GVA HTTP API Draft 4.5

[Introduction 3](#_Toc15552992)

[Migration 3](#_Toc15552993)

[Authorization/Authentication 3](#_Toc15552994)

[Active Directory 3](#_Toc15552995)

[Authorization 3](#_Toc15552996)

[Authentication 4](#_Toc15552997)

[Flow (VP & OBT) 4](#_Toc15552998)

[Sequence 4](#_Toc15552999)

[Notes 4](#_Toc15553000)

[Shipment Statuses 5](#_Toc15553001)

[Deinstrumentation 5](#_Toc15553002)

[Implementation 6](#_Toc15553003)

[Locks 7](#_Toc15553004)

[Network-related ID Generation 7](#_Toc15553005)

[GW WSN ID, WSN PAN ID, WSN Channel ID 8](#_Toc15553006)

[TAG WSN ID 8](#_Toc15553007)

[Dates & Times 8](#_Toc15553008)

[Limits 8](#_Toc15553009)

[Errors 9](#_Toc15553010)

[Statuses 9](#_Toc15553011)

[Body JSON Structure 9](#_Toc15553012)

[Common 10](#_Toc15553013)

[Status-related 10](#_Toc15553014)

[Endpoints 12](#_Toc15553015)

[POST /session 12](#_Toc15553016)

[DELETE /session 12](#_Toc15553017)

[GET /session 13](#_Toc15553018)

[POST /shipments 13](#_Toc15553019)

[GET /shipments 16](#_Toc15553020)

[GET /shipments/:id 17](#_Toc15553021)

[PUT /shipments/:id/provision 21](#_Toc15553022)

[PUT /shipments/:id/provision/reset 22](#_Toc15553023)

[POST /shipments/:id/provision/verify 22](#_Toc15553024)

[PUT /shipments/:id/monitor 23](#_Toc15553025)

[PUT /shipments/:id/monitor/config 24](#_Toc15553026)

[PUT /shipments/:id/monitor/updatePackageId 26](#_Toc15553027)

[PUT /shipments/:id/receive 26](#_Toc15553028)

[PUT /shipments/:id/receive/reset 27](#_Toc15553029)

[PUT /shipments/:id/receive/accept 27](#_Toc15553030)

[PUT /shipments/:id/receive/reject 28](#_Toc15553031)

[PUT /shipments/:id/receive/deinstrument 28](#_Toc15553032)

[PUT /shipments/:id/end 29](#_Toc15553033)

[PUT /shipments/:id/photos/:type 29](#_Toc15553034)

[DELETE /shipments/:id/photos/:type 30](#_Toc15553035)

[DELETE /shipments/:id 31](#_Toc15553036)

[DELETE /shipments 31](#_Toc15553037)

[GET / sensordata /:id 32](#_Toc15553038)

[POST /gateway/:id/reboot/mqtt 34](#_Toc15553039)

[POST /gateway/:id/reboot/sms 34](#_Toc15553040)

[POST /gateway/:id/calibrate 35](#_Toc15553041)

[POST /gateway/:id/channelchange/:newchannel 35](#_Toc15553042)

# Introduction

This API will support two initial client types: Visibility Portal (VP) browser-based application and OBT Android-based application.

The intent is to both support field trial use cases and make architectural space for additional client types and GVA behaviors.

## Migration

This API will require a shift from a direct application-to-database topology to an application-to-web-service topology. Instead of tight coupling to a specific database solution and direct access to the data's JSON representation, the database solution will be abstracted by design. Specifically, while JSON may be accepted by a given endpoint, the data is not necessarily stored in that type of structure. Design discussions will shift from centering on monolithic database schemas to HTTP endpoints and the data fragments/shapes they expose based on need.

An effort has been made to reduce the migration scope through means such as retaining as many field name spellings as possible and minimizing the endpoints.

# Authorization/Authentication

## Active Directory

For December, the GVA will not rely on Active Directory behind the scenes. Instead, the GVA will store all user account information in its internal database. This shift will require no changes for the VP or OBT clients.

The rationale for this shift is to support only the VP/OBT UX that is required for December. Iterating on a custom user management implementation, with only the minimal functionality to support basic usernames, passwords, and roles (Desk Agent or Dock Worker) is the simpler, more controllable approach.

Also, the GVA team is unaware of any concrete business requirement for the original selection of Active Directory beyond its availability in Azure. For example, there were no communicated business requirements such as the ability to connect the GVA's Azure Active Directory resources to a customer's on-premises directories, supporting a single-sign-on UX for a customer's employees, etc.

Post December, such business requirements around user management and security can be fully resolved into technical requirements.

## Authorization

This API does not express any concept of "VP-only endpoints" or "OBT-only endpoints" in its structure or naming conventions.

Those distinctions will only surface in how endpoint access is granted or denied to a given role (e.g. Desk Agent or Dock Worker).

Whether or not a given username/password pair is accepted during login to the VP/OBT, and whether or not the logged-in session can access a given endpoint, depends on the above criteria.

## Authentication

Security policies may change based on future recommendations by Tom Hill @ 12 Bridges, who is currently tasked with researching/compiling the security practices and Azure tools we should eventually use. However, due to time constraints we will move forward with the below general concepts and integrate recommendations on a case-by-case basis as time permits.

The authentication flow will align with VP and OBT wireframes, relying on username/password pairs.

Initial credentials, and their associated authorization attributes, will be prepared by the GVA team and shared with the VP and OBT teams. There will not, for example, be any requirement of either team to perform any personal self-service management tasks in the Azure portal or other service. One of the goals of this HTTP API is to abstract the underlying user management system(s) for clients. In short, clients should not know or care about the use of Active Directory or similar behind the scenes.

## Flow (VP & OBT)

### Sequence

1. Use **POST /session** for a login request and store the token included in the JSON response.
2. For every subsequent request, include the token in a standard [**Authorization**](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Authorization) header using **OAuth** as the type value. Example: **Authorization: OAuth YWxhZGRpbjpvcGVuc2VzYW1l**
3. Use **DELETE /session** for a logout request.

### Notes

* No version of the OAuth spec is implemented in the API for December. The above HTTP header example refers to OAuth because:
  + It is a sane value for the HTTP header.
  + It is a sane future-proof selection given OAuth’s prevalence and the assumption that future development phases would continue to abstract implementation details (e.g. support for Active Directory credentials) behind a native implementation of a standard like OAuth.
* The token does not expire. Concerns like formal token management (expiration, refreshing, etc.) are out of scope for December and would need to be derived from full requirements for both security and user management.

# Shipment Statuses

The full specification document used as the initial source-of-truth is Reference Point **propertyStatus property overview.docx** provide by the OBT team. It includes additional content such as more detail on the impact of each status on specific user roles (Desk Worker, Dock Worker, etc.) that will not be reproduced here (to avoid specification drift).

The only goal of the table below is to indicate which statuses are processed/known/supported by this API and brief descriptions to distinguish them.

It’s important to note that while this API and the OBT currently have implementation-level knowledge of these statuses, the VP may not expose all of them in its UX. Specifically, for December, only a subset of them will be displayed in the UX: "New", "Active", "In route", "Completed", etc.

The VP team has agreed to handle the mapping of the statuses below to their user-facing forms due to the mapping being a purely presentational, client-specific concern.

Abbreviations used in the table: Desk Agent (DA), Dock Worker (DW).

|  |  |  |
| --- | --- | --- |
| **Status** | **Description (from OBT team)** | **Introduced** |
| new | A shipment that a DA has created that has never been selected by a DW for provisioning. | Draft 2 |
| inProvision | A new shipment that a DW selects with OBT to provision, but has not finished placing all instruments and linking them to package IDs yet. | Draft 2 |
| inMonitoring | A shipment that a DW finishes provisioning and approves for shipment. | Draft 2 |
| inReceiving | A new shipment that a DW selects with OBT to receive. | Draft 2 |
| accepted | A shipment that a DW approves for acceptance and uploads a photo of the signed waybill for a DA to review. | Draft 2 |
| rejected | A shipment that a DW rejects for acceptance. | Draft 2 |
| completed | A shipment that was accepted by a DW and then approved by a DA. | Draft 2 |
| acceptedDeinstrumented | A shipment that was accepted and then deinstrumented by a DW. | Draft 3 |
| rejectedDeinstrumented | A shipment that was rejected and then deinstrumented by a DW. | Draft 3 |
| completedDeinstrumented | A shipment that was accepted by a DW, completed by a DA, and also deinstrumented by a DW. The final two steps could have happened in either order. | Draft 3 |

## Deinstrumentation

Draft 3 addresses the requirement for the OBT to record the deinstrumentation of a shipment. The initial problem posed by that requirement was that, in the case of an accepted shipment, the Dock Worker must be allowed to deinstrument the shipment before or after the Desk Agent reviews the proof-of-delivery and (from that role's perspective) complete the shipment. However, from the Dock Worker's perspective, deinstrumentation is the final task (for both accepted and rejected shipments).

The GVA team implemented the solution by adding three status values to the state machine: acceptedDeinstrumented, rejectedDeinstrumented, and completedDeinstrumented.

### Implementation

#### Visualization

* See Reference Point **GVA HTTP API deinstrumentation support.pptx**

#### OBT FAQ

* How does the OBT know which shipments are eligible for deinstrumentation?
  + GET /shipments?status=**accepted**&status=**completed**&status=**rejected**
* How does the GVA know which status should result from PUT /shipments/:id/receive/deinstrument?
  + It chooses **acceptedDeinstrumented** if the current status is **accepted**.
  + It chooses **completedDeinstrumented** if the current status is **completed**.
  + It chooses **rejectedDeinstrumented** if the current status is **rejected**.
* What are the work paths based each status?
  + **receive**
    - DW can accept
    - DW can reject
  + **accepted**/**rejected**/**completed**
    - DW can deinstrument
  + {**accepted**,**rejected**,**completed**}**Deinstrumented**
    - DW has no work

#### VP FAQ

* How does the VP know which shipments are eligible for the proof-of-delivery/approval step?
  + GET /shipments?status=**accepted**&status=**acceptedDeinstrumented**
* How does the GVA know which status should result from PUT /shipments/:id/end?
  + It chooses **acceptedCompleted** if the current status is **accepted**.
  + It chooses **completedDeinstrumented** if the current status is **acceptedDeinstrumented**.
* What are the work paths based each status?
  + **receive**/**rejected**/{**rejected**,**completed**}**Deinstrumented**/**completed**
    - DA has no work
  + **accepted**/**acceptedDeinstrumented**
    - DA can review proof-of-delivery and complete the shipment from the VP perspective

#### General FAQ

* Post-December, how should the GVA handle these race condition scenarios?
  + Scenario: Both clients perform requests that result in the shipment transitioning to **completedDeinstrumented**.
    - OBT: PUT /shipments/:id/receive/deinstrument on a shipment that is **completed**.
    - VP: PUT /shipments/:id/end on a shipment that is **acceptedDeinstrumented**.
    - Resolution: The race "loser" receives a success response as if there was no race.
    - Rationale: The final status would have been the same in either event.
  + Scenario: The shipment is **accepted** and both clients perform requests that result in different transitions.
    - OBT: PUT /shipments/:id/receive/deinstrument
    - OBT: PUT /shipments/:id/end
    - Resolution: The race "loser" receives an error response (HTTP 409, code **status\_conflict**, with a **details** section in the JSON that explains what happened).
    - Rationale: The final status differs depending on the "winner," and the "loser" should not receive a signal that its request was fulfilled as expected. In both scenarios, the "loser" can simply try the same request again.

#### Design FAQ

* Why not use two status variables per shipment?
  + **Assumption**: Fewer fields to update and fewer state machines would lead to fewer opportunities for code errors, documentation inaccuracies, miscommunications about transitions, etc.
  + **Assumption**: It's easier to answer the simple question "what is the status of shipment X?" with a simple answer.
  + **Assumption**: It's more future-proof to set the precedent of minimizing the state machine count when possible.
  + **Assumption**: It may turn out that acceptedDeinstrumented, rejectedDeinstrumented, and completedDeinstrumented add value by distinguishing the final real-world statuses and help to support unanticipated use cases.

## Locks

The Reference Point **propertyStatus property overview.docx** specifies that the GVA should "[block] all other Dock Workers from choosing the same shipment" for provision/receiving once an OBT has transitioned the shipment into inProvision or inReceiving status.

* HTTP 403 responses will be returned when an OBT user attempts to transition a shipment without owning the lock.
* PUT /shipments:/id/{provision,receive}/reset endpoints unlock the shipment.
* PUT /shipments:/id/receive/{accept,reject} endpoints also unlock the shipment.
* PUT /shipments:/id/receive/deinstrument does not inspect the lock status. It is assumed that after the shipment is accepted or rejected, any OBT is free to deinstrument the shipment.
* PUT /shipments:/id/end does not inspect the lock status. It is assumed that after the shipment is accepted, the VP is free to complete the shipment.

# Network-related ID Generation

During shipment creation (POST /shipments), several IDs are generated by the GVA, stored in its database, and exposed to clients in shipments’ JSON representations.

Like the Shipment Statuses section, the table below is a summarization intended to describe the GVA’s behavior regarding these IDs. Its content does not cover, for example, how the IDs are used by other systems.

The Reference Points used to create this summary:

* System Identities.pptx
* GVA HTTP API Draft 2 (WIP) jim notes.docx

## GW WSN ID, WSN PAN ID, WSN Channel ID

"Gateway JSON Field" values below are found in responses from endpoints such as GET /shipments/:id.

|  |  |  |
| --- | --- | --- |
| **Name** | **Gateway JSON Field** | **Description** |
| GW WSN ID | **wsnId** | Starts at 0x8000, rolls over to 0x0001 |
| WSN PAN ID | **panId** | Starts at 0x4000, rolls over to 0x0001 |
| WSN Channel ID | **channelId** | Hard-coded as 0x01 for December (unless a more appropriate hard-coded value is desired) |

## TAG WSN ID

The OBT will need to continue generating this ID because the GVA will not know the number of tags by the time the OBT requests these network-related IDs in the sequence illustrated by these Reference Points:

* OBDsequenceDiagram\_WIFI.pdf
* Sequence Diagram V10-12-2017.pdf

Specifically, by the time an initial GET /shipments/:id or PUT /shipments/:id/provision request occurs, the GVA will not know the number of tags. At shipment creation, the VP provides the GVA with the shipping unit count but not the tag count. The GVA first learns of the tag count when it receives the tag GUIDs in PUT /shipments/:id/monitor, which would be too late in the provisioning process for the OBT to receive the TAG WSN ID values (e.g. in the response JSON).

The GVA must receive tag WSN IDs during PUT /shipments/:id/monitor when it also receives fields such as tag UUIDs.

# Dates & Times

All dates & times, both sent to the GVA and received from the GVA, should be UTC.

In order to support VP/OBT clients in any time zone, each client will need to convert to UTC when sending a time to the GVA and localize when receiving a time from the GVA.

# Limits

* Request body size: 5 megabytes
  + Rationale: estimated as large enough to support PUT /shipments/:id/photo/:type uploads.

# Errors

## Statuses

This API's error handling responses will focus on these status codes.

* 400
  + If any required JSON element is missing/empty or basic validation checks (e.g. two entity counts are unequal but expected to be equal) fails.
  + If the request attempts to create an entity that already exists, e.g. an ID conflict was detected.
  + If credentials were provided but incomplete.
* 401
  + If credentials were not provided.
* 403
  + If the credentials are otherwise valid but the role associated with them was not granted access to this action.
* 404
  + If the identified resource does not exist (or is not visible to the user based on access grants).
* 409
  + If the request cannot be fulfilled because doing so would conflict with the current state of the resource.
* 500
  + If the GVA fails to process the request for any reason not covered by the above codes.

## Body JSON Structure

* **"requestId"**: <string>
  + Request ID to assist correlation between bug reports and log inspection.
* **"status"**: <number>
  + HTTP status code
* **"code"**: "machine\_readable\_string"
  + Provides application-level codes (versus HTTP codes) to describe the error in more detail and also provide stable strings for conditional logic (as an alternative to regex on human-readable messages).
* **"message"**: "human readable string which is free to change at any time"
  + Provides client developers more information in prose, with the freedom to include things such as URLs to issues/documentation/etc. and also the freedom to change the content at any time.
* **"detail"**: <object>
  + Provides either an empty object or one of arbitrary (likely shallow) depth with additional itemized/organized information to augment the code and message, e.g. names of fields received in the request (as the keys) and their validation errors).

## Common

For each of these codes, the accompanying message/detail fields in the JSON should provide more detail about the specific scenario.

#### bad\_request

This code will be returned in HTTP 400 scenarios.

#### unauthorized

This code will be returned in HTTP 401 scenarios.

#### forbidden

This code will be returned in HTTP 403 scenarios.

#### server\_error

This code will be returned in HTTP 5XX scenarios.

#### input\_validation\_failed

This code will be returned in HTTP 400 scenarios such as:

* Missing/empty fields whose values are required.
* Values with unacceptable formats, lengths, etc. based on criteria such as business rules or database schema constraints.

## Status-related

#### status\_transition\_forbidden

This code will be returned in HTTP 4XX scenarios such as:

* PUT /shipments/:id/provision from OBT X while OBT Y is still in the process of provisioning the same shipment.
* PUT /shipments/:id/receive from OBT X while OBT Y is still in the process of receiving the same shipment.

#### status\_transition\_invalid

This code will be returned in HTTP 4XX scenarios such as:

* PUT /shipments/:id/monitor while the shipment’s status is "new".
* PUT /shipments/:id/accept while the status is "complete".

#### status\_conflict

This code will be returned in HTTP 4XX scenarios such as:

* PUT /shipments/:id/monitor/config while the shipment’s status is not "inMonitoring".

# Endpoints

## POST /session

* Create a login session for the VP or OBT.
* Authorized Role(s)
  + Desk Agent
  + Dock Worker
* Response
  + Success
    - Status: 201
    - The token will be used uniformly by both the VP and OBT.
      * For example, the VP will not pass an HTTP cookie when performing an XHR. It will use the same Authorization header (see "Authentication Flow (VP & OBT)" above) as the OBT.
  + Errors (beyond those in the "Errors > Common" section)
    - None.
* Idempotent
  + Yes. If a session already exists, the currently valid token will be returned instead of a new one. To force the creation of a new one, first use DELETE /session.

|  |
| --- |
| Request Structure |
| {  "username": "Erica",  "password": "SomethingSecure"  } |

|  |
| --- |
| Response Structure |
| {  "token": "YWxhZGRpbjpvcGVuc2VzYW1l"  } |

## DELETE /session

* Log out of the VP or OBT.
* Authorized Role(s)
  + Desk Agent
  + Dock Worker
* Warnings
  + There is only one valid token per user account. If any users are sharing an account, all of them will be logged out by this operation. This is due to the current limitation of the basic authentication implementation. As noted in POST /session, the generated token is only invalidated by this DELETE endpoint, and in the meantime the same token will be returned.
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - None.
* Idempotent
  + Yes.

## GET /session

* Get basic information about the session.
* Authorized Role(s)
  + Desk Agent
  + Dock Worker
* Response
  + Success
    - Status: 200
  + Errors (beyond those in the "Errors > Common" section)
    - None.

|  |
| --- |
| Response Structure |
| {  "username": "Erica"  } |

## POST /shipments

* Creates a new shipment.
* Authorized Role(s)
  + Desk Worker
* Shipment Status Transition
  + Initialize state as "new".
* All IDs described in the "Network-related ID Generation" section are generated by the GVA while processing this request.
* Request
  + For December, additional "shippingUnit" nodes will be added to match the "shippingUnitCount". This allows the OBT to receive a JSON structure with them already represented instead of having to add them in addition to the normal preparation for sending PUT /shipments/:id/monitor.
  + Validation
    - Phone numbers
      * For December, phone number strings will not be validated due to lack of requirements.
      * Post-December, we recommend migration to strict validation using the E.164 standard used by services such as Twilio and AWS Simple Notification Service.
    - shipmentId: None. (API Feedback Discussion Nov 2.docx)
    - uShipmentId: Must be unique across all shipments. (API Feedback Discussion Nov 2.docx)
    - referenceId: None. (API Feedback Discussion Nov 2.docx)
    - shipmentName: None. (API Feedback Discussion Nov 2.docx)
* Response
  + Success
    - Status: 201
  + Errors (beyond those in the "Errors > Common" section)
    - None.
* Idempotent
  + No. GVA will attempt to create a new resource for every request. If a conflicting shipment already exists, an error response will be returned and error logs/events may be stored/emitted.
* Reference Points
  + Castle Canyon Web Portal Wireframes-MVP\_FT-KJ (004).pptx
    - We expect the fields to be collected by the client-side app during the flow on slides 7-10.
    - We expect the HTTP request to happen as a result of a click to the "Save This Shipment" button on slide 10.

|  |
| --- |
| Request Structure |
| {  "shipmentId":"<string>",  "uShipmentId":"<string>",  "shipmentName":"<string>",  "referenceId":"<string>",  "shippingUnitCount": <integer>,  "shipmentNote": "<string>",  "customerName":"<string>",  "customerEmail": "<string>",  "customerAddress": {  "line1":"<string>",  "city":"<string>",  "state":"<string>",  "pin":"<string>",  "country":"<string>",  "phone":"<string>",  },  "earliestPickup":"<YYYY-MM-DD HH:MM:SS>",  "pickupAddress": {  "line1":"<string>",  "city":"<string>",  "state":"<string>",  "pin":"<string>",  "country":"<string>",  "phone":"<string>",  },  "latestDelivery":"<YYYY-MM-DD HH:MM:SS>",  "deliveryAddress": {  "line1":"<string>",  "city":"<string>",  "state":"<string>",  "pin":"<string>",  "country":"<string>",  "phone":"<string>",  },  "tag2GwReportingTime":<integer, # of seconds>,  "gw2CloudReportingTime":<integer, # of seconds>,  "gateways": [  {  "shippingUnits": [  {  "tags": [  {  "thresholds": {  "temperature": {  "min":<float>,  "max":<float>  },  "humidity": {  "min":<float>,  "max":<float>  },  "light": {  "min":<float>,  "max":<float>  },  "pressure": {  "min":<float>,  "max":<float>  },  "tilt": {  "max":<float>  },  "shock": {  "max":<float>  },  "battery": {  "min":<float>  }  }  }  ]  }  ]  }  ]  "geofences": [  {  "type": "source/destination/ocean",  "shape": "circle",  "coordinates": [  {  "latitude": "0.0",  "longitude": "0.0",  "radius": "<kilometers>"  }  ],  "breachAction": [  {  "action": "SMS alert",  "enable": true,  "customerPhone": true,  "pickupPhone": true,  "deliveryPhone": true  },  {  "action": "Enable APM",  "duration": "<minutes>"  },  {  "action": "Calibrate Shipment",  }  ]  }  ]  } |

|  |
| --- |
| **Response Structure** |
| **{**  **"id": <integer>**  **}** |

## GET /shipments

* Return a list of shipments.
  + To allow the VP to provide the Desk Agent with a list of received shipments to review.
  + To allow the OBT to provide the Dock Worker with a list of shipments from to select for a provisioning/receiving flow.
* Authorized Role(s)
  + Desk Agent
  + Dock Worker
* Request
  + Query Arguments
    - status: Filter on the shipment's status.
      * Only return shipments with a status of "new": "?status=new
      * Only return shipments with a status of "new" **OR** "inMonitoring": "?status=new&status=inMonitoring"
* Response
  + Success
    - Status: 200
    - Sort: Most recently created shipments are listed first. Until the monitoring-related requirements of the VP have stabilized, we cannot sort based on any monitoring-related data field.
    - For December, all time-based filtering will be performed by the client to fit its specific use cases (e.g. OBT will only display shipments which align with its own "same-day" logic).
    - See GET /shipments/:id for more information on the internal structure of each shipment’s JSON representation.
  + Errors (beyond those in the "Errors > Common" section)
    - None
* Reference Points
  + Castle Canyon Web Portal Wireframes-MVP\_FT-KJ (004).pptx
    - Slide 19, "All Received Shipments View"
* Pagination is not supported for December due to the low volume of entities.

|  |
| --- |
| Response Structure |
| {  "shipments": [  <object with GET /shipments/:id response structure>,  <object with GET /shipments/:id response structure>,  <object with GET /shipments/:id response structure>  ]  } |

## GET /shipments/:id

* Purpose
  + To allow the VP to provide the Desk Agent with a received shipment's summary to review prior to ending the shipment.
  + To allow the OBT to provide the Dock Worker a shipment's details during the provisioning or receiving flow of the application.
* Authorized Role(s)
  + Desk Agent
  + Dock Worker
* Response
  + Success
    - Status: 200
    - Phone numbers will be provided exactly as entered on the VP. Purely presentational, client-specific transformations are not handled by the GVA.1
  + Errors (beyond those in the "Errors > Common" section)
    - None
  + All IDs in the response JSON (except top-level shipment "id", a database-generated, auto-incrementing, unique integer) are externally defined (e.g. input by user on VP, device/hardware-specific UUIDs, etc.) or generated by the GVA based on specifications (see Network-related ID Generation). They are not values such as database-specific auto-incrementing keys or similar. Rather, they are included in responses to meet a specific use case of at least one client.
  + Telemetry
    - Any value which has not yet been updated will be null.
* Reference Points
  + Castle Canyon Web Portal Wireframes-MVP\_FT-KJ (004).pptx
    - Slide 20, "All Received Shipments View > Individual Shipment Summary View"
  + OBT POC overview and Beta Release documentation v2 .docx
    - Pages 1, 2, 7-9, selection of a shipment by its ID

|  |
| --- |
| Response Structure |
| {  "id": "<string>",  "status":"<string>",  "shipmentId":"<string>",  "uShipmentId":"<string>",  "shipmentName":"<string>",  "referenceId":"<string>",  "shippingUnitCount": <integer>,  "shipmentNote": "<string>",  "customerName":"<string>",  "customerEmail": "<string>",  "customerAddress": {  "line1":"<string>",  "city":"<string>",  "state":"<string>",  "pin":"<string>",  "country":"<string>",  "phone":"<string>",  },  "earliestPickup":"<YYYY-MM-DD HH:MM:SS>",  "pickupAddress": {  "line1":"<string>",  "city":"<string>",  "state":"<string>",  "pin":"<string>",  "country":"<string>",  "phone":"<string>",  },  "latestDelivery":"<YYYY-MM-DD HH:MM:SS>",  "deliveryAddress": {  "line1":"<string>",  "city":"<string>",  "state":"<string>",  "pin":"<string>",  "country":"<string>",  "phone":"<string>",  },  "tag2GwReportingTime":<integer, # of seconds>,  "gw2CloudReportingTime":<integer, # of seconds>,  "photos": {  "documentation": {  "url": "<string>",  "note": "<string>"  },  "proof\_of\_delivery": {  "url": "<string>",  "note": "<string>"  }  },  "telemetry": {  "reportingTime": "<YYYY-MM-DD HH:MM:SS or null>",  "location": {  "latitude": <float or null>,  "longitude": <float or null>  }  },  "gateways": [  {  "id": "<UUID>",  "wsnId": <integer>,  "panId": <integer>,  "channelId": <integer>,  "shippingUnits": [  {  "packageId":"<string>",  "tags": [  {  "id":"<UUID>",  "wsnId: <integer>,  "thresholds": {  "temperature": {  "min":<integer>,  "max":<integer>  },  "humidity": {  "min":<integer>,  "max":<integer>  },  "light": {  "min":<integer>,  "max":<integer>  },  "pressure": {  "min":<integer>,  "max":<integer>  },  "tilt": {  "max":<integer>  },  "shock": {  "max":<integer>  },  "battery": {  "min":<integer>  }  }  }  },  {  "id":"<string>",  "thresholds": {  {  "temperature": {  "min":<integer>,  "max":<integer>  },  "humidity": {  "min":<integer>,  "max":<integer>  },  "light": {  "min":<integer>,  "max":<integer>  },  "pressure": {  "min":<integer>,  "max":<integer>  },  "tilt": {  "max":<integer>  },  "shock": {  "max":<integer>  },  "battery": {  "min":<integer>  }  }  ]  }  ]  }  ]  }  "geofences": [  {  "type": "source/destination/ocean",  "shape": "circle",  "enable": true/false,  "coordinates": [  {  "latitude": "0.0",  "longitude": "0.0",  "radius": "<kilometers>"  }  ],  "breachAction": [  {  "action": "SMS alert",  "enable": true,  "customerPhone": true,  "pickupPhone": true,  "deliveryPhone": true  },  {  "action": "Enable APM",  "duration": "<minutes>"  }  ]  }  ]  } |

## PUT /shipments/:id/provision

* Transition the shipment’s status.
* Authorized Role(s)
  + Dock Worker
* Shipment Status Transition
  + From "new" to "inProvision"
* Shipment Status Lock
  + OBT user acquires the lock.
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - status\_transition\_forbidden
    - status\_transition\_invalid
* Idempotent
  + Yes, if the user was the same one who made the first effective request. GVA will avoid side-effects if the shipment is already in the requested state. GVA will return the same endpoint-standard success response.
  + This api supports OBT and non-OBT flow.. for non-OBT flow the Request Structure is mandatory.
* Reference Points
  + OBT POC overview and Beta Release documentation v2 .docx (page 2)

|  |
| --- |
| Request Structure |
| {  "gateways": ["<UUID>", "<UUID>", …]  } |

## PUT /shipments/:id/provision/reset

* Transition the shipment’s status.
* Authorized Role(s)
  + Dock Worker
* Shipment Status Transition
  + From "inProvision" back to "new" to allow the current Dock Worker to "quit" the shipment before completing the task and allow a different Dock Worker to select it. The GVA does not restrict how many times that cycle can happen.
* Shipment Status Lock
  + OBT user releases the lock.
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - status\_transition\_forbidden
    - status\_transition\_invalid
* Idempotent
  + Yes. GVA will avoid side-effects if the shipment is already in the requested state. GVA will return the same endpoint-standard success response.
* Reference Points
  + propertyStatus property overview.docx

## POST /shipments/:id/provision/verify

* Optional REST API that can be called by the dock worker (OBT), to validate if any of the devices are already in use with an existing shipment
* Authorized Role(s)
  + Dock Worker
* Shipment Status Transition
  + None
* Shipment Status Lock
  + No Change
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - 404 – incase ShipmentID is not found
    - 400 – incase device is used by
* Idempotent
  + No.
* Notes:
  + Both the “gateways” and “tags” field in the JSON structure are mandatory,

|  |
| --- |
| Request Structure |
| {  "gateways": ["<UUID>", "<UUID>", …],  "tags": ["<UUID>", "<UUID>", …]  } |

|  |
| --- |
| Error Response example for 400 |
| {  "status": 400,  "message": "One or more input fields was not accepted.",  "details": {  "12353": "tag [12353] is in use by shipment [ID: 204]",  "12346785-2": "gateway [12346785-2] is in use by shipment [ID: 204]"  },  "code": "input\_validation\_failed",  "requestId": "f2826297-1670-4c9b-9bf0-3c8e5c18a00a"  } |

## PUT /shipments/:id/monitor

* Transition the shipment’s status.
* Authorized Role(s)
  + Dock Worker
* Shipment Status Transition
  + From "inProvision" to "inMonitoring"
* Shipment Status Lock
  + OBT user releases the lock.
* "Tag 0" support
  + For scenarios where the gateway itself is acting as a tag, it will be represented as a tag object under one of the "shippingUnit" nodes. The OBT will set the tag's "id" to the gateway's UUID and the tag's "wsnId" to the gateway's WSN ID.
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - status\_transition\_forbidden
    - status\_transition\_invalid
* Idempotent
  + Yes. GVA will avoid side-effects if the shipment is already in the requested state. GVA will return the same endpoint-standard success response.
* Reference Points
  + propertyStatus property overview.docx (row 4)

|  |
| --- |
| Request Structure |
| {  "gateways": [  {  "id": "<UUID>",  "shippingUnits": [  {  "id": "<tracking ID>",  "tags": [  {  "id": "<UUID>"  "wsnId": <integer>  }  ]  }  ]  }  ]  } |

## PUT /shipments/:id/monitor/config

* Update the shipment's reporting time intervals, sensor thresholds, etc. while it is in the "inMonitoring" status.
* Authorized Role(s)
  + Desk Agent
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - status\_conflict
* Idempotent
  + No. See North-to-South Side Effects below.
* North-to-South Side Effects
  + "Change multiple configuration" command from GVA to GW (JSON Format for the data between North and South.pdf)
* Reference Points
  + propertyStatus property overview.docx

|  |
| --- |
| Request Structure |
| {  "tag2GwReportingTime":<integer, # of seconds>,  "gw2CloudReportingTime":<integer, # of seconds>,  "gateways": [  {  "id": "<UUID>",  "shippingUnits": [  {  "id": "<tracking ID>",  "tags": [  {  "id": "<UUID>"  "thresholds": {  "temperature": {  "min":<integer>,  "max":<integer>  },  "humidity": {  "min":<integer>,  "max":<integer>  },  "light": {  "min":<integer>,  "max":<integer>  },  "pressure": {  "min":<integer>,  "max":<integer>  },  "tilt": {  "max":<integer>  },  "shock": {  "max":<integer>  },  "battery": {  "min":<integer>  }  }  }  ]  }  ]  }  ]  } |

## PUT /shipments/:id/monitor/updatePackageId

* Update the each shipping units Package ID. while it is in the "inMonitoring" status. Typically used in non-OBT flows
* Authorized Role(s)
  + Desk Agent, Dock Worker
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - status\_conflict
* Idempotent
  + None

|  |
| --- |
| Request Structure |
| {     "tagUuid-1":  "desired PackageId",     "tagUuid-2":  "desired  PackageId",     "gwUuid":      "desired  PackageId"  } |

## PUT /shipments/:id/receive

* Transition the shipment’s status.
* Authorized Role(s)
  + Dock Worker
* Shipment Status Transition
  + From "inMonitoring" to "inReceiving"
* Shipment Status Lock
  + OBT user acquires the lock.
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - status\_transition\_forbidden
    - status\_transition\_invalid
* Idempotent
  + Yes, if the user was the same one who made the first effective request/ GVA will avoid side-effects if the shipment is already in the requested state. GVA will return the same endpoint-standard success response.
* Reference Points
  + OBT POC overview and Beta Release documentation v2 .docx (page 2)
  + propertyStatus property overview.docx (row 7)

## PUT /shipments/:id/receive/reset

* Transition the shipment’s status.
* Authorized Role(s)
  + Dock Worker
* Shipment Status Transition
  + From "inReceiving" back to "inMonitoring" to allow the current Dock Worker to "quit" the shipment before completing the task and allow a different Dock Worker to select it. The GVA does not restrict how many times that cycle can happen.
* Shipment Status Lock
  + OBT user releases the lock.
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - status\_transition\_forbidden
    - status\_transition\_invalid
* Idempotent
  + Yes. GVA will avoid side-effects if the shipment is already in the requested state. GVA will return the same endpoint-standard success response.
* Reference Points
  + propertyStatus property overview.docx

## PUT /shipments/:id/receive/accept

* Transition the shipment’s status.
* Authorized Role(s)
  + Dock Worker
* Shipment Status Transition
  + From "inReceiving" to "accepted"
* Shipment Status Lock
  + OBT user releases the lock.
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - status\_transition\_forbidden
    - status\_transition\_invalid
* Idempotent: Yes
  + GVA will avoid side-effects if the shipment is already in the requested state. GVA will return the same endpoint-standard success response.
* Reference Points
  + propertyStatus property overview.docx (row 8)

## PUT /shipments/:id/receive/reject

* Transition the shipment’s status.
* Authorized Role(s)
  + Dock Worker
* Shipment Status Transition
  + From "inReceiving" to "rejected"
* Shipment Status Lock
  + OBT user releases the lock.
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - status\_transition\_forbidden
    - status\_transition\_invalid
* Idempotent: Yes
  + GVA will avoid side-effects if the shipment is already in the requested state. GVA will return the same endpoint-standard success response.
* North-to-South Side Effects
  + "Disassociation request" command from GVA to GW (JSON Format for the data between North and South.pdf)
* Reference Points
  + propertyStatus property overview.docx (row 8)

## PUT /shipments/:id/receive/deinstrument

* Transition the shipment’s status.
* Authorized Role(s)
  + Dock Worker
* Shipment Status Transition
  + From "accepted" to "acceptedDeinstrumented"
  + From "rejected" to "rejectedDeinstrumented"
  + From "completed" to "completedDeinstrumented"
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - status\_transition\_forbidden
    - status\_transition\_invalid
* Idempotent: Yes
  + GVA will avoid side-effects if the shipment is already in the requested state. GVA will return the same endpoint-standard success response.
* Reference Points
  + GVA HTTP API deinstrumentation support.pptx

## PUT /shipments/:id/end

* Transition the shipment’s status.
* Authorized Role(s)
  + Desk Agent via VP
* Shipment Status Transition
  + From "accepted" to "completed"
  + From "acceptedDeinstrumented" to "completedDeinstrumented"
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - status\_transition\_forbidden
    - status\_transition\_invalid
* Idempotent
  + Yes. GVA will avoid side-effects if the shipment is already in the requested state. GVA will return the same endpoint-standard success response.
* Reference Points
  + Castle Canyon Web Portal Wireframes-MVP\_FT-KJ (004).pptx (slide 20)
  + propertyStatus property overview.docx (row 10)

## PUT /shipments/:id/photos/:type

* Create or replace a shipment’s photo and associated note.
* Authorized Role(s)
  + Dock Worker
* Supported types
  + proof\_of\_delivery
  + documentation
* The note is assumed to be a caption for the photo.
  + For example, if a photo already exists and a new one is uploaded with an empty "note" value, a non-empty original note would also be replaced (by the final empty "note" value).
* Warnings
  + This API will not perform any post-processing.
    - The image uploaded by the OBT should be suitable for display on the VP as-is or after basic client-side scaling.
  + Only one total image per shipment will be stored.
  + At the request of the OBT team, for December there will be no validation of the shipment's status before allowing changes to any photo. Rationale:
    - It allows the OBT team to test this feature more easily.
    - The OBT team will need to drive requirements on the validation logic before it is implemented in the GVA.
  + The filenames in URLs are currently based on a hash of the file data. During testing, avoid sharing the input photos with others who are testing. If one shipment's photo is deleted, other shipments which shared the same input photo will then have stale photo URLs that will 404.
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - None.

|  |
| --- |
| Request Structure |
| {  "data": "<Base64 of photo binary>",  "contentType": "<e.g. image/jpeg>",  "note": "<free-form string>"  } |

|  |
| --- |
| **Response Structure** |
| **{**  **"url": "<string>",**  **"action": "<string, 'create' or 'replace'>"**  **}** |

## DELETE /shipments/:id/photos/:type

* Remove the shipment photo and its note.
* Authorized Role(s)
  + Dock Worker
* Supported types
  + See: PUT /shipments/:id/photos/:type
* The note is assumed to be a caption for the photo.
  + Both the photo and its note will be removed.
* At the request of the OBT team, for December there will be no validation of the shipment's status before allowing changes to any photo. Rationale:
  + It allows the OBT team to test this feature more easily.
  + The OBT team will need to drive requirements on the validation logic before it is implemented in the GVA.
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section.
    - None.
* Idempotent
  + Yes. GVA will avoid side-effects if the shipment is already in the requested state. GVA will return the same endpoint-standard success response.

## DELETE /shipments/:id

* Delete a shipment.
* This endpoint is not available by default. The service must be explicitly configured to expose it for development/integration testing purposes.
* Authorized Role(s)
  + Desk Agent
  + Dock Worker
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - None.
* Idempotent: No. A 404 response should be returned after deletion.

## DELETE /shipments

* Delete all shipments.
* This endpoint is not available by default. The service must be explicitly configured to expose it for development/integration testing purposes.
* Authorized Role(s)
  + Desk Agent
  + Dock Worker
* Response
  + Success
    - Status: 204
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - None.
* Idempotent: Yes.

## GET / sensordata /:id

* Retrieve all data received by the cloud from a gw attached to a particular shipment ID. This will include data and command messages for a particular shipment. While it is in the "inMonitoring", “Accepted”, “Rejected”, “Deinstrument” or “End” status. User will need to rely on “messageType” to identify the particular message.
* Authorized Role(s)
  + Desk Agent
* Response
  + Success
    - Status: 200
    - Body: mentioned below.
  + Errors (beyond those in the "Errors > Common" section)
    - status\_conflict

|  |
| --- |
| Response Structure |
| {      "sensordata":[         {            "\_id":<String: mongodb ID>          "receivedTime":<Integer: time in MS>,          "messageType":"com.intel.wsn.sensorData", <msg Type>          "message":{               "time":<String: time in MS>,             "gatewayId":<String: gateway ID>,             "shipmentId":<String>,             "status":<string>,             "messageToken":"token",             "location":{                  "latitude":45.5419135,                "longitude":-122.96147279999998,                "altitude":-1,                "positionUncertainty":1593,                "locationMethod":"GLA-From-GVA",                "timeOfPosition":1522264736030,                "cellTowers":[                     {                        "cellId":14683346,                      "locationAreaCode":33981,                      "mobileCountryCode":310,                      "mobileNetworkCode":410,                      "signalStrength":5                   }                ],                "wifiAccessPoints":[                   ],                "glaPosition":{                     "location":{                        "lat":45.5419135,                      "lng":-122.96147279999998                   },                   "accuracy":1593,                   "timeOfPosition":1522264736030                }             },             "isEncrypted":"no",             "payload":[                  {                     "tagId":"15371",                   "sensorData":[                        {                           "type":"light",                         "isAnalysis":"true",                         "isAnomaly":"false",                         "currentValue":"0",                         "anomalyValue":"0"                      },                      {                           "type":"humidity",                         "isAnalysis":"true",                         "isAnomaly":"false",                         "currentValue":"31",                         "anomalyValue":"0"                      },                      {                           "type":"temperature",                         "isAnalysis":"true",                         "isAnomaly":"false",                         "currentValue":"26",                         "anomalyValue":"0"                      },                      {                           "type":"battery",                         "isAnalysis":"true",                         "isAnomaly":"false",                         "currentValue":"3009",                         "anomalyValue":"0"                      },                      {                           "type":"shock",                         "isAnalysis":"true",                         "isAnomaly":"true",                         "currentValue":"17400",                         "anomalyMinValue":"1000",                         "anomalyMaxValue":"1000",                         "anomalyCount":"1"                      },                      {                           "type":"tilt",                         "isAnalysis":"true",                         "isAnomaly":"false",                         "currentValue":"92",                         "anomalyValue":"0"                      }                   ]                }          }             ]          }       }    ] } |

## POST /gateway/:id/reboot/mqtt

* Reboot a Gateway via the mqtt link.
* Authorized Role(s)
  + Desk Agent
* Response
  + Success
    - Status: 200
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - 204 with reason.

## POST /gateway/:id/reboot/sms

* Reboot a Gateway via the SMS link
* Authorized Role(s)
  + Desk Agent
* Response
  + Success
    - Status: 200
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - 204 with reason.

## POST /gateway/:id/calibrate

* Send Calibrate request message to Gateway
* Authorized Role(s)
  + Desk Agent
* Response
  + Success
    - Status: 200
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - 204 with reason.

## POST /gateway/:id/channelchange/:newchannel

* Send Calibrate request message to Gateway
* Authorized Role(s)
  + Desk Agent
* NewChannel valid input: 11 to 26
* Response
  + Success
    - Status: 200
    - Body: None.
  + Errors (beyond those in the "Errors > Common" section)
    - 204 with reason.