Track & Trace Shipping Orders API

Shipping Orders (Outbound)

Supports BarTender Track & Trace 12.2 and Later Versions

Contents

Overview

The Shipping Order API enables external systems to create, manage, monitor, and reconcile outbound shipments in BarTender Track & Trace.

A Shipping Order represents the expected content of an outbound shipment. It is typically generated by a Warehouse Management System (WMS), Enterprise Resource Planning (ERP) system, EDI provider, or integration middleware. Once created, the Shipping Order becomes available in the Track & Trace mobile application, where operators validate goods before dispatch.

Typical use cases

Use the Shipping Order API when an external system needs to:

- Create outbound shipments instructions in Track & Trace
- Prepare expected outbound content before dispatch
- Validate shipped goods against expected quantities or serialized items
- Retrieve shipped results after outbound processing
- Compare expected content with shipped content
- Search for shipping orders by status, source, destination, transaction ID, or timestamp
- Cancel or update shipping orders before outbound validation is completed

1. Base URL

Use the tenant-specific base URL for all requests:

`https://<tenant>.bartender-tt.com`

2. Authentication

All requests require authentication headers.

Required headers

```
apiKey: <your-api-key>
Authorization: Basic <credentials>
x-tenant: <tenant-code>
Content-Type: application/json
Accept: application/json
```

Header descriptions

Header	Required	Description
`ApiKey`	Yes	API key associated with the tenant and authenticated user or integration.
`Authorization`	Yes	Basic authorization credentials.

Header	Required	Description
`x-tenant`	Yes	Tenant code required when using Basic Authorization.
`Content-Type`	For requests with a body	Must be `application/json`.
`Accept`	Recommended	Use `application/json` to request JSON responses.

Example

```
apiKey: abc123examplekey
Authorization: Basic <credentials>
x-tenant: DEMOTT
Content-Type: application/json
Accept: application/json
```

3. Core concepts

3.1 What is a Shipping Order

A Shipping Order is an outbound shipment instruction that defines what goods must be shipped from a source location to a destination location.

A Shipping Order typically includes:

- Shipment reference number
- Source site
- Destination site or customer
- Product identifiers, such as SKU, GTIN, PID, or serialized items
- Expected quantities
- Optional packaging structure
- Optional carrier or tracking information

3.2 Shipping Order lifecycle status

Status	Description
`available`	The Shipping Order has been created and is available for outbound validation.
`in_progress`	Outbound validation has started but is not complete.
`done`	Outbound validation has been completed.
`canceled`	The Shipping Order has been canceled and is no longer available for operational processing.

3.3 Content formats

The `contentFormat` field defines how shipment content is structured.

Content format	Description
`tag`	Serialized item-level tracking using EPC/tag data.
`quantity`	Aggregated quantity by PID or GTIN.
`sku-quantity`	Aggregated quantity by SKU.

4. Typical integration workflow

A common integration flow is:

Step 1: Create a Shipping Order

The external system sends outbound shipment instructions to Track & Trace.

```
PUT /logistics/shiporder
```

Step 2: Search for Shipping Orders

The external system searches for shipping orders based on status, location, transaction ID, or timestamps.

```
POST /logistics/shiporder
```

Step 3: Monitor Shipping Order status

The external system checks whether outbound validation is still in progress or completed.

```
GET /logistics/shiporder/status/{soId}
```

Step 4: Retrieve a Shipping Order

The external system retrieves full Shipping Order details.

```
GET /logistics/shiporder/{soId}
```

Step 5: Retrieve shipped results

After completion, the external system retrieves shipped items.

```
GET /logistics/shiporder/result/{soId}
```

Step 6: Compare expected and shipped content

The external system retrieves discrepancy data for reconciliation.

```
GET /logistics/ shiporder/compare/{soId}
```

5. Best practices

- Synchronize Product Master Data before creating Shipping Orders that reference products by PID, GTIN, or SKU.
- Use a consistent transactionId between the WMS/ERP and Track & Trace.
- Avoid updating Shipping Orders after outbound validation has started.
- Use quantity or sku-quantity for ERP/WMS reconciliation.
- Use tag only when serialized outbound validation is required.
- Use the status endpoint for polling instead of repeatedly retrieving the full Shipping Order payload.
- Avoid high-frequency polling in production environments.
- Use pagination when searching for Shipping Orders.
- Filter searches by status, source, or destination whenever possible.
- For major structural changes after processing begins, cancel and recreate the Shipping Order if business rules allow it.

6. Endpoints

6.1 Create a Shipping Order

Method and path

```
PUT /logistics/ shiporder
```

Description

Creates a new Shipping Order in Track & Trace, representing an expected outbound shipment to be prepared and validated before dispatch.

Once created, the Shipping Order becomes available in the outbound workflow of the Track & Trace mobile application. Operators can then scan goods against the expected shipment content.

When to use

Use this endpoint when:

- A WMS or ERP generates an outbound shipment that must be validated in Track & Trace.
- Outbound instructions, including expected items and quantities, must be sent to a site before dispatch.
- A Shipping Order must be created before outbound validation begins.

Request headers

```
apiKey: <your-api-key>
Authorization: Basic <credentials>
```

```
x-tenant: <tenant-code>
Content-Type: application/json
Accept: application/json
```

Request body

The request body defines the expected outbound shipment.

Sample request

```
{
  "contentFormat": "quantity",
  "transactionId": "SHIP-002-251009",
  "destination": "urn:mjx:site:loc:DEMOTT.00002.0",
  "source": "urn:mjx:site:loc:DEMOTT.00004.0",
  "extensions": {
    "ext1": "val1",
    "ext2": "val2"
  },
  "containers": [
    {
      "content": [
        {
          "format": "quantity",
          "quantity": 2,
          "pid": "03663328100103"
        }
      ]
    }
  ]
}
```

Request field summary

Field	Type	Required	Description
`contentFormat`	string	Yes	Defines how shipment content is structured. Supported values include tag, quantity, and sku-quantity.
`transactionId`	string	Recommended	External shipment reference used by the WMS, ERP, or integration system.
`destination`	string	Yes	Destination business location for the shipment.
`source`	string	Yes	Source business location from which goods are shipped.

Field	Type	Required	Description
`extensions`	object	No	Optional custom metadata.
`containers`	array	Yes	Defines the expected outbound content and optional packaging hierarchy.
`containers[].content`	array	Yes	List of expected items inside the container.
`format`	string	Yes	Content item format. Usually matches the selected content format.
`quantity`	number	Required for quantity formats	Expected quantity for the product.
`pid`	string	Required for PID/GTIN quantity format	Product identifier used for GTIN/PID-based validation.
`sku`	string	Required for SKU quantity format	SKU used for SKU-level validation.
`epc`	string	Required for tag format when `hexa` is not provided	Serialized EPC value.
`hexa`	string	Required for tag format when `epc` is not provided	Hexadecimal tag value.

Success response

201 Created

```
{
  "soId": "8133384547531811"
}
```

Possible responses

Code	Meaning
201	Shipping Order created successfully. Returns the new `soId`.
204	Request processed successfully with no body returned.
400	Invalid request body or malformed input.
401	Authentication failed or credentials are missing.
403	The client does not have access rights to the content.
404	Related resource not found.

Code	Meaning
502	Upstream or gateway error.

6.2 Retrieve a Shipping Order

Method and path

```
GET /logistics/shiporder/{soId}
```

Description

Retrieves a Shipping Order by its unique identifier. This endpoint returns the complete Shipping Order structure, including metadata, lifecycle status, and expected outbound content.

When to use

Use this endpoint when:

- Displaying Shipping Order details in an integration or operational interface.
- Verifying expected outbound content before dispatch.
- Checking Shipping Order metadata and lifecycle status.
- Reconciling outbound validation results with upstream systems.

Path parameters

Parameter	Type	Required	Description
soId	string	Yes	Unique Shipping Order identifier generated by Track & Trace at creation time.

Request headers

```
apiKey: <your-api-key>
Authorization: Basic <credentials>
x-tenant: <tenant-code>
Accept: application/json
```

Sample request

```
GET /logistics/shiporder/8133384547531811
```

Success response

200 OK


```
{
  "soId": "8133384547531811",
  "contentFormat": "quantity",
  "transactionId": "SHIP-002-251009",
  "creationTime": "2025-10-09T15:20:54.735Z",
  "updateTime": "2025-10-09T15:20:54.798Z",
```

```
  "expirationTime": null,
  "lastStatusChange": "2025-10-09T15:20:54.735Z",
  "status": "available",
  "destination": "urn:mjx:site:loc:DEMOTT.00002.0",
  "source": "urn:mjx:site:loc:DEMOTT.00004.0",
  "extensions": {
    "ext1": "val1",
    "ext2": "val2"
  },
  "containers": [
    {
      "content": [
        {
          "format": "quantity",
          "quantity": 2,
          "pid": "03663328100103"
        }
      ]
    }
  ]
}
```


Response field summary

Field	Type	Description
`soId`	string	Unique Shipping Order identifier.
`contentFormat`	string	Content representation format.
`transactionId`	string	External shipment reference.
`creationTime`	string	Timestamp when the Shipping Order was created.
`updateTime`	string	Timestamp when the Shipping Order was last updated.
`expirationTime`	string or null	Optional time after which the Shipping Order is no longer relevant.
`lastStatusChange`	string	Timestamp of the most recent status transition.
`status`	string	Current lifecycle status.
`destination`	string	Destination business location.
`source`	string	Source business location.
`extensions`	object	Optional custom metadata.
`containers`	array	Expected outbound content and packaging structure.

Possible responses

Code	Meaning
200	Shipping Order returned successfully.
400	Invalid request.
401	Authentication failed or credentials are missing.
403	The client does not have access rights to the content.
404	Shipping Order not found.
502	Upstream or gateway error.

6.3 Retrieve the Shipping Order Status

Method and path

```
GET /logistics/shiporder/status/{soId}
```

Description

Retrieves the current lifecycle status of a Shipping Order. This endpoint provides a lightweight way to monitor outbound progress without retrieving the full Shipping Order payload.

When to use

Use this endpoint when:

- Polling outbound progress during shipment validation.
- Checking whether a Shipping Order is completed before retrieving results.
- Checking whether a Shipping Order is completed before running comparison.
- Monitoring lifecycle transitions in an integration workflow.
- Avoiding full payload retrieval for performance reasons.

Path parameters

Parameter	Type	Required	Description
soId	string	Yes	Unique Shipping Order identifier generated by Track & Trace at creation time.

Request headers

```
apiKey: <your-api-key>
Authorization: Basic <credentials>
x-tenant: <tenant-code>
Accept: application/json
```

Sample request

```
GET /logistics/shiporder/status/8133384547531811
```

Success response

200 OK

```
{
  "soId": "8133384547531811",
  "status": "in_progress",
  "lastStatusChange": "2025-10-09T16:14:22.100Z"
}
```

This sample is therefore illustrative.

Response field summary

Field	Type	Description
`soId`	string	Unique Shipping Order identifier.
`status`	string	Current lifecycle status.
`lastStatusChange`	string	Timestamp of the most recent status transition.

Possible responses

Code	Meaning
200	Current Shipping Order status returned successfully.
400	Invalid request.
401	Authentication failed or credentials are missing.
403	The client does not have access rights to the content.
404	Shipping Order not found.
502	Upstream or gateway error.

6.4 Retrieve Shipped Items

Method and path

`GET /logistics/shiporder/result/{soId}`

Description

Retrieves the items scanned so far for a specific Shipping Order. This endpoint can be used while outbound validation is still in progress or after completion. It returns the current scanned results in the requested format.

When to use

Use this endpoint when:

- Monitoring outbound validation progress.

- Retrieving scanned shipment results after completion.
- Exporting shipped quantities back to a WMS or ERP.
- Retrieving serialized tag-level shipment data.

Path parameters

Parameter	Type	Required	Description
`soId`	string	Yes	Unique Shipping Order identifier generated by Track & Trace at creation time

Query parameters

Parameter	Type	Required	Default	Description
`result_format`	string	No	`tag`	Defines whether returned results include all tags or aggregated product/quantity content.

Supported **`result_format`** values

Value	Description
`tag`	Returns serialized item-level result data.
`quantity`	Returns aggregated PID/GTIN quantity data.
`sku-quantity`	Returns aggregated SKU quantity data.

Request headers

```
apiKey: <your-api-key>
Authorization: Basic <credentials>
x-tenant: <tenant-code>
Accept: application/json
```

Sample request

```
GET /logistics/shiporder/result/8133384547531811?result_format=quantity
```

Success response example: quantity

200 OK


```
{
  "soId": "8133384547531811",
  "resultFormat": "quantity",
  "results": [
    {
      "pid": "03663328100103",
      "quantity": 2
    }
  ]
}
```

Success response example: tag format

These examples are illustrative.

Response field summary

Field	Type	Description
`soId`	string	Unique Shipping Order identifier.
`resultFormat`	string	Format used for the returned result.
`results`	array	List of shipped results.
`pid`	string	Product identifier for PID/GTIN-based quantity results.
`sku`	string	SKU for SKU-level results.
`quantity`	number	Shipped quantity.
`epc`	string	EPC value for serialized item results.
`hexa`	string	Hexadecimal tag value for serialized item results.

Possible responses

Code	Meaning
`200`	Shipping Order returned successfully.
`400`	Invalid request.
`401`	Authentication failed or credentials are missing.
`403`	The client does not have access rights to the content.
`404`	Shipping Order not found.
`502`	Upstream or gateway error.

6.5 Compare Shipped vs Expected

Method and path

```
GET /logistics/ shiporder/compare/{soId}
```

Description

Compares the scanned items of a Shipping Order with the expected outbound content defined at creation time. This endpoint supports outbound reconciliation by identifying discrepancies between expected and shipped goods. It can show matching items, missing items, and unexpected items.

When to use

Use this endpoint when:

- Outbound validation is completed and discrepancies must be identified.

- Reconciliation results must be exported to a WMS or ERP.
- Outbound accuracy must be audited at tag level or aggregated quantity level.
- Shipment completeness must be verified before closing an outbound process.

Path parameters

Parameter	Type	Required	Description
`soId`	string	Yes	Unique Shipping Order identifier generated by Track & Trace.

Query parameters

Parameter	Type	Required	Default	Description
`as_quantity`	boolean	No	`false`	Forces comparison against PID/quantity instead of individual tags.
`as_sku_quantity`	boolean	No	`false`	Returns discrepancies in SKU format.

Request headers

```
apiKey: <your-api-key>
Authorization: Basic <credentials>
x-tenant: <tenant-code>
Accept: application/json
```

Sample request

```
GET /logistics/ shiporder/compare/8133384547531811?as_quantity=true
```

Success response example

200 OK

```
{
  "soId": "8133384547531811",
  "comparisonFormat": "quantity",
  "matches": [
    {
      "pid": "03663328100103",
      "expected": 2,
      "shipped": 2
    }
  ],
  "unders": [],
  "overs": []
}
```

This sample is illustrative.

Response field summary

Field	Type	Description
soId	string	Unique Shipping Order identifier.
comparisonFormat	string	Format used for comparison.
matches	array	Items where expected and shipped content match.
unders	array	Items expected but not fully shipped.
overs	array	Items shipped but not expected or shipped in excess.
expected	number	Expected quantity.
shipped	number	Shipped quantity.

Possible responses

Code	Meaning
200	Shipping Order returned successfully.
400	Invalid request.
401	Authentication failed or credentials are missing.
403	The client does not have access rights to the content.
404	Shipping Order not found.
502	Upstream or gateway error.

6.6 Search for Shipping Orders

Method and path

`POST /logistics/shiporder`

Description

Searches for Shipping Orders using filter criteria and optional sorting. This endpoint supports operational monitoring and integration workflows by allowing external systems to query Shipping Orders by status, source, destination, transaction ID, and timestamps.

When to use

Use this endpoint when:

- Retrieving Shipping Orders available for outbound processing.
- Filtering Shipping Orders by source or destination site.
- Monitoring Shipping Orders that are in progress or completed.
- Supporting dashboards, reconciliation workflows, or integration monitoring.

Query parameters

Parameter	Type	Required	Description
`from`	integer	No	Starting position from which matching Shipping Orders are returned.
`size`	integer	No	Number of Shipping Orders returned in the response.

Request headers

```
apiKey: <your-api-key>
Authorization: Basic <credentials>
x-tenant: <tenant-code>
Content-Type: application/json
Accept: application/json
```

Request body

The request body contains search criteria, filters, and optional ordering.

Sample request

```
{
  "filters": [
    {
      "property": "status",
      "operator": "EQ",
      "values": [
        "available",
        "in_progress"
      ]
    },
    {
      "property": "destination",
      "operator": "EQ",
      "values": [
        "urn:mjx:site:loc:DEMOTT.00002.0"
      ]
    }
  ],
  "order": {
    "property": "creationTime",
    "direction": "DESC"
  }
}
```

Request field summary

Field	Type	Required	Description
`filters`	array	No	List of filter criteria used to search Shipping Orders.
`filters[].property`	string	Yes, when filter is used	Field to filter by, such as `status`, `destination`, `source`, or `transactionId`.
`filters[].operator`	string	Yes, when filter is used	Filter operator, such as `EQ`.
`filters[].values`	array	Yes, when filter is used	Values used by the filter.
`order`	object	No	Optional ordering configuration.
`order.property`	string	Yes, when order is used	Field used for sorting, such as `creationTime`.
`order.direction`	string	Yes, when order is used	Sort direction, such as `ASC` or `DESC`.

Success response example

200 OK


```
{
  "from": 0,
  "size": 2,
  "results": [
    {
      "soId": "8133384547531811",
      "transactionId": "SHIP-002-251009",
      "status": "available",
      "destination": "urn:mjx:site:loc:DEMOTT.00002.0",
      "source": "urn:mjx:site:loc:DEMOTT.00004.0"
    },
    {
      "soId": "8133384547531812",
      "transactionId": "SHIP-002-251010",
      "status": "in_progress",
      "destination": "urn:mjx:site:loc:DEMOTT.00002.0",
      "source": "urn:mjx:site:loc:DEMOTT.00004.0"
    }
  ]
}
```

Possible responses

Code	Meaning
200	Shipping Order returned successfully.
400	Invalid request.
401	Authentication failed or credentials are missing.
403	The client does not have access rights to the content.
404	Shipping Order not found.
502	Upstream or gateway error.

6.7 Update a Shipping Order

Method and path

```
PUT /logistics/shiporder/{soId}
```

Description

Updates an existing Shipping Order in Track & Trace. This endpoint allows modification of outbound content, metadata, or lifecycle status before outbound processing is completed. It is typically used to correct expected quantities, adjust packaging content, update source or destination details, or cancel an outbound shipment before dispatch validation.

When to use

Use this endpoint when:

- Outbound shipment content changes before validation begins.
- Expected quantities or items must be corrected.
- Source or destination `bizLocation` must be updated.
- Shipping Order status must be updated, for example to `canceled`.
- Expiration time must be set or modified.

Path parameters

Parameter	Type	Required	Description
`soId`	string	Yes	Unique Shipping Order identifier generated by Track & Trace.

Request headers

```
apiKey: <your-api-key>
Authorization: Basic <credentials>
x-tenant: <tenant-code>
Content-Type: application/json
Accept: application/json
```

Request body

Only the fields included in the request body are updated. All fields are optional.

Sample request

```
{
  "contentFormat": "quantity",
  "transactionId": "SHIP-002-251009",
  "status": "available",
  "destination": "urn:mjx:site:loc:DEMOTT.00002.0",
  "source": "urn:mjx:site:loc:DEMOTT.00004.0",
  "extensions": {
    "ext1": "val1",
    "ext2": "val2"
  },
  "containers": [
    {
      "content": [
        {
          "format": "quantity",
          "quantity": 2,
          "pid": "03663328100103"
        }
      ]
    }
  ]
}
```

Special note for `tag` content format:

If the Shipping Order uses tags in the content instead of SKU or PID with quantities, each content element must include either hexa or epc. Tags are stored as EPCs. If only hexa is provided, the EPC is generated from the hexadecimal value.

Request field summary

Field	Type	Required	Description
`contentFormat`	string	No	Updates the content representation format.
`transactionId`	string	No	Updates the external shipment reference.
`status`	string	No	Updates the Shipping Order lifecycle status.
`destination`	string	No	Updates the destination business location.
`source`	string	No	Updates the source business location.
`extensions`	object	No	Updates optional custom metadata.

Field	Type	Required	Description
`containers`	array	No	Updates expected outbound content and packaging structure.

Success response example

204 No Content

No response body is returned.

Possible responses

Code	Meaning
204	Shipping Order updated successfully.
400	Invalid request.
401	Authentication failed or credentials are missing.
403	The client does not have access rights to the content.
404	Shipping Order not found.
502	Upstream or gateway error.

6.8 Delete a Shipping Order

Method and path

```
DELETE /logistics/shiporder/{soId}
```

Description

Removes an existing Shipping Order from Track & Trace before it is processed. This endpoint cancels an expected outbound shipment, so it is no longer available in the outbound workflow.

When to use

Use this endpoint when:

- A Shipping Order has been canceled before outbound validation begins.
- A Shipping Order was created in error and must be removed.
- A replacement Shipping Order will be created with corrected information.

Path parameters

Parameter	Type	Required	Description
`soId`	string	Yes	Unique Shipping Order identifier generated by Track & Trace.

Request headers

```
apiKey: <your-api-key>
Authorization: Basic <credentials>
x-tenant: <tenant-code>
Accept: application/json
```

Sample request

```
DELETE /logistics/shiporder/8133384547531811
```

Success response

204 No Content

No response body is returned.

Possible responses

Code	Meaning
204	Shipping Order updated successfully.
400	Invalid request.
401	Authentication failed or credentials are missing.
403	The client does not have access rights to the content.
404	Shipping Order not found.
502	Upstream or gateway error.

7. Models

7.1 Shipping Order

Field	Type	Description
soId	string	Unique identifier generated by Track & Trace.
contentFormat	string	Defines how shipment content is represented.
transactionId	string	External shipment reference from WMS, ERP, or integration system.
creationTime	string	Timestamp when the Shipping Order was created.
updateTime	string	Timestamp when the Shipping Order was last updated.
expirationTime	string or null	Optional time after which the Shipping Order becomes obsolete.
lastStatusChange	string	Timestamp of the most recent lifecycle status transition.
status	string	Current Shipping Order lifecycle status.

Field	Type	Description
`destination`	string	Destination business location.
`source`	string	Source business location.
`extensions`	object	Optional custom metadata.
`containers`	array	Expected outbound content and optional packaging structure.

7.2 Container

Field	Type	Description
`content`	array	List of content elements within the container.

7.3 Content element

Field	Type	Description
`format`	string	Content format for the item.
`quantity`	number	Expected or shipped quantity.
`pid`	string	Product identifier for PID/GTIN-based quantity validation.
`sku`	string	SKU used for SKU-level quantity validation.
`epc`	string	EPC value for serialized item validation.
`hexa`	string	Hexadecimal tag value for serialized item validation.

7.4 Shipping Order Status

Field	Type	Description
`soId`	string	Unique Shipping Order identifier.
`status`	string	Current lifecycle status.
`lastStatusChange`	string	Timestamp of the most recent status transition.

7.5 Search request

Field	Type	Description
`filters`	array	List of filter objects that define search criteria.
`order`	object	Optional ordering criteria for the result set.

7.6 Filter object

Field	Type	Description
`property`	string	Field name to filter on, such as `status` or `destination`.
`operator`	string	Comparison operator, such as `EQ`.
`values`	array	Values used by the filter.

8. Error handling

Recommended error body

```
{
  "error": "Bad Request",
  "message": "The request body is invalid.",
  "details": [
    {
      "field": "destination",
      "issue": "Destination is required."
    }
  ]
}
```

Common response codes

Code	Meaning
400	The request is invalid or malformed.
401	Authentication failed or required credentials were not provided.
403	The client does not have access rights to the content.
404	The requested ASN or related resource could not be found.
502	An upstream or gateway error occurred while processing the request.

9. Example end-to-end flow

9.1 Create an ASN

```
PUT /logistics/shiporder
{
  "contentFormat": "quantity",
  "transactionId": "SHIP-002-251009",
  "destination": "urn:mjx:site:loc:DEMOTT.00002.0",
  "source": "urn:mjx:site:loc:DEMOTT.00004.0",
  "containers": [
    {
      "content": [
        {
          "format": "quantity",
          "quantity": 2,
          "pid": "03663328100103"
        }
      ]
    }
```

```
  ]
}
```

9.2 Check status

```
GET /logistics/shiporder/status/8133384547531811
{
  "soId": "8133384547531811",
  "status": "done",
  "lastStatusChange": "2025-10-09T16:25:10.000Z"
}
```

9.3 Retrieve shipped results

```
GET /logistics/shiporder/result/8133384547531811?result_format=quantity
{
  "soId": "8133384547531811",
  "resultFormat": "quantity",
  "results": [
    {
      "pid": "03663328100103",
      "quantity": 2
    }
  ]
}
```

9.4 Compare expected and shipped content

```
GET /logistics/shiporder/compare/8133384547531811?as_quantity=true
{
  "soId": "8133384547531811",
  "comparisonFormat": "quantity",
  "matches": [
    {
      "pid": "03663328100103",
      "expected": 2,
      "shipped": 2
    }
  ],
  "unders": [],
  "overs": []
}
```

BarTender

Related Documentation

Technical Documents

To view and download technical documents, visit:

https://www.bartendersoftware.com/resources/library

User Guides

Support Articles

- Getting Started with Track & Trace
- Getting an Overview with the Track & Trace Homepage
- Setting Up the Track & Trace Mobile App
- Keep Track of Inventory Assets with Track & Trace

Documentation

- Reviewing Assets in Track & Trace
- Managing Track & Trace
- Working with Track & Trace

Other Resources

Please visit the BarTender website at https://www.bartendersoftware.com.