <quantity_ml></quantity_ml>
 </plant_additive >
 <success>1</success>
</xml>
```