MDP - WebHook platform proposal

Contents

[1.](#_heading=h.3znysh7) Introduction 2

[2.](#_heading=h.tyjcwt) Gateway API 3

[2.1.](#_heading=h.3dy6vkm) Configure Webhook 3

[2.2.](#_heading=h.1t3h5sf) Booking Events 4

[2.3.](#_heading=h.2s8eyo1) Webhook Payload 4

[3.](#_heading=h.3rdcrjn) Event Sourcing MicroService 6

[3.1](#_heading=h.26in1rg) Configuration 6

[4.](#_heading=h.cneubajsgqd3) Event Delivery MicroService 7

[4.1.](#_heading=h.lnxbz9) Configuration 7

[5.](#_heading=h.35nkun2) Solution diagram 8

# Introduction

WebHook platform proposal is a service that sends notifications to handlers provided by partners. Notifications will be triggered at the moment of a new booking event created, modified or canceled.

The service allows to specify exactly what the partner wants to receive. Filters can be applied to the booking event (accounting, reservation...), event type (creation, cancelation...) and additionally for maximum flexibility use the JSON filtering syntax.

The service APIs are organized in terms of types and fields, not endpoints. Access to the full capabilities is provided within a single endpoint for each partner's subscription.

WebHook platform proposal is a service in corporate cloud that consists of three components:

* API gateway for partners to manage the details of the subscription
* MicroService to source booking events based on corporate Queue
* Microservice to deliver events to partners

# Gateway API

The role of the API gateway is:

* Configure partner Callback Url to let be notified when certain events happen
* Set up an event map to collect only important data
* Enable Access Secret and other credentials

### Configure Webhook

Configure a webhook object to represent your webhook endpoint. Configure it with the URL of your webhook.

#### **REQUEST URL**

| **POST** | **/api/v1/webhooks** |
| --- | --- |

#### **REQUEST BODY**

|  | Parameter | Type |  |
| --- | --- | --- | --- |
| 1. | **name** | string | required |
|  | Name for webhook | | |
| 2. | **payload\_url** | string | required |
|  | URL of the target to which to POST event batches. Only ports 80 for http and 443 for https can be set. | | |
| 3. | **secret** | string | required |
|  | API Key Secret  obtained within registration | | |
| 4. | **events** | string array | required |
|  | Array of booking events this webhook will send. Use the endpoint to list the booking events. | | |
| 5. | isActive | bool |  |
|  | The status of the webhook. When false, the target will no longer receive batches. | | |

### Booking Events

* Accounting
* Reservation
* Loyalty

### Webhook Payload

Each webhook event payload also contains properties unique to the event. You can find the unique properties in the individual event type sections.

But common properties are:

|  | Parameter | Type | Location |
| --- | --- | --- | --- |
| 1. | **signature** | string | header |
|  | This header is sent if the webhook is configured with a secret. This is the HMAC hex digest of the request body, and is generated using the SHA-256 hash function and the **secret** as the HMAC key | | |
| 2. | **created** | The timestamp when the webhook was created | body |
| 3. | **event** | Reason for triggering | body |
| 3. | **data** | {object} | body |
|  | URL of the target to which to POST event batches. Only ports 80 for http and 443 for https can be set. | | |

# Event Sourcing MicroService

The task of this service is to listen to the topic of the corporate message queue, pick events in which partners are interested, group them by content in order to prepare a set of messages to deliver to partners.

### Configuration

At the global configuration level, the following is defined:

* Credentials to access Kafka
  + Cluster Name
  + Broker Id, Host & Port
  + Consumer Group
  + List of involved Topics

# Event Delivery MicroService

The task of this service is to continuously and instantly deliver events to partners through assigned channels, repeat delivery attempts in the event of an external system failure, and notice failed points.

### Configuration

At the global configuration level, the following is defined:

* Retry Policy Settings
  + Time Limit

Any webhook batch that does not receive an HTTP 200 response will be retried for a total of this period in minutes before the data is discarded.

* + Frag Limit

Any webhook batch that does not receive an HTTP 200 response will be retried for a total of this time before the data is discarded.

# Solution diagram