Volume

3

BIO-TECH MEDICAL SOFTWARE, INC.

BioTrackTHC XML API

Washington State
Liquor Control Board

BioTrackTHC
1-800-779-4094

BioTrackTHC API

For questions regarding this API,
please call 1-800-779-4094 or email waquestions@biotrackthc.com

BIO-TECH MEDICAL SOFTWARE, INC.

BioTrackTHC XML API

Fort Lauderdale, FL
Phone 800.779.4094
waquestions@biotrackthc.com

Washington State Liquor Control Board

BioTrackTHC

Table of Contents

Prefix: About This Document

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. This document is being released to the public in draft form ahead of schedule to expedite the integration process for commercial entities that intend to serve the producer, processor and retail establishments within the state of Washington.

Please note: There WILL be changes to this document. This may include paring down of existing structures or additions to the specification based on legal requirements.

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.

That being said, we look forward to working with the industry to finalize and solidify the world's first official marijuana traceability API. For questions regarding this API, please call 1-800-779-4094 or email waquestions@biotrackthc.com.

Chapter
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 (URL to follow) of the type text/xml. The result will also be of text/xml type.

login

When registering with the WSLCB, 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	variable length text field

```
<xml>
  <API>4.0</API>
  <action>login</action>
  <password>foobar</password>
  <license_number>123456789</license_number>
  <username>username@domain.com</username>
</xml>
```

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

```
<xml>
  <admin>1</admin>
<sessionid>qXs2iECVlWXoy7erZ6e1pMNZJ8+JqrlN/kdWCfDXyh
YLK0opQHox93NA3pQpNymIx4CnPeOVKBpWw28AYsL1Kw
</sessionid>
  <time>1384323370</time>
  <success>1</success>
</xml>
```

Returned Parameters:

admin	Boolean value
sessionid	sha512 base64 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>123456789</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>qXs2iECVlWXoy7erZ6e1pMNZJ8+JqrlN/kdWCfDXyh
YLK0opQHox93NA3pQpNymIx4CnPeOVKBpWw28AYsL1Kw
</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. In addition, it will also carry an <errorcode>1234</errorcode> for reference. This document does not **currently** have a detailed list of error codes. That will be forthcoming in a future draft for ease of debugging efforts. 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>
<plant_remove_schedule>1</plant_remove_schedule>
<plant_remove>1</plant_remove>
<plant_remove_schedule_undo>1
</plant_remove_schedule_undo>
<plant_remove_undo>1</plant_remove_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_derivative_weigh>1</plant_derivative_weigh>
<plant_derivative_weigh_undo>1
</plant_derivative_weigh_undo>
<plant_new>1</plant_new>
<plant_new_undo>1</plant_new_undo>
<plant_convert_to_clone>1</plant_convert_to_clone>
<plant_convert_to_clone_undo>1
</plant_convert_to_clone_undo>
<plant_derivative_collect>1</plant_derivative_collect>
<plant_derivative_collect_undo>1
</plant_derivative_collect_undo>
<plant_cure>1</plant_cure>
<plant_cure_undo>1</plant_cure_undo>
<plant_move>1</plant_move>
<plant_location_move>1</plant_location_move>
<plant_yield_modify>1</plant_yield_modify>
<plant_additive_apply>1</plant_additive_apply>
<plant_additive_apply_undo>1</plant_additive_apply_und
o>
<inventory_new>1</inventory_new>
<inventory_new_undo>1</inventory_new_undo>
<inventory_transfer>1</inventory_transfer>
```

```
<inventory_transfer_undo>1</inventory_transfer_undo>
<inventory_audit>1</inventory_audit>
<inventory_adjust>1</inventory_adjust>
<inventory_remove_schedule>1
</inventory_remove_schedule>
<inventory_remove_schedule_undo>1
</inventory_remove_schedule_undo>
<inventory_convert>1</inventory_convert>
<inventory_convert_undo>1</inventory_convert_undo>
<inventory_combine>1</inventory_combine>
<inventory_combine_undo>1</inventory_combine_undo>
<inventory_check>1</inventory_check>
<inventory_remove>1</inventory_remove>
<inventory_move>1</inventory_move>
<inventory_remove_undo>1</inventory_remove_undo>
<inventory_transfer_schedule>1</inventory_transfer_sched
ule>
<inventory_transfer_schedule_undo>1</inventory_transfer
_schedule_undo>
<user_add>1</user_add>
<user_modify>1</user_modify>
<user_remove>1</user_remove>
<location_add>1</location_add>
<location_modify>1</location_modify>
<location_remove>1</location_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>
</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

Chapter 2: Locations

In this chapter, you'll learn how to:

✓ **Add, modify and remove locations**

location_add

Every organization can be divided into discrete locations, each with their own set of inventory, rooms, etc. This can facilitate real separation (e.g. a different address) or even a different part of a building, if necessary. An organization can exist with only one location and, in many instances, can leave this value null when requested to indicate such.

Parameters:

action	variable length text field
name	variable length text field
address1	variable length text field
address2	variable length text field
city	variable length text field
state	variable length text field
zip	variable length text field
phone	variable length text field
license	variable length text field
medical	Boolean value
id	integer value that uniquely identifies the location for future requests

```
<xml>
  <API>4.0</API>
  <action>location_add</action>
  <name>Default Location</name>
  <address1>1234 Address Way</address1>
  <address2></address2>
  <city>Seattle</city>
  <state>WA</state>
  <zip>98101</zip>
  <phone>253-555-5555</phone>
  <license>12345678</license>
  <medical>0</medical>
  <id>1</id>
</xml>
```

Returned Parameters:

success	Boolean value

location_modify

This function should be used to update an existing location.

Parameters:

action	variable length text field
name	variable length text field
address1	variable length text field
address2	variable length text field
city	variable length text field
state	variable length text field
zip	variable length text field
phone	variable length text field
license	variable length text field
medical	Boolean value
id	integer value that uniquely identifies the location for future requests

```
<xml>
  <API>4.0</API>
  <action>location_modify</action>
  <name>Default Location</name>
  <address1>1234 Address Way</address1>
  <address2></address2>
  <city>Seattle</city>
  <state>WA</state>
  <zip>98101</zip>
  <phone>253-555-5555</phone>
  <license>12345678</license>
  <medical>0</medical>
  <id>1</id>
</xml>
```

Returned Parameters:

success	Boolean value

location_remove

This function should be used to remove a location. If a location is accidentally deleted, a simple call to location_modify can restore it, if necessary.

Parameters:

action	variable length text field
id	integer value that uniquely identifies the location

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

Returned Parameters:

success	Boolean 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	integer value
id	integer value

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

Returned Parameters:

success	Boolean 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	integer value
id	integer value

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

Returned Parameters:

success	Boolean value

plant_room_remove

Plant rooms can be removed with this function.

Parameters:

action	variable length text field
location	integer value
id	integer value

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

Returned Parameters:

success	Boolean value

inventory_room_add

Inventory rooms represent a way to logically segregate inventory in a specific location. These can include e.g. placing some inventory in a safe or on the shelf. 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.

Parameters:

action	variable length text field
name	variable length text field
location	integer value

id	integer value

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

Returned Parameters:

success	Boolean 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	integer value
id	integer value

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

Returned Parameters:

success	Boolean value

inventory_room_remove

Inventory rooms can be removed with this function.

Parameters:

action	variable length text field
location	integer value
id	integer value

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

Returned Parameters:

success	Boolean value

Chapter 4: Plants

In this chapter, you'll learn how to:

- ✓ Add and remove plants
- ✓ Harvest and cure plants
- ✓ Collect plant derivatives (e.g. shake, kief, etc.)
- ✓ Apply additives, pesticides, etc.
- ✓ …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, strain type, quantity, location, new room, whether from seed (0 will indicate clone) and parent identification number (in the case of a clone this would be the mother plant and in the case of a seed this would be the identification number attached to their seeds in inventory).

Parameters:

action	variable length text field
strain	variable length text field
strain_type	variable length text field
location	integer value
room	integer value
parentid	text field
quantity	integer value
group	optional, can create a logical grouping of the new plants

```
<xml>
  <API>4.0</API>
  <action>plant_new</action>
  <group>-1</group>
  <parentid>2288954595338316</parentid>
  <quantity>2</quantity>
  <room>1</room>
  <seed>0</seed>
  <strain>Blueberry</strain>
  <strain_type>Indica</strain_type>
</xml>
```

Return example:

```
<xml>
  <barcode_id>6853296789574115</barcode_id>
  <barcode_id>6853296789574116</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 not only for audit purposes but also serve the purpose of fixing simple mistakes that are made in the course of normal system use. Most submission methods support an "undo" method, as well, for such instances. Under more complex circumstances, as will be seen further in the chapter, there are methods available for direct modification of submitted data. However, even in those instances, the transaction id is needed. In other words, caveat lector: Do not lose your transaction id.

plant_new_undo

The plant_new_undo function will allow a cultivator to remove plants that were accidentally added incorrectly without penalizing them with respect to a destruction event. This function, however, will only work for plants that are accidentally added. In other words, if a user adds a batch of plants and then applies nutrients, pesticides, etc and then attempts an undo; this will be denied. Undo functions are built-in with safeguards to prevent abuse.

Parameters:

action	variable length text field
transactionid	integer value

```
<xml>
  <API>4.0</API>
  <action>plant_new_undo</action>
  <transactionid>3278</transactionid>
</xml>
```

Return example:

```
<xml>
  <sessiontime>1384476955</sessiontime>
  <success>1</success>
  <transactionid>3279</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>
  <barcodeid>6853296789574115</barcodeid>
  <barcodeid>6853296789574116</barcodeid>
  <room>2</room>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

plant_location_move

The plant_location_move function will allow a cultivator to move plants from one location to another.

Parameters:

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

```
<xml>
  <API>4.0</API>
  <action>plant_location_move</action>
  <barcodeid>6853296789574115</barcodeid>
  <barcodeid>6853296789574116</barcodeid>
  <location>2</location>
  <room>5</room>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

plant_additive_apply

The plant_additive_apply function will allow a cultivator to apply additives, pesticides, etc. to a plant or set of plants.

Parameters:

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

applied_quantity	indicates the total amount of the additive applied
applied_quantity_uom	variable length text field
concentration	indicates the concentration of additive
concentration_uom	variable length text field

```
<xml>
  <API>4.0</API>
  <action>plant_additive_apply</action>
  <barcodeid>6853296789574115</barcodeid>
  <barcodeid>6853296789574116</barcodeid>
  <applied_quantity>1</applied_quantity>
  <applied_quantity_uom>liter</applied_quantity_uom>
  <concentration>0.05</concentration>
  <concentration_uom>µg/L</concentration_uom>
  <additive>Pestacide #2</additive>
  <additive_time>1384476985</additive_time>
</xml>
```

Returned Parameters:

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

plant_additive_apply_undo

The plant_additive_apply_undo function will revert an additive that has been applied to a plant or set of plants.

Parameters:

action	variable length text field
transactionid	integer value

```
<xml>
  <API>4.0</API>
  <action>plant_additive_apply_undo</action>
  <transactionid>3279</transactionid>
</xml>
```

Returned Parameters:

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

plant_remove_schedule

The plant_remove_schedule function will allow a licensee to schedule for destruction a plant or set of plants. This event will begin a 72-hour waiting period before a plant_remove function may be called on the plant(s).

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_remove_schedule</action>
  <barcodeid>6853296789574115</barcodeid>
  <barcodeid>6853296789574116</barcodeid>
  <reason>Mold</reason>
</xml>
```

Returned Parameters:

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

plant_remove_schedule_undo

The plant_remove_schedule_undo function will reverse a plant or set of plants that have been scheduled for removal but have not been removed yet.

Parameters:

action	variable length text field
transactionid	integer value

```
<xml>
  <API>4.0</API>
  <action>plant_remove_schedule_undo</action>
  <transactionid>3279</transactionid>
</xml>
```

Returned Parameters:

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

plant_remove

The plant_remove function will allow a licensee to destroy (remove) a plant or set of plants. Plants may only be removed after the waiting period has expired.

Parameters:

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

```
<xml>
  <API>4.0</API>
  <action>plant_remove</action>
  <barcodeid>6853296789574115</barcodeid>
  <barcodeid>6853296789574116</barcodeid>
</xml>
```

Returned Parameters:

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

plant_remove_undo

The plant_remove_undo function will reverse a plant or set of plants that have been scheduled for removal but have not been removed yet.

Parameters:

action	variable length text field
transactionid	integer value

```
<xml>
  <API>4.0</API>
  <action>plant_remove_undo</action>
  <transactionid>3279</transactionid>
</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>
  <barcodeid>6853296789574115</barcodeid>
  <barcodeid>6853296789574116</barcodeid>
</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 reverse a plant or set of plants that have been scheduled for harvest but have not been harvested yet.

Parameters:

action	variable length text field
transactionid	integer value

```
<xml>
  <API>4.0</API>
  <action>plant_harvest_schedule_undo</action>
  <transactionid>3280</transactionid>
</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 or set of plants. This will move said plants from the "growing" phase to the "drying" phase. During this process, a cultivator must take, at a minimum, a wet weight of the plant(s). In addition, a cultivator may also gather additional derivatives defined by their inventory type. There may be additional inventory types added later.

Inventory Types

Inventory Types	
0	Vegetation Trim
1	Trim
2	Stems
3	Sugar (Sweet) Leaf
4	Shake
5	Kief

6	Flower
7	Clone (for sale)
8	Fan Leaf
9	Other (Root ball, etc.)
10	Seed

The traceability system also supports the concept of delayed collection. This allows for cultivators to submit data in a multitude of ways. Plant weights can be taken individually or in batches. Individual weights can be taken and collected at a later point. 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.

Parameters:

action	variable length text field
collectiontime	Optional, Unix 32-bit integer timestamp, defaults to current time
barcodeid	Array of 1 or more text fields representing the plants
weights	Array of 1 or more nodes containing weight information
amount	decimal value
collected	integer value of either 0, 1 or 2. 0 represents that the user will collect (batch) the item later. 1 will batch now and issue a new identifier to the derivative and place it in inventory. 2 will discard the derivative as waste.
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.
room	integer, room the collection occurred in
location	integer, location the collection occurred in

Example:

```
<xml>
 <API>4.0</API>
 <action>plant_harvest</action>
 <barcodeid>9318094993507695</barcodeid>
 <barcodeid>9330604318166731</barcodeid>
 <barcodeid>9992776458335982</barcodeid>
 <collectadditional>0</collectadditional>
 <location>1</location>
 <room>2</room>
 <new_room>3</new_room>
 <weights>
  <amount>250.00</amount>
  <collected>1</collected>
  <invtype>1</invtype>
  <uom>g</uom>
 </weights>
 <weights>
  <amount>500.00</amount>
  <collected></collected>
  <invtype>6</invtype>
  <uom>g</uom>
 </weights>
 <weights>
  <amount>125.00</amount>
  <collected>0</collected>
```

```
    <invtype>2</invtype>
    <uom>g</uom>
  </weights>
</xml>
```

Returns:

```
<xml>
  <derivatives>
    <barcode_id>0358560579655604</barcode_id>
    <barcode_type>1</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. These will be generated for any derivatives that have set collected to 1.
barcode_id	New identifier for the inventory specified by barcode_type.
barcode_type	Specifies the type of derivative.

The collected value for input type 6 (Flower) can be null; it is discarded. If derivatives are set to batch later (collected is set to 0), they must be accounted for at a later point (see the function plant_derivative_account_for). The flexibility to batch later can be a time saver for cultivators who, for example, collect stems from every plant for the day and only wish to issue one batch identifier from the entire lot as opposed to each batch for the day.

Some cultivators will find that they don't necessarily take weights from individual plants or even individual batches but from the entire collected amount for a time period. For example, a kief collector below a processing area might take a while to fill from various batches during the day and only weighed after a certain amount has been collected. In this scenario, a cultivator might benefit more from the plant_derivative_weigh function which allows for derivative collect from any number of plants, regardless of state (so long as they have not been removed).

plant_harvest_undo

The plant_harvest_undo function will reverse a harvest process as long as additional actions have not been taken against the plant(s) that were processed within the selected plant_harvest. In other words, if said plants have not been processed through the plant_cure function yet and any derivatives that were entered have not been transferred, sold, etc. the plant_harvest_undo function can be called. If a mistake is caught much later and a simple plant_harvest_undo function can no longer be called, the user will want to consider instead calling the plant_yield_modify function. That function allows direct modification of entered values based on the inventory type and transactionid.

Parameters:

action	variable length text field
transactionid	integer value

```
<xml>
  <API>4.0</API>
  <action>plant_harvest_undo</action>
  <transactionid>3284</transactionid>
</xml>
```

Returned Parameters:

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

plant_derivative_weigh

The plant_derivative_weigh function will allow a cultivator to weigh and account for derivatives on plants without changing the state of the plant. This can be useful in a variety of instances and lends flexibility to cultivators so that the traceability system can accept their input in a manner most efficient for their business logic.

The inputs and return values for this function are similar to the plant_harvest function with a couple exceptions. Derivatives of type 6 (Flower) cannot be processed in this manner. Any weights taken of type 6 will be ignored; they must be taken through the due course of the harvest and cure process. The other exceptions include that the new_room parameter does not exist and collectadditional is also irrelevant and, thus, not used.

Parameters:

action	variable length text field
collectiontime	Optional, Unix 32-bit integer timestamp, defaults to current time
barcodeid	Array of 1 or more text fields representing the plants
weights	Array of 1 or more nodes containing weight information
amount	decimal value
collected	integer value of either 0, 1 or 2. 0 represents that the user will collect (batch) the item later. 1 will batch now and issue a new identifier to the derivative and place it in inventory. 2 will discard the derivative as waste.
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.
room	integer, room the collection occurred in
location	integer, location the collection occurred in

Example:

```
<xml>
  <API>4.0</API>
  <action>plant_derivative_weigh</action>
  <barcodeid>9318094993507695</barcodeid>
  <barcodeid>9330604318166731</barcodeid>
  <barcodeid>9992776458335982</barcodeid>
  <location>1</location>
  <room>2</room>
  <weights>
    <amount>250.00</amount>
    <collected>1</collected>
    <invtype>1</invtype>
    <uom>g</uom>
  </weights>
  <weights>
    <amount>125.00</amount>
    <collected>0</collected>
    <invtype>2</invtype>
    <uom>g</uom>
  </weights>
</xml>
```

Returns:

```
<xml>
  <derivatives>
    <barcode_id>0358560579655604</barcode_id>
    <barcode_type>1</barcode_type>
  </derivatives>
  <sessiontime>1384487873</sessiontime>
  <success>1</success>
  <transactionid>3286</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. These will be generated for any derivatives that have set collected to 1.

barcode_id	New identifier for the inventory specified by barcode_type.
barcode_type	Specifies the type of derivative.

Much like the plant_harvest function, any derivatives that have set collected to 0 must be accounted for at a later point.

plant_derivative_weigh_undo

The plant_derivative_weigh_undo function will reverse an ad-hoc derivative collection. If any of the collected derivatives have been transferred, sold, etc. they will need to instead be modified through the plant_yield_modify function. That function allows direct modification of entered values based on the inventory type and transactionid.

Parameters:

action	variable length text field
transactionid	integer value

```
<xml>
  <API>4.0</API>
  <action>plant_derivative_weigh _undo</action>
  <transactionid>3286</transactionid>
</xml>
```

Returned Parameters:

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

plant_cure

The plant_cure function will begin the process of curing a plant or set of plants. This will move said plants from the drying phase to inventory. During this process, a cultivator must take, at a minimum, a dry weight of the plant(s). In addition, a cultivator may also gather additional derivatives defined by their inventory type.

The inventory type 6 (Flower) can be batched now or batched later at this point. It cannot be discarded through this function. If batched later, it will need to be accounted for at a later point.

If the cultivator is doing a partial harvest/cure, the plant(s) can pass through this function again to accumulate additional dry weight(s). 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 1 or more text fields representing the plants
weights	Array of 1 or more nodes containing weight information
amount	decimal value
collected	integer value of either 0, 1 or 2. 0 represents that the user will collect (batch) the item later. 1 will batch now and issue a new identifier to the derivative and place it in inventory. 2 will discard the derivative as waste.
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 dry weight of the plant(s) at a later point that will compound to the original dry weight.
room	integer, room the collection occurred in
location	integer, location the collection occurred in

Example:
```
<xml>
  <API>4.0</API>
  <action>plant_harvest</action>
  <barcodeid>9318094993507695</barcodeid>
  <barcodeid>9330604318166731</barcodeid>
  <barcodeid>9992776458335982</barcodeid>
  <collectadditional>0</collectadditional>
  <location>1</location>
  <room>2</room>
  <weights>
    <amount>250.00</amount>
    <collected>1</collected>
    <invtype>1</invtype>
    <uom>g</uom>
  </weights>
  <weights>
    <amount>500.00</amount>
    <collected>1</collected>
    <invtype>6</invtype>
    <uom>g</uom>
  </weights>
  <weights>
    <amount>125.00</amount>
    <collected>0</collected>
    <invtype>2</invtype>
    <uom>g</uom>
  </weights>
</xml>
```

Returns:

```
<xml>
  <derivatives>
    <barcode_id>0358560579655604</barcode_id>
    <barcode_type>1</barcode_type>
  </derivatives>
  <derivatives>
    <barcode_id>0358560579655605</barcode_id>
    <barcode_type>6</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. These will be generated for any derivatives that have set collected to 1.
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 reverse a cure process as long as additional actions have not been taken against the plant(s) that were processed within the selected plant_cure. In other words, if said or derivatives from said plants have not been transferred, sold, etc. the plant_cure_undo function can be called. If a mistake is caught much later and a simple plant_cure_undo function can no longer be called, the user will want to consider instead calling the plant_yield_modify function. That function allows direct modification of entered values based on the inventory type and transactionid.

Parameters:

action	variable length text field
transactionid	integer value

```
<xml>
  <API>4.0</API>
  <action>plant_cure_undo</action>
  <transactionid>3290</transactionid>
</xml>
```

Returned Parameters:

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

plant_derivative_account_for

The plant_derivative_account_for function will allow a cultivator to accounted for derivatives that were previously batched later.

Parameters:

action	variable length text field
collectiontime	Optional, Unix 32-bit integer timestamp, defaults to current time
transactionid	Array of 1 or more integer fields representing the transactions to collect from
invtype	integer representing the inventory type being collected
collect	integer value, either 1 or 2. 1 will batch the item, 2 will discard it as waste
quantity	decimal value representing the weight of the resultant collection. This may be less than the sum of the original values due to moisture loss, etc.
quantity_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_derivative_account_for</action>
  <transactionid>3290</transactionid>
  <transactionid>3291</transactionid>
  <invtype>2</invtype>
  <collect>1</collect>
  <quantity>120.00</quantity>
  <quantity_uom>g</quantity_uom>
</xml>
```

Returns:

```
<xml>
  <derivatives>
    <barcode_id>0358560579655608</barcode_id>
    <barcode_type>2</barcode_type>
  </derivatives>
  <sessiontime>1384487873</sessiontime>
  <success>1</success>
  <transactionid>3301</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. These will be generated for any derivatives that have set collected to 1.
barcode_id	New identifier for the inventory specified by barcode_type.
barcode_type	Specifies the type of derivative.

Any items discarded as waste will not, of course, receive a new unique identifier.

plant_derivative_account_for_undo

The plant_derivative_account_for_undo function will reverse items that have been accounted for. If the items in question have already been transferred, sold, etc. the cultivator will need to, instead, call the plant_yield_modify function. That function allows direct modification of entered values based on the inventory type and transactionid.

Parameters:

action	variable length text field
transactionid	integer value

```
<xml>
  <API>4.0</API>
  <action>plant_cure_undo</action>
  <transactionid>3290</transactionid>
</xml>
```

Returned Parameters:

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

plant_convert_to_clone

The plant_convert_to_clone function will allow a licensee to convert a plant that is growing 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 7 (clone).

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_clone</action>
  <barcodeid>6853296789574125</barcodeid>
  <barcodeid>6853296789574126</barcodeid>
</xml>
```

Returned Parameters:

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

plant_convert_to_clone_undo

The plant_convert_to_clone_undo function will reverse a plant or set of plants that have been converted to inventory clones. This undo function can take either an individual identifier, set of identifiers or a transactionid (to process all items within the convert transaction).

Parameters:

action	variable length text field
transactionid	Optional if barcodeid is specified, integer value
barcodeid	Optional if transactionid is specified, integer value

```
<xml>
  <API>4.0</API>
  <action>plant_convert_to_clone_undo</action>
  <transactionid>3298</transactionid>
</xml>
```

Returned Parameters:

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

Specifying a specific identifier as opposed to a specific transactionid can be useful if multiple items were processed in one transaction but, for example, one of the items in the transaction has already been transferred, sold, etc.

plant_yield_modify

The plant_yield_modify function will allow direct access to modify previously stored values for harvest, cure or separate derivative collections. The user will need to specify only 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.

The collection values can be changed through this function as well as values. If an item was previously collected as 2 (discarded), and should be changed to 1 (batch now) or 0 (batch later), amount can be left null and the user can simply provide a different collect value.

Use of this function on a regular basis is strongly discouraged and highly circumspect. Most simple mistakes should be correctable through the use of undo functions.

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
collected	Optional, integer value of either 0, 1 or 2. 0 represents that the user will collect (batch) the item later. 1 will batch now and issue a new identifier to the derivative and place it in inventory. 2 will discard the derivative as waste.
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

Chapter 5

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.

Parameters:

action	variable length text field
barcodeid	inventory identifier
quantity	integer value, new quantity
uom	variable length text field. Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each. If weighable, grams are assumed if omitted. If non-weighable, each is assumed.
reason	reason for the removal or addition of inventory
theft	Boolean value, indicates if the adjustment is due to theft
health	Boolean value, indicates if the adjustment is due to health concerns

roomdata	Optional, array of 1 or more nodes containing room allocation information
room	Optional, integer value, represents the identification number of a room
qty	Optional, integer value, represents the quantity currently in the associated room

```
<xml>
  <API>4.0</API>
  <action>inventory_adjust</action>
  <barcodeid>6647455983218747</barcodeid>
  <quantity>690</quantity>
  <reason>Testing</reason>
</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

If an item is to be zeroed out, and it is not due to theft, a user should call the inventory_remove function instead. This also carries with it, however, the need to call the inventory_remove_schedule function which carries with it a holding period.

inventory_audit

The inventory_audit function will allow a licensee to review multiple items at once through the course of regular auditing of their inventory. This function shouldn't be used if items need to be removed due to health concerns or theft; those parameters are not accepted for this function.

Parameters:

action	variable length text field
data	Array of 1 or more nodes containing inventory information
barcodeid	inventory identifier
quantity	integer value, new quantity
uom	variable length text field. Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.
reason	reason for the removal or addition of inventory
roomdata	Optional, array of 1 or more nodes containing room allocation information
room	Optional, integer value, represents the identification number of a room
qty	Optional, decimal value, represents the quantity currently in the associated room

```
<xml>
  <API>4.0</API>
  <action>inventory_audit</action>
  <data>
    <barcodeid>7480211204033809</barcodeid>
    <quantity>100.00</quantity>
  </data>
  <data>
    <barcodeid>1002205938403155</barcodeid>
    <quantity>95.00</quantity>
  </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

inventory_remove_schedule

The inventory_remove_schedule function will notify the traceability system of intent to remove an inventory item. This function will usually be called in the instance of a health issue with an inventory item.

Parameters:

action	variable length text field
barcodeid	Array of 1 or more text fields representing the plants
reason	reason for the removal or addition of inventory
health	Boolean value, indicates if the adjustment is due to health concerns

```
<xml>
  <API>4.0</API>
  <action>inventory_remove_schedule</action>
  <barcodeid>6853296789574115</barcodeid>
  <barcodeid>6853296789574116</barcodeid>
  <reason>Mold</reason>
  <health>1</health>
</xml>
```

Returned Parameters:

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

inventory_remove_schedule_undo

The inventory_remove_schedule_undo function will reverse an inventory item that has been scheduled for removal.

Parameters:

action	variable length text field
transactionid	integer value

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_remove_schedule_undo</action>
  <transactionid>3350</transactionid>
</xml>
```

Returned Parameters:

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

inventory_remove

The inventory_remove function will allow a licensee to remove an item that has been previously quarantined and scheduled for removal.

Parameters:

action	variable length text field
barcodeid	inventory identifier
reason	reason for the removal or addition of inventory
health	Boolean value, indicates if the removal is due to health concerns

```
<xml>
  <API>4.0</API>
  <action>inventory_remove</action>
  <barcodeid>6647455983218747</barcodeid>
  <reason>Testing</reason>
   <health>0</health>
```

```
</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_remove_undo

The inventory_remove_undo function will reverse an inventory item that has been removed.

Parameters:

action	variable length text field
transactionid	integer value

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_remove_undo</action>
  <transactionid>3570</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 room data 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
roomdata	Optional, array of 1 or more nodes containing room allocation information
room	Optional, integer value, represents the identification number of a room
qty	Optional, decimal value, represents the quantity currently in the associated room

```
<xml>
 <API>4.0</API>
 <action>inventory_move</action>
 <data>
   <barcodeid>7480211204033809</barcodeid>
   <roomdata>
       <room>1</room>
       <qty>50.00</qty>
       <room>2</room>
       <qty>25.00</qty>
       </roomdata>
 </data>
 <data>
   <barcodeid>7480211204033808</barcodeid>
   <roomdata>
       <room>1</room>
       <qty>1.00</qty>
       <room>2</room>
       <qty>3.50</qty>
       </roomdata>
 </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 inventory_transfer. It will pull various pieces of inventory on the inventory identifiers specified in the request. This information can include: strain, quantity available, whether or not the item requires weighing, the harvest time, the license number of the entity that currently possesses the identifier and any additives/pesticides that were applied back to the plant level.

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>
  <barcodeid>6853296789574115</barcodeid>
  <barcodeid>6853296789574116</barcodeid>
</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
quantity	decimal value
requiresweighing	Boolean value, indicates if the inventory item is weighable or non-weighable

ismedicated	Indicates if the item is medicated or not
usableweight	If the item is not weighable, this will indicate the amount of usable product per unit.
inventorytype	integer value based on pre-defined inventory types
harvest_time	Unix 32-bit integer timestamp
license_number	variable length text field, indicates who currently possesses the inventory item
additives	Array of 1 or more nodes containing additive information, if applicable
name	variable length text field, indicates name of pesticide
time	Unix 32-bit integer timestamp, indicates when the pesticide was applied
applied_quantity	indicates the total amount of the additive applied
applied_quantity_uom	variable length text field
concentration	indicates the concentration of additive
concentration_uom	variable length text field

Return example:

```
<xml>
 <data>
  <additives>
   <applied_quantity>1</applied_quantity>
   <applied_quantity_uom>liter</applied_quantity_uom>
   <concentration>0.05</concentration>
   <concentration_uom>µg/L</concentration_uom>
   <name>Pestacide #2</name>
   <time>1298368298</time>
  </additives>
```

```
    <additives>
      <applied_quantity>1</applied_quantity>
    <applied_quantity_uom>gallon</applied_quantity_uom>
      <concentration>0.03</concentration>
      <concentration_uom>µg/L</concentration_uom>
      <name>Pestacide #1</name>
      <time>1298368398</time>
    </additives>
    <barcode_id>8919990967962719</barcode_id>
    <harvest_time>1298368498</harvest_time>
    <invtype>6</invtype>
    <is_medicated>1</is_medicated>
    <license_number>12345</license_number>
    <quantity>51.20</quantity>
    <requires_weighing>1</requires_weighing>
    <strain>Blueberry</strain>
    <usable_weight>51.20</usable_weight>
  </data>
  <success>1</success>
</xml>
```

inventory_new

The inventory_new function can be used to create new inventory not previously entered into the system.

Parameters:

action	variable length text field
location	integer
data	Array of 1 or more nodes containing new inventory information
strain	variable length text field
strain_type	variable length text field
quantity	decimal value
uom	variable length text field. Valid values are: g, mg, kg, oz, lb, each. These

	represent: grams, milligrams, kilograms, ounces, pounds, each.
is_medicated	Boolean value, indicates whether the item is medicated
requires_weighing	Boolean value, indicates whether the item requires weighing
usable_weight	Optional, if the item is non-weighable this field is mandatory
usable_weight_uom	Optional, if the item is non-weighable this field is mandatory
invtype	integer, corresponds to the inventory type system
vendor_license	variable length text field

```
<xml>
 <API>4.0</API>
 <action>inventory_new</action>
 <data>
  <invtype>6</invtype>
  <is_medicated>1</is_medicated>
  <quantity>100.00</quantity>
  <requires_weighing>1</requires_weighing>
  <strain>Blueberry</strain>
  <strain_type>Indica</strain_type>
  <vendor_license>1000000000</vendor_license>
 </data>
 <location>1</location>
</xml>
```

Return example:

```
<xml>
 <barcode_id>6853296789574115</barcode_id>
 <barcode_id>6853296789574116</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_new_undo

The inventory_new_undo function will reverse an inventory item that has been created with the inventory_new function; provided it has not been sold out of, transferred, etc.

Parameters:

action	variable length text field
transactionid	integer value

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_new_undo</action>
  <transactionid>3570</transactionid>
</xml>
```

Returned Parameters:

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

inventory_transfer_schedule

The inventory_transfer_schedule 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. For internal transfers (e.g. from one location to another), there is no need to quarantine and schedule.

Parameters:

action	variable length text field

barcodeid	Array of 1 or more text fields representing the plants

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_transfer_schedule</action>
  <barcodeid>6853296789574115</barcodeid>
  <barcodeid>6853296789574116</barcodeid>
</xml>
```

Returned Parameters:

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

inventory_transfer_schedule_undo

The inventory_transfer_schedule_undo function will reverse an inventory item or set of items that have been scheduled for transfer.

Parameters:

action	variable length text field
transactionid	integer value

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_transfer_schedule_undo</action>
  <transactionid>3350</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

sessiontime	Unix 32-bit integer timestamp

inventory_transfer

The inventory_transfer function can be used to transfer inventory that already exists in the system.

Parameters:

action	variable length text field
location	integer
items	Array of 1 or more nodes containing transfer inventory information
barcodeid	inventory identifier
strain	variable length text field
strain_type	variable length text field
quantity	decimal value
uom	variable length text field. Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.
is_medicated	Boolean value, indicates whether the item is medicated
requires_weighing	Boolean value, indicates whether the item requires weighing
usable_weight	Optional, if the item is non-weighable this field is mandatory
usable_weight_uom	Optional, if the item is non-weighable this field is mandatory
invtype	integer, corresponds to the inventory type system
vendor_license	Optional if internal transfer, variable length text field
internal_location	Optional if external transfer, integer
direction	Boolean value, 0 indicates outbound, 1 indicates inbound

```
<xml>
  <API>4.0</API>
  <action>inventory_transfer</action>
  <data>
    <invtype>6</invtype>
    <is_medicated>1</is_medicated>
    <quantity>100.00</quantity>
    <requires_weighing>1</requires_weighing>
    <strain>Blueberry</strain>
    <strain_type>Indica</strain_type>
    <vendor_license>1000000000</vendor_license>
    <is_partial>1</is_partial>
  </data>
  <data>
    <invtype>6</invtype>
    <is_medicated>1</is_medicated>
    <quantity>200.00</quantity>
    <requires_weighing>1</requires_weighing>
    <strain>Purple Kush</strain>
    <strain_type>Indica</strain_type>
    <vendor_license>1000000000</vendor_license>
    <is_partial>0</is_partial>
  </data>
 <direction>0</direction>
  <location>1</location>
</xml>
```

Return example:

```
<xml>
  <barcode_id>6853296789584125</barcode_id>
  <sessiontime>1384476925</sessiontime>
  <success>1</success>
  <transactionid>3778</transactionid>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value

sessiontime	Unix 32-bit integer timestamp
barcode_id	Optional, array of 1 or more text fields representing new unique identifiers attached to any items transferred as partial transfers

inventory_transfer_undo

The inventory_transfer_undo function will reverse an inventory item that has been transferred with the inventory_transfer function; provided it has not been received by the other party or processed in any other way.

Parameters:

action	variable length text field
transactionid	integer value

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_transfer_undo</action>
  <transactionid>3570</transactionid>
</xml>
```

Returned Parameters:

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

inventory_combine

The inventory_combine function will allow a user to combine multiple items into one. It's generally a good idea to use this function as little as possible; but, it is here if needed.

Parameters:

action	variable length text field
strain	variable length text field
combined_quantity	decimal value, new quantity of combined items
combined_quantity_uom	variable length text field. Valid values are: g, mg, kg, oz, lb, each. These

	represent: grams, milligrams, kilograms, ounces, pounds, each.
requires_weighing	Boolean value, indicates whether or not the newly combined item requires weighing
usable_weight	Optional, decimal value required if the new item does not require weighing
usable_weight_uom	Optional, Valid values are: g, mg, kg, oz, lb. These represent: grams, milligrams, kilograms, ounces, pounds.
invtype	integer, indicates inventory type of new item
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.
roomdata	Optional, array of 1 or more nodes containing room allocation information
room	Optional, integer value, represents the identification number of a room
qty	Optional, decimal value, represents the quantity currently in the associated room

```
<xml>
  <API>4.0</API>
  <action>inventory_combine</action>
  <combined_quantity>945</combined_quantity>
  <data>
```

```
    <barcodeid>6647455983218747</barcodeid>
    <remove_quantity>693.00</remove_quantity>
  </data>
  <data>
    <barcodeid>5723224643296982</barcodeid>
    <remove_quantity>252.00</remove_quantity>
  </data>
  <invtype>6</invtype>
  <ismedicated>1</ismedicated>
  <requiresweighing>1</requiresweighing>
  <strain>Blueberry</strain>
</xml>
```

Return example:

```
<xml>
  <sessiontime>1384476925</sessiontime>
  <barcode_id>5723224643296983</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 field representing new unique identifier

inventory_combine_undo

The inventory_combine_undo function will reverse an inventory item that has been created from the inventory_combine function; provided it has not been sold, transferred, adjusted, etc.

Parameters:

action	variable length text field
transactionid	integer value

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_combine_undo</action>
  <transactionid>3570</transactionid>
</xml>
```

Returned Parameters:

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

inventory_convert

The inventory_convert function will allow a user to convert one type of item to another. This function has a wide variety of uses. It can be used to convert one weighable item into many weighable items (e.g. 1000 grams into 10 smaller 100 grams increments). Or, it can be used to convert 56 grams into 2 pre-packaged ounces (weighable to non-weighable). A user could then convert those two pre-packaged 2 ounces into four ½ ounce pre-packaged items (non-weighable to non-weighable). Finally, a non-weighable item could then be converted back into weighable product by converting one of the ½ ounce pre-packaged items into 14 grams of weighable product.

Parameters:

action	variable length text field
barcodeid	inventory identifier
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.
old_quantity	decimal value, quantity of old product before conversion
new_quantity	decimal value, quantity of old product after conversion
new_quantity_uom	Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.

derivative_quantity	decimal value, quantity of new product produced
derivative_quantity_uom	Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.
derivative_inventory_type	integer value defined by inventory typing system
derivative_strain	variable length text field
location	integer value
serialize	Boolean value, 0 indicates one identifier for the new batch of product, 1 indicates a new identifier for each new unit. Only applies to non-weighable items (e.g. pre-packaged).

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_convert</action>
  <barcodeid>6647455983218747</barcodeid>
  <waste>0.00</waste>
  <old_quantity>56.00</old_quantity>
  <new_quantity>28.00</new_quantity>
  <derivative_quantity>1.00</derivative_quantity >
<derivative_quantity_uom>each</derivative_quantity_uom
>
<derivative_inventory_type>6</derivative_inventory_type>
<derivative_strain>Blueberry</derivative_strain>
<derivative_requires_weighing>0
</derivative_requires_weighing>
<location>1</location>
<serialize>0</serialize>
</xml>
```

Returned Parameters:

success	Boolean value
transactionid	integer value
sessiontime	Unix 32-bit integer timestamp
barcode_id	text field representing new unique identifier

inventory_convert_undo

The inventory_convert_undo function will reverse an inventory item that has been converted from one item to another using the inventory_convert function; provided it has not been sold, transferred, adjusted, etc.

Parameters:

action	variable length text field
transactionid	integer value

Example:

```
<xml>
  <API>4.0</API>
  <action>inventory_convert_undo</action>
  <transactionid>3570</transactionid>
</xml>
```

Returned Parameters:

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

Chapter
6

Chapter 6: Sales

In this chapter, you'll learn how to:

- ✓ **Deduct inventory for a sale**
- ✓ **Void a sale**
- ✓ **Refund a sale**

sale_dispense

The sale_dispense function will allow a user to deduct items from inventory through the sales process.

Parameters:

action	variable length text field
data	Array of 1 or more nodes containing inventory information
barcodeid	inventory identifier
quantity	integer value, quantity to remove.
quantity_uom	variable length text field. Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.
usable_weight	decimal value, usable amount of item being sold.
usable_weight_uom	variable length text field. Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.
location	integer value

Example:

```
<xml>
 <API>4.0</API>
 <action>sale_dispense</action>
 <data>
  <barcodeid>6647455983218747</barcodeid>
  <quantity>1.00</quantity>
  <quantity_uom>each</quantity_uom>
  <usable_weight>14.00</usable_weight>
  <usable_weight_uom>g</usable_weight_uom>
  </data>
 <data>
  <barcodeid>6647455983218749</barcodeid>
  <quantity>1.00</quantity>
  <quantity_uom>each</quantity_uom>
  <usable_weight>7.00</usable_weight>
  <usable_weight_uom>g</usable_weight_uom>
  </data>
 <location>1</location>
</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

sale_void

The sale_void function will reverse items that have been sold to a customer and return the items to inventory. This function should be used in a similar manner to the undo functions whereby this function is used to 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_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. You must specify both a transactionid and one or more identifiers. This function allows you to either restock the items or remove them and schedule them for waste removal.

Parameters:

action	variable length text field
data	Array of 1 or more nodes containing inventory information
barcodeid	inventory identifier
quantity	integer value, quantity to bring in.
quantity_uom	variable length text field. Valid values are: g, mg, kg, oz, lb, each. These represent: grams, milligrams, kilograms, ounces, pounds, each.

restock	Boolean value, 1 will return the item into inventory whereas 0 will mark the item for destruction.
location	integer value

Example:

```
<xml>
  <API>4.0</API>
  <action>sale_refund</action>
  <data>
    <barcodeid>6647455983218747</barcodeid>
    <quantity>1.00</quantity>
    <quantity_uom>each</quantity_uom>
    <restock>0</restock>
   </data>
  <data>
    <barcodeid>6647455983218749</barcodeid>
    <quantity>1.00</quantity>
    <quantity_uom>each</quantity_uom>
    <restock>1</restock>
   </data>
 <location>1</location>
</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

Chapter 7: Testing

In this chapter, you'll learn how to:

✓ Send lab results directly from a laboratory

Reserved

Chapter 8: Synchronization

In this chapter, you'll learn how to:

- ✓ 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

Reserved