Clock Base / PMS API

07/01/15	Initial release
25/02/15	Added : Charge Creation method and example; Clock PMS API
12/03/15	Added : Booking API and examples
15/09/15	Added: Custom Payment Purchase
06/11/15	Added: Create Payment; HTTP/S PUSH; New message channels: booking_new, booking_update, booking_expected, booking_checked_in, booking_checked_out, booking_canceled, booking_no_show
06/01/16	Added: Door Key API
13/01/16	Added: Payment Create Form Example,
19/01/16	Removed: Booking.notes Added: Booking note, housekeeping_note, meals_note, client_request Added: Booking ToDos
15/02/16	API Users (Authentication) – help changed.
21/06/16	Added: Rooms – Update housekeeping status and warning Time zone changes: Old account - a new setting for UTC/account time zone. New account are in account time zone only.
27/07/16	Data filtering – more help
29/08/16	Rate restrictions added to Rate and Availability
03/01/17	Fixes on NIL/NOT NIL data filtering examples
07/03/17	Added: POS orders api and new order notification
29/09/17	Added: Booking – Adyen Credit Card Tokens; Folio Close; Folio Fiscalization; RoomStatuses; Booking – Registration Cards; Revenue Categories; Charge Templates/Menu Items; Meals; POS API – Folio Orders, Folio Order Charges
17/10/17	Added: Rate - derived_rate_type, base_rate_id; Root - created_at
13/11/17	Added: occupancy_forecasts end point

Table of Contents

Introduction

Clock **Base** API is a REST API for read-only access to **accounting information** in cloud based products: Clock PMS and Clock POS. Additionally, there is an API for **creating new charges and payments** in folios and methods for **Custom Payment Service** implementation

Clock **PMS** API is a REST API for creating and retrieving bookings, availability and rate requests and product searches from Clock PMS.

Clock **POS** API is a REST API for accessing Orders.

Responses can be rendered in an **XML, JSON or YAML format**. The default format is JSON. Add '.xml', '.yaml' or '.json' at the end of the URL to get the corresponding format.

Message Channels. Clock API can give you a notification for certain events like booking creation, folio closing, etc. Notifications can be received by two methods: **HTTP/S Push** (Clock server makes a POST to a URL on your server) or **Long Pooling** (Your server listens on a URL on our server)

Authentication

To access the API, you should provide a **'user_name'** and **'api_key'** using the 'Digest access authentication' method. Request your 'user_name' and 'api_key' from the system administrator of the Hotel or Restaurant.

This help text is for Clock users.

To give access to your API (full financial information) of your property to a developer, company or product :

- Log in to Clock as 'Administrator' or a subscription owner.
- Go to 'Settings' – 'API Users'.
- Select 'Add'. In the '**User Name**' field, fill in the name of the company, product or developer that will use the API.
- Select the Accounts that the API can access.
- Grant the rights required for performing given operations only. Check each API section for specific right requirements. Granting all rights possible may expose sensitive information to third parties. If you are not sure about the rights needed, do not grant any rights initially.
- 'Save'
- Open the newly created user ('Edit')
- Click the 'Create API key' button.
- Give your developer the 'User Name' and 'API key'.
- Save

To stop the access to your API.

- Open the user you have created for API access purposes.

- If you want to stop the access to a certain account only – uncheck it.
- If you want to stop the access permanently – click 'Destroy API key' and 'Deactivate!'.
- Save

Data formats and standards

DateTime	In JSON and XML, datetime is in ISO 8601 (with time zone): "2001-12-30T06:34:58.623Z" or "2001-12-30T08:34:58.623+02:00" In YAML, a 'space separated', UTC only format is used : "2001-12-30 06:34:58.6236040 Z". All datetimes contains time zone, so they can be converted to any other time zone without data lost.
Date	A date is rendered in ISO 8601: '2001-12-30' All dates are converted from datetime of an event to the accoount time zone and then time is stripped. Example: If a folio is closed on 31 Dec 2001 at 22:00 UTC and your time zone is +3, the folio date of closing will be 01 Jan 2002.
String	Up to 256 characters
Text	No size limitation
Decimal	Precision is set for each attribute. Example: 15,4
Integer	Large-range integer : -9223372036854775808 to 9223372036854775807
Boolean	'true' / 'false'
'*_cents' fields	All money field values are presented in sub-units in an integer format. Divide the value by 100 to get the currency unit value . Example: price_cents: 9091; currency: EUR is a 90.91 EUR. There are several exceptions where 1 unit is not equal to 100 sub-units: BHD, CLF, IQD, JPY, KWD, LYD, MGA, MRO, OMR, TND, VND, VUV. In these cases, use their specific sub-unit ratio.
'*_currency' fields	All currency fields are ISO 4217, three-letter codes.
Country fields	Country fields are ISO 3166-1 alpha-2 (two-letter country code)

API rate limit

The rate limit is 5 calls per second per account (hotel, restaurant). 'HTTP/ 429 Too Many Requests' error will be returned, when this limit is exceeded.
Don't extract the full list of objects. Use filter parameters to get only the data that you need.
Thank you for your cooperation

Using parameters for data filtering

All URLs support parameter passing for filtering the result. Searching by 'Equal' (eq), 'Not equal' (not_eq), 'Greater' (gt), 'Greater and equal' (gteq), 'Less' (lt), 'Less and equal' (lteq), and similar operators are supported. For the full list of operators check this article :
http://rubydoc.info/github/rails/arel/master/Arel/Predications

Searching for an empty and NIL value is also supported.

Examples:

1. Searching for folios closed after a certain date-time using 'Greater or Equal' operator:

base_api/:subscription_id/:account_id/folios?closed_at.gteq=2014-12-30T06:34:58.623Z

Or (any time zone)

base_api/:subscription_id/:account_id/folios?closed_at.gteq=2014-12-30T08:34:58.623+02:00

2. Searching for folios with ID greater than 123:

base_api/:subscription_id/:account_id/folios?id.gt=123

3. Searching for folios in a list of invoice numbers "456, 457, 458":

base_api/:subscription_id/:account_id/folios?
invoice_number[]=456&**invoice_number[]**=457&**invoice_number[]**=458

4. Searching for Open (close_date is NIL), Valid (voided_at is NIL) folios

base_api/:subscription_id/:account_id/folios?close_date.eq&voided_at.eq

5. Search for DoorKeys requested for deletion (deleted_at is NIL and delete_requested_at is NOT NIL)

pms_api/:subscription_id/:account_id/door_keys?deleted_at.eq&delete_requested_at.not_eq

6. Search for Room Type WBE Rates (wbe is True and bookable_type is "Pms::RoomType")

/pms_api/:subscription_id/:account_id/rates?
wbe.eq=true&bookable_type.eq=Pms::RoomType

!! Only attributes at the first object level can be used for filtering.

Clock Base API

URL list

```
root: "/base_api/:s_id/:a_id"
folio_charges: "/base_api/:s_id/:a_id/folios/:folio_id/charges"
folio_charge: "/base_api/:s_id/:a_id/folios/:folio_id/charges/:id"
folio_credit_items: "/base_api/:s_id/:a_id/folios/:folio_id/credit_items"
folio_credit_item: "/base_api/:s_id/:a_id/folios/:folio_id/credit_items/:id"
folio_folio_taxes: "/base_api/:s_id/:a_id/folios/:folio_id/folio_taxes"
folio_folio_tax: "/base_api/:s_id/:a_id/folios/:folio_id/folio_taxes/:id"
close_folio: "/base_api/:s_id/:a_id/folios/:id/close"
folios: "/base_api/:s_id/:a_id/folios"
folio: "/base_api/:s_id/:a_id/folios/:id"
charge_logs: "/base_api/:s_id/:a_id/:revenue_date/charge_logs"
charge_log: "/base_api/:s_id/:a_id/:revenue_date/charge_logs/:id"
credit_item_logs: "/base_api/:s_id/:a_id/:revenue_date/credit_item_logs"
credit_item_log: "/base_api/:s_id/:a_id/:revenue_date/credit_item_logs/:id"
exchange_rates: "/base_api/:s_id/:a_id/exchange_rates"
exchange_rate: "/base_api/:s_id/:a_id/exchange_rates/:id"
users: "/base_api/:s_id/:a_id/users"
user: "/base_api/:s_id/:a_id/users/:id"
closures: "/base_api/:s_id/:a_id/closures"
closure: "/base_api/:s_id/:a_id/closures/:id"
companies: "/base_api/:s_id/:a_id/companies"
company: "/base_api/:s_id/:a_id/companies/:id"
custom_payment_purchase: "/base_api/:s_id/:a_id/custom_payment_purchases/:id"
charge_templates: "/base_api/:s_id/:a_id/charge_templates"
charge_template: "/base_api/:s_id/:a_id/charge_templates/:id"
revenue_categories: "/base_api/:s_id/:a_id/revenue_categories"
revenue_category: "/base_api/:s_id/:a_id/revenue_categories/:id"
statistics: "/base_api/:s_id/:a_id/statistics"

:s_id - Subscription ID, :a_id - Account ID, :id - ID of the object

root url example: 'https://sky-eu1.clock-software.com/base_api/2020/1695.xml'
```

Folios

Folios are the main financial objects in Clock. They are used for recording **charges**, **credit items** (payments) and calculating **taxes** (VAT, etc.).

Name (method)	**URL**	**Description**
Index (GET)	/base_api/:subscription_id/:account_id/ folios	Index of all folios. Returns ID list. Accepts parameters ('folio' attributes) for filtering.
Example **:** `https://sky-eu1.clock-software.com/base_api/2020/1695/folios.xml?id.gt=8888`		
Show (GET)	/base_api/:subscription_id/:account_id/ folios/:id	Single 'folio' instance.
Example **:** `https://sky-eu1.clock-software.com/base_api/2020/1695/folios/8888.xml`		
Close (POST)	/base_api/:subscription_id/:account_id/ folios/:id/**close**	Closing an open folio. The API user needs the right “Folio: Close” granted. Additionally the right “Folio: Close folio with outstanding balance” may be granted depending on business needs. Optional parameter “document_type_id” can be passed to close folio as invoice.

Example (closing as folio) **:**
`curl -H "Content-Type: application/json" -X POST --digest --user your_api_user:your_api_key -d '' https://sky-eu1.clock-software.com/base_api/2020/1695/folios/8888/close.json`
 	
Example (closing as invoice – document_type_id) **:**
`curl -H "Content-Type: application/json" -X POST --digest --user your_api_user:your_api_key -d '{"document_type_id": "1"}' https://sky-eu1.clock-software.com/base_api/2020/1695/folios/8888/close.json`		

Closed status: a folio can be 'open' or 'closed'

- **Open folios** : folio attributes and charges can be changed. An open folio can be considered to be an active customer account or open bill. **'close_date' is nil.**
- **Closed folio** : folio attributes and charges cannot be changed. Closed folios can be considered to be issued documents, an invoice or closed bill. **'close_date' is not nil**.

Validity status: a folio can be 'voided' or 'valid'

- **Voided folios** : invalidated folios by users. Usually these folios are issued by mistake and therefore voided. '**voided_at' is not nil.**

- **Valid folios : 'voided_at' is nil.**

Charges

Charges (Services, Products, etc.) related to a Folio. Charges represent the CURRENT state of attributes for a posted service/product. Previous versions (changes) of charges are not present in the Charges object. Find the full history of charges in the 'Charge Log' object.

Use the POST method with the same URL to create a new charge.

- The right 'Charges: Create' should be granted to the API user.
- For the attributes required for a new charge – check the document “Clock Base API – Appendix Object attributes”.
- Upon the successful charge creation, the new charge itself will be retuned. Otherwise the error code 500 and its description will be returned.
- The Folio 'Closed Status' should be 'Open' and 'Validity status' should be 'Valid', so that charges can be accepted .

Name (method)	**URL**	**Description**
Index (GET)	/base_api/:subscription_id/:account_id/ folios/:folio_id/charges	Index of all folio charges for folio ID.
Example **:** https://sky-eu1.clock-software.com/base_api/2020/1695/folios/8888/charges.xml		
Create (POST)	/base_api/:subscription_id/:account_id/ folios/:folio_id/charges	Creates a new charge in a folio
Example **: See Appendix: Charge Create – form example**		

Modifiers (for each charge record)
Each charge can contain several 'modifiers'. Modifiers are elements INCUDED in the charge price.

Examples:Modifiers in PMS can be the City Tax included in the price but not posted as a separate charge. Modifiers in POS can be a 'Extra cheese' price add-on to a 'Pizza' product.

Often modifiers can have purely a 'Descriptive' meaning and no price. As in POS: 'Medium beaked' used as a steak descriptor.

Credit Items (payments)

Payments posted to a Folio.

Name (method)	**URL**	**Description**
Index	/base_api/:subscription_id/:account_id/ folios/:folio_id/credit_items	Index of all folio payments for folio ID.

(GET)		
Example : https://sky-eu1.clock-software.com/base_api/2020/1695/folios/8888/credit_items.xml		
Create (POST)	/base_api/:subscription_id/:account_id/folios/:folio_id/credit_items	Creates a new Payment in the folio
Required parameters: `value, currency, payment_type.` Optional parameters: `payment_sub_type, reference_id, reference_text, text.` Check Appendix for more information about the attributes and payment types. Check the example POST form		

Taxes

Taxes calculated for this folio (VAT, GST, etc.). There is a separate record for each tax rate.

Name (method)	URL	Description
Index (GET)	/base_api/:subscription_id/:account_id/folios/:folio_id/folio_taxes	Index of folio taxes for folio ID.
Example : https://sky-eu1.clock-software.com/base_api/2020/1695/folios/8888/folio_taxes.xml		

Custom Payment Purchase

With the custom payment service API, hotels can perform their own payment services for collecting payments from their guests. This way a hotel can develop an interface for a payment service provider or bank not supported by the standard Clock PMS payment interfaces.

Use the 'Show' method to find details for a specific purchase.

Use the 'Success' method to confirm the purchase. A correct pms_token parameter is required.

Check the “Clock PMS Payment API” for more information.

Name (method)	URL	Description
Show (GET)	/base_api/:subscription_id/:account_id/custom_payment_purchases/:id	Find each custom payment purchase by ID
Example :		

https://sky-eu1.clock-software.com/base_api/2020/1695/custom_payment_purchases/44.xml		
Success (PUT)	/base_api/:subscription_id/:account_id/ custom_payment_purchases/:id	Complete the purchase. Use this method for successful payments. Required parameters: **pms_token** (found in redirect URL) **transaction_reference** (some id/number from payment service for reference)

Fiscalization

Clock PMS/POS Folio supports 3 dedicated fields for local fiscal requirements (fiscalization). The fields are dedicated for integration with fiscal interfaces. The fields are:

- **fiscal_request_code** – free text field (no length limitations).
  - Printed on the bottom of the folio.
  - The label of the field can be customized by the user in Clock PMS settings.
- **fiscal_response_code** – free text field (no length limitations).
  - Printed on the bottom of the folio.
  - The label of the field can be customized by the user in Clock PMS settings.
  - If this field is not null, the folio is considered as fiscalized by the system.
  - The setting “Wait for folio fiscalization before printing” in Clock PMS/POS, depends on this field also.
- **qr_code_text** – free text field (no length limitations).
  - Printed on the bottom of the folio as QR code image (Clock PMS only).

A push notification “**folio_update”** will be sent on any update of the fields.

Update (PUT)	/base_api/:subscription_id/:account_id/ folios/:id	Updating fiscal folio fields.
Optional parameters: fiscal_request_code, fiscal_response_code, qr_code_text Example: `curl -H "Content-Type: application/json" -X PUT --digest --user your_api_user:your_api_key -d '{"fiscal_request_code": "some_text1", "fiscal_response_code": "some_text2", "qr_code_text": "some_text3_or_url"}' https://sky-eu1.clock-software.com/base_api/2020/1695/folios/8888.json`		

Operation Log

The Operation log objects represent all changes to a charge or payment in incremental chronology. All charge or payments reports in Clock are based on the charge/payment log.

Logs are accessible for a certain revenue date.

Charge Log

Chronology of changes to charges. The structure of a charge log is the same as the charge itself.

Name (method)	URL	Description
Index (GET)	/base_api/:subscription_id/:account_id/ :revenue_date/charge_logs	All charge changes for a certain revenue date.
Example : https://sky-eu1.clock-software.com/base_api/2020/1695/ 2014-12-30/charge_logs.xml		

Credit Item Log (payments log)

Chronology of changes to payments . The structure of a credit log is the same as the credit item itself.

Name (method)	URL	Description
Index (GET)	/base_api/:subscription_id/:account_id/ :revenue_date/credit_item_logs	All payment changes for a certain revenue date.
Example : https://sky-eu1.clock-software.com/base_api/2020/1695/ 2014-12-30/credit_item_logs.xml		

Other

Companies

List of companies registered.

The basic company contact info is located in the 'contact_info' object.

The Billing info information for invoicing purposes is located in the 'billing_info' object.

Name (method)	URL	Description
Index (GET)	/base_api/:s_id/:a_id/companies	Index of all companies. It returns an ID list. It can be filtered by parameters ('company' attributes).

Example **:** `https://sky-eu1.clock-software.com/base_api/2020/1695/companies.xml?updated_at.gt=2014-10-17T12:16:37Z`		
Show (GET)	/base_api/:s_id/:a_id/companies/:id	Single 'company' instance.
Example **:** `https://sky-eu1.clock-software.com/base_api/2020/1695/companies/99.xml`		

Exchange rates

Current exchange rates of all currencies supported by the subscription.

Name (method)	**URL**	**Description**
Index (GET)	/base_api/:subscription_id/:account_id/credit_item_logs	All currencies and rates
Example **:** `https://sky-eu1.clock-software.com/base_api/2020/1695/exchange_rates.xml`		

Users

Users of the system.

Name (method)	**URL**	**Description**
Index (GET)	/base_api/:subscription_id/:account_id/users	All users
Example **:** `https://sky-eu1.clock-software.com/base_api/2020/1695/users.xml`		
Show (GET)	/base_api/:subscription_id/:account_id/users/:id	One User by ID
Example **:** `https://sky-eu1.clock-software.com/base_api/2020/1695/users/66.xml`		

Closures

Cashier or Day (POS) closures performed. To get the data between two closures: find created_at for each closure and use as a filter for folios, charge_log or credit_item_log.

Name (method)	URL	Description
Index (GET)	/base_api/:subscription_id/:account_id/closures	Cashier/Day closures
Example : https://sky-eu1.clock-software.com/base_api/2020/1695/closures.xml?created_at.gt=2014-10-17T12:16:37Z		

Revenue Categories

A list of all revenue categories in the account.

Name (method)	URL	Description
Index (GET)	/base_api/:subscription_id/:account_id/revenue_categories	All revenue categories
Example : https://sky-eu1.clock-software.com/base_api/2020/1695/revenue_categories.xml		
View (GET)	/base_api/:subscription_id/:account_id/revenue_categories/:id	A revenue group by ID.
Example : https://sky-eu1.clock-software.com/base_api/2020/1695/revenue_categories/345.xml		

Example response:

```
<object>
  <id type="integer">345</id>
  <name>SPA</name>
  <revenue_group>extra</revenue_group>
  <created_at type="dateTime">2014-09-11T09:29:24+03:00</created_at>
  <updated_at type="dateTime">2014-09-11T09:29:24+03:00</updated_at>
  <subscription_id type="integer">2</subscription_id>
</object>
```

Charge Templates / Menu Items

Charge templates (PMS) / Menu Items (POS) for an account.

Name (method)	URL	Description
Index (GET)	/base_api/:subscription_id/:account_id/charge_templates	All charge templates
Example : https://sky-eu1.clock-software.com/base_api/2020/1695		

/charge_templates.xml		
View (GET)	/base_api/:subscription_id/:account_id/ charge_templates/:id	A charge template by ID.

Example : https://sky-eu1.clock-software.com/base_api/2020/1695/charge_templates/456.xml

Example response:

```
<object>
  <id type="integer">456</id>
  <text>Parking</text>
  <revenue_group>other</revenue_group>
  <visual_group_text/>
  <plain_price_cents type="integer">400</plain_price_cents>
  <currency>EUR</currency>
  <tax_rate type="float">0.2</tax_rate>
  <account_id type="integer">2</account_id>
  <created_at type="dateTime">2017-05-05T13:35:40+03:00</created_at>
  <updated_at type="dateTime">2017-09-27T16:26:06+03:00</updated_at>
  <store_id nil="true"/>
  <inventory_code>PARKING</inventory_code>
  <revenue_category>PARKING</revenue_category>
  <tax_code/>
  <qty nil="true"/>
  <print_text/>
  <default_order_group nil="true"/>
  <sort_order type="integer">2</sort_order>
  <color>#ffffff</color>
  <capacity type="integer">20</capacity>
</object>
```

Clock PMS API

URL list

root: /pms_api/:s_id/:a_id

default_booking_folios:
"/pms_api/:s_id/:a_id/bookings/:booking_id/folios/default"
booking_folios: "/pms_api/:s_id/:a_id/bookings/:booking_id/folios"
booking_booking_self_service_to_dos:
"/pms_api/:s_id/:a_id/bookings/:booking_id/booking_self_service_to_dos"
booking_booking_self_service_to_do:
"/pms_api/:s_id/:a_id/bookings/:booking_id/booking_self_service_to_dos/:id"
booking_registration_cards:
"/pms_api/:s_id/:a_id/bookings/:booking_id/registration_cards"
booking_registration_card:
"/pms_api/:s_id/:a_id/bookings/:booking_id/registration_cards/:id"
booking_adyen_tokens: "/pms_api/:s_id/:a_id/bookings/:booking_id/adyen_tokens"
bookings: "/pms_api/:s_id/:a_id/bookings"
booking: "/pms_api/:s_id/:a_id/bookings/:id"
room_type_availability_adjustments:
"/pms_api/:s_id/:a_id/room_types/:room_type_id/availability_adjustments"
room_types: "/pms_api/:s_id/:a_id/room_types"
room_type: "/pms_api/:s_id/:a_id/room_types/:id"
room_bookings: "/pms_api/:s_id/:a_id/rooms/:room_id/:date/bookings"
rooms: "/pms_api/:s_id/:a_id/rooms"
room: "/pms_api/:s_id/:a_id/rooms/:id"
rate_rate_prices: "/pms_api/:s_id/:a_id/rates/:rate_id/rate_prices"
rates: "/pms_api/:s_id/:a_id/rates"
rate: "/pms_api/:s_id/:a_id/rates/:id"
rates_availability: "/pms_api/:s_id/:a_id/rates_availability"
products: "/pms_api/:s_id/:a_id/products"
config: "/pms_api/:s_id/:a_id/config"
door_keys: "/pms_api/:s_id/:a_id/door_keys"
door_key: "/pms_api/:s_id/:a_id/door_keys/:id"
phone_calls: "/pms_api/:s_id/:a_id/phone_calls"
room_statuses: "/pms_api/:s_id/:a_id/room_statuses"
meals: "/pms_api/:s_id/:a_id/meals"

:s_id - Subscription ID, :a_id - Account ID, :id - ID of the object

root url example: 'https://sky-eu1.clock-software.com/pms_api/2020/1695.xml'

Booking

Booking

Name (method)	URL	Description
Index (GET)	/pms_api/:subscription_id/:account_id/ bookings	Index of all bookings. Returns ID list. Accepts parameters ('booking' attributes) for filtering.
Example **:** `https://sky-eu1.clock-software.com/pms_api/2020/1695/bookings.xml?id.gt=8888`		
Show (GET)	/pms_api/:subscription_id/:account_id/ bookings/:id	Single 'booking' instance.
Example **:** `https://sky-eu1.clock-software.com/pms_api/2020/1695/bookings/8888.xml`		
Create (POST)	pms_api/:subscription_id/:account_id/ bookings	Creates a new booking. Check bellow for “notes” parameters. The rights: 'Booking: Create or Edit' and 'Charges: Create' should be granted to the API user.
Example: `curl -H "Content-Type: application/json" -X POST --digest --user` **`your_api_user:your_api_key`** `-d '{"booking": {"arrival": "2015-10-31", "departure": "2015-11-05", "arrival_room_type_id": "39", "arrival_room_id": "2160", "guest_last_name": "Smith", "guest_e_mail": "smith@test.com", "guest_phone_number": "442 987 6574"}}'` **`https://sky-eu1.clock-software.com/pms_api/2020/1695.json`** Example HTML form: Check the Appendix.		
Update (PUT)	/pms_api/:subscription_id/:account_id/ bookings/:id	Updates an existing booking. Check bellow for “notes” parameters.
Example (update reference_number and guest_language): `curl -H "Content-Type: application/json" -X PUT --digest --user` **`your_api_user:your_api_key`** `-d '{"booking": {"reference_number": "1234567890", "guest_language": "en"}}'` **`https://sky-eu1.clock-software.com/pms_api/2020/1695/bookings/8888.json`**		

Example (update meals):

```
curl -H "Content-Type: application/json" -X PUT --digest --user your_api_user:your_api_key -d '{"booking": { "meals":[{"id":8},{"id":9}] }}' https://sky-eu1.clock-software.com/pms_api/2020/1695/bookings/8888.json
```

Example (remove all meals):

```
curl -H "Content-Type: application/json" -X PUT --digest --user your_api_user:your_api_key -d '{"booking": { "meals":[{}] }}' https://sky-eu1.clock-software.com/pms_api/2020/1695/bookings/8888.json
```

Example HTML form: Check the Appendix.

Status: The Booking status can be: 'expected', 'checked_in', 'checked_out', 'canceled' or 'no_show'. Bookings having the last two statuses ('canceled' or 'no_show') are considered not to be valid bookings. Such bookings are not taken into availability. The status is fully reversible and can be changed at any time (examples: from 'cancelled' to 'expected'; from 'checked_out' to 'checked_in', etc.)

Notes: In order to add a note to the booking use the following parameters in POST/PUT methods: *booking[note], booking[housekeeping_note], booking[meals_note], booking[client_request]*

Default Booking Folio

Use this booking's method to get a single open, valid folio suitable for charging.

Name (method)	**URL**	**Description**
Index (GET)	/pms_api/:subscription_id/:account_id/ bookings/:booking_id/folios/default	Returns the default folio (for more details - check Folio object and object attributes from Base API)
Example **:** `https://sky-eu1.clock-software.com/pms_api/2020/1695/bookings/8888/folios/default.yaml`		

Booking Folios

Use this Booking method to get all booking folios. Keep in mind that some of them can be voided (not valid), empty or closed ones.

Name (method)	**URL**	**Description**
Index	/pms_api/:subscription_id/:account_id/	Returns ID list of booking folios (for

(GET)	bookings/:booking_id/folios	more details - check Folio object and object attributes from Base API)
Example **:** `https://sky-eu1.clock-software.com/pms_api/2020/1695/bookings/8888/folios.yaml`		

Bookings by Room

Use this Room method to get bookings related to a Room on a selected Date. Room changes (changes of a room associated with a booking) are considered by this method too. Additional parameter '**show_departures**' can be passed to include bookings departing on selected date.

Name (method)	**URL**	**Description**
Index (GET)	/pms_api/:subscription_id/:account_id/ rooms/:room_id/:date/bookings	Returns bookings

Example **:**
`https://sky-eu1.clock-software.com/pms_api/2020/1695/rooms/36/2015-05-01/bookings.yaml`
Example - Departures included:
`https://sky-eu1.clock-software.com/pms_api/2020/1695/rooms/36/2015-05-01/bookings.yaml?show_departures`		

Booking ToDos

ToDos assigned to the booking. ToDos are shown in the home screen.

Name (method)	**URL**	**Description**
Index (GET)	/pms_api/:subscription_id/:account_id/ bookings/:booking_id/ booking_self_service_to_dos	Index of all booking's todos.
Example **:** `https://sky-eu1.clock-software.com/pms_api/2020/1695/bookings/8888/booking_self_service_to_dos.xml`		
Show (GET)	/pms_api/:subscription_id/:account_id/ bookings/:booking_id/ booking_self_service_to_dos/:id	Single 'todo' instance.
Example **:**		

`https://sky-eu1.clock-software.com/pms_api/2020/1695/bookings/8888/booking_self_service_to_dos/1234.xml`		
Create (POST)	/pms_api/:subscription_id/:account_id/ bookings/:booking_id/ booking_self_service_to_dos	Creates a todo.
Required parameters: self_service_to_do[text] **Optional parameters:** self_service_to_do[date]		

Booking – Adyen Credit Card Tokens

Use this end-point to add an Adyen tokenized credit card to a Booking. Adyen token is an identification of a credit card priorly stored in Adyen Payment Gateway by you. The identification is consist by two parameters: "**shopperReference**" (Clock param: "**token**") and "**shopperEmail**" (Clock param: "**email**"). More information about Adyen API for Tokenization (Recurring contract): **Here** or **Here** . For Adyen test account, contact Adyen support.

Name (method)	**URL**	**Description**
Create (POST)	/pms_api/:subscription_id/:account_id/ bookings/:booking_id/adyen_tokens	Adding an Adyen tokenized card to a Booking.
Required parameters: token; email **Example:** `curl -H "Content-Type: application/json" -X POST --digest --user your_api_user:your_api_key -d '[{"token":"adyen_token_32_symbols_long","email":"shoper@email.com"}]' https://sky-eu1.clock-software.com/pms_api/2020/1695/bookings/8888/adyen_tokens.json`		

Booking – Registration Cards

Registration Cards for a Booking. Custom defined filed are included also.

This end point requires the "Registration Card: View personal information" right granted to the API user.

Name (method)	**URL**	**Description**
Index (GET)	/pms_api/:subscription_id/:account_id/ bookings/:booking_id/registration_cards	Registration cards added to a booking.
Example **:**		

https://sky-eu1.clock-software.com/pms_api/2020/1695/bookings/8888/registration_cards.xml

Example response:

```
<object type="array">
  <object>
    <id type="integer">316629</id>
    <id_number>some ID</id_number>
    <birth_country>AU</birth_country>
    <birth_city>Birth City</birth_city>
    <age>30</age>
    <document_type>Document</document_type>
    <document_number>Number</document_number>
    <document_issued_by>Issued by</document_issued_by>
    <visa_type>Type</visa_type>
    <checkpoint_name>Checkpoint</checkpoint_name>
    <next_destiantion>Next</next_destiantion>
    <car>Car</car>
    <car_number>No</car_number>
    <visit_purpose>Purpose</visit_purpose>
    <workplace_company>Company</workplace_company>
    <workplace_position>Position</workplace_position>
    <accompany>With</accompany>
    <complete type="boolean">true</complete>
    <birth_date_str>2017-09-26</birth_date_str>
    <document_issue_date_str>2017-09-26</document_issue_date_str>
    <document_expire_date_str>2017-09-26</document_expire_date_str>
    <visa_issue_date_str>2017-09-26</visa_issue_date_str>
    <visa_expire_date_str>2017-09-26</visa_expire_date_str>
    <checkpoint_date_str>2017-09-26</checkpoint_date_str>
    <gender_code>M</gender_code>
    <number nil="true"/>
    <required_images nil="true"/>
    <custom>
      <text>Text</text>
      <info_two>Info 2</info_two>
      <my_text>my text</my_text>
    </custom>
    <contact_info>
      <first_name>First</first_name>
      <middle_name>Middle</middle_name>
      <last_name>Last</last_name>
      <country>AF</country>
      <city>City</city>
      <address>Address</address>
      <zip_code>postcode</zip_code>
      <phone_number>Phone</phone_number>
      <e_mail>email@gmail.com</e_mail>
      <fax_number>Fax</fax_number>
      <language>en</language>
      <state>State</state>
      <person_title_id type="integer">2</person_title_id>
    </contact_info>
  </object>
</object>
```

Rate and Availability Requests

Occupancy forecast

This request consumes a high amount of application resources. Please use it with caution.

The Occupancy Forecast is used to display room type (including virtual room types if any) availability day-by-day for selected period. The same object is used by “Occupancy Forecast” screen in Clock PMS.

Request: **Period** (length of the period is restricted to maximum of 365 days)

Result: For each **Room Type** and each **Date**, you can get:

- **capacity**. Physical rooms in hotel.
- **adjustment**. Manually added or subtracted rooms to/from capacity.
- **oos**. Out of Service rooms.
- **bookings**. Rooms occupied by Bookings.
- **blocks**. Rooms blocked for events.
- **free**. Free rooms = Capacity -/+ Adjustment – OOS – Bookings – Blocks.

Calculated **total** for each date:

- **total_occupied**. Occupied rooms for all room types.
- **total_occupied_percentage**. Percentage of occupation.
- **total_free**. Free rooms for all room types.

Name (method)	**URL**	**Description**
Index (GET)	/pms_api/:subscription_id/:account_id/occupancy_forecasts	Required parameters: • from: Date • to: Date
Example request **:** `https://sky-eu1.clock-software.com/pms_api/2020/1695/occupancy_forecasts.yaml?range[from_date]=2017-12-01&range[to_date]=2017-12-03`		

Example respond (yaml) :
`:forecasts:`
`- :room_type: Double`
`  :results:`
`  - :date: 2017-12-01`
`  :capacity: 68`
`  :adjustment: 0`
`  :oos: 1`
`  :bookings: 0`		

```
    :blocks: 0
    :free: 67
  - :date: 2017-12-02
    :capacity: 68
    :adjustment: 0
    :oos: 1
    :bookings: 0
    :blocks: 0
    :free: 67
- :room_type: Suite
  :results:
  - :date: 2017-12-01
    :capacity: 10
    :adjustment: 0
    :oos: 0
    :bookings: 0
    :blocks: 0
    :free: 10
  - :date: 2017-12-02
    :capacity: 10
    :adjustment: 0
    :oos: 0
    :bookings: 0
    :blocks: 0
    :free: 10
:totals:
- :date: 2017-12-01
  :total_occupied: 0
  :total_occupied_percentage: 0%
  :total_free: 77
- :date: 2017-12-02
  :total_occupied: 0
  :total_occupied_percentage: 0%
  :total_free: 77
```

Rate and Availability

This request consumes a highest amount of application resources. Please use it with caution.

The Rate and Availability Request is used by Clock PMS to display free rooms and prices for a selected period. The Calendar in the WRS is based on this request.

Request for : **Period, Selected Rates, Selected Rooms or Room Types**

Result: For each Room or Room Type / For each Rate / For each Date:

- free: Boolean. General conclusion about the availability, the price and the restrictions.
- price: cents (integer) and currency. Calculated price if any.
- room_type_free_rooms: Integer. How many free rooms for related room type has.
- errors: Explanation about restriction validations.
- rate_restriction: All restrictions set for this rate/date. Restrictions are:
  - min_adults – Integer
  - min_children – Integer
  - max_adults – Integer
  - max_children – Integer

- min_stay – Integer
- max_stay – Integer
- min_free_rooms – Integer
- max_free_rooms – Integer
- close_for_arrival – Boolean
- close_for_departure – Boolean
- stop_from_sale – Boolean
- min_days_before_arrival – Integer
- max_days_before_arrival – Integer
- active_from_date – Date
- active_to_date - Date

Name (method)	**URL**	**Description**
Index (GET)	/pms_api/:subscription_id/:account_id/rates_availability	Required parameters: • from: Date • to: Date • rates: Array of Rate IDs • room_types: Array of Room Type IDs • rooms: Array of Room IDs Depending of you needs use ether 'room_types' or 'rooms'. One of them is required.

Example request **:**

```
https://sky-eu1.clock-software.com/pms_api/2020/1695/rates_availability.yaml
?from=2015-05-01
&to=2015-05-02
&rates[]=859
&rates[]=861
&room_types[]=39
&room_types[]=82
```

Example respond (yaml) :

```
- :type: RoomType
  :id: 39
  :rates:
    859:
      2015-05-01:
        :free: true
        :price:
          :cents: 10000
          :currency: EUR
        :room_type_free_rooms: 10
        :errors: {}
        :rate_restriction:
          max_adults: 2
          max_children: 0
          min_stay: 2
          max_stay:
```

```
          min_free_rooms:
          max_free_rooms:
          close_for_arrival: false
          close_for_departure: false
          stop_from_sale: false
          min_adults:
          min_children:
          min_days_before_arrival:
          max_days_before_arrival:
          active_from_date:
          active_to_date:
active_to_date:
      2015-05-02:
        :free: true
        :price:
          :cents: 10000
          :currency: EUR
        :room_type_free_rooms: 10
        :errors: {}
        :rate_restriction:
          max_adults: 2
          max_children: 0
          min_stay: 2
          max_stay:
          min_free_rooms:
          max_free_rooms:
          close_for_arrival: false
          close_for_departure: false
          stop_from_sale: false
          min_adults:
          min_children:
          min_days_before_arrival:
          max_days_before_arrival:
          active_from_date:
          active_to_date:
- :type: RoomType
  :id: 82
  :rates:
    861:
      2015-05-01:
        :free: true
        :price:
          :cents: 15000
          :currency: EUR
        :room_type_free_rooms: 2
        :errors: {}
        :rate_restriction:
          max_adults:
          max_children:
          min_stay:
          max_stay:
          min_free_rooms:
          max_free_rooms:
          close_for_arrival: false
          close_for_departure: false
          stop_from_sale: false
          min_adults:
          min_children:
          min_days_before_arrival:
          max_days_before_arrival:
          active_from_date:
          active_to_date:
      2015-05-02:
```

```
        :free: true
        :price:
          :cents: 15000
          :currency: EUR
        :room_type_free_rooms: 2
        :errors: {}
        :rate_restriction:
          max_adults:
          max_children:
          min_stay:
          max_stay:
          min_free_rooms:
          max_free_rooms:
          close_for_arrival: false
          close_for_departure: false
          stop_from_sale: false
          min_adults:
          min_children:
          min_days_before_arrival:
          max_days_before_arrival:
          active_from_date:
          active_to_date:
```

Products

The Product search is used by Clock PMS to present available products for certain Arrival,Departure, etc. The Product Search evaluates rate restrictions and present available options for a booking. The s**econd** page in Clock PMS is based on the Product Search.

Request: For Arrival and Departure and Selected Rates. Additional filters can be set for Room, Room Types, Adults, Children, etc.

Response: For each Room Type (or Room) / For each Rate:

- room_type_free_rooms: Integer. Number of free rooms available for the respective Room Type.
- available: Boolean. General conclusion about the product availability.
- price (cents, currency). Price of the product
- errors: Text. Restrictions validation result.

Name (method)	**URL**	**Description**
Index (GET)	/pms_api/ :subscription_id/ :account_id/ products	Required parameters: • product_search[arrival]: Date • product_search[departure]: Date • rates: Array of Rate IDs Additional params: • product_search[adult_count]: Integer • product_search[children_count]: Integer • product_search[room_type_id]: Array of Room Type IDs • product_search[room_id]: Array of Room IDs

		• product_search[bonus_code]: String • product_search[room_count]: Integer. Number of free rooms needed. • show_rooms: If a parameterused, the result will be returned for each of the rooms, instead of room types.

Example request **:**

```
https://sky-eu1.clock-software.com/pms_api/2020/1695/products.yaml
?rates[]=859
&rates[]=861
&product_search[arrival]=2015-05-01
&product_search[departure]=2015-05-10
&product_search[adult_count]=2
&product_search[children_count]=1
&product_search[room_type_id][]=39
&product_search[room_type_id][]=82
&product_search[bonus_code]=SUPEROFFER2
&product_search[room_count]=1
```

Example respond (yaml) :

```
- :type: RoomType
  :id: 39
  :rates:
    859:
    - :room_type_free_rooms: 10
      :available: true
      :price:
        :cents: 90000
        :currency: EUR
      :errors: {}
- :type: RoomType
  :id: 82
  :rates:
    861:
    - :room_type_free_rooms: 2
      :available: true
      :price:
        :cents: 135000
        :currency: EUR
      :errors: {}
```

Room Statuses

The Room Status Request is used by Clock PMS to display a list of free room numbers for a period. Free room numbers are used for the room allocation process (allocating room numbers to bookings). Prior to the room number allocation, the Booking has a room type only (e.g. “Double”). After the allocation, the Booking is linked to a room number (101 / Double for example). A room is available if there are no bookings for the period and its status is OK and not OOS (Out Of Service) or Disabled (Not active).

Request for : **From, To.** Please note, it is not about Arrival and Departure dates. The request refers to the dates of nights in interest (including both From and To dates). **Room Type** (optional).

Result: For each **Room Type** / For each **Room**:

- id: Integer. ID of the Room.
- number: String. Number/Name of the room (example: '101')
- available: Boolean. true – The Room is free/available for allocation. False – The Room is occupied, OOS or disabled.
- housekeeping_status: String. Empty status means – Clean. Check Room attributes list for more information.
- housekeeping_warning: String. Free text for hotel/staff usage. Not to be displayed to the guests/clients.

Name (method)	**URL**	**Description**
Index (GET)	/pms_api/ :subscription_id/ :account_id/ room_statuses	Required parameters: • from: Date • to: Date Optional parameters: • room_type_id: Integer

Example request (overnight stay on January 1) **:**

```
https://sky-eu1.clock-software.com/pms_api/2020/1695/room_statuses.yaml
?from=2018-01-01
&to=2018-01-01
```

Example respond (yaml) :

```
- room_type_id: 1
  room_type: Double
  rooms:
  - id: 65
    number: '103'
    available: false
    housekeeping_status: ''
    housekeeping_warning: ''
  - id: 64
    number: '102'
    available: true
    housekeeping_status: checked_out
    housekeeping_warning: ''
  - id: 66
    number: '104'
    available: true
    housekeeping_status: inspect
    housekeeping_warning: 'TV remote control battery'
- room_type_id: 2
  room_type: Suite
  rooms:
  - id: 166
    number: 10A
    available: true
    housekeeping_status:
    housekeeping_warning:
```

Other

Room Type

Room (or Unit) Types defined in the account.

Name (method)	**URL**	**Description**
Index (GET)	/pms_api/:subscription_id/:account_id/ room_types	Index of all room type in the account.
Example **:** `https://sky-eu1.clock-software.com/pms_api/2020/1695/room_types.xml`		
Show (GET)	/pms_api/:subscription_id/:account_id/ room_types/:id	Single room type
Example **:** `https://sky-eu1.clock-software.com/pms_api/2020/1695/room_types/2.xml`		

Rooms

Room (Apartments, Beds, Villas, Units) defined in the account. For 'Bookings by Room' see Booking section.

Name (method)	**URL**	**Description**
Index (GET)	/pms_api/:subscription_id/:account_id/ rooms	Index of all rooms (apartments, beds, villas, units) in the account.
Example **:** `https://sky-eu1.clock-software.com/pms_api/2020/1695/rooms.xml`		
Show (GET)	/pms_api/:subscription_id/:account_id/ rooms/:id	Single room instance
Example **:** `https://sky-eu1.clock-software.com/pms_api/2020/1695/rooms/101.xml`		
Update (PUT)	/pms_api/:subscription_id/:account_id/ rooms/:id	Update Housekeeping status and Housekeeping warning

Optional parameters: room[housekeeping_warning], room[housekeeping_status] Values for housekeeping_status: If empty, the room is clean and ready. Else, one of the following : 'inspect', 'progress', 'checked_out'. The Checked Out status is set automatically on the Booking Checkout.

Rates

List of Rates defined in the account.

Name (method)	**URL**	**Description**
Index (GET)	/pms_api/:subscription_id/:account_id/ rates	Index of all rates in the account.
Example **:** `https://sky-eu1.clock-software.com/pms_api/2020/1695/rates.xml`		
Show (GET)	/pms_api/:subscription_id/:account_id/ rates/:id	Single rate instance
Example **:** `https://sky-eu1.clock-software.com/pms_api/2020/1695/rates/66.xml`		

Meals

A list of all meals in the account.

Name (method)	**URL**	**Description**
Index (GET)	/base_api/:subscription_id/:account_id/ meals	All meals
Example **:** https://sky-eu1.clock-software.com/base_api/2020/1695/meals.xml		

Example response:

```
<object type="array">
  <object>
    <id type="integer">8</id>
    <name>Breakfast</name>
    <order_tag type="integer">1</order_tag>
    <standard type="boolean">true</standard>
    <first_meal type="boolean">false</first_meal>
    <created_at type="dateTime">2013-09-18T15:33:46+03:00</created_at>
    <updated_at type="dateTime">2013-09-18T15:33:46+03:00</updated_at>
  </object>
```

```
  <object>
    <id type="integer">9</id>
    <name>Dinner</name>
    <order_tag type="integer">2</order_tag>
    <standard type="boolean">true</standard>
    <first_meal type="boolean">false</first_meal>
    <created_at type="dateTime">2013-09-18T15:33:55+03:00</created_at>
    <updated_at type="dateTime">2013-09-18T15:33:55+03:00</updated_at>
  </object>
</object>
```

Door Keys

With the Clock Door Key API, hotels can connect a custom door key system to Clock PMS. Using this API and the native door key system API, the hotel can develop its own software “adapter” (interface) between the two systems. Check the “Clock PMS Door Key API“ document for more information.

Clock POS API

URL list

```
root: /pos_api/:s_id/:a_id
```

Base API +

```
orders: /pos_api/:s_id/:a_id/orders
order: /pos_api/:s_id/:a_id/orders/:id
order_charges: /pos_api/:s_id/:a_id/orders/:id/charges
```

```
:s_id - Subscription ID, :a_id - Account ID, :id - ID of the object
root url example: 'https://sky-eu1.clock-software.com/pms_api/2020/1695.xml'
```

Orders

Order

The Order Object in Clock POS represents a **transaction** of adding or removing charges to/from folios. The Order itself is used in Clock POS for printing of a customer receipt for newly added charges to a client's folio. If you add three menu items (Coke, Coffee and Snack) to a table, the Order will contain these three charges. Orders are also created for Voids and Transfers (between folios or to PMS). The Transfers are the most complicated case as the order will contain charges from different folios (negative ones from source folios, and positive ones to destination folios). For this reason, the order doesn't have a folio_id attribute - it can have multiple related folios. Check Folio Orders end-point also.

Name (method)	**URL**	**Description**
Index (GET)	/pos_api/:subscription_id/:account_id/ orders	Index of all orders. Returns ID list. Accepts parameters (id, created_at, user_created_id) for filtering.
Example **:** `https://sky-eu1.clock-software.com/pos_api/2020/1695/orders.xml?id.gt=8888`		
Show (GET)	/pos_api/:subscription_id/:account_id/ orders/:order_id	Single 'order' instance.
Example **:** `https://sky-eu1.clock-software.com/pos_api/2020/1695/orders/8888.xml`		

Folio attributes are: id; number; created_at; updated_at; account_id; transaction_id; notes; user_created_id: user_updated_id.

A notification is fired on order creation. More details in “Message Channels” section.

Order Charges

List of order charges. The Charge object is described in the Base API (Folio charge). Charge attributes can be found in the “Clock_Base_API_Appendix_Object_attributes” document.

One order can contain charges from different folios (transferred charges).

Name (method)	**URL**	**Description**
Index (GET)	/pos_api/:subscription_id/:account_id/ orders/:order_id/charges	Index of all order charges. Returns list of charge ID and attributes of each charge.
Example **:** `https://sky-eu1.clock-software.com/pos_api/2020/1695/orders/8888/charges`		

Folio Orders

This end-point provides the Orders related to a Folio. If you have a Folio ID and you need to find related Orders - use this end-point.

Name (method)	**URL**	**Description**
Index (GET)	/pos_api/:subscription_id/:account_id/ folios/:folio_id/orders	Index of all orders related to a folio. Returns list of orders IDs.
Example **:** `https://sky-eu1.clock-software.com/pos_api/2020/1695/folios/9999/orders`		
View (GET)	/pos_api/:subscription_id/:account_id/ folios/:folio_id/orders/:order_id	Single order by ID
Example **:** `https://sky-eu1.clock-software.com/pos_api/2020/1695/folios/9999/orders/567`		

Folio Order Charges

Charges related both to a single folio and a single order.

Name (method)	**URL**	**Description**
Index	/pos_api/:subscription_id/:account_id/ folios/:folio_id/orders/:order_id/charges	A list of all charges. Each charge with its attributes.

(GET)		
Example **:** `https://sky-eu1.clock-software.com/pos_api/2020/1695/folios/9999/orders/567/charges`		

Message Channels

With the Message Channels in Clock API, you can get instant notifications of selected events. Two notification methods are used: Push and Pull.

Push

In order to use push events, you need a paid account. The configuration of push events is a paid service. Contact our support team for terms and conditions if you need push events on demo/trial account.

HTTP or HTTPS POST will be submitted to a URL on your server.

HTTP/S Push notification also has a retry feature in case your URL is not reachable at the moment.

A separate URL is required for each account (hotel/restaurant). To this end, a parameter, subdomain or domain for each account URL can be used. Example URL: https://push.myhotel.com?account=123

The Push service uses the Amazon SNS service (more info). The POST request payload body contains an AWS SNS notification structure (json). An SNS notification contains:

- **Subject** – contains the event type. Example: "folio_close", "booking_new", etc.
- **Message** – message is in a **json** formatted string and contains ID of the object: booking_id, folio_id, etc. Example: "Message" : "{\"booking_id\":999999}"

With the object ID you can get object details using base/pms API.

Examples of the messages can be found in Appendix in this document.

Setup of the push events

Pre-requirements:

1. An URL for each account (hotel/restaurant) on your server.
2. Administrative access to the your Web server logs. You should be able to view POST payload body.

Setup:

1. You: Send to Clock Support a list of: Account ID, Account Name, URL on your server.
2. Clock Support: Will send you a confirmation POST request to each of your URL's.
3. You: For each URL check your Web server log, find the confirmation POST request. In the payload body, find the '**SubscribeURL**' parameter.
   (an example of the request is located in Appendix of this document)
4. You: Paste the '**SubscribeURL**' in your browser and confirm the request. Notify Clock Support for successful URL confirmation.
5. Clock Support: Completes the configurations. Sends you an e-mail for completion.
6. You: Test the event notifications creating a test booking.

Pull

The 'Long polling' method is used.

“Long polling is itself is not a true push; the long polling is a variation of the traditional polling technique, but it allows to emulate a push mechanism when a real push is not possible, e.g.: sites with security policies that require the rejection of incoming HTTP/S Requests.With long polling, the client requests information from the server exactly as in normal polling, except it issues its HTTP/S requests (polls) at a much slower frequency. If the server does not have any information available for the client when the poll is received, instead of sending an empty response, the server holds the request open and waits for response information to become available. Once it does, the server immediately sends an HTTP/S response to the client, completing the open HTTP/S Request. In this way the usual response latency (the time between when the information first becomes available and the next client request) otherwise associated with polling clients is eliminated.” - Wikipedia.

If no messages are available, **304** http code will be returned in 30 seconds. In case of event – http code **200** will be returned.

When the Message Channel is used, the 'Check-After-Job' techniques implementation is strongly recommended: After finishing with the tasks related with the received event, check for new events with the REST methods.

Example: If you get a 'Folio Closed' event and export it to the accounting software, after the export itself check for newly closed folios using Folios Get URL.

The Actual Event URL can be found by calling the Root URL.

Events

Event	Pull - example URL (The actual pull event url can be found by calling the root url)	Push - SNS notification parameters
folio_close (on folio closing/document issuing)	https://httpush.clock-software.com/sub/b38a1a3e-8ce2-430e-9cd0-e395a7c04f02-1-folio_close	Subject: "folio_close" Message: "{\"folio_id\":888888}"
housekeeping_changed (on room houskeeping status change)	https://httpush.clock-software.com/sub/b38a1a3e-8ce2-430e-9cd0-e395a7c04f02-1-housekeeping_changed	Subject: "housekeeping_changed" Message: "{\"room_id\":1111}"
booking_new (on booking creation)	https://httpush.clock-software.com/sub/b38a1a3e-8ce2-430e-9cd0-e395a7c04f02-1-booking_new	Subject: "booking_new" Message: "{\"booking_id\":999999}"
booking_update (on any booking first level attribute change)	https://httpush.clock-software.com/sub/b38a1a3e-8ce2-430e-9cd0-e395a7c04f02-1-booking_update	Subject: "booking_update" Message: "{\"booking_id\":999999}"
booking_expected (on booking status changed to 'expected')	https://httpush.clock-software.com/sub/b38a1a3e-8ce2-430e-9cd0-e395a7c04f02-1-booking_expected	Subject: "booking_expected" Message: "{\"booking_id\":999999}"
booking_checked_in (on booking status changed to 'checked_in')	https://httpush.clock-software.com/sub/b38a1a3e-8ce2-430e-9cd0-e395a7c04f02-1-booking_checked_in	Subject: "booking_checked_in" Message: "{\"booking_id\":999999}"
booking_checked_out (on booking status changed to 'checked_out')	https://httpush.clock-software.com/sub/b38a1a3e-8ce2-430e-9cd0-e395a7c04f02-1-booking_checked_out	Subject: "booking_checked_out" Message: "{\"booking_id\":999999}"
booking_canceled	https://httpush.clock-software.com/sub/b38a1a3e-8ce2-430e-	Subject: "booking_canceled" Message: "{\"booking_id\":999999}"

(on booking status changed to 'canceled')	9cd0-e395a7c04f02-1-booking_canceled	
booking_no_show (on booking status changed to 'no_show')	https://httpush.clock-software.com/sub/b38a1a3e-8ce2-430e-9cd0-e395a7c04f02-1-booking_no_show	Subject: "booking_no_show" Message: "{\"booking_id\":999999}"
pos_order_new (on order creation in POS)	https://httpush.clock-software.com/sub/b38a1a3e-8ce2-430e-9cd0-e395a7c04f02-1-pos_order_new	Subject: "pos_order_new" Message: "{\"order_id\":1517248}"

APPENDIX

Tax Methods

Each account has its own default tax method. The tax method is persisted in each folio.

Support article: //www.clock-hotel-software.com/en/support-center/clock-pms/504-tax-settings

no_tax	No tax calculation
in_prices_on_row	Tax included in prices, rounding - per line
add_on_row	Tax not included in prices, rounding - per line
add_on_total	Tax not included in prices, rounding – total

Revenue Groups

Revenue groups are fixed. Each application has different revenue groups depending on its business specifics.

Application (Account type)	Revenue groups
PMS	rooms, f&b, extra, other, tax, packages, deposit, surcharge, surcharge_transfer
POS	food, beverage, service, other, discount, surcharge, surcharge_transfer
GOLF	green_fees, cart_fees, merchandise, services, surcharge, surcharge_transfer

Payment Types

Payment types are fixed: **cash, card, bank, debit, on-line, check, vaucher, other, transfer, cross_account_transfer**

Object attributes

Separate document “Clock Base API – Appendix Object attributes”

Charge Create – form example

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>
      Charge POST Example
    </title>
  </head>
  <body>
    <h1>Clock PMS - Base API - Charge POST Example</h1>
```

```
    <hr/>
    <form action="https://sky-eu1.clock-
software.com/base_api/2020/1695/folios/8888/charges.xml" method="post">
      <h2>Required fields</h2>
      Text: <input type="text" name="charge[text]" value="Test Charge"><br>
      Price: <input type="text" name="charge[price]" value="1.50"><br>
      Currency: <input type="text" name="charge[currency]" value="EUR"><br>
      Service Date: <input type="text" name="charge[service_date]" value="2015-
10-31"><br>
      Revenue Group: <input type="text" name="charge[revenue_group]"
value="extra"><br>

      <h2>Optional fields</h2>
      Qty: <input type="text" name="charge[qty]" value="1"><br>
      Tax Percent: <input type="text" name="charge[tax_rate]" value="20"><br>
      Tax Code: <input type="text" name="charge[tax_code]" value="700-700"><br>
 Revenue Category: <input type="text" name="charge[revenue_category]"
value="Test Revenue Category"><br>
 Inventory Code: <input type="text" name="charge[inventory_code]"  value="TEST-
00001"><br>

      <hr/>
      <input type="submit" value="Submit">
    </form>
  </body>
</html>
```

Payment Create – form example

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>
      Payment POST Example
    </title>
  </head>
  <body>
    <h1>Clock PMS - Base API - Payment POST Example</h1>
    <hr/>
    <form action="https://sky-eu1.clock-
software.com/base_api/2020/1695/folios/8888/credit_items.xml" method="post">
      <h2>Required fields</h2>
      price: <input type="text" name="credit_item[value]" value="5.55"><br>
      currency: <input type="text" name="credit_item[currency]" value="USD"><br>
      payment_type: <input type="text" name="credit_item[payment_type]"
value="cash"><br>

      <h2>Optional fields</h2>
      text: <input type="text" name="credit_item[text]" value="Test Payment by
API"><br>
      payment_sub_type: <input type="text" name="credit_item[payment_sub_type]"
value="Some Sub Payment"><br>
      reference_id: <input type="text" name="credit_item[reference_id]"
value="Some reference identifiers"><br>
      reference_text: <input type="text" name="credit_item[reference_text]"
value="Some reference text"><br>
      <hr/>
      <input type="submit" value="Submit">
    </form>
  </body>
```

```
</html>
```

Booking Create – form example

```
<!DOCTYPE html>
<html lang="en">
 <head>
 <meta charset="utf-8">
 <title>
 Booking POST Example
 </title>
 </head>
 <body>
 <h1>Clock PMS - PMS API - Booking POST Example</h1>
 <hr/>
 <form action="https://sky-eu1.clock-
software.com/pms_api/2020/1695/bookings.yaml" method="post">
 <h2>Required fields</h2>
 Arrival: <input type="text" name="booking[arrival]" value="2015-10-31"><br>
 Departure: <input type="text" name="booking[departure]" value="2015-11-05"><br>
 <hr/>
 RoomTypeID: <input type="text" name="booking[arrival_room_type_id]"
value="39"><br>
 <h3>AND/OR</h3>
 RoomID: <input type="text" name="booking[arrival_room_id]" value="2160"><br>
 <hr/>
 GuestLastName: <input type="text" name="booking[guest_last_name]"
value="Smith"><br>
 <h3>AND/OR</h3>
 GuestEMail: <input type="text" name="booking[guest_e_mail]"
value="smith@test.com"><br>
 <h3>AND/OR</h3>
 GuestPhoneNumber: <input type="text" name="booking[guest_phone_number]"
value="442 987 6574"><br>
 <hr/>
 <input type="submit" value="Submit">
 </form>
 </body>
 </html>
```

Booking Update – form example

```
<!DOCTYPE html>
<html lang="en">
 <head>
 <meta charset="utf-8">
 <title>
 Booking PUT Example
 </title>
 </head>
 <body>
 <h1>Clock PMS - PMS API - Booking PUT Example</h1>
 <hr/>
 <form action="https://sky-eu1.clock-
software.com/pms_api/2020/1695/bookings/5181.yaml" method="post">
    <input name="_method" type="hidden" value="patch">
    Arrival: <input type="text" name="booking[arrival]" value="2015-10-31"><br>
    Departure: <input type="text" name="booking[departure]" value="2015-11-
05"><br>
    <hr/>
```

```
    <input type="submit" value="Submit">
  </form>
  </body>
 </html>
```

HTTP/S Push – URL Confirmation

```
POST / HTTP/1.1
x-amz-sns-message-type: SubscriptionConfirmation
x-amz-sns-message-id: 165545c9-2a5c-472c-8df2-7ff2be2b3b1b
x-amz-sns-topic-arn: arn:aws:sns:us-west-2:123456789012:MyTopic
Content-Length: 1336
Content-Type: text/plain; charset=UTF-8
Host: example.com
Connection: Keep-Alive
User-Agent: Amazon Simple Notification Service Agent

{
  "Type" : "SubscriptionConfirmation",
  "MessageId" : "165545c9-2a5c-472c-8df2-7ff2be2b3b1b",
  "Token" :
"2336412f37fb687f5d51e6e241d09c805a5a57b30d712f794cc5f6a988666d92768dd60a747ba6f
3beb71854e285d6ad02428b09ceece29417f1f02d609c582afbacc99c583a916b9981dd2728f4ae6
fdb82efd087cc3b7849e05798d2d2785c03b0879594eeac82c01f235d0e717736",
  "TopicArn" : "arn:aws:sns:us-west-2:123456789012:MyTopic",
  "Message" : "You have chosen to subscribe to the topic arn:aws:sns:us-west-
2:123456789012:MyTopic.\nTo confirm the subscription, visit the SubscribeURL
included in this message.",
  "SubscribeURL" : "https://sns.us-west-2.amazonaws.com/?
Action=ConfirmSubscription&TopicArn=arn:aws:sns:us-west-
2:123456789012:MyTopic&Token=2336412f37fb687f5d51e6e241d09c805a5a57b30d712f794cc
5f6a988666d92768dd60a747ba6f3beb71854e285d6ad02428b09ceece29417f1f02d609c582afba
cc99c583a916b9981dd2728f4ae6fdb82efd087cc3b7849e05798d2d2785c03b0879594eeac82c01
f235d0e717736",
  "Timestamp" : "2012-04-26T20:45:04.751Z",
  "SignatureVersion" : "1",
  "Signature" :
"EXAMPLEpH+DcEwjAPg8O9mY8dReBSwksfg2S7WKQcikcNKWLQjwu6A4VbeS0QHVCkhRS7fUQvi2egU3
N858fiTDN6bkkOxYDVrY0Ad8L10Hs3zH81mtnPk5uvvolIC1CXGu43obcgFxeL3khZl8IKvO61GWB6jI
9b5+gLPoBc1Q=",
  "SigningCertURL" : "https://sns.us-west-
2.amazonaws.com/SimpleNotificationService-f3ecfb7224c7233fe7bb5f59f96de52f.pem"
  }
```

HTTP/S Push – Event Message Example (booking_new)

```
POST / HTTP/1.1
x-amz-sns-message-type: Notification
x-amz-sns-message-id: 22b80b92-fdea-4c2c-8f9d-bdfb0c7bf324
x-amz-sns-topic-arn: arn:aws:sns:us-west-2:123456789012:MyTopic
x-amz-sns-subscription-arn: arn:aws:sns:us-west-2:123456789012:MyTopic:c9135db0-
26c4-47ec-8998-413945fb5a96
Content-Length: 773
Content-Type: text/plain; charset=UTF-8
Host: example.com
Connection: Keep-Alive
User-Agent: Amazon Simple Notification Service Agent
{
 "Type" : "Notification",
 "MessageId" : "22b80b92-fdea-4c2c-8f9d-bdfb0c7bf324",
```

```
 "TopicArn" : "arn:aws:sns:us-west-2:123456789012:MyTopic",
 "Subject" : "booking_new",
 "Message" : "{\"booking_id\":693926}",
 "Timestamp" : "2015-11-17T09:44:23.421Z",
 "SignatureVersion" : "1",
 "Signature" :
"OcBCD+Wh4T1eWQxebQo8CnhM3CZJMtGSSYVw9OgSk5bera4QXO10yp0/x0XEHsL3hi5Z5B4daIJXEg0
SeybsjgtCfqJ0z8WnJ/np5ErWLMK2ymPMZxJWVEQXM76shLUNBDvYEWhfeMPTCylGJVDD2YeaWZXh5M/
oW6dQcxpetmoZkiahjmDsN3BQFvVbCpQ5zrfZZKBkULCo716KLKOlTMDRqK1uSSdZAbi95jOvZo9coq7
vLwtn/rL2iKmmnOOh9C3FmxegblNvKz12B8Vxeh5l98zULb/6YhWoUVD+eAnu7dKmGuHhFhjvy+Fl1SF
5TN0yI+C6/m+IEIHzi2tZVQ==",
 "SigningCertURL" : "https://sns.eu-west-
1.amazonaws.com/SimpleNotificationService-bb750dd426d95ee9390147a5624348ee.pem",
 "UnsubscribeURL" : "https://sns.eu-west-1.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:eu-west-
1:006467213368:PUSH_Ro_Test:fc851d44-40ec-40d5-88fc-acccfc7b5250"
}
```