**Klook API Specification**

This guide outlines the API specifications and offers additional resources and recommendations to help developers integrate with Klook's API efficiently.

**About Klook API for Merchants**

This specification is intended for merchants, reservation systems & channel managers who are looking to integrate with Klook.

The Klook API enables seamless integration with a fully-featured tours and experiences booking platform or can be integrated with your existing travel booking system. It offers a comprehensive set of endpoints that support real-time operations for retrieving product information, pricing, and availability.

The API supports the retrieval of product content, availability, and pricing, both in bulk and real-time. It also allows for dynamic pricing calculations for different traveler types and product options. Additionally, it includes functionality to manage availability holds and price locking during booking and cancellation processes.

**Background**

The "Things to Do" market, valued at over $1 trillion globally, represents a significant vertical for Klook. This diverse sector includes Attractions, Tours & Activities, Theme Parks & Water Parks, Food & Beverage, Wi-Fi & SIM Cards, and Transportation Passes. By expanding our offerings within this vertical, we aim to simplify access to unique experiences worldwide.

**Vision**

Our vision for the Open API is to establish a fully connected travel ecosystem, empowering suppliers and channel managers to integrate effortlessly with Klook. Through real-time data exchange and streamlined booking processes, we strive to enhance the customer journey, drive technological innovation, and foster industry-wide collaboration. This API enables our partners to scale efficiently while aligning with our shared commitment to growth and exceptional service in travel.

**Challenges**

While implementing an Open API presents significant opportunities, several challenges must be addressed to ensure seamless integration and optimal performance:

1. **Integration Complexity** Suppliers and channel managers may encounter complexities in aligning their systems with Klook’s platform. Each partner has unique technical requirements, which can introduce delays and elevate development costs. We are committed to providing resources and support to help streamline these integration efforts.
2. **Consistency in Data Management** Ensuring uniformity in data formats and response structures is crucial for smooth operations. Inconsistencies can cause errors in inventory and pricing updates, as well as issues in the booking process. To mitigate this, we prioritize standardization and clear documentation to facilitate accurate data handling.
3. **Performance Monitoring** Reliable and responsive API performance is key to a successful integration. We maintain continuous monitoring of API activity to detect and resolve any downtime or latency issues swiftly, safeguarding the quality of supplier partnerships and end-user satisfaction.

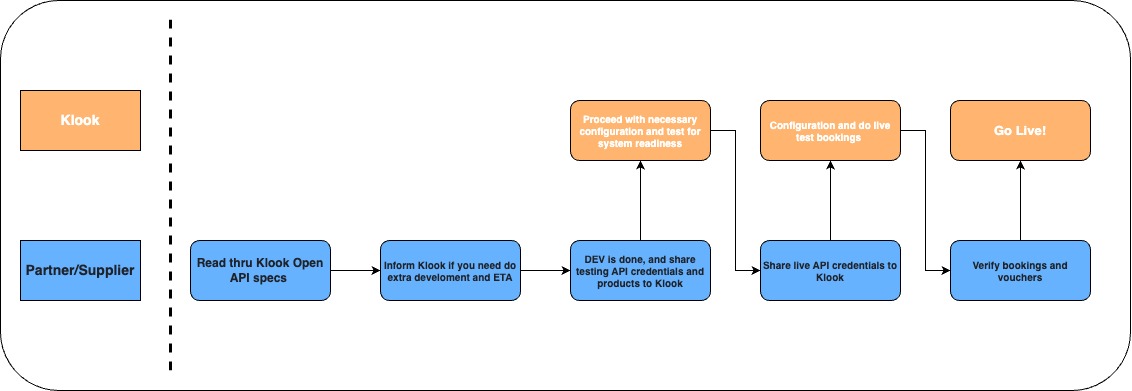
**Opportunities**

The API specification unlocks numerous benefits for suppliers and channel managers, offering new ways to expand their reach and optimize operations:

1. **Enhanced Collaboration** The API fosters a closer collaboration between suppliers and Klook, paving the way for innovative offerings, bundled services, and enriched customer experiences. By working more effectively together, we can create value for all stakeholders in the travel ecosystem.
2. **Expanded Market Reach** Through this API specifications, suppliers can connect to a wide range of distribution channels, including our website, mobile app, Redbook, TikTok, and more. This expanded visibility helps suppliers reach a global audience, driving bookings and boosting revenue from diverse markets worldwide.
3. **Real-Time Inventory Management** With real-time inventory and pricing updates, suppliers can ensure accurate availability across all Klook channels. This reduces the risk of overbooking, enhances operational efficiency, and allows for agile adjustments in response to market demand.
4. **Data-Driven Insights** By leveraging API usage data and customer interaction insights, suppliers and channel managers can gain a deeper understanding of market trends and customer preferences. These insights can inform strategic business decisions and inspire future product innovations.

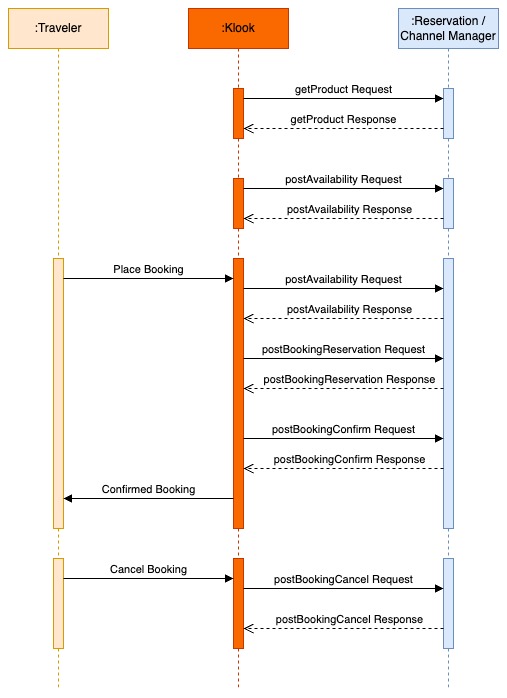
**Integration Process**

**Integration & Go-Live Process**



1. **Review API Specifications:** The partner or supplier reviews Klook's Open API specifications to understand integration requirements.
2. **Confirm Development Needs:** Inform the Klook team if any additional development is necessary, including an estimated time of completion.
3. **Development & Testing Setup:** Once development is completed, share test API credentials and product data with Klook for initial validation. Klook will expect both URL endpoints with the format of **https://*{your endpoint name}*/octo/*{path}*** & ***API keys*** to be generated by you.
4. **System Readiness Configuration** Klook performs necessary configurations and preliminary tests to confirm system compatibility and readiness.
5. **Provide Live Credentials** After successful testing, the supplier provides Klook with live API credentials for final configuration.
6. **Live Test Bookings** Klook conducts live test bookings to confirm end-to-end functionality. This includes testing booking creation, voucher issuance, and system response.
7. **Verify Bookings and Vouchers**
   1. The Klook BD team shares booking details and vouchers (tickets) with the supplier for verification.
   2. The supplier verifies the booking details, including product information, date/timeslots, and participant data. They also scan the voucher’s QR code or barcode to confirm its validity.
8. **Go Live** Once all checks are complete and the live environment is verified, the integration is officially launched.

Overview of API Flow Diagram:



**Glossary of Terms**

OCTO defines core terms that are re-used throughout the specification. Below we include some of the key terms you need to know before getting started. Refer to [Schemas](https://klook.gitbook.io/openapi/getting-started/schemas) for detailed definitions of fields used in the specification.

Term

Difinition

**Reseller**

The reseller, connecting to the Supplier via the API, to further distribute their Products

**Supplier**

The provider of Products, which Reseller is connecting to.

**Product**

The attraction, activity or tour offered by Supplier.

**Option**

A variant of the Product. All products must have at least one option.

**Unit**

The ticket type, e.g. Adult, Child, Senior, etc.

**Unit Item**

A line item per unit within the booking.

**Booking**

A booking made for a specific Product and Option and one or more Unit Items.

**Voucher**

A single admission document (barcode, QRcode, PDF, etc.) that can be used for the entire booking.

**Ticket**

An admission document (barcode, QRcode, PDF, etc.) that can be used per unit item.

**Errors**

Your API should respond to every request with either a 200 OK if everything went ok or 400 Bad Request if it didn't. In the case of the 400 Bad Request, the response body should similar to this:

Copy

{

"error": "INVALID\_PRODUCT\_ID",

"errorMessage": "The Product ID was invalid or missing",

"productId": "123"

}

Error response should always provide error and errorMessage defined as:

FIELD

DESCRIPTION

error

The error code, a table of possible error codes is shown below.

errorMessage

A human readable error message which will be translated depending on the language provided by the Accept-Language header.

Depending on the error code we also may pass additional fields which can make it easier to understand what's wrong with your request. In the example above we provide productId and pass the value that was sent in the request, indicating that the productId of 123 is not valid.

**List of Error Codes**

Below is a list of the error codes and a description of what each means. Further down this page we also provide an example request body for all the error codes that provide additional attributes.

**Headers**

You must supply a set of HTTP headers when making a request. These headers can include both standard HTTP headers as well as some custom OCTO headers. Below are the standard request headers and OCTO custom headers that are required to use:

**Request Headers**

Header

Required

Description

Authorization

Required

Your [Authentication](https://klook.gitbook.io/openapi/getting-started/authentication) Bearer token.

Content-Type

Required

This must be application/json for all POST PATCH and DELETE requests.

Octo-Capabilities

Required

A list of the Capabilities (their IDs) to be included in the response. See [Endpoints and Capabilities](https://klook.gitbook.io/openapi/getting-started/endpoint-and-capabilities) to learn more about Capabilities.

**Response Headers**

Header

Required

Description

Content-Type

Required

application/json

Octo-Capabilities

Required

A list of the Capabilities (their IDs) initialized with your request. See [Endpoints and Capabilities](https://klook.gitbook.io/openapi/getting-started/endpoint-and-capabilities) to learn more about Capabilities.

**Authentication**

OCTO uses Bearer authentication. To authenticate requests, an API key must be sent as a Bearer token in the Authorization [header](https://klook.gitbook.io/openapi/getting-started/headers) of your request:

Note: The API keys will need to be generated by system provider who's integrating to Klook

Copy

GET /supplier HTTP/1.1

Host: {host}

Authorization: Bearer {your\_API\_key}

For security reasons, it's recommended to use a single unique API key per reseller-supplier relationship.

All API requests must be made over HTTPS. Calls made over plain HTTP should fail. API requests without authentication will also fail. If the token is invalid or is deactivated by Supplier a 403 Forbidden error should be returned.

**Endpoint & Capabilities**

A number of [Core Endpoints](https://klook.gitbook.io/openapi/getting-started/endpoint-and-capabilities#octo-api-core-endpoints) that are applicable to most use cases and are **required** for the implementation.

Since not all use cases can be fulfilled with just the core endpoints, OCTO provides a concept of [Capabilities](https://klook.gitbook.io/openapi/getting-started/endpoint-and-capabilities#capabilities), allowing for enhancement to the integrations based on specific needs. Capabilities are **optional** for implementation and to enhance integration by, for example, adding additional information about pricing, content, pickups, etc.

Note: URL endpoints with the format of **https://*{your endpoint name}*/octo/*{path}*** Eg: https://abctourgroup/octo/products

**Open API Core Endpoints**

[**Supplier**](https://klook.gitbook.io/openapi/api-core-mandatory/supplier)

Name

Method

Description

[Get Supplier](https://klook.gitbook.io/openapi/api-core-mandatory/supplier#get-supplier)

GET

Returns a single Supplier and associated details for a given Supplier ID.

[**Products**](https://klook.gitbook.io/openapi/api-core-mandatory/products)

[Get Products](https://klook.gitbook.io/openapi/api-core-mandatory/products#get-products)

GET

Returns a list of Products and associated details.

[Get Product](https://klook.gitbook.io/openapi/api-core-mandatory/products#get-product)

GET

Returns a single Product and associated details for a given Product ID.

[**Availability**](https://klook.gitbook.io/openapi/api-core-mandatory/availability)

[Availability Calendar](https://klook.gitbook.io/openapi/api-core-mandatory/availability#availability-calendar)

POST

Returns availability for a given Product & Option as a single object per day. Optimized to be queried for large date ranges and to populate an availability calendar.

[Availability Check](https://klook.gitbook.io/openapi/api-core-mandatory/availability#availability-check)

POST

Returns availability for a given Product & Option as a single object per start time (or day). You have to perform this step to retrieve an availabilityId required for Bookings.

[**Bookings**](https://klook.gitbook.io/openapi/api-core-mandatory/bookings)

[Booking Reservation](https://klook.gitbook.io/openapi/api-core-mandatory/bookings#booking-reservation)

POST

Creates a booking that reserves the availability (e.g. while you collect payment and contact information from the customer) for a given availabilityId. The booking will remain with the status ON\_HOLD until Booking Confirmation or when the reservation hold expires.

[Booking Confirmation](https://klook.gitbook.io/openapi/api-core-mandatory/bookings#booking-confirmation)

POST

Confirms previously placed Booking Reservation, finalizing the booking and making it ready to be used.

[Booking Cancellation](https://klook.gitbook.io/openapi/api-core-mandatory/bookings#booking-cancellation)

POST

Cancels your booking. You can only cancel a booking if booking.cancellable is TRUE, and is within the booking cancellation cut-off window.

[Get Booking](https://klook.gitbook.io/openapi/api-core-mandatory/bookings#get-booking)

GET

Returns the status and details of your existing booking.

[Get Bookings](https://klook.gitbook.io/openapi/api-core-mandatory/bookings#get-bookings)

GET

Returns the list of the bookings you made for the given filters.

**Capabilities**

Name

ID

Description

[Pricing](https://klook.gitbook.io/openapi/capabilities/pricing)

octo/pricing

Adds pricing to most endpoints giving you advanced static and dynamic pricing capabilities.

octo/content

Extends the core product, option, and unit schemas to provide rich content and images.

Coming Soon

octo/pickups

Adds hotel pickup options to the booking allowing the guest to be picked up and set which pickup location they want to be picked up from.

Coming Soon

octo/webhooks

Allows you to programmatically create webhooks in Supplier's Ventrata system in order to listen out for booking updates or availability updates.

You control which capabilities you want to enable by using the Octo-Capabilities [header](https://klook.gitbook.io/openapi/getting-started/headers). For example:

Copy

GET /availability HTTP/1.1

Host: {host}

Authorization: Bearer {your\_API\_key}

Octo-Capabilities: octo/content, octo/offers

It's also possible to use the \_capabilities query parameter if you're unable to use headers:

Copy

GET /availability?\_capabilities=octo/content,octo/pricing HTTP/1.1

Host: {host}

Authorization: Bearer {your\_API\_key}

You can list all the capabilities with a comma to separate each one.