Hyperwallet Mirakl Connector
Solution Guide
Version 5.0

Table of Contents

Terms & acronyms

Project term	Mirakl term	Hyperwallet term	Description
HMC			Hyperwallet Mirakl Connector
Business Account	Professional account	Business Payee/User	A company selling products on the marketplace
Individual account	Standard account	Individual Payee/User	An individual selling products on the marketplace.
Invoice	Accounting document, Payment file	Payment, Payment request	The automatically generated accounting document used for calculating the payout to the seller and operator at the end of a billing cycle in Mirakl. Support for other types of accounting documents in Mirakl (e.g. credit notes) is currently on the product roadmap.
KYC	KYC	Payee verification	Process to verify the identity of individual account holder or business stakeholder (for business accounts) by checking data and documents and assessing their suitability.
Operator	Operator	Organisation	Marketplace owner/operator providing the platform for sellers to sell products and services. Establishes rules for the marketplace operation and sets up all the relevant configurations (countries, languages, currencies, fees, security settings etc.).
Seller	Seller / Shop	Payee / User	An individual or business entity selling products and services via the marketplace.
Payout	Payment	Payment	Paying Mirakl accounting documents (i.e. invoices) via HMC to corresponding payees in Hyperwallet.
Multi-program hierarchy		Multiple Programs Level 2 & 3 program hierarchy	Functionality to support use cases for more complex payout structure where the operator payout can be directed to different child programs. For more information, please refer to Hyperwallet documentation: docs.hyperwallet.com/content/program-hierarchy/v1/multiple-programs.

Introduction

This document is for the use of PayPal/Hyperwallet integration teams and their customers. It describes the core features of the Hyperwallet Mirakl Connector and how to operate it together with the Mirakl and Hyperwallet platforms. This document supplements, but does not replace, the official Hyperwallet and Mirakl platform documentation or any other solution guidance provided by Hyperwallet and Mirakl.

Solution Overview

The **Hyperwallet Mirakl Connector (HMC)** is a drop-in service that mediates between a Mirakl marketplace solution and the Hyperwallet payout platform. It implements the common integration points between Mirakl and Hyperwallet, saving new operators significant effort on setting up their marketplace system. The HMC is designed specifically to support the widest range of marketplace use cases and PSP combinations and should enable a functional Mirakl-Hyperwallet integration with little to no extra technical work needed.

This document is kept up to date with the latest version of the HMC solution.

For detailed technical setup and deployment instructions, please also consult the README file that is distributed with the HMC source code, located in the public GitHub repository at:

`https://github.com/paypal/mirakl-hyperwallet-connector`

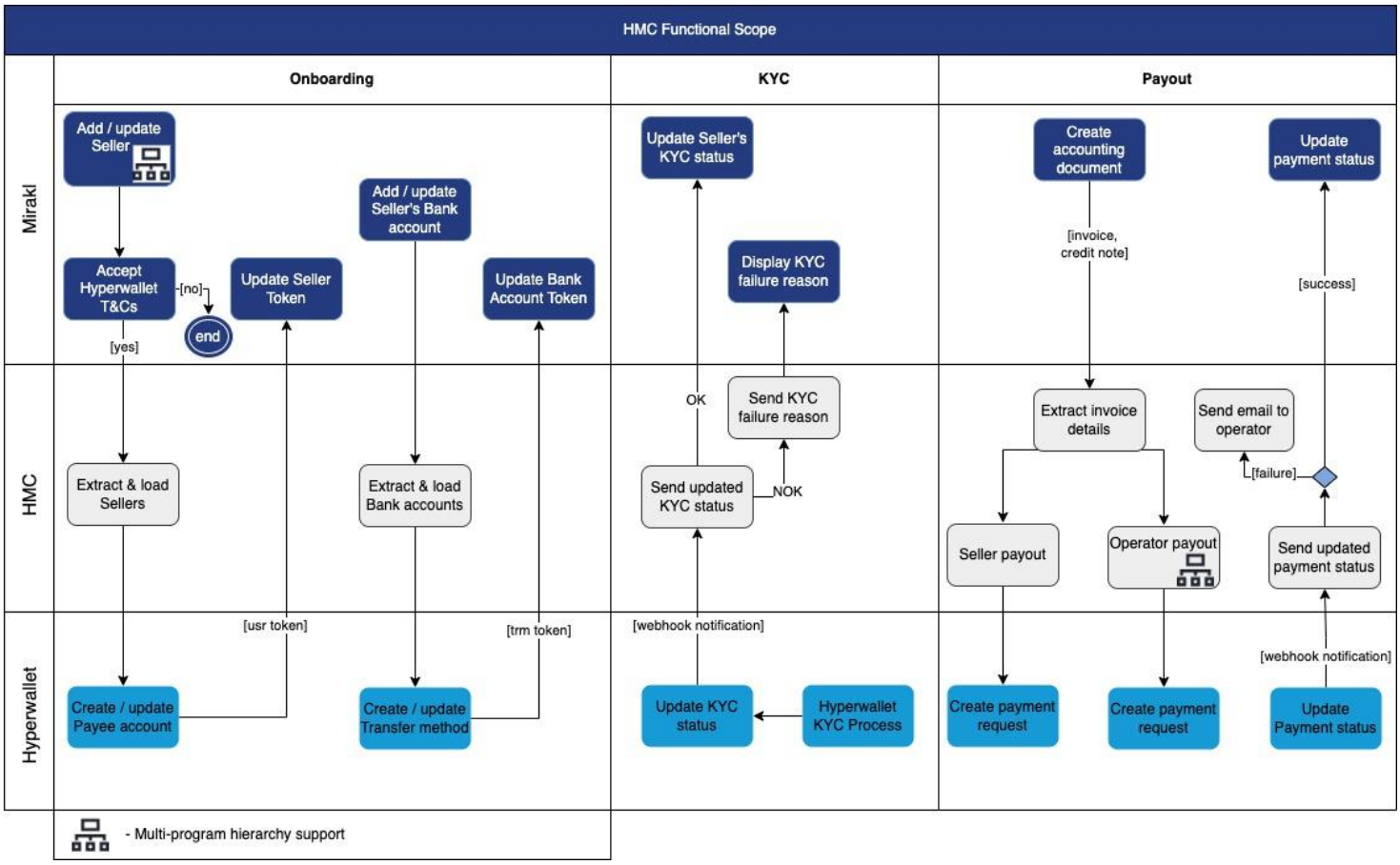
Pay-in provider compatibility

HMC facilitates payout to Sellers and Operators irrespective of the source of funds and in “pay-in agnostic” way. This means that HMC

- Can work with Braintree-Mirakl pay-in connectors and function as part of the overall PayPal ecosystem for pay-in and payout services; or
- Can be integrated with any pay-in PSP and work alongside it to provide the payout services.

1.1 Functional Scope

The HMC encompasses a number of features that are described in the below section.



1.2 Features

Feature	Connector->Mirakl Actions	Connector->Hyperwallet Actions
1. Onboarding		
Create payee account for Individual Hyperwallet User	▪ Retrieve shop details from Mirakl for newly onboarded sellers trading as individuals. ▪ Obtains all shops added to operator marketplace since the previous extract. ▪ Mirakl shop record updated with generated Hyperwallet User token (Additional Mirakl custom field). ▪ Assign the shop to a relevant level within the operator program hierarchy (e.g. if the operator works through separate legal entities in different jurisdictions).	▪ Create a new User in Hyperwallet with type of INDIVIDUAL ▪ One Hyperwallet User created for each newly onboarded Individual seller obtained from Mirakl. ▪ The shop will be assigned to the corresponding operator program within the merchant program hierarchy in Hyperwallet.
Create payee account for Business	▪ Retrieve shop details from Mirakl for newly onboarded sellers trading as businesses.	▪ Create a new User in Hyperwallet with type of BUSINESS.

Hyperwallet User	▪ Obtains all shops added to operator marketplace since the previous extract. ▪ Mirakl shop record updated with generated. Hyperwallet User token (Additional Mirakl custom field). ▪ Assign the shop to a relevant level within the operator program hierarchy (e.g. if the operator works through separate legal entities in different jurisdictions).	▪ One Hyperwallet User created for each newly onboarded Business seller obtained from Mirakl. ▪ The shop will be assigned to the corresponding operator program within the merchant program hierarchy in Hyperwallet.
Accept Hyperwallet T&Cs	▪ Only retrieve shops that accepted the Hyperwallet T&Cs.	▪ Only shops that have accepted Hyperwallet T&Cs will be sent to Hyperwallet. ▪ Rejecting the T&Cs means the shop will not be eligible for any of the Hyperwallet features including payout.
Update existing Hyperwallet Users	▪ Retrieve the details of all existing shops in Mirakl that have been updated since the last extract. ▪ Custom field containing the Hyperwallet User token indicates the Shop already exists in Hyperwallet.	▪ Update the existing Hyperwallet User with all shop data retrieved from Mirakl. ▪ The user token will be used to update the relevant Payee record in Hyperwallet.
Create Bank account	▪ Retrieve bank account details for newly onboarded sellers. The following Mirakl bank account types are supported: o IBAN (EUR, CHF); o U.S. ABA (USD); o CANADIAN (CAD, USD); o United Kingdom account (GBP). ▪ Mirakl shop record updated with generated Hyperwallet bank account token (Additional Mirakl custom field). ▪ **Note:** There is not separate payment method in Mirakl for UK bank accounts, hence this cannot be implemented at this stage.	▪ Bank account (i.e. Transfer Method) created after the User account is successfully created for the seller in Hyperwallet. ▪ A single bank account created for the existing Hyperwallet User account. ▪ Bank Account is the only transfer method supported in HMC.
Update Bank account details	▪ Retrieve the details of bank accounts that have been updated since the last extract. ▪ Custom field containing the Hyperwallet TRM token indicates the bank account for the shop already exists in Hyperwallet.	▪ Update the existing bank accounts with new details from Mirakl, i.e. update existing Transfer method.
Configurable seller data extract frequency	▪ Operator can configure the frequency of the shop details/bank account retrieval from Mirakl.	▪ Process to extract shop data from Mirakl is triggered based on configured frequency
2. KYC (Managed by Hyperwallet)		
KYC status updates	▪ Update Mirakl with KYC status changes received from Hyperwallet via HMC.	▪ Receive a notification of changes in KYC status as part of the payee verification process. This will be

	▪ KYC status is displayed as part of the shop record in Mirakl	received by the Connector via a webhook and sent through to Mirakl.
KYC failure reasons	▪ Send a predefined failure reasons based on the KYC status received. ▪ Failure reason is displayed as part of the shop record in Mirakl	▪ Receive KYC status update from Hyperwallet
3. Payout		
Payout to sellers	▪ Retrieve details of all seller invoices generated since the last extract. ▪ Retrieves invoices generated by the Mirakl operator billing cycle, or manual credit notes, that are eligible for payout and: o have not yet been paid; o are not in draft state.	▪ Create a payment request for paying out the seller based on the details of the invoice. ▪ All seller payouts are from a single operator funding account. ▪ Seller payouts made to the seller bank account stored in Hyperwallet
Payout to Operator	▪ Retrieve details of the operator commission and subscription fee amounts due for each seller invoice. ▪ Operators can configure whether to turn off this automated payout process.	▪ Create a new operator payout request to payout the individual commission and subscription fee amounts due to the operator for each seller invoice ▪ If operator chooses to turn off the automated payout, their commission and subscription will not be processed/paid and will remain in the main funding account in Hyperwallet. ▪ Send the corresponding program tokens based on which entity/level of merchant hierarchy the seller belongs to. ▪ Payout will be directed to the corresponding operator bank account associated with the relevant merchant program in Hyperwallet.
Payout status updates	▪ Send a failure notification to the operator by email in case of a payout failure (seller payout or operator payout).	▪ Payout status will be received by HMC via a webhook.
4. Technical enablement		
API failure and error alerts	▪ Send email alerts to the operator in case of any API failures or other errors occur (e.g. mandatory data missing, timeout, authentication failures).	

1.3 HMC release history

v1.0 (December 2020)

Category	Features
Onboarding	▪ Create and update payee account (Individual, Business payees) ▪ Create and update payee bank accounts ▪ Accept Hyperwallet T&Cs
KYC	▪ KYC status updates
Payout	▪ Payout to Sellers (IBAN, US ABA, Canadian) ▪ Payout to Operator ▪ Payout status updates
Technical enablement	▪ API failure and error alerts

v2.0 (May 2021)

Category	Features
Programs	▪ Support for Multi-Program Hierarchies
Security	▪ Support for Payload Encryption (Layer7)
KYC	▪ KYC failure reasons
Payout	▪ Payout in GBP to UK bank accounts

v3.0 (June 2021)

Category	Features
Payout	▪ Support for paying out manual credit notes

v3.1 (July 2021)

Category	Features
Technical enablement	▪ Provides a Postman collection in the repository, to ease the testing/deployment process

v4.0 (August 2021)

Category	Features
Technical enablement	▪ Introduced a .env file for use in the Docker deployment process, and for optionally collecting configuration values

	▪ Migration of remaining user-configurable values to Environment Variables, away from being contained in module-specific property files (Note: requires specific upgrade tasks, see notes in the repository) ▪ Extensive revision of the solution README to provide more specific & detailed guidance

v4.1 & v.4.2 (September 2021)

Category	Features
Technical enablement	▪ Update the example Postman file with latest HMC endpoints

v4.3 (September 2021)

Category	Features
Seller onboarding	▪ Added 2 custom fields for supporting the Business Registration Country and Business Registration State/Province fields in Hyperwallet
Technical enablement	▪ Use Mirakl operator time zone to ensure dates retain the exact day/month/year value entered in the Mirakl backoffice

v4.4 (November 2021)

Category	Features
Technical enablement	**Removal of RabbitMQ message queue as a technical dependency**

v4.5 (December 2021)

Category	Features
KYC Improvements	▪ **Removal of Employer ID custom field** ▪ **Improved logic so that the Hyperwallet Platform is always informed when the Seller and their stakeholders are ready for verification**

v4.6 (December 2021)

Category	Features
Seller onboarding	▪ **Reduction of fields replicated when creating and updating Business Sellers**

v4.7 (January 2022)

Category	Features
Seller payout	▪ **Enhance the seller payout diagram including successful and failed notifications from HW**

v4.8 (January 2022)

Category	Features
Operator payout	▪ **Enhance the operator payout diagram including successful and failed notifications from HW**

V4.9 (February 2022)

Category	Features
Webhook improvements	▪ **Develop a mechanism in order to retry webhook notifications.**

V5.0 (March 2022)

Category	Features
Notification storage and filter	▪ **Develop a functionality to store all the webhooks received from Hyperwallet in the database so that they can be later filtered out discarding duplicated and obsolete notifications.** ▪ **Include new currencies for the Bank Accounts creation.** ▪ **Enhance the general overview diagram, including the Hyperwallet tokens update into Mirakl**

2 Operator onboarding

2.1 Overview

Onboarding of Marketplace Operators on to the Hyperwallet program(s) will be handled differently than seller onboarding. Operators do not usually exist in Mirakl or Hyperwallet as account entities.

The operator details required for participation in the Hyperwallet program will be provided by the PayPal team as part of the onboarding process and configured in the HMC properties. This includes program tokens that will be used to identify the operator on the Hyperwallet platform to enable operator payouts for orders made on the marketplace.

The configuration instructions in the README include the environment variables for setting the necessary program tokens. The default setup conforms to a Single-level program hierarchy structure in Hyperwallet.

Refer to section 2.3 below regarding the multi-program hierarchy structure set up.

For further details regarding setting up Operators in Hyperwallet and defining the required program hierarchy please refer to Hyperwallet documentation that will be provided to you by the PayPal team as part of the onboarding process.

2.2 Webhook Processing

HMC supports receiving and processing event notifications sent by Hyperwallet via webhook. This is accomplished using a built-in webhook listener that handles all supported webhook notification types and works with both basic authentication and payload encryption. The supported webhook notification types are specified in the various feature-related sections in this document.

During the operator onboarding process, the Hyperwallet team will enable webhook notifications by registering the webhook listener endpoint URL.

The endpoint for the webhook listener is on the path: `/webhooks/notifications`. This path is used by default, and no properties or configuration are used for enabling or setting up the webhook listener.

The endpoint must be accessible from Hyperwallet IP ranges specified in the official documentation page https://docs.hyperwallet.com/content/webhooks/v1/integration

For example, if the HMC is deployed to the URL: `https://hmc.example.com`, then the full webhook listener URL will be: `https://hmc.example.com/webhooks/notifications`.

2.3 Multi-program hierarchy

In certain scenarios a Marketplace Operator may wish to support use cases for more complex payout structure where the operator payout can be directed to different child programs. This may depend on a number of factors including payee types, organisational structure, funding needs and regional distinctions.

The appropriate setup should be defined in collaboration with the PayPal team as part of the onboarding process. For more information about supporting multiple programs in Hyperwallet please refer to Hyperwallet documentation: docs.hyperwallet.com/content/program-hierarchy/v1/multiple-programs.

When a multi-program hierarchy structure is defined in Hyperwallet for the Operator this will need to be reflected in HMC and Mirakl as described below.

HMC configs for multi-program hierarchy:

The properties described in the above section should be updated to include multiple Issuing Store tokens. See "Multiple Issuing Store configurations" section of the README file that is distributed with the connector source code.

Mirakl settings for multi-program hierarchy:

The *"Hyperwallet Program"* custom field in Mirakl should be updated to allow selecting the relevant program that a Seller shop will belong to. This will be done by extending the single-value list under this field to include the multiple programs defined as part of the operator program structure in Hyperwallet.

For example, if a Marketplace Operator *Greenfield* has been set up with several programs to support regional markets (e.g. UK, Europe, USA) this single value list should be extended to include the following values that will represent different child programs under the parent merchant program:

- Greenfield – UK;
- Greenfield – Europe;
- Greenfield – USA.

When creating a new Seller shop the Operator should select which program the Seller should be linked to by selecting the relevant Hyperwallet Program from the above single-value list. This can only be done by the Operator as the Hyperwallet Program field is hidden and not displayed to the Sellers as part of the Store details in Mirakl.

Note: it is possible to change Hyperwallet Program when updating the Seller details in the future. However, you will only be able to change to another operator child program at the same level of hierarchy and under the same parent merchant program as described in the Hyperwallet documentation (see *programToken* field under the Update User section: docs.hyperwallet.com/content/api/v4/resources/users/update). Please contact your Hyperwallet account representatives for any questions about setting up and managing multi-program hierarchy.

Bank accounts

It is possible to configure bank accounts for the operator child programs (e.g. Greenfield – UK, Greenfield – USA). This is defined as part of the Operator onboarding program with the PayPal team.

Once the hierarchy structure is defined the appropriate TRM tokens will need to be added to the connector properties a described in "Multiple Issuing Store configurations" section.

For further details how to configure this please see the "Multiple Issuing Store configurations" section of the README file that is distributed with the connector source code.

3 Seller onboarding

3.1 Overview

Sellers are trading entities that sell products and services via their online shops on a Mirakl marketplace managed by a marketplace Operator. A Seller can represent an individual, if trading as a person, or a business entity, if trading as a company. For the purpose of this guide, Sellers are equivalent to Mirakl shops/stores.

For sellers that create multiple shops in Mirakl for selling with multiple currencies, each shop must be treated as a new entity. Each shop has its own account holder and needs to be onboarded separately.

Seller information is managed in the Mirakl marketplace, which is the source of all Seller data, including the details used for the Payout. Each Seller must have a corresponding user account and bank account in Hyperwallet in order to receive payments from the Mirakl marketplace Operator.

The following section describes onboarding of Sellers from Mirakl to Hyperwallet via the HMC. This includes creation and update of Seller accounts and their corresponding bank accounts through a set of automated synchronisation jobs performed by the HMC at regular configurable intervals.

Onboarding of marketplace Operators and the payee verification process for Operators are not part of the current HMC scope and are not covered by this document. For these and other out-of-scope topics, Hyperwallet will provide a parallel solution to the marketplace Operators.

3.2 Create/Update Seller

3.2.1 Overview

HMC provides two jobs to perform creation of new sellers and update of existing sellers:

- Professional sellers extract – extracts all business sellers from Mirakl.
- Individual sellers extract – extracts all individual sellers from Mirakl.

These are implemented as two separate jobs because the data model and behaviours in Mirakl & Hyperwallet are different between individual sellers and business sellers.

These jobs are scheduled to run on a configurable frequency. Refer to section 3.2.4 Timed Job Frequency below for further information about configuring the timed jobs.

In the Contact Details section in the Mirakl seller page, the “Professional” boolean attribute is used to determine the seller type.

Upon successful creation of a seller’s user account in Hyperwallet:

- The user account can be found under the Account listing screen in Hyperwallet.
- The user account token will be stored under the corresponding seller’s shop record in Mirakl – see section 7.5.1 Seller/Store Custom Fields
    - **Note:** this field can only be viewed by the marketplace Operator and will be hidden for Sellers.

T&C Acceptance

The HMC will only create Hyperwallet payee accounts for new sellers when the sellers have accepted the Hyperwallet Terms & Conditions in Mirakl. If T&Cs are not accepted the rest of this guide does not apply. Acceptance is confirmed by setting the "Hyperwallet Ts & Cs and Privacy Policy consent" custom field to "Yes".

3.2.2 REST API Endpoints

Platform	Endpoint	Documentation
Mirakl	(GET) S20	help.mirakl.net/help/api-doc/operator/mmp.html#S20
Hyperwallet	(POST) Create User	docs.hyperwallet.com/content/api/v4/resources/users/create
Hyperwallet	(PUT) Update User	docs.hyperwallet.com/content/api/v4/resources/users/{user-token}

3.2.3 Data Requirements

Countries & Currencies

Refer to 3.3 Create/Update Seller Bank Account below regarding supported bank account types.

Data mapping

The following data is used by the HMC when replicating seller accounts from Mirakl into Hyperwallet.

All fields are required except where explicitly stated.

Note that different fields are replicated depending on whether the seller is of the Business or Individual type.

Hyperwallet field	Mirakl field	Notes	Mirakl Custom field Type
Both Business & Individual sellers			
clientUserId	shopId – autogenerated when creating a shop in Mirakl		
profileType	profileType - Check "Professional" for Business Seller type or leave unchecked for an Individual seller type.		
email	contact_informations.email		
addressLine1	contact_informations.street1		
addressLine2	contact_informations.street2		
city	contact_informations.city		
stateProvince	contact_informations.state (Only 2 letter code for US)		
country	contact_informations.country		
postalCode	contact_informations.postalCode		

programToken		Value stored within connector	
businessName	shopName		
Business Seller - Additional fields for Business Seller type			
businessRegistrationId	pro_details.identification_number		
businessOperatingNames	pro_details.corporate_name		
Individual Seller - Additional fields for Individual Seller type			
firstName	firstName: contact_informations.civility + contact_informations.firstname		
middleName		Not required	
lastName	lastName		
gender		Not required	
phoneNumber	contact_informations.phone		
mobilePhone	contact_informations.phone_secondary		

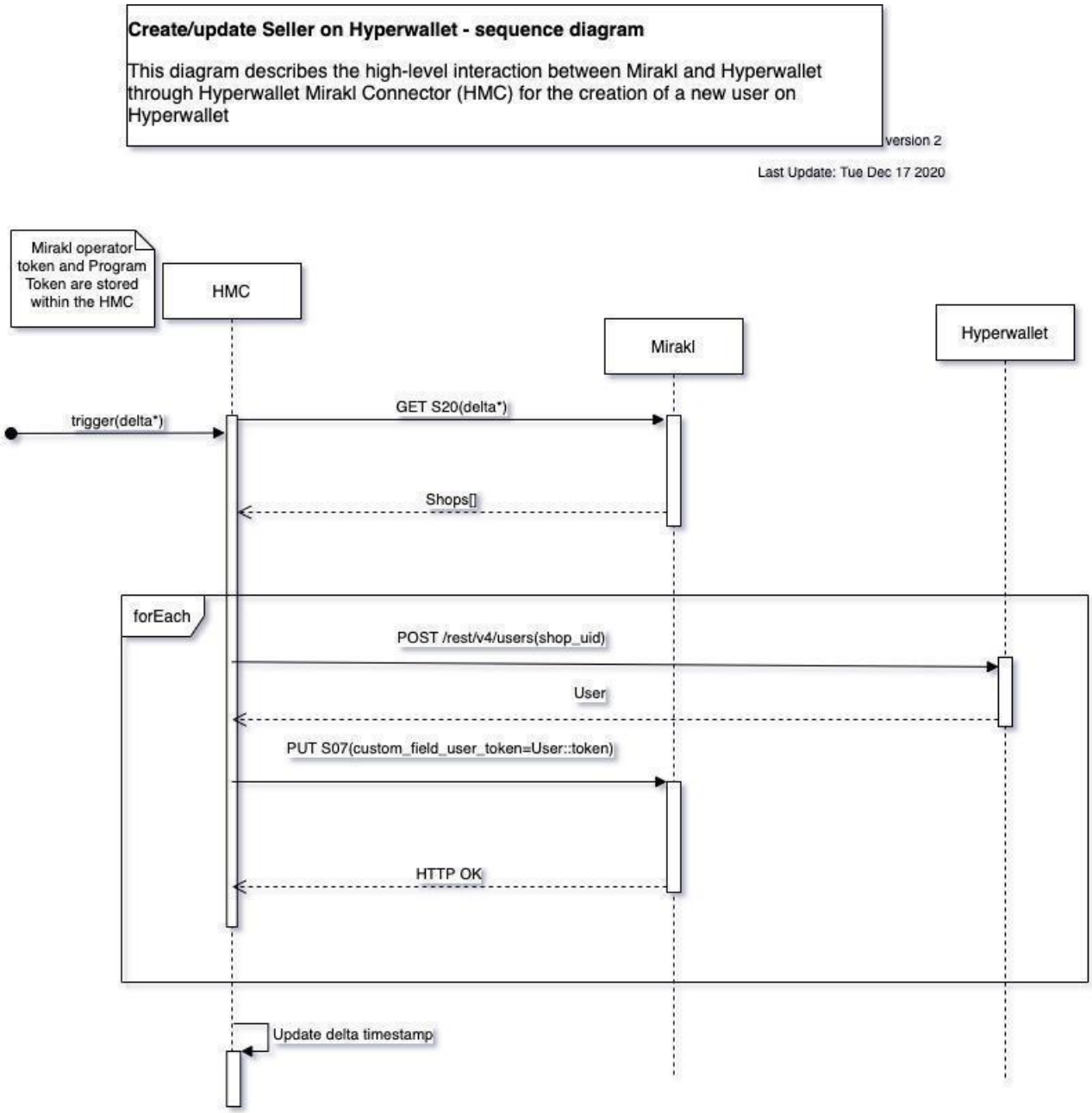
3.2.4 Timed Job Frequency

The frequency for the Individual and Professional Seller Extract jobs is configurable. By default, these jobs are scheduled to run at midnight (00.00) daily. The frequency can be changed if required, using the following Environment Variables (see README for further details such as format and default value):

- PAYPAL_HYPERWALLET_EXTRACT_SELLERS_CRON_EXPRESSION
- PAYPAL_HYPERWALLET_EXTRACT_PROFESSIONAL_SELLERS_CRON_EXPRESSION

Please note, defining a very high frequency (e.g. every few minutes) may not deliver much benefit, as Seller records are not expected to be created/updated that often by the Operator. However, it will most likely impact performance. We suggest defining the frequency to once a day, if possible.

3.2.5 Technical Flow



3.3 Create/Update Seller Bank Account

3.3.1 Overview

The following bank account schemes and currencies from Mirakl are currently supported:

- IBAN (EUR, CHF)
- U.S. ABA (USD)

- CANADIAN (USD, CAD)
- United Kingdom account (GBP)

Additional schemes and currencies are planned to be released in the future when prioritised for implementation.

The following HMC job performs creation and update of seller bank accounts:

- *Bank accounts extract*

Upon successful creation of a bank account in Hyperwallet:

- The bank account details can be found in the corresponding user account record in Hyperwallet, under *"View – Consumer Account"* > *"Transfer"* tab;
- The bank account token (TRM) will be stored under the corresponding seller's shop record in Miracle – see *"Hyperwallet Additional Custom Fields"* section. **Note:** this field can only be viewed by the marketplace Operator and will be hidden for Sellers.

3.3.2 REST API Endpoints

Platform	Endpoint	Documentation
Mirakl	(GET) S20	help.mirakl.net/help/api-doc/operator/mmp.html#S20
Hyperwallet	(POST) Create Bank Account	docs.hyperwallet.com/content/api/v4/resources/bank-accounts/create

3.3.3 Data Requirements

Supported payment methods (bank account types)

Each supported payment method consists of a predefined set of attributes.

Attempting to use different countries or currencies than those listed here will likely result in a failure when records are synchronised into Hyperwallet during the Seller Extract job or during Payout payment creation. Please check the logs if you encounter any unexpected errors or missing data (see section 7 for logging & the log file location).

Data mapping

The following data will be entered when creating Seller's stakeholders in Mirakl and replicating it into Hyperwallet via HMC. **All fields are required except where explicitly stated**.

Hyperwallet field	Mirakl field	Notes
"profileType":	"BUSINESS" or "INDIVIDUAL", based on "Professional" attribute	
"transferMethodCountry":	Seller Country from contact details (using 2 char ISO code as per HW API spec)	
"transferMethodCurrency":	currency_iso_codefield from S20 API	
"type":	N/A	Fixed value "BANK_ACCOUNT" – will be

		automatically added to the feed by HMC
"businessName":	“Company name” field under Seller Contact Details	
"country":	“Country” field under Seller Contact Details	
"addressLine1":	“Address” field under Seller Contact Details	
"addressLine2"	“Address (continued)” field under Seller Contact details.	This is optional and should be passed if this data exists.
"city":	“City” field in Seller Bank Account Details	
“postalCode”:	“Postcode” field in Seller Bank Account Details	
“stateProvince”:	Custom field: hw-bankaccount-state.	See custom field setup guidance in Section 7.5.
"bankAccountRelationship": "OWN_COMPANY"	N/A	This field will not be sent.
IBAN Payment method		
"bankId":	BIC	
"bankAccountId"	IBAN	
UK Payment Method		
"bankId":	BIC	
"bankAccountId"	IBAN	
U.S. ABA Payment method		
branchId	Routing number (ABA).	
bankAccountId	Bank account number	
bankAccountPurpose	N/A	Fixed value “CHECKING” - will be automatically added to the feed by HMC
U.S. ABA Payment method		
bankId	3 digit bank code	
branchId	5 digit transit number	
bankAccountId	Bank account number	

3.3.4 Timed Job Frequency

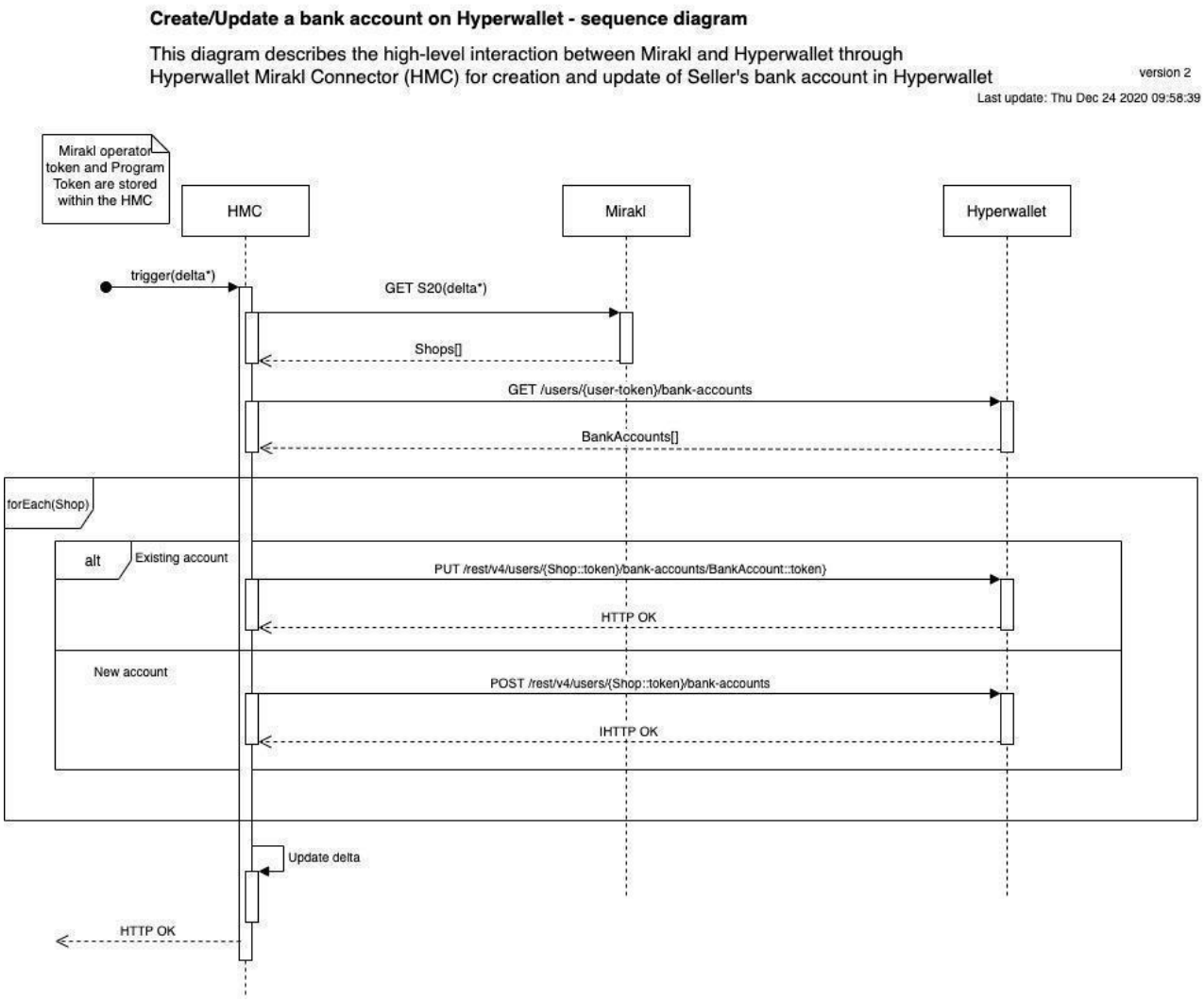
As with Seller extract jobs, this job is scheduled to run at certain configurable frequency, by default at 00.30 AM.

The frequency of the Bank account extract job is configured by the following Environment Variable (see README for further details such as format and default value):

- PAYPAL_HYPERWALLET_BANK_ACCOUNT_EXTRACT_CRON_EXPRESSION

Please note, defining a very high frequency (e.g. every minute) may not deliver much benefit, as Seller records are not expected to be created/updated that often by the Operator. However, it will most likely impact performance. We suggest defining the frequency no higher than once a day, if possible.

3.3.5 Technical Flow



3 KYC

4.1 Overview

The Know Your Customer (KYC) process aims to verify customer accounts to ensure that the customers are genuinely who they claim to be. This involves verification of customer identity and, in the case of the Business Sellers, additional information about the business and its stakeholders.

KYC checks are mandatory in most countries that are the target market for Mirakl and Hyperwallet and so support for KYC processes exists on both platforms.

The Hyperwallet Mirakl Connector supports a managed flow for processing KYC verification, where seller details are added directly into a Hyperwallet-managed KYC form

The 'KYC Status' feature should be enabled in Mirakl in order to enable KYC processing. The operator should request this feature to be enabled by contacting Mirakl support.

4.2 Managed KYC Flow

By default, HMC supports KYC verification via Hyperwallet-managed KYC forms. Please contact your Hyperwallet support team for instructions on how to conduct this process.

When the Hyperwallet platform sends KYC status update notifications, HMC will read these notifications and update the status on the relevant seller record(s) in Mirakl, thereby completing the KYC verification flow across Hyperwallet and Mirakl. See Section 4.3 "Update KYC Status" for more information about this process.

4.3 Update KYC status

The Seller KYC status update will be received from HW via a webhook notification and updated on Mirakl as per the mapping below.

Note that there is a difference in the mapping for Individual and Business account types.

Individual Account

HW KYC status	Mirakl KYC status
UNDER_REVIEW	Awaiting KYC verification (PENDING_APPROVAL)
VERIFIED	KYC Passed (APPROVED)
REQUIRED	Awaiting KYC data (REFUSED)
NOT REQUIRED	KYC Passed (Approved)

Business Account

The 3 statuses from HW (*verificationStatus, businessStakeholderVerificationStatus, letterOfAuthorizationStatus*) will be compared to derive the overall KYC status updated in Mirakl.

The order of priority will be as follows, based on the following precedence rules:

- REQUIRED (most restrictive)

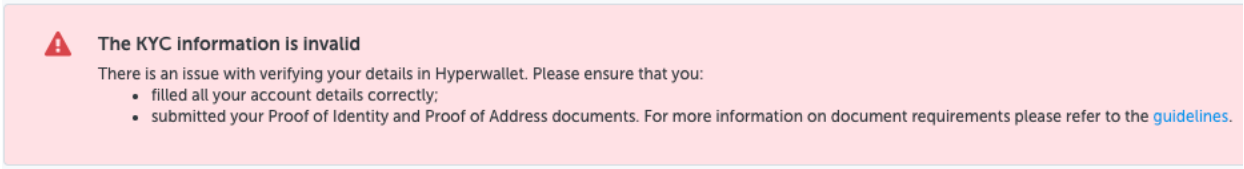
- UNDER REVIEW
- VERIFIED
- NOT_REQUIRED (least restrictive)

When ALL of these statuses in HW: - verificationStatus - businessStakeholderVerificationStatus - letterOfAuthorizationStatus have reached at least:	Mirakl KYC status
NOT REQUIRED	KYC Passed (APPROVED)
VERIFIED	KYC Passed (APPROVED)
UNDER_REVIEW	Awaiting KYC verification (PENDING_APPROVAL)
REQUIRED	Awaiting KYC data (REFUSED)

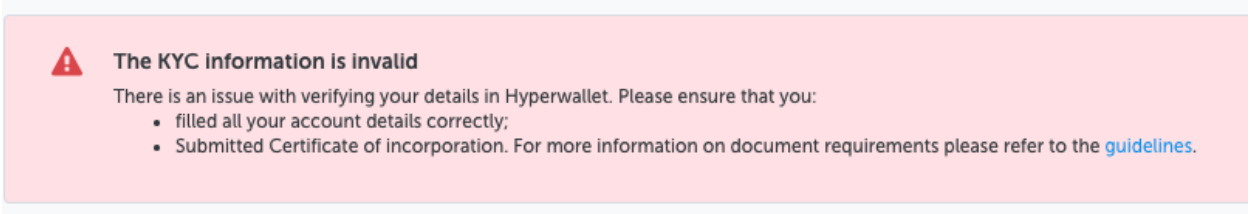
4.7 KYC Failure reason message

Upon updating KYC status if the status received from Hyperwallet is still REQUIRED, HMC will send a failure reason message to Mirakl to be displayed on the shop details page. Below are the possible failure messages depending on the type of notification received from Hyperwallet.

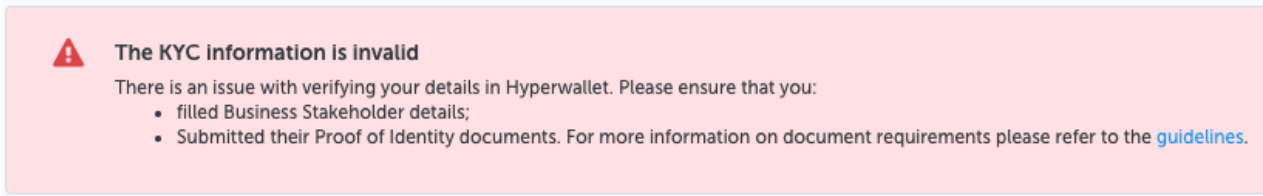
- **Individual Seller:** notification received is `verificationStatus=REQUIRED`



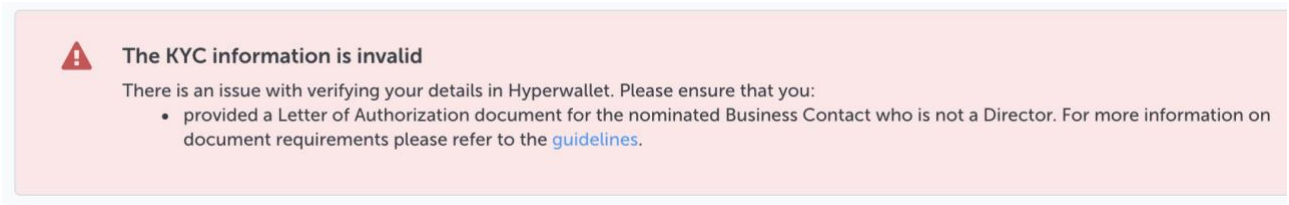
- **Business Seller:** notification received is `verificationStatus=REQUIRED`



- **Business Seller:** notification received is `businessStakeholderVerificationStatus=REQUIRED`



- **Business Seller:** notification received is `letterOfAuthorizationStatus=REQUIRED`


The KYC information is invalid
There is an issue with verifying your details in Hyperwallet. Please ensure that you:
provided a Letter of Authorization document for the nominated Business Contact who is not a Director. For more information on document requirements please refer to the guidelines.

5 Payout

The Payout feature in HMC covers payment of Mirakl accounting documents (i.e. invoices) to corresponding payees via Hyperwallet. This means that payout depends on the creation of Invoices in the Mirakl platform.

Mirakl runs the regular process of creating invoices on a billing cycle. The billing cycle is configurable, and depending on the operator's use cases can be set to run anywhere from monthly, specific day(s) of a month, or even daily. In addition, the billing cycle can be configured at the Seller level, meaning different sellers can have a different billing cycle. HMC was implemented to avoid the complexity of relying on Mirakl billing cycles. Instead, when running an invoice extract, the HMC queries for any new invoices generated since the last extract and processes the payouts for each qualifying invoice.

The following criteria determine if an invoice is eligible for automated payout via HMC:

- Both automatically generated invoices and manual credit notes can be extracted from Mirakl and sent for payout to Hyperwallet via HMC.
- Invoices with DRAFT or GENERATED state are not processed, HMC will only extract/process the invoices in COMPLETE state;
- Invoices with PAID payment_status are also omitted as no payout is required.

HMC supports 2 types of payees – Marketplace Sellers and Operators:

- Sellers receive payments for the goods and services sold (***transfer_amount*** on Mirakl invoice)
- Operator receive their operator fees from transactions placed on the marketplace (***total_commissions_incl_tax*** + ***total_subscription_incl_tax***).

4.4 Seller Payout

As per the Hyperwallet API spec, the TRM token (seller's bank account) and programme token will be used to target the correct Seller account.

Seller accounts and bank accounts will be created automatically via HMC, as described in section 3 Seller onboarding. No further configuration is required for HMC to enable Seller payout.

Payout is triggered through the *Extract Invoices* job, which extracts eligible invoices from Mirakl and creates payment requests in Hyperwallet. This job handles both Seller and Operator payout.

4.4.1 REST API Endpoints

Platform	Endpoint	Documentation
Mirakl	**(GET) IV01**	help.mirakl.net/help/api-doc/operator/mmp.html#IV01
Hyperwallet	**(POST) Create Payment**	docs.hyperwallet.com/content/api/v4/resources/payments/create

4.4.2 Data Requirements

Seller, Bank account and Invoice

- The Seller accounts should exist in Mirakl and be set up with one of the supported Bank Account payment methods and currencies, as described in section 3.3 Create/Update Seller Bank Account;
- Accounting documents (i.e. Invoices) should be created in Mirakl for the above sellers in order to trigger Payout feature.

Data mapping

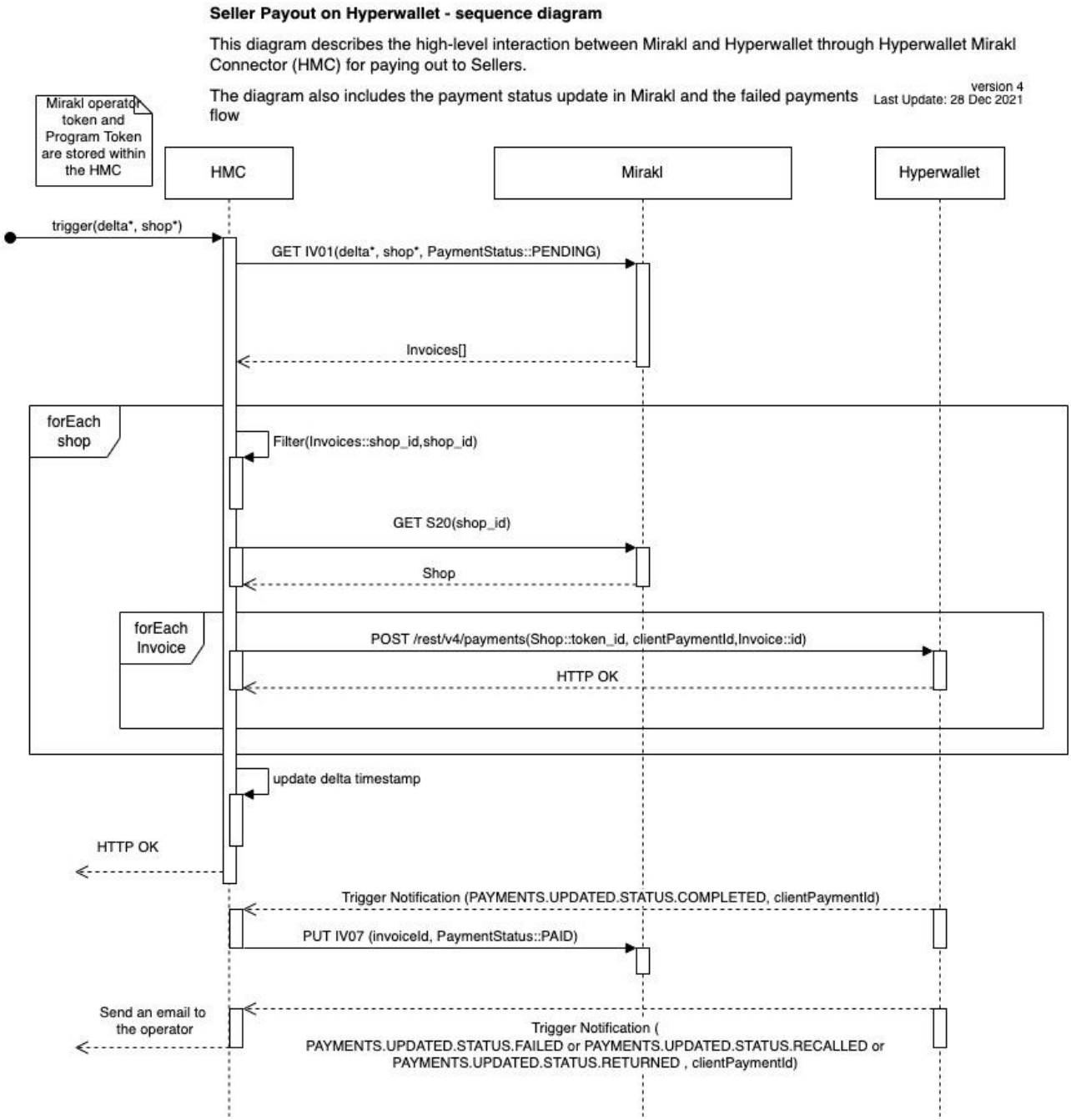
Mirakl IV01 API field	HW API field	Notes
total_charged_amount	"amount"	
invoice_id	"clientPaymentId"	
"currency_iso_code"	"currency"	
TRM token	"destinationToken"	The seller TRM token denoting the bank account in HW Embedded Experience.
Operator token in the Connector	"programToken"	Operator token
n/a	"purpose": "OTHER"	We will use OTHER for all payouts.

4.4.3 Timed Job Frequency

As with Seller extract jobs, this job is scheduled to run at certain configurable frequency, by default at 01.00 AM. The frequency can be changed if required, using the following Environment Variables (see README for further details such as format and default value):

- PAYPAL_HYPERWALLET_EXTRACT_INVOICES_CRON_EXPRESSION

4.4.4 Technical Flow



4.5 Operator Payout

The Operator is entitled to receive a payout for the commission and subscription charges that are applied to sellers' activities on the marketplace.

The Operator may choose not to receive their Operator fee, in which case their funds will be kept within the main funding account of the program in Hyperwallet. This can be configured in HMC properties as a toggle (on/off). See section 5.3.5 Enabling or disabling the operator payout below.

Only one marketplace operator exists for each instance of Mirakl. Setting up operator account Hyperwallet credentials in HMC will be handled manually. These details (including the program tokens and TRM tokens) should be provided to the Operator by the Hyperwallet team as part of the onboarding process.

Configuring operator payout details is described in section 2.3.1 Operator onboarding.

4.5.1 REST API Endpoint

Platform	Endpoint	Documentation
Mirakl	(GET) IV01	help.mirakl.net/help/api-doc/operator/mmp.html#IV01
Hyperwallet	(POST)	docs.hyperwallet.com/content/api/v4/resources/payments/create

4.5.2 Data Requirements

- The Seller accounts should exist in Mirakl and be set up with one of the supported Bank Account payment methods and currencies, as described in 3.3 Create/Update Seller Bank Account;
- Accounting documents (i.e. Invoices) should be created in Mirakl for the above sellers in order to trigger Payout feature.
- The Operator account should be created in Hyperwallet and relevant settings (user token, TRM) should be copied over to HMC properties.

4.5.3 Timed Frequency

Operator payout is processed as part of the same Payout job (*Extract Invoices*) as for the Seller Payout. See 5.2.4 in Seller Payout section.

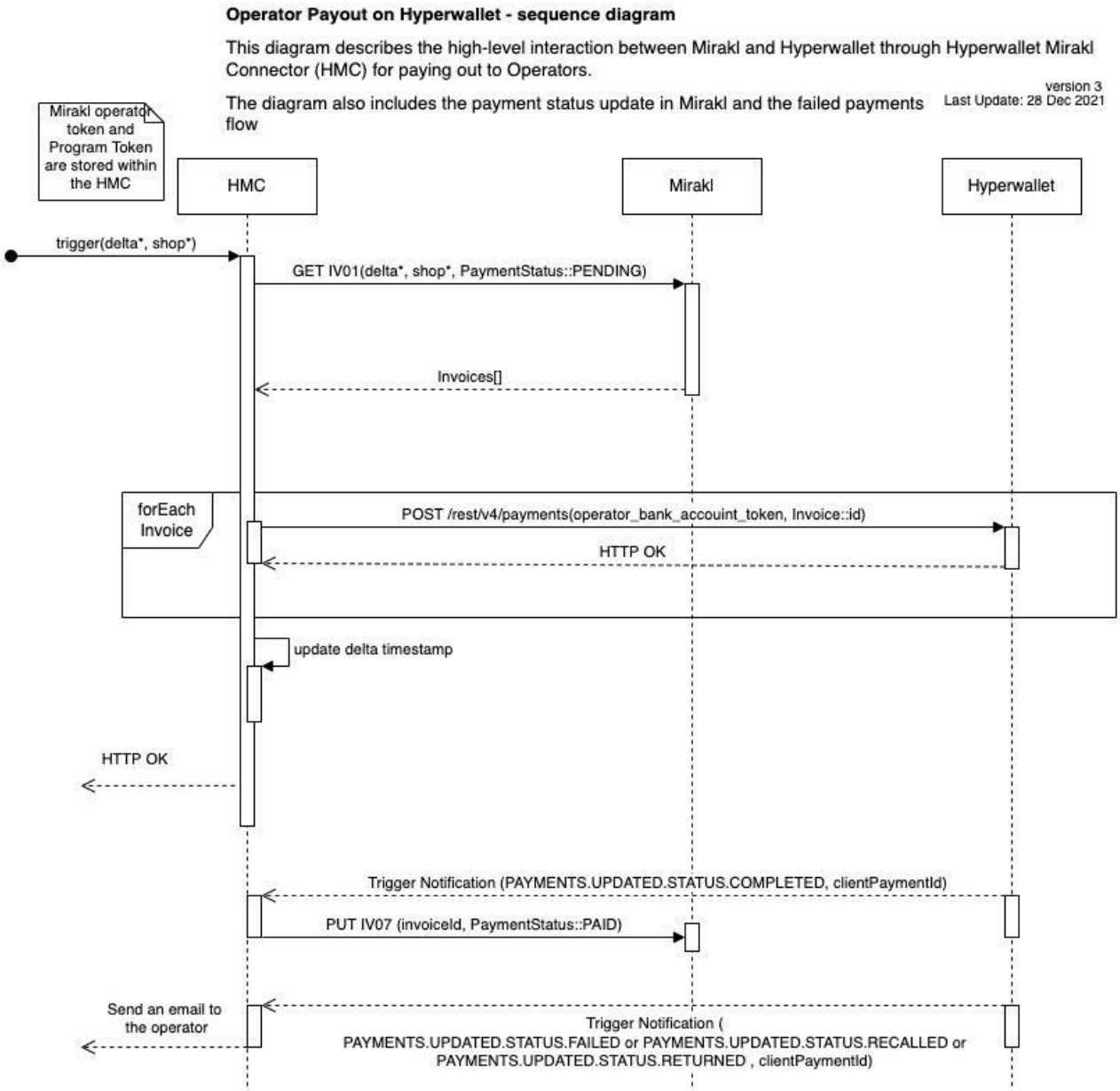
4.5.4 Enabling or disabling the operator payout

The following Environment Variable can be used to enable or disable the payout of the operator commission into the operator's bank account (see README for further details such as format and default value):

- PAYPAL_HYPERWALLET_OPERATOR_COMMISSIONS_ENABLED

This setting is ON by default, meaning that the operator's commission will be paid to the corresponding bank account set up for the operator. If disabled, the commission will be kept within the main funding account of the program in Hyperwallet.

4.5.5 Technical Flow



4.6 Payout Notifications

Payout notifications are received from Hyperwallet via a webhook.

Success notifications

When the payment status is changed to COMPLETED in Hyperwallet HMC will automatically set the status on the corresponding accounting document in Mirakl to PAID.

The payment confirmation is received by the HMC via a Payment webhook notification sent from Hyperwallet, with the status: PAYMENT.UDATED.STATUS.COMPLETED.

In regard to Operator payout, there is no Mirakl end-point for confirming that the operator has had commissions/subscriptions paid out for an invoice. It is assumed that the Hyperwallet platform will be responsible for notifying the operators about successful payouts.

Failure notifications

Handling of failed payout notifications applies to the following statuses received by HMC from Hyperwallet via the webhook:

- FAILED
- RECALLED
- RETURNED

When one of those statuses is received via the webhook, the HMC will send an email to the operator's email address (configured in properties). Below is the content of the email message:

> *Subject: Payment issue - <invoice ID>*
> *Body: There was an issue with payment of <invoice ID> invoice. The payment status is <HW STATE>. Please login to Hyperwallet to view and resolve the payment issue.*

Note: <invoice ID> will be provided in the format to distinguish the operator payment (e.g. "12345-operatorFee").

4.6.1 REST API Endpoints

Platform	Endpoint	Documentation
Mirakl	(PUT) IV07	help.mirakl.net/help/api-doc/operator/mmp.html#IV07
Hyperwallet	Webhook	docs.hyperwallet.com/content/webhooks/v1/notification-types/payments

4.7 Manual Credit Notes

HMC supports processing payments from the Operator to the Sellers via the creation of Manual Credit Notes in Mirakl. The processing works the same as for automatically generated invoice documents and as described in Section 5.1.

This is provided only to facilitate payment testing and has important consequences when used.

Since Mirakl includes manual credit notes in its billing cycle calculations, manual credit note amounts will be included in automatic invoices. **If the connector has manually paid a manual credit note and a seller has had orders/refunds in the same billing cycle, the amount in the credit note will be included and**

paid a second time as part of an automatic invoice. For this reason, it is strongly urged to only use manual credit notes in test environments while testing, to ensure that payments can be made in an end-to-end setup between Mirakl, the HMC, and Hyperwallet.

The Manual Credit Note processing feature is **disabled** by default, but can optionally be configured through the following Environment Variable (see README for further details such as format and default value):

- PAYPAL_HYPERWALLET_OPERATOR_CREDIT_NOTES_ENABLED

Note on Refunds:

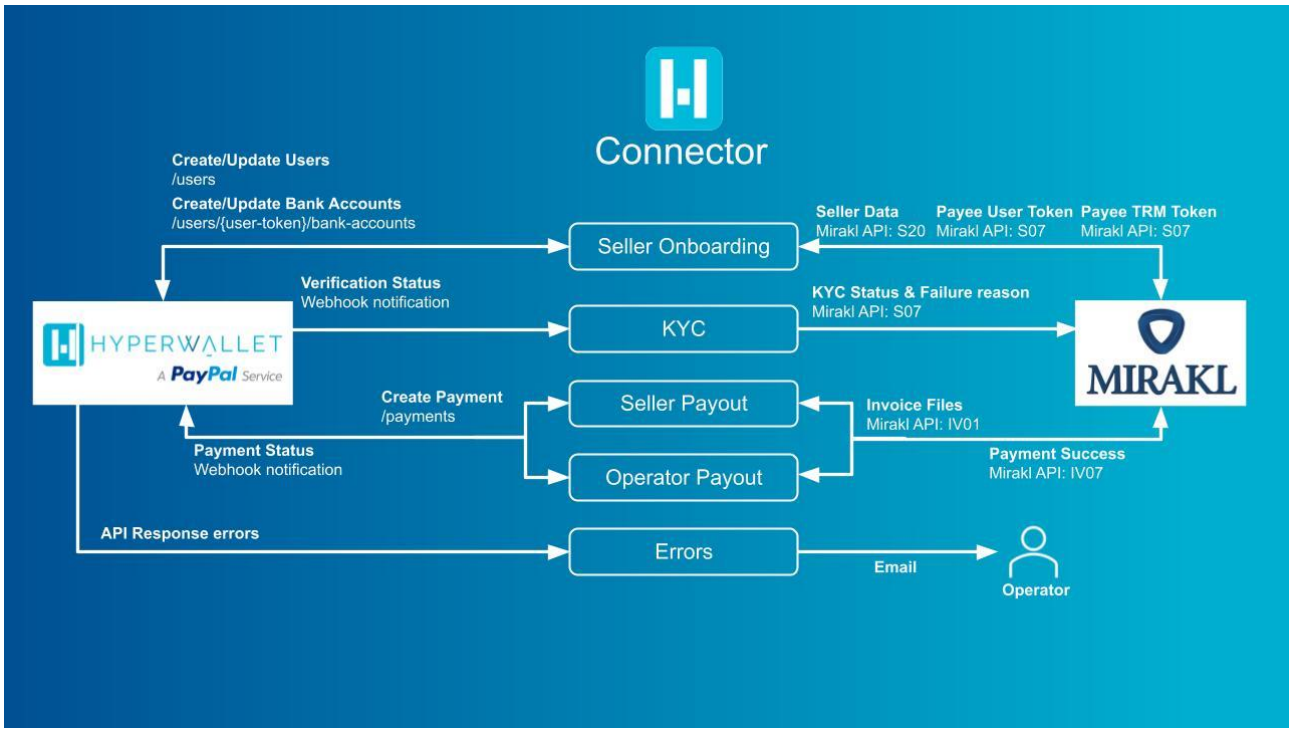
Please note that HMC is responsible for facilitating payout to the Sellers and Operator, which includes the money that is owed to the Seller or Operator.

Customer refunds are processed as part of the pay-in functionality and are covered by the Braintree-Mirakl connector scope. This is a separate connector service provided as part of the overall PayPal product offering.

In instances when the Seller balance in Mirakl becomes negative (e.g. to enable the Seller to honour customer refunds) Mirakl allows creation of Manual credit notes. This allows replenishment of the Seller bank account balance, which involves a payout element.

5 Hyperwallet-Mirakl Technical Integration

5.1 Overall Architecture



5.2 Error Handling

API call failures and other unexpected errors (e.g. missing mandatory fields in the payload) are handled by the HMC. When such errors occur, the operator will receive an email message to the email address configured in the HMC properties.

Error Types:

- Mirakl: Issue detected getting shops in Mirakl
- Mirakl: Issue detected updating KYC information in Mirakl
- Hyperwallet: Issue detected when creating seller in Hyperwallet
- Hyperwallet: Issue detected when updating seller in Hyperwallet
- Hyperwallet: Issue detected when creating bank account in Hyperwallet
- Hyperwallet: Issue detected when updating bank account in Hyperwallet

Error example (email to operator):

```
Subject: Issue detected when updating seller in Hyperwallet

Body: There was an error, please check the logs for further information:
Error updating user with clientUserId [2214]
{exceptionMessage=A system error has occurred. Please try again. If you continue to receive
this error, contact customer support for assistance (Ref ID: usr-dc7bf083-310d-4f04-a9b7-
13561f011e85).,error=CONSTRAINT_VIOLATIONS[[code=CONSTRAINT_VIOLATIONS,fieldName=<null>,messa
ge=A system error has occurred. Please try again. If you continue to receive this error,
contact customer support for assistance (Ref ID: usr-dc7bf083-310d-4f04-a9b7-
13561f011e85).,relatedResources=<null>][code=CONSTRAINT_VIOLATIONS,fieldName=<null>,message=,
relatedResources=<null>]]}
```

If the Seller Extract or Bank Account Extract jobs end in failure, any data that wasn't processed due to the failure will be processed during next extract.

Re-run previously failed actions

HMC has built-in auto-recovery functionality for unexpected technical failures such as API timeout and IO exceptions. If any of such failures occur during creation and update of user accounts (both Individual and Busines) or bank accounts HMC will attempt to re-run the previously failed action.

This action will be attempted once, and if the error occurs again on the re-run action it will be logged and communicated to the operator as described in this section above.

When communicating with Hyperwallet HMC is using a default timeout set in the Hyperwallet SDK. The timeout for communicating with Mirakl API is up to 90 seconds (30 seconds for connection and 60 seconds for reading the records).

5.3 Payload Encryption

Hyperwallet provides payload encryption for webhooks and API communication which is also supported by HMC. This involves a process where webhook and API data is signed and encrypted when sent from Hyperwallet to HMC and vice versa. The receiving application (Hyperwallet or HMC, depending on where the data is sent to) is then responsible for validating the signature and decrypting the data.

For further information on payload encryption please refer to the Hyperwallet Payload Encryption specification docs.hyperwallet.com/content/api/v4/overview/payload-encryption.

The Payload Encryption feature is by default disabled in HMC upon installation and can be enabled using the following property adding the profile `encrypted` to the Environment Variable `PAYPAL_SPRING_PROFILE_ACTIVE`.

5.4 External Technical Documentation

- Mirakl

API docs: help.mirakl.net/Customers/topics/home_pages/api_docs.htm

SDK/Connectors: help.mirakl.net/Customers/topics/home_pages/connectors.htm

- Hyperwallet

API docs: docs.hyperwallet.com/content/hyperwallet-payout-documentation

SDK: docs.hyperwallet.com/content/api/v4/overview/sdk

5.5 Storing and filtering notifications

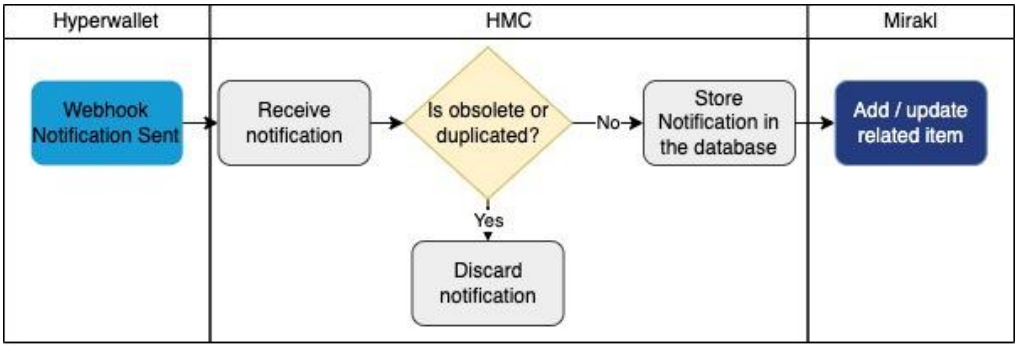
Hyperwallet API (as any other existing API) cannot guarantee the order of the notifications received for a specific item (seller, stakeholder, payment…), therefore, this could cause issues changing the KYC status of sellers that were already verified by Hyperwallet moving them back to be verified again or not properly updating the payments, causing inconsistencies in the info between Hyperwallet and Mirakl platforms.

To avoid such possible issues, the HMC includes a functionality to store all the notifications received and filter out the obsolete and duplicated notifications for a specific item, processing only the necessary notifications.

Functionality behaviour

As soon as a notification is received in the HMC, it will be stored in the database, except for two different cases where notifications will be skipped:

- Duplicated notification: the same notification already received and stored in the database
- Obsolete notification: a notification for the same item (a specific seller, stakeholder...), but created before the existing one already stored in the database.



The following information is stored in the database in the NotificationEntity table:

Database field	Data type	Notes
Id	Long	Autogenerated ID
webhookToken	String	Token of the notification
objectToken	String	Token of the related item
creationDate	Date	Creation date of the notification on HW
receptionDate	Date	Date the notification was received in the HMC
notificationType	Date	Type of the notification, containing one of the following possible values: USR, PMT, STK, TRM, UNK

The notification type is not received in the Hyperwallet webhook, however, the three first characters of the token contain the type of the notification.

The possible values that will be stored in the database are:

- USR → Sellers
- STK → Stakeholders
- PMT → Paymets (invoices)

- TRM → Bank account
- UNK → Unknown

UNK value will not be received in the webhook, however if the HMC receives an unknown notification type (not included in the recognised values USR, STK, PMT or TRM), this value will be stored to avoid any failure cause the notification type is not a "valid" one.

Notifications querying and housekeeping

Two different endpoints are available to query or to remove the notifications stored in the database.

To query the notifications currently stored in the database, an endpoint using the GET HTTP method is available under /webhooks/notifications path.

To avoid the database size increasing without any limit, an endpoint using the DELETE HTTP method is available under /webhooks/notifications path.

Both endpoints include two parameters ("from" and "to") so that a range of dates are provided in order to query/remove the notifications received within such period.
Both parameters are mandatory to avoid problems removing notifications accidentally.

HMC has not created any job to remove the notifications, and the customer is responsible on triggering (manually or automatically) the mentioned endpoint to remove the notifications (if needed) based on their company policies.

Example of valid execution request for both querying and removing notifications:

```
curl --location --request GET 'http://localhost:8080/webhooks/notifications?from=2021-04-
27T10:30:00.000-00:00&to=2023-04-27T10:30:00.000-00:00'
```

```
curl --location --request DELETE 'http://localhost:8080/webhooks/notifications?from=2021-04-
27T10:30:00.000-00:00&to=2023-04-27T10:30:00.000-00:00'
```

5.6 Notifications retry mechanism

In order to improve the reliability of the HMC, a mechanism has been implemented to retry the processing of the notifications received from Hyperwallet just in case it is not possible to upload the information in Mirakl for any reason.

The issue rarely happens, and it can be easily identified. This could not cause a severe problem with the payment notifications as the workaround would be as simple as updating the payment status in the Mirakl backoffice, however for the KYC notifications there is not a friendly solution and using an external tool to execute a request to one of the Mirakl API's is needed.

To avoid these manual workarounds and automatically solve the inconsistences produced due to the failure update on Mirakl, the retry notifications mechanism can be enabled in the HMC.

Sensitive information is never stored in the database nor displayed in the logs, only the tokens are used to identify the notifications.

Functionality behaviour

The tokens of the notifications failing when trying to update information on Mirakl will be stored in the database, so that a periodical job retrieve such notifications from Hyperwallet and then try to update Mirakl again until they reach the maximum number of attempts allowed (configurable using an environment variable as explained below)

Following scenarios could happen when a received notification fails and then checks if it already exists in the failed notifications information:

- **Already existing notification** → the notification is skipped as it is identical to the previous one.
- **Updated existing notification** → the same notification with a creation date later than the existing one is received and then the previous one is removed from the database.
- **Outdated existing notification** → the same notification with a creation date before the existing one is received and then the notification is skipped.

When a specific notification is retried and fails, reaching the maximum number of allowed attempts, such notification is removed from the database and an email will be sent to the operator.

Retrying the notifications

All the failed notifications stored in the database are retried periodically using a cronjob that will be executed every 15 minutes by default (configurable using an environment variable as explained below).

The job execution will check all failed notifications stored in the database trying to process each one again. If the notification is properly processed successfully updating the information on Mirakl, the notification will be removed from the database, otherwise, the number of retries will be increased for such notification unless it reaches the maximum number of attempts causing the notification won't be retried anymore removing it from the database.

Configuring the feature

The retry mechanism can be configured using the following environment variables:

- **PAYPAL_HYPERWALLET_RETRY_NOTIFICATIONS** → This variable can be set to ***"true"*** to enable the notifications retry mechanism, or ***"false"*** to deactivate it. Set to ***"true"*** by default.
- **PAYPAL_HYPERWALLET_MAX_AMOUNT_OF_NOTIFICATION_RETRIES** → this variable determines the number of maximum attempts allowed for each notification. Set to ***"5"*** by default
- **PAYPAL_HYPERWALLET_RETRY_FAILED_NOTIFICATIONS_CRON_EXPRESSION** → used to **configure the periodicity of the cron job**

Database type

Data	Database field	Data type	Notes
Notification token	notificationToken	String	Token of the specific notification
Notification type	type	String	Payment, user or stakeholder update, etc…
Target token	target	String	Token of the notification object

Program token	programToken	String	
Number of retries	retryCounter	Int	Current number of times the failed notification has been retried
Creation date	creationDate	Date	Creation date of the notification on HW

Notify the failures

In order to notify the operator about the failed notifications, an email will be sent every time a notification reaches its maximum number of attempts, including the following details:

> ***Subject**: [HMC] Technical error occurred when processing the notification N*
>
> ***Body**: There was an error processing the notification 'N' and the operation could not be completed. The maximum number of attempts 'M' has been reached, therefore it will not try to re-process the notification anymore. Please check the logs for further information*

Logs also include traces for the following cases

- A failed notification if going to be stored and retried.
- An outdated notification has been received and will be skipped.
- An updated notification has been received, and the previous one will be removed.
- A notification has exceeded the maximum number of allowed attempts.

6 Deployment and setup

6.1 Getting Started with HMC

HMC can be deployed as Java/Spring service or as a Docker container.

For setup and build details please refer to the *README* file in the main HMC repository.

6.2 Setting up and running jobs

There 5 jobs in HMC:

Job	Description	HTTP method	Endpoint
Individual sellers extract	Extract new individual seller data from Mirakl and perform create/update on Hyperwallet.	POST	/job/sellers-extract

Professional sellers extract	Extract new professional seller data from Mirakl and perform create/update on Hyperwallet.	POST	/job/professional-sellers-extract
Bank Accounts extract	Extract bank account data from sellers in Mirakl and create/update a bank account on Hyperwallet associated to the corresponding user in Hyperwallet	POST	/job/bank-accounts-extract
Invoices extract	Extract new invoices and create payment requests in Hyperwallet for Seller and Operator payout.	POST	/job/invoices-extract

Those jobs are currently setup with the following Environment Variables, as follows:

Environment Variable	Default Cron expression
PAYPAL_HYPERWALLET_EXTRACT_SELLERS_CRON_EXPRESSION	0 0 0 1/1 * ? *
PAYPAL_HYPERWALLET_EXTRACT_PROFESSIONAL_SELLERS_CRON_EXPRESSION	0 0 0 1/1 * ? *
PAYPAL_HYPERWALLET_BANK_ACCOUNT_EXTRACT_CRON_EXPRESSION	0 30 0 1/1 * ? *
PAYPAL_HYPERWALLET_EXTRACT_INVOICES_CRON_EXPRESSION	1 0 0 1/1 * ? *

These jobs are scheduled to run at regular intervals (time of day, as per the cron expression above).

Jobs can also be executed manually through their endpoints.

All of the endpoints support 2 optional parameters:

Param	Description	Format
delta	When filled the extract jobs will filter values to be updated/created from this date onwards. Using a value far into the past runs the risk of having the connector re-process many old entries, which will likely cause the Hyperwallet platform to return numerous 'ID already exists' errors	yyyy-MM-dd'T'HH:mm:ss.SSSXXX
name	When filled the job instance running will be assigned the given name	String

Example of valid execution request:

```
curl --location --request POST 'http://localhost:8080/job/bank-accounts-extract?delta=2020-11-22T11:52:00.000-00:00&name=bankAccountExtractJob'
```

The HMC repository contains an example Postman collection in the /docs folder, which can be imported and used as a starting point for running jobs manually using the Postman API tool.

7.2.1 Access Rules

The exposed connector endpoints should only be accessible by operator-network locations, and for the purposes of triggering jobs manually.

The webhook listener described in section 2.2 should only be accessible from Hyperwallet's IP range described in the official documentation page

7.2.2 Data initialization

It is recommended to run first time all the jobs with a delta time set in a time before the creation of any data in Mirakl to ensure all the existing data in Mirakl environment is exported into Hyperwallet

7.2.3 Data storage

The connector has 3 different database connections, all these database connections are currently setup as embedded H2 Database files. The files are stored in the directory: data.

The data stored on those databases are:

- Last delta execution time for a specific job
- Mirakl Seller/shop ID's when a seller can't be created in Hyperwallet due to connectivity issues during a job execution. This data will be used in the next job execution for retrying the import process for those specific Mirakl sellers.

7.2.4 Time Zones

Mirakl stores all dates, including Date custom fields, with times and time zones. For example, when a user in France enters a date of birth for a business stakeholder of '1980-01-01', Mirakl will store this as '1980-01-01-00:00:00 UTC+1'. When the connector retrieves these dates from Mirakl, it needs to be able to provide the time zone that will ensure the intended day is preserved, and not translated into an unintended value such as '1979-12-31 23:00:00 UTC'.

To ensure that the correct dates are always preserved, the connector can be configured with the time zone of the Mirakl operator instance, in order to adjust any times that are retrieved via the Mirakl API.

The time zone configuration is set with the following Environment Variable:

- `PAYPAL_MIRAKL_OPERATOR_TIME_ZONE`

Setting this value to a TZDB region (e.g. Europe/London) is preferred as this will help to avoid issues with DST.

6.3 Logging & log file

By default, the connector produces detailed log entries for every action that it takes, including both successful and failed actions. The logs can therefore be a useful troubleshooting tool while testing & running the connector.

Logfile	Location
application.log	paypal/logs/application.log

Each log line contains a timestamp, the module and code-level class that produced it, and a descriptive text explaining what has occurred.

Any errors that come from Hyperwallet or Mirakl are reproduced exactly in HMC's logs.

Example of log lines:

```
2021-04-27 14:08:42.553 ERROR 1 --- [eduler_Worker-1]
ctHyperwalletBankAccountRetryApiStrategy : Bank account not created or updated for
seller with clientId [41342]
2021-04-27 14:08:42.553 ERROR 1 --- [eduler_Worker-1]
ctHyperwalletBankAccountRetryApiStrategy : {exceptionMessage=Branch Sorting Code
Invalid
entry..,error=CONSTRAINT_VIOLATIONS[[code=CONSTRAINT_VIOLATIONS,fieldName=bankId,messag
e=Branch Sorting Code Invalid entry..,relatedResources=<null>]]}
```

6.4 Deployment profiles

HMC has the option to run under different execution profiles, which determine whether certain features are enabled or disabled (such as payload encryption) or if the connections to certain system components (for example webhook processing) are real or mocked (faked).

Profiles are set using the **PAYPAL_SPRING_PROFILE_ACTIVE** environment variable and are documented in the README file contained in the HMC repository.

The general execution profiles are **dev**, **qa**, and **prod**

For production and UAT testing purposes, it is strongly encouraged to only use the prod profile, along with any of the optional features that need to be enabled.

6.5 Mirakl Configuration

7.5.1 Seller Custom Fields

During the initial setup, the custom fields and sections described here must be created in Mirakl, in the Settings > Advanced parameters > Shops section of the Mirakl Operator back office.

The Mirakl documentation describes how to create and manage custom fields:
help.mirakl.net/Customers/topics/Mirakl/mmp/Operator/config_custom_fields/config_custom_fields.html

Note: all the custom fields listed in this section need to be configured in Mirakl in order for HMC to function properly. The last column ("Required/Optional") indicated whether the value in this field need to be provided in Mirakl or not, based on the data requirements outlined in Hyperwallet documentation:

- Required – the value must be provided for this field.
- Optional – the value is not required but can be provided.
- n/a – the value should not be provided by the user and will be auto-generated.

7.5.1.1 Section 1 – Hyperwallet Seller/Payee Details – Operator-only fields

Section Name: Hyperwallet Seller/Payee Details - Operator-only fields

Section Description: The fields in this section are for assisting in the operator's administration & operation tasks. All fields herein should be Invisible to sellers.

Section Fields:

Code	Label	Description	Type	Values	Shop permissions	Required/ Optional
hw-program	Hyperwallet Program	Your Hyperwallet implementation may consist of one or more programs based on your payout needs. Select the appropriate program for this Seller/Payee.	Single value list	Values configured by the operator based on the program hierarchy agreed with Hyperwallet. This is set to DEFAULT for a single-level merchant hierarchy. New values can be added in case of multi-level program hierarchy. Please refer to Multi-program hierarchy section of this document.	Invisible	Required
hw-user-token	Hyperwallet User Token	Auto-generated, DO NOT change this value. This is a unique identifier for this Seller/Payee in Hyperwallet.	Text	Autogenerated	Invisible	n/a
hw-bankaccount-token	Hyperwallet Bank Account Token	Auto-generated, DO NOT change this value. This is a unique identifier for this Seller/Payee's bank account in Hyperwallet.	Text	Autogenerated	Invisible	n/a
hw-bankaccount-state	Bank Account State/Province	Seller/Payee's Bank Account State/Province	Text	Free text	Invisible	Required

7.5.1.2 Section 5 – Hyperwallet Ts & Cs and Privacy Policy consent

Note: this section is placed at the end of the Seller custom fields section, before the Contact details on the Seller Details page.

Section Name: Hyperwallet Ts & Cs and Privacy Policy consent

Section Description: Payment Services for your Store will be provided by Hyperwallet, a PayPal company. In order to use this marketplace, you are required to accept the Hyperwallet Terms of Services and Privacy Policy

Section Fields:

Code	Label	Description	Type	Values	Shop permissions	Required/ Optional
hw-terms-consent	I accept the Hyperwallet Terms of Services and Privacy Policy: https://hyperwallet.com/agreements-terms	By accepting these terms I also agree that Hyperwallet may contact me directly to request additional information to verify my account. I will contact the marketplace operator if I need to revoke this.	Yes/No		Read Write	required

7.5.2 Mirakl Features to Activate

The following features need to be activated on the Operator's Mirakl environments. Please contact Mirakl Support in order to activate each of these features:

"KYC status": help.mirakl.net/Customers/topics/Mirakl/PSP_Project/topics/validating_kyc_psp.htm

"Payment Confirmation": help.mirakl.net/Customers/topics/Mirakl/integrating_mirakl/pay_sellers/payment_confirmation.htm

"Multiple Currencies": help.mirakl.net/Customers/topics/Mirakl/mmp/Operator/config_languages_currencies/Currencies/config_currencies.html