Offline Payment API (OPA)

+

Marketing API (MAP)

Version 2.1.18

Last updated on 15 Aug 2025

RAZER MERCHANT SERVICES

J-39-1, Block J, Persiaran Multimedia, i-City,
40000 Shah Alam, Selangor, Malaysia.

+(603) - 5521 8438

support@Fiuu.com

Fiuu.com

SCAN ME

Social Networks

https://twitter.com/FiuuPayment
https://facebook.com/FiuuPayment
https://youtube.com/@FiuuPayment
https://instagram.com/FiuuPayment
https://linkedin.com/company/FiuuPayment

Developer Platforms

https://t.me/FiuuDeveloperForum (for tech support)
https://github.com/FiuuPayment
Mobile XDK, seamless and in-page checkout, and many shopping carts payment plugin/module/addon/extension are available

PCI DSS COMPLIANT

ISO 27001

Revision

Date	Version	Author(s)	Description
18/07/2016	1.2	MOL	• Standardized signature. • Renamed Reversal & Refund endpoints. • Renamed ResponseCode to StatusCode • Renamed UserData to PayerId • Standardized MOL Transaction Id to molTransactionId. • Extended StatusCode for Reversed/Refunded. • Added Error Code list • Standardized Refund/Reversal Reference Id to paymentReferenceId • Added message example
30/08/2016	1.2.1	MOL	• Added “businessDate” on payment request. • Renamed “TransactionDate” to “BusinessDate” on recon file.
05/09/2016	1.2.2	MOL	• Added “ChannelId” to recon file and rearrange the sequence.
07/09/2016	1.2.3	MOL	• Added “StoreId” & “TerminalId” to transaction recon file. • Added summary recon file.
13/09/2016	1.2.4	MOL	• Updated “businessDate” into example for Payment, Refund and Reversal
06/10/2016	1.2.5	MOL	• Added new Store Summary Reconciliation File.
14/04/2017	1.2.6	MOL	• Added BusinessDate to store summary reconciliation file
06/06/2017	1.2.6.a	MOL	• Customized reconciliation to group by Merchant ID instead of App Code.
23/06/2017	1.3.0	MOL	• Updated sandbox URL. • Fixed 5.1 Signature Generation sample. • Removed reversal & refund flow diagram to avoid unnecessary confusion. • Removed offline to online flow to avoid unnecessary confusion. • Renamed transaction status “unknown” to “pending”. • Removed transaction status “pending authorize” to avoid unnecessary confusion. • Moved transaction status “reversed/refunded” to new error code. • Removed MOLWallet channel to avoid unnecessary confusion.
28/06/2017	1.3.1	MOL	• Fixed 5.1 Signature Generation sample.
28/06/2017	1.4.0	MOL	• Added a new optional parameter “hashType” to support signature generated using HMAC-SHA256. • Added HMAC-SHA256 signature generation example.

21/11/2017	1.4.1	MOL	• Added One2pay channel.
15/12/2017	1.4.2	MOL	• Update recon file sample screenshot
12/02/2018	1.4.3	MOL	• Added WeChat Pay channel.
17/04/2018	1.4.4	MOL	• Added Payment Flow Overview • Added Pre-Create Transaction QR Code • Added Payment Notification
04/05/2018	1.4.5	MOL	• Added status code (pending authorize)
31/05/2018	1.4.6	MOL	• Added E-Commerce Module • Update “ChannelId” parameters as optional parameter in payment service
28/06/2018	1.4.7	MOL	• Replaced One2pay with Razer Pay
16/10/2018	1.4.8	MOLPay	This API cutover to production on MOLPay system
17/10/2018	1.4.8	Fatihi	• Standardize all the referenceId length to 40 • Added WeChat Pay MY channel.
13/02/2019	1.4.9	Shang Qin	• Added new parameter for preCreate ○ Request ■ imageFormat ■ imageSize ○ Response ■ customImageURL
10/04/2019	1.4.10	Shang Qin	• Add new **wallet** channels:- ○ Touch `n Go Digital ○ Boost ○ Maybank QRPay
27/05/2019	1.5.0	Shang Qin	• Add new **wallet** channel - Alipay Pre-Auth • Change ***channelId*** to mandatory field (Payment API)
20/08/2019	1.5.1	Shang Qin	• Add new **wallet** channels - GrabPay merchant presented QRC
11/09/2019	1.5.2	Shang Qin	• Fixing reversal signature bug during error happens (will be deployed and effective any time once major partners/merchants have applied the resolution)
03/12/2019	1.6.0	Hafizi	• Add new ***MAP (Marketing API)*** for e-voucher & loyalty program
02/02/2020	2.0.0	Shang Qin	• Add v2 for all OPA API (Refer to section Version) by adding in channelId in all responses
08/04/2020	2.1.0	Shang Qin	• Revamp the reconciliation report for merchant. No longer using sftp. Merchants will need to call API to retrieve the daily records.
10/06/2020	2.1.1	Chen Yaau	• Add new **wallet** channels - ShopeePay

24/06/2020	2.1.2	Shang Qin	• Update all URLs to Razer's subdomain
16/10/2020	2.1.3	Shang Qin	• Add new **wallet** channels - UnionPay

11/08/2021	2.1.4	Chen Yaau	• Add new **online banking**/**wallet** channels:
			  ○ DuitNow QR
			  ○ Alipay+
			• Remove Razer Pay **wallet** channel

06/10/2021	2.1.5	Chen Yaau	• Adding new **BNPL** channel - Atome
28/02/2022	2.1.6	Hafizi	• Fix Signature Calculation Sample (HMAC)
26/08/2022	2.1.7	Chen Yaau	• Adding ShopeePay merchant presented mode
30/11/2022	2.1.8	Chen Yaau	• Adding/modifying error codes
02/05/2023	2.1.9	Chen Yaau	• Adding extraInfo
11/07/2023	2.1.10	Safwan	• Adding PayNow channel
06/11/2023	2.1.11	Chen Yaau	• Adding validityDuration
08/05/2024	2.1.12	Choo	• Adding Channel Validity Duration table
29/05/2024	2.1.13	Safwan	• Adding QRPH and KBank QR channel
12/08/2024	2.1.14	Yao Song	• Adding Error code 1015
06/02/2024	2.1.15	Hafiz	• Replace latest production service URL
20/03/2024	2.1.16	Puvaan	• Adding metadata in v4
08/04/2024	2.1.17	Yao Song	• Extend PayNow Min & Max wait time
15/08/2025	2.1.18	Chen Yaau	• Adding **40010** error code.

Abbreviation

RMS / Fiuu	Razer Merchant Services, a payment gateway business unit under Razer Fintech group
PG	payment gateway
OPA	Offline Payment API
TNG-D	Touch'n Go Digital
MY, CN	Country name or code, i.e. Malaysia, China
MYR, CNY	Currency code, i.e. Malaysia Ringgit, China Yen or Renminbi

Table of Contents

1. Introduction

The Offline Payment API offers merchants an integration platform to collect e-wallet in-store payment using both merchant presented and buyer presented Payment Barcode/QR Code.

The use cases supported are POS integration, payment terminal integration, or mobile APP integration. Retailers, chain stores especially F&B outlets, vending machine providers, unmanned store operators, parking management operators and many more are the most popular merchants and partners in e-wallet acceptance.

This Offline Payment API provides a secure payment experience with server-to-server communication over a secure socket layer (SSL) and conforms to Representational State Transfer (RESTful) architectural style and uses JSON as its data representation format.

Following are list of payment providers supported by Offline Payment API:

- ## Razer Pay

  Razer Pay is an online mobile payment processing and money transfer e-wallet application that is intended to replace the physical wallet with a mobile phone. For more information about Razer Pay, please visit https://pay.razer.com/.

Channel Id	15
Payment Currency	MYR, SGD

- ## Alipay

  Alipay.com is one of the largest online payment platforms in China. It was launched in China in 2004 by Alibaba Group and its founder Jack Ma. For more information about Alipay, please visit www.alipay.com.

Channel Id	16
Payment Currency	MYR

- ## Touch `n Go Digital

The Touch 'n Go eWallet is an electronic wallet (e-wallet) that holds electronic money (e-money). This service via mobile application is offered by TNG Digital Sdn.Bhd. (TNGD). For more information about Touch n Go digital, please visit https://www.tngdigital.com.my/.

Channel Id	17
Payment Currency	MYR

- ## Alipay Pre-Auth

Pre-authorization is also well known as authorization hold which is within the banking industry of verifying electronic transactions initiated with the account and rendering this balance as unavailable until the merchant clears the transaction. As the trade initiates, the merchant sends a pre-auth request to Alipay for a certain amount of funds. Successful pre-auth means that the user authorizes the merchant to acquire the funds. At the closure of this trade, the merchant can capture the funds according to the actual costs and fees.

Channel Id	18
Payment Currency	MYR

- ## Boost

Boost is a mobile wallet app that aims to revolutionize the way you transact in today's increasingly digital and mobile world. For more information about Boost, please visit https://www.myboost.com.my/.

Channel Id	19
Payment Currency	MYR

- ## MAE by Maybank2u

MAE Scan & Pay (previously known as QRPay) is a cashless payment solution that enables customers to make payments to merchants using a unique two-dimensional quick-response (QR) code.

Channel Id	20
Payment Currency	MYR

- ## GrabPay

GrabPay is a safe, convenient and flexible mobile wallet to pay both for services on the Grab app and in stores and restaurants. This launch is teamed with partnerships with local champions like Maybank, KLIA Ekspres and merchants such as Tealive, ensure GrabPay is not only accepted nationwide in eight cities, but can also be used for a variety of services.

Channel Id	21
Payment Currency	MYR

- ## UnionPay

The “UnionPay” mobile application is the unified mobile payment portal of China’s banking industry, developed and launched by UnionPay together with commercial banks and payment institutions in China. The app integrates the mobile payment functions, special services and benefits of various banks and institutions, providing secure and convenient one-stop mobile payment services for its users. https://www.unionpayintl.com/en/servicesProducts/products/innovativeProducts/mobilePayment/

Channel Id	22
Payment Currency	MYR

- ## ShopeePay

ShopeePay is Shopee's official in-app ewallet, it helps to store money from your refunds and top-up which can then be used to pay for your next order via online (from Shopee App or Web) or even Offline Deals.

Channel Id	23
Payment Currency	MYR

- ## DuitNow QR

DuitNow QR is Malaysia's National QR Standard established by PayNet under the BNM's Interoperable Credit Transfer Framework. Through DuitNow QR's interoperability, any compliant QR Code can take payments from any participating Banks and e-Wallets.

Channel Id	24
Payment Currency	MYR

- ## Alipay+ (Cross-border)

Alipay+ is a payment method provided by Ant Financial that allows user make payment via all wallets under Alipay+ network such as KakaoPay (Korea), TrueMoney (Thailand), EZLink (Singapore), Dana (Indonesia), and etc..

Channel Id	25
Payment Currency	MYR

Atome

Atome is a leading “Buy Now, Pay Later" brand in Asia. It provides short-term payment plans that allow individuals to make a purchase and pay for it at a later time with no interest or service fees.

Channel Id	26
Payment Currency	MYR

WeChat Pay (Cross-border)

WeChat Pay is one of the largest online payment platforms in China. It is a digital wallet service incorporated into WeChat, which allows users to perform mobile payments and send money between contacts. For more information about WeChat Pay, please visit pay.weixin.qq.com.

Channel Id	36
Payment Currency	MYR

WeChat Pay Malaysia

WeChat Pay Malaysia is for the Malaysia Wallet. It is a digital wallet service incorporated into WeChat, which allows users to perform mobile payments and send money between contacts.

Channel Id	37
Payment Currency	MYR

- ## PayNow

  PayNow is a popular electronic payment method in Singapore that enables individuals and businesses to make quick and secure transactions using their mobile number, National Registration Identity Card (NRIC), or Unique Entity Number (UEN).

Channel Id	38
Payment Currency	SGD

- ## KBank QR

  KBank QR is a convenient and secure payment method offered by Kasikornbank in Thailand. It allows individuals and businesses to make instant transactions by scanning QR codes with their mobile banking app, facilitating quick and cashless payments directly from their bank accounts.

Channel Id	39
Payment Currency	THB

- ## QRPH

  QRPH (QR Ph) is an innovative electronic payment system in the Philippines that enables individuals and businesses to conduct fast and secure transactions using QR codes. By scanning the QR code with their mobile banking or e-wallet app, users can effortlessly make payments directly from their accounts, promoting a cashless and efficient payment experience.

Channel Id	40
Payment Currency	PHP

2. Security Features

Offline Payment API service is protected for only authorized merchants with a hardened platform to secure payment data transmission.

- ## Secure Sockets Layer (SSL) data transport

  It's required to use HTTPS for all interchange messages between merchants and the payment gateway. This is to prevent any sensitive data being revealed by an unauthorized party during message exchange.

- ## Transport Layer Security (TLS)

  It's a protocol that provides authentication, privacy, and data integrity between two communicating computer applications. Used for web browsers and other applications that require data to be securely exchanged over a network. **The required TLS for this API is TLS 1.2 & above.**

- ## IP Address Filtering

  Merchant or partner is recommended to register and get their server's static IP address whitelisted at the payment gateway if heavy traffic is expected.

- ## Data Message Protection (Signature)

  This is an application layer security in ensuring data integrity. All data in the message exchange will be hashed using a unique Secret Key and output as Signature. Secret Key is assigned to merchants during account creation. Payment gateway will validate this Signature to prevent any data tampering during the message exchange. It's also STRONGLY recommended for merchants to perform the same validation for all response messages received from payment gateway.

3. Get Started

3.1 Merchant Application Account

Before merchants start integrating with Offline Payment API, merchants must register an application account with a payment gateway. For every Application Code provided it will have its own secret key. If merchants already have an account then they may skip this step.

Payment gateway will provide the account information as below for merchant integration and production processing purposes.

Application Code	Unique code to identify merchant applications which integrate with Offline Payment API.
Secret Key	A server-side shared secret key which will be used to generate signatures for API communication.

3.2 IP Address Whitelist

This is optional and applicable only for merchants or partners that process huge volumes (more than 10 transactions per second). After the merchant obtains the Offline Payment API account, the merchant is recommended to provide outgoing IP address(es) of the merchant server for whitelisting purposes. This is to ensure only servers authorized by merchants are able to connect to the payment gateway.

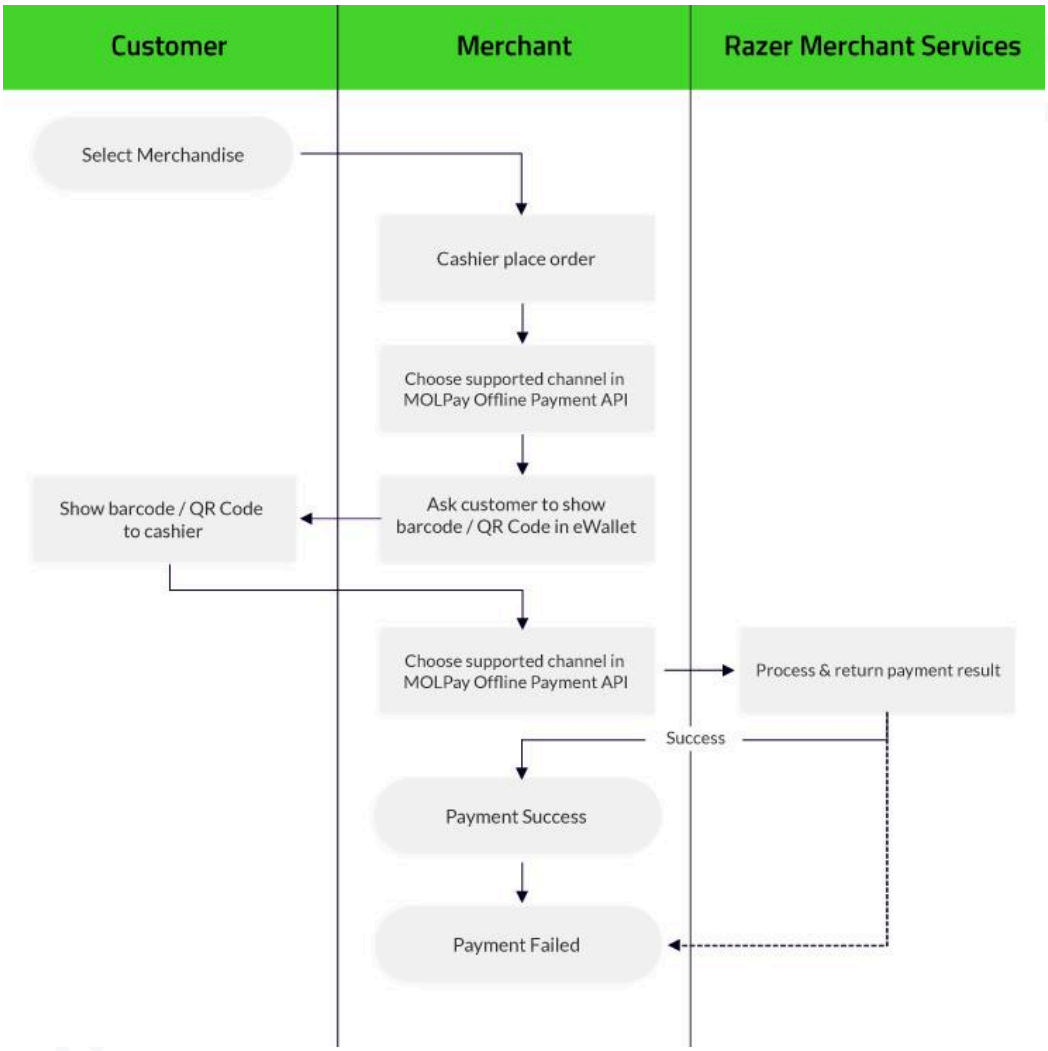
3.3 Version

The version represents the functionality or enhancement provided by each iteration. Use the version that best fits your integration.

- v1
    - Add **payment** API
    - Add **inquiry** API
    - Add **reversal** API
    - Add **refund** API
    - Add **precreate** API
- v2
    - Add **channelId** into the response parameter for **all** API.
        - **payment** API
        - **inquiry** API
        - **reversal** API
        - **refund** API
        - **precreate** API
    - Fix **errorCode** not included in response signature calculation when **reversal** failed.
    - Fix missing **errorCode** not included in response when statusCode is 99 for **refund API.**
    - Remove **md5 hashing algorithm as an option to generate** signatures**.**
- v3
    - Adding **extraInfo** in response for **all** API.
        - **payment** API
        - **inquiry** API
        - **reversal** API
        - **refund** API
        - **precreate** API
- v4
    - Add metadata in request and response for below API.
        - precreate API
        - payment API

4. Payment Flow Overview

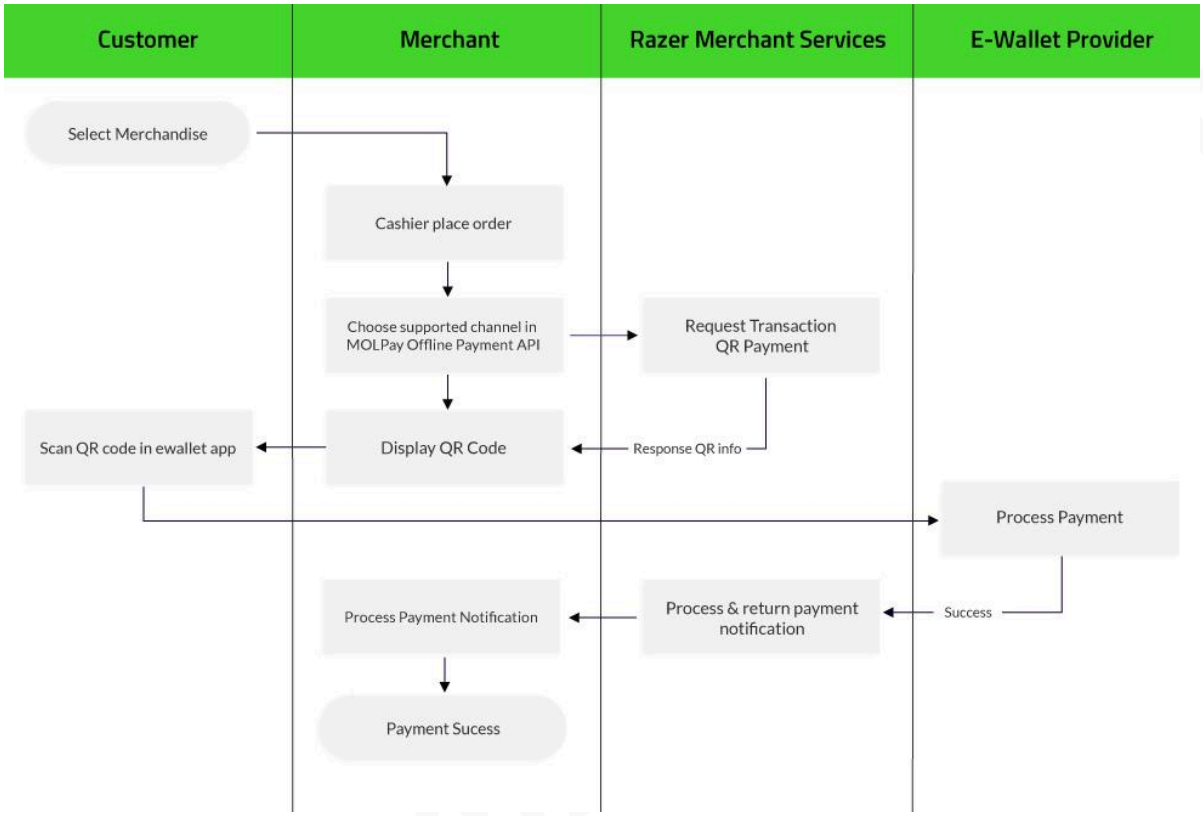
4.1 Merchant Scan Overview



1. Customers select merchandise products.
2. Cashier places the order and chooses the payment channel.
3. Cashiers scan QR/barcode from customer e-wallet apps.

4. Merchant servers capture QR/barcode and request **payment** service to the payment gateway server.
5. Merchant process payment upon receiving of **payment** response from payment gateway.

4.2 Customer Scan Overview



1. Customers select merchandise products.
2. Cashier places the order and chooses the payment channel.
3. Merchant server requests **Pre-Create Transaction QR Code** service to PG server.
4. Merchant displays QR on screen while getting **Pre-Create Transaction QR Code** response from PG server.
5. Customer scan the on screen QR code with e-wallet apps and complete the payment in apps
6. PG will callback payment notification to the merchant server upon receiving a success callback from the provider.
7. Merchant process payment upon receive **payment notification** from PG

Merchant required to perform **payment reversal if **NOT** getting payment notification from PG **after 1 minute or 60 seconds**.*

5. Offline Payment Services

- **Payment**

  Merchant initiates this request to the payment gateway to request payment using barcode scanned from customer eWallet.

- **Inquiry**

  Merchant initiates this request to the payment gateway to check payment transaction status that previously made.

- **Reversal**

  Merchant initiates this request to the payment gateway to void payment transactions that previously made due to connection timeout or unknown response. Only able to reverse a transaction on the same day.

- **Refund**

  Merchant initiates this request to the payment gateway to refund payment transactions that were previously made. Able refund payment transaction up to 90 days except Touch ‘n Go Digital eWallet, which is up to 30 days only.

- **Pre-Create Transaction QR Code**

  Merchant initiates this request to payment gateway to pre-create payment QR code to display at merchant side

- **Payment Notification**

  Payment gateway will notify merchant upon payment is completed by user

5.1 Channel Availability

Channel Name	Channel ID	Customer Presented (Payment API)	Merchant Presented (Precreate API)
RazerPay	15	❌	❌
Alipay	16	✔	✔
TNG-D	17	✔	✔
Alipay Pre-Auth	18	✔	✔
Boost	19	✔	✔
MAE by Maybank2u ** Only one option is allowed	20	✔	✔
GrabPay	21	✔	✔
UnionPay	22	✔	✔
ShopeePay	23	✔	✔
DuitNow QR	24	❌	✔
Alipay+	25	✔	✔
Atome	26	✔	✔
WeChatPay (CN)	36	✔	✔
WeChatPay (MY)	37	✔	✔
PayNow	38	❌	✔
KBank QR	39	❌	✔
QRPH	40	❌	✔

✔ - Available
❌ - Not available

5.2 Payment

This service provides functionality to initiate payment on the requested channel (payment method).

Environment	Service URL
Sandbox	https://**sandbox-payment.fiuu.com/RMS**/API/MOLOPA/payment.php
Production	https://**opa.fiuu.com/RMS**/API/MOLOPA/payment.php

Request Header	
HTTP Method	POST
Content Type	x-www-form-urlencoded

Request Body Message

Parameter Name	Data Type (Size)	M/O/C	Description
applicationCode	ans{1..32}	M	***Application Code*** is a unique identifier provided by PG. A merchant can have many stores and each store will have its own unique applicationCode
version	ans{1..3}	M	***Version*** of Offline Payment API starts with prefix "v" followed by the version number. When a merchant makes a HTTP request to PG, the version must be specified in the parameter. **Refer to section** Version
referenceId	ans{1..40}	M	***Reference Id*** is a unique identifier generated by a merchant for each distinct transaction. This serves as the transaction identifier for reconciliation.
channelId	n{2}	O	***channelId*** is a unique identifier of the payment provider specified for the payment. This is for buyer presented barcode or QR code payment acceptance. **Refer to section** Channel Availability

			**System will auto detect the payment channel via authorizationCode when merchants pass an empty string in this parameter.*
authorizationCode	an{200}	M	***authorizationCode*** is captured by scanning the barcode generated from user eWallet.
currencyCode	a{3}	M	***Currency Code*** refers to **ISO-4217** currency code of the transacted amount, e.g. MYR
amount	ns{10,2}	M	***Amount*** is the payment amount to collect from the user. **Format** : Positive number. **Always 2 decimal places with "." as a decimal point.**
description	ans{1..50}	O	***description*** of transaction details.
storeId	ans{4..20}	M	***Store Id*** is a unique identifier provided by a merchant for each distinct store.
terminalId	ans{4..20}	M	***Terminal Id*** is a unique identifier provided by a merchant for each distinct terminal.
businessDate	ans{10}	O	***Business date*** is a business date on the merchant side to which the transaction belongs. It eases the reconciliation when a merchant has different business cutoff time with PG. Leave empty will follow the transaction date at the PG system. **Format:** yyyy-MM-dd
validityDuration	n{1..3}	O	***validityDuration*** is a payment expiration time that the merchant can set to prevent the buyer from continuing making payment after the specific period. **Refer to section** Channel Validity Duration **Format:** 60 **(second)**
hashType	ans{3..11}	O	***Hash Type*** is the hashing algorithm used to generate signatures. Left empty will default as md5 (only for v1). **Value** / **Description** **md5** / MD5 **hmac-sha256** / HMAC-SHA256

signature	ans{32}	M	All parameters required for signature creation (*refer to* Generate Signature)
metadata	text	O	**metadata** refers to the additional transaction information generated by the merchant (specifically for v4). This data should be passed in a JSON-encoded format.

Response Body Message

** Original parameter value passed from merchant, merely for reference purpose

Parameter Name	Data Type (Size)	M/O/C	Description
applicationCode	ans{1..32}	M	***Application Code*** is a unique identifier provided by PG. A merchant can have many stores and each store will have its own unique applicationCode.
version	ans{1..3}	M	***Version*** of Offline Payment API starts with prefix "v" followed by the version number. When a merchant makes an HTTP request to PG, the version must be specified in the parameter. **Refer to section Version**
referenceId	ans{1..40}	M	***Reference Id*** is a unique identifier generated by a merchant for each distinct transaction. This serves as the transaction identifier for reconciliation.
authorizationCode	n{50}	M	***authorizationCode*** is captured by scanning the barcode generated from user eWallet.
channelId	n{2}	O	***channelId*** is a unique identifier of the payment provider specified for the payment.
currencyCode	a{3}	M	***Currency Code*** refers to currency of the transacted amount.
amount	ns{10,2}	M	***Amount*** is the payment amount to collect from the user. **Format** : Positive number. **Always 2 decimal places with "." as a decimal point.**
molTransactionId	n{10}	M	***PG Transaction id*** is an unique identifier given by payment gateway for transaction reference purpose.
payerId	ans{100}	O	***Payer Id*** is a piece of user data returned from a payment provider which may be used to print on receipt. (only available if transaction is success)
exchangeRate	ns{10,4}	C	***exchangeRate*** refers to the rate of conversion from the given currency to the currency in the user eWallet. (only available if transaction is success)

baseCurrencyCode	a{3}	C	***baseCurrencyCode*** is the currency of the user eWallet. (only available if transaction is success)
baseAmount	ns{10,2}	C	***baseAmount*** is the payment amount in user eWallet currency. (only available if transaction is success) **Format** : Positive number. **Always 2 decimal places with "." as a decimal point.**
extraInfo	object	O	**extraInfo** contains the additional information of the payment transaction.
statusCode	n{2}	M	***Status Code*** refers to the status indicator for payment transactions. (refer to Status Code)
errorCode	n{4}	C	***Error Code*** refers to details. (refer to Error Code) (if success will return empty value for errorCode)
transactionDateTime	ans{19}	M	***transaction date time*** refers to the transaction date time at PG. **Format** : yyyy-MM-dd**T**HH:mm:ss
hashType	ans{3..11}	O	***Hash Type*** is the hashing algorithm used to generate signatures. Left empty will default as md5 (in v1 only). **Value** / **Description** **md5** / MD5 **hmac-sha256** / HMAC-SHA256
signature	ans{32}	M	All parameters required for signature creation (*refer to* Generate Signature)
metadata	text	O	**metadata** refers to the additional transaction information generated by the merchant (specifically for v4). This data should be passed in a JSON-encoded format.

Example

HTTP Method	POST /payment.php
Request Parameters in HTTP Body (*x-www-form-urlencoded format*)	**amount**=10.00&**applicationCode**=3f2504e04f8911d39a0c0305e82c3301&**authorizationCode**=123456789123456789&&**businessDate**=2016-08-01&**channelId**=16&**currencyCode=**MYR&**description**=Retail&**referenceId**=2016072010291101&**storeId**=1022&**terminalId**=1022001&**version**=V1&**signature**=b09233f9950cba483aabeadb476ae8ca

Response (*JSON format*)	**200 OK** (refer to HTTP Status Code)
	{
	"**amount**": 10.00,
	"**applicationCode**": "3f2504e04f8911d39a0c0305e82c3301",
	"**authorizationCode**": "123456789123456789",
	"**baseAmount**": 16.64,
	"**baseCurrencyCode**": "CNY",
	"**channelId**": "16",
	"**currencyCode**": "MYR",
	"**errorCode**": "9999",
	"**exchangeRate**": 1.66,
	"**molTransactionId**": “152688223",
	"**payerId**": "*kev12@*.com",
	"**referenceId**": "2016072010291101,
	"**statusCode**": "99",
	"**transactionDateTime**": "2016-07-20T10:29:15",
	"**version**": "V1",
	"**signature**" : "c7d8b24ad11dbff5ab1831dcf321e94e"
	}

5.3 Inquiry

Merchant shall use this function to check and query on the payment status within the past 60 minutes only.

Environment	Service URL
Sandbox	https://**sandbox-api.fiuu.com/RMS**/API/MOLOPA/inquiry.php
Production	https://**api.fiuu.com/RMS**/API/MOLOPA/inquiry.php

Request Header	
HTTP Method	GET
Content Type	x-www-form-urlencoded

Request Body Message

Parameter Name	Data Type (Size)	M/O/C	Description
applicationCode	ans{1..32}	M	***Application Code*** is uniquely identifying merchant applications which integrate with Offline Payment API. A merchant company could have up to 1 merchant account and multiple application accounts.
version	ans{1..3}	M	***Version*** of Offline Payment API starts with prefix "v" followed by the version number. When a merchant makes a HTTP request to PG, the version must be specified in the parameter. **Refer to section Version**
referenceId	ans{1..40}	M	***Reference Id*** is a unique identifier generated by a merchant for each distinct transaction. This serves as the transaction identifier for reconciliation. **molTransactionID from payment response can also be used as the transaction identifier for inquiry and reconciliation purposes.**
hashType	ans{3..11}	O	***Hash Type*** is the hashing algorithm used to generate

			signatures. Left empty will default as md5 (in v1). **Value** / **Description** **md5** — MD5 **hmac-sha256** — HMAC-SHA256
signature	ans{32}	M	All parameters required for signature creation (*refer to* Generate Signature)

Response Body Message

** Original parameter value passed from merchant, merely for reference purpose

Parameter Name	Data Type (Size)	M/O/C	Description
applicationCode	ans{1..32}	M	***Application Code*** is a unique identifier provided by PG. A merchant can have many stores and each store will have its own unique applicationCode.
version	ans{1..3}	M	***Version*** of Offline Payment API starts with prefix "v" followed by the version number. When a merchant makes an HTTP request to PG, the version must be specified in the parameter. **Refer to section Version**
referenceId	ans{1..40}	M	***Reference Id*** is a unique identifier generated by a merchant for each distinct transaction. This serves as the transaction identifier for reconciliation.
authorizationCodeType	n{1}	C	***authorizationCodeType*** is to specify the type of the given authorization code. The supporting types depend on the channel passed. (Not available for pre-create transaction)
authorizationCode	n{50}	C	***authorizationCode*** is captured by scanning the barcode generated from user eWallet. (Not available for pre-create transaction)
currencyCode	a{3}	M	***Currency Code*** refers to currency of the transacted

			amount.
amount	ns{10,2}	M	***Amount*** is the payment amount to collect from the user. **Format** : Positive number. **Always 2 decimal places with "." as a decimal point.**
channelId	n{2}	O	***channelId*** is a unique identifier of the payment provider specified for the payment.
molTransactionId	n{10}	M	***PG Transaction id*** is an unique identifier given by payment gateway for transaction reference purpose.
payerId	ans{100}	O	***Payer Id*** is a piece of user data returned from a payment provider which may be used to print on receipt. (only available if transaction is success)
exchangeRate	ns{10,4}	C	***exchangeRate*** refers to the rate of conversion from the given currency to the currency in the user eWallet. (only available if transaction is success)
baseCurrencyCode	a{3}	C	***baseCurrencyCode*** is the currency of the user eWallet. (only available if transaction is success)
baseAmount	ns{10,2}	C	***baseAmount*** is the payment amount in user eWallet currency. (only available if transaction is success) **Format** : Positive number. **Always 2 decimal places with "." as a decimal point.**
extraInfo	object	O	**extraInfo** contains the additional information of the payment transaction.
statusCode	n{2}	M	***Status Code*** refers to the status indicator for payment transactions. (refer to Status Code)
errorCode	n{4}	C	***Error Code*** refers to details. (refer to Error Code) (if success will return empty value for errorCode)
transactionDateTime	ans{19}	M	***transaction date time*** refers to the transaction date

			time at PG. **Format** : yyyy-MM-dd**T**HH:mm:ss
hashType	ans{3..11}	O	***Hash Type*** is the hashing algorithm used to generate signatures. Left empty will default as md5 (in v1 only). **Value** / **Description** **md5** — MD5 **hmac-sha256** — HMAC-SHA256
signature	ans{32}	M	All parameters required for signature creation (*refer to* Generate Signature)

Example

HTTP Method	**GET /inquiry.php**
Request Parameters in HTTP Body (*x-www-form-urlencoded format*)	https://api.fiuu.com/MOLPay/API/MOLOPA/inquiry.php?**applicationCode**=3f2504e04f8911d39a0c0305e82c3301&**referenceId**=2016072010291101&**version**=V1&**signature**=960674ae5b451e1f1811e221eac45d1c

Response (*JSON format*)	**200 OK** (refer to HTTP Status Code)
	{
	"**amount**": 10.00,
	"**applicationCode**": "3f2504e04f8911d39a0c0305e82c3301",
	"**authorizationCode**": "123456789123456789",
	"**baseAmount**": 16.64,
	"**baseCurrencyCode**": "CNY",
	"**channelId**": "16",
	"**currencyCode**": "MYR",
	"**exchangeRate**": 1.66,
	"**molTransactionId**": "152688223",
	"**referenceId**": "2016072010291101,
	"**statusCode**": "00",
	"**transactionDateTime**": "2016-07-20T10:29:15",
	"**version**": "V1",
	"**signature**" : "c7d8b24ad11dbff5ab1831dcf321e94e"
	}

5.4 Reversal

Reversal is to void a payment transaction within the same day. However it is not applicable to void the refund request.

Merchant can send a void request upon an unknown payment status transaction when there is a poor network connectivity or system loading issue to avoid charging to the customer e-wallet.

Environment	Service URL
Sandbox	https://**sandbox-api.fiuu.com/RMS**/API/MOLOPA/reversal.php
Production	https://**api.fiuu.com/RMS**/API/MOLOPA/reversal.php

Request Header	
HTTP Method	POST
Content Type	x-www-form-urlencoded

Request Body Message

Parameter Name	Data Type (Size)	M/O/C	Description
applicationCode	ans{1..32}	M	***Application Code*** is a unique identifier provided by PG. A merchant can have many stores and each store will have its own unique applicationCode.
version	ans{1..3}	M	***Version*** of Offline Payment API starts with prefix "v" followed by the version number. When a merchant makes an HTTP request to PG, the version must be specified in the parameter. **Refer to section Version**
referenceId	ans{1..40}	M	***Reference Id*** is a unique identifier generated by a merchant for each distinct transaction. This serves as the transaction identifier for reconciliation.
paymentReferenceId	n{10}	M	***Payment Reference Id*** is the original payment

			transaction made previously.
businessDate	ans{10}	O	***Business date*** is a business date on the merchant side to which the transaction belongs. This is to ease the reconciliation when merchants have different business cutoff times with PG. Leave empty will follow the transaction date at the PG system. **Format:** yyyy-MM-dd
hashType	ans{3..11}	O	***Hash Type*** is the hashing algorithm used to generate signatures. Left empty will default as md5 (in v1 only). **Value** / **Description** **md5** / MD5 **hmac-sha256** / HMAC-SHA256
signature	ans{32}	M	All parameters required for signature creation (*refer to* Generate Signature)

Response Body Message

** Original parameter value passed from merchant, merely for reference purpose

Parameter Name	Data Type (Size)	M/O/C	Description
applicationCode	ans{1..32}	M	***Application Code*** is a unique identifier provided by PG. A merchant can have many stores and each store will have its own unique applicationCode.
version	ans{1..3}	M	****Version*** of Offline Payment API starts with prefix "v" followed by the version number. When a merchant makes an HTTP request to PG, the version must be specified in the parameter. **Refer to section Version**
referenceId	ans{1..40}	M	****Reference Id*** is a unique identifier generated by a merchant for each distinct transaction. This serves as the transaction identifier for reconciliation.

<table>
<tr><td>paymentReferenceId</td><td>n{10}</td><td>M</td><td>Payment Reference Id is the original payment transaction made previously.</td></tr>
<tr><td>channelId</td><td>n{2}</td><td>O</td><td>channelId is a unique identifier of the payment provider specified for the payment.</td></tr>
<tr><td>molTransactionId</td><td>n{10}</td><td>M</td><td>PG Transaction id is an unique identifier given by payment gateway for transaction reference purpose.</td></tr>
<tr><td>extraInfo</td><td>object</td><td>O</td><td>extraInfo contains the additional information of the payment transaction.</td></tr>
<tr><td>statusCode</td><td>n{2}</td><td>M</td><td>Status Code refers to the status indicator for payment transactions. (refer to Status Code)</td></tr>
<tr><td>errorCode</td><td>n{4}</td><td>C</td><td>Error Code refers to details. (refer to Error Code)

(if success will return empty value for errorCode)</td></tr>
<tr><td>transactionDateTime</td><td>ans{19}</td><td>M</td><td>transaction date time refers to the transaction date time at PG.

Format : yyyy-MM-ddTHH:mm:ss</td></tr>
<tr><td>hashType</td><td>ans{3..11}</td><td>O</td><td>Hash Type is the hashing algorithm used to generate signature. Left empty will default as md5 (in v1 only).

<table><tr><th>Value</th><th>Description</th></tr><tr><td>md5</td><td>MD5</td></tr><tr><td>hmac-sha256</td><td>HMAC-SHA256</td></tr></table></td></tr>
<tr><td>signature</td><td>ans{32}</td><td>M</td><td>All parameters required for signature creation (refer to Generate Signature)

** KNOWN BUG & MITIGATION PLAN
BUG: signature does not include errorCode when statusCode=99 (i.e. failed to reverse)

Solution: Refer to section Version. Apply version 2</td></tr>
</table>

Example

HTTP Method	**POST /reversal.php**
Request Parameters in HTTP Body (*x-www-form-urlencoded format*)	**applicationCode**=3f2504e04f8911d39a0c0305e82c3301&**businessDate**=2016-08-01&**paymentReferenceId**=2016072010291101&**referenceId**=2016072010291102&**version**=V1&**signature**=c90220bf7e46438737d2f8b13d9cdb88

Response (*JSON format*)	**200 OK** (refer to HTTP Status Code)
	{
	"**applicationCode**": "3f2504e04f8911d39a0c0305e82c3301",
	"**channelId**": "16",
	"**molTransactionId**": "152688223",
	"**paymentReferenceId**": "2016072010291101",
	"**referenceId**": "2016072010291102,
	"**statusCode**": "00",
	"**transactionDateTime**": "2016-07-20T10:29:15",
	"**version**": "V1",
	"**signature**" : "c7d8b24ad11dbff5ab1831dcf321e94e"
	}

5.5 Refund

This service provides functionality to refund payment transactions up to 90 days except Touch 'n Go Digital eWallet, which is up to 30 days only. Same day refund or void please use the "Reversal" function in the previous section.

**** MDR will not be refunded for certain channels. May check with your sales PIC.**

Environment	Service URL
Sandbox	https://**sandbox-api.fiuu.com/RMS**/API/MOLOPA/refund.php
Production	https://**api.fiuu.com/RMS**/API/MOLOPA/refund.php

Request Header	
HTTP Method	POST
Content Type	x-www-form-urlencoded

Request Body Message

Parameter Name	Data Type (Size)	M/O/C	Description
applicationCode	ans{1..32}	M	***Application Code*** is a unique identifier provided by PG. A merchant can have many stores and each store will have its own unique applicationCode.
version	ans{1..3}	M	***Version*** of Offline Payment API starts with prefix "v" followed by the version number. When a merchant makes an HTTP request to PG, the version must be specified in the parameter. **Refer to section Version**
referenceId	ans{1..40}	M	***Reference Id*** is a unique identifier generated by a merchant for each distinct transaction. This serves as the transaction identifier for reconciliation.
paymentReferenceId	n{10}	M	***Payment Reference Id*** is the original payment

			transaction made previously.
currencyCode	a{3}	M	***Currency Code*** refers to **ISO-4217** currency code of the transacted amount, e.g. MYR
amount	ns{10,2}	M	***Amount*** is the payment amount to collect from the user. **Format** : Positive number. **Always 2 decimal places with "." as a decimal point.**
description	ans{1..50}	O	***description*** to describe payment transaction.
businessDate	ans{10}	O	***Business date*** is a business date on the merchant side to which the transaction belongs. This is to ease the reconciliation when merchants have different business cutoff time with PG. Leave empty will follow the transaction date at the PG system. **Format:** yyyy-MM-dd
hashType	ans{3..11}	O	***Hash Type*** is the hashing algorithm used to generate signatures. Left empty will default as md5 (in v1 only). **Value** / **Description** **md5** / MD5 **hmac-sha256** / HMAC-SHA256
signature	ans{32}	M	All parameters required for signature creation (*refer to* Generate Signature)

Response Body Message

** Original parameter value passed from merchant, merely for reference purpose

Parameter Name	Data Type (Size)	M/O/C	Description
applicationCode	ans{1..32}	M	***Application Code*** is a unique identifier provided by PG. A merchant can have many stores and each store will have its own unique applicationCode.

version	ans{1..3}	M	***Version*** of Offline Payment API starts with prefix "v" followed by the version number. When a merchant makes an HTTP request to PG, the version must be specified in the parameter. **Refer to section Version**
referenceId	ans{1..40}	M	***Reference Id*** is a unique identifier generated by a merchant for each distinct transaction. This serves as the transaction identifier for reconciliation.
paymentReferenceId	n{10}	M	***Payment Reference Id*** is the original payment transaction made previously.
currencyCode	a{3}	M	***Currency Code*** refers to currency of the transacted amount.
amount	ns{10,2}	M	***Amount*** is the payment amount to collect from the user. **Format** : Positive number. **Always 2 decimal places with "." as a decimal point.**
channelId	n{2}	O	***channelId*** is a unique identifier of the payment provider specified for the payment.
molTransactionId	n{10}	M	***PG Transaction id*** is an unique identifier given by payment gateway for transaction reference purpose.
payerId	ans{100}	O	***Payer Id*** is a piece of user data returned from a payment provider which may be used to print on receipt. (only available if transaction is success)
exchangeRate	ns{10,4}	C	***exchangeRate*** refers to the rate of conversion from the given currency to the currency in the user eWallet. (only available if transaction is success)
baseCurrencyCode	a{3}	C	***baseCurrencyCode*** is the currency of the user eWallet. (only available if transaction is success)
baseAmount	ns{10,2}	C	***baseAmount*** is the payment amount in user eWallet currency. (only available if transaction is success) **Format** : Positive number. **Always 2 decimal places**

<table>
<tr><td></td><td></td><td></td><td>with “.” as a decimal point.</td></tr>
<tr><td>extraInfo</td><td>object</td><td>O</td><td>extraInfo contains the additional information of the payment transaction.</td></tr>
<tr><td>statusCode</td><td>n{2}</td><td>M</td><td>Status Code refers to the status indicator for payment transactions. (refer to Status Code)</td></tr>
<tr><td>errorCode</td><td>n{4}</td><td>C</td><td>Error Code refers to details. (refer to Error Code)

(if success will return empty value for errorCode)</td></tr>
<tr><td>transactionDateTime</td><td>ans{19}</td><td>M</td><td>transaction date time refers to the transaction date time at PG.

Format : yyyy-MM-ddTHH:mm:ss</td></tr>
<tr><td>hashType</td><td>ans{3..11}</td><td>O</td><td>Hash Type is the hashing algorithm used to generate signatures. Left empty will default as md5 (in v1 only).

<table><tr><th>Value</th><th>Description</th></tr><tr><td>md5</td><td>MD5</td></tr><tr><td>hmac-sha256</td><td>HMAC-SHA256</td></tr></table></td></tr>
<tr><td>signature</td><td>ans{32}</td><td>M</td><td>All parameters required for signature creation (refer to Generate Signature)</td></tr>
</table>

Example

HTTP Method	POST /refund.php
Request Parameters in HTTP Body (*x-www-form-urlencoded format*)	**amount**=10.00&**applicationCode**=3f2504e04f8911d39a0c0305e82c3301&**businessDate**=2016-08-01&**currencyCode**=MYR&**description**=Refund&**paymentReferenceId**=2016072010291101&**referenceId**=2016072010291102&**version**=V1&**signature**=de3e87068a930f816b0be312f5019643

Response (*JSON format*)	**200 OK** (refer to HTTP Status Code)
	{
	"**amount**": 10.00,
	"**applicationCode**": "3f2504e04f8911d39a0c0305e82c3301",
	"**baseAmount**": 16.64,
	"**baseCurrencyCode**": "CNY",
	"**channelId**": "16",
	"**currencyCode**": "MYR",
	"**exchangeRate**": 1.66,
	"**molTransactionId**": "152688225",
	"**payerId**": "*kev12@*.com",
	"**referenceId**": "2016072010291102,
	"**paymentReferenceId**": "2016072010291102,
	"**statusCode**": "00",
	"**transactionDateTime**": "2016-07-20T10:29:15",
	"**version**": "V1",
	"**signature**" : "761922c12d2415cbed81a745de0d959e"
	}

5.6 Pre-Create Transaction QR Code

This service provides functionality to pre-create payment transaction QR Code. It is a one-time merchant presented dynamic QR code with a short validity, normally within minutes or hours.

Environment	Service URL
Sandbox	~~https://**sandbox-payment.fiuu.com/RMS**/API/MOLOPA/precreate.php~~
Production	https://**opa.fiuu.com/RMS**/API/MOLOPA/precreate.php

Request Header	
HTTP Method	POST
Content Type	x-www-form-urlencoded

Request Body Message

Parameter Name	Data Type (Size)	M/O/C	Description
applicationCode	ans{1..32}	M	***Application Code*** is a unique identifier provided by PG. A merchant can have many stores and each store will have its own unique applicationCode.
version	ans{1..3}	M	***Version*** of Offline Payment API starts with prefix "v" followed by the version number. When a merchant makes an HTTP request to PG, the version must be specified in the parameter. **Refer to section** Version
referenceId	ans{1..40}	M	***Reference Id*** is a unique identifier of payment provider specified for the payment.
channelId	n{2}	M	***channelId*** is a unique identifier of the payment provider specified for the payment. This is merchant QR code. **Refer to section** Channel Availability

<table>
<tr><td>currencyCode</td><td>a{3}</td><td>M</td><td>Currency Code refers to ISO-4217 currency code of the transacted amount, e.g. MYR</td></tr>
<tr><td>amount</td><td>ns{10,2}</td><td>M</td><td>Amount is the payment amount to collect from the user.

Format : Positive number. Always 2 decimal places with “.” as a decimal point.</td></tr>
<tr><td>description</td><td>ans{1..50}</td><td>O</td><td>description to describe the transaction.</td></tr>
<tr><td>storeId</td><td>ans{4..20}</td><td>M</td><td>Store Id is a unique identifier provided by a merchant for each distinct store.</td></tr>
<tr><td>terminalId</td><td>ans{4..20}</td><td>M</td><td>Terminal Id is a unique identifier provided by a merchant for each distinct terminal.</td></tr>
<tr><td>imageFormat</td><td>a{5}</td><td>O</td><td>Image Format is the output of the QR picture merchant desires. The default format given will be in PNG format.

Only one custom image will be generated according to the format requested in the response parameter “customImageURL” if imageFormat is specified.

<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>png</td><td>.PNG</td></tr>
<tr><td>jpg</td><td>.JPG, JPEG</td></tr>
<tr><td>bmp</td><td>.BMP</td></tr>
</table>
</td></tr>
<tr><td>imageSize</td><td>ans{4..10}</td><td>O</td><td>Image Size is the custom QR image resolution merchant request in pixel.

Format: WidthxHeight (250x150)
Smallest: 200x150
Largest: 2000x2000</td></tr>
<tr><td>businessDate</td><td>ans{10}</td><td>O</td><td>Business date is a business date on the merchant side to which the transaction belongs. This is to ease the reconciliation when merchants have different business cutoff time with PG. Leave empty will follow the transaction date at the PG system.

Format: yyyy-MM-dd</td></tr>
</table>

<table>
<tr><td>validityDuration</td><td>n{1..3}</td><td>O</td><td>validityDuration is the QR validity period that the merchant can set to prevent the buyer from continuing to scan and pay after a specific period.

Refer to section Channel Validity Duration
Format: 60 (second)</td></tr>
<tr><td>hashType</td><td>ans{3..11}</td><td>O</td><td>Hash Type is the hashing algorithm used to generate signatures. Left empty will default as md5 (in v1 only).

<table><tr><th>Value</th><th>Description</th></tr><tr><td>md5</td><td>MD5</td></tr><tr><td>hmac-sha256</td><td>HMAC-SHA256</td></tr></table></td></tr>
<tr><td>signature</td><td>ans{32}</td><td>M</td><td>All parameters required for signature creation (refer to Generate Signature)</td></tr>
<tr><td>metadata</td><td>text</td><td>O</td><td>metadata refers to the additional transaction information generated by the merchant (specifically for v4). This data should be passed in a JSON-encoded format.</td></tr>
</table>

Response Body Message

** Original parameter value passed from merchant, merely for reference purpose

Parameter Name	Data Type (Size)	M/O/C	Description
applicationCode	ans{1..32}	M	***Application Code*** is a unique identifier provided by PG. A merchant can have many stores and each store will have its own unique applicationCode.
version	ans{1..3}	M	***Version*** of Offline Payment API starts with prefix "v" followed by the version number. When a merchant makes an HTTP request to PG, the version must be specified in the parameter. **Refer to section Version**
referenceId	ans{1..40}	M	***Reference Id*** is a unique identifier generated by a merchant for each distinct transaction. This serves as the transaction identifier for reconciliation.
currencyCode	a{3}	M	***Currency Code*** refers to currency of the transacted amount.
amount	ns{10,2}	M	***Amount*** is the payment amount to collect from the user. **Format** : Positive number. **Always 2 decimal places with "." as a decimal point.**
molTransactionId	n{10}	M	***PG Transaction id*** is an unique identifier given by payment gateway for transaction reference purpose.
channelId	n{2}	O	***channelId*** is a unique identifier of the payment provider specified for the payment.
authorizationCode	ans{1..200}	M	***authorizationCode*** is the information to generate QR image.
ImageUrl	ans{200}	M	***ImageUrl*** refers to a URL to retrieve the QR image. *Perform HTTP Request (HTTP GET) to retrieve the image file from this URL.
ImageUrlBig	ans{200}	M	***ImageUrlBig*** refers to a URL to retrieve the large QR image.

<table>
<tr><td></td><td></td><td></td><td>*Perform HTTP Request (HTTP GET) to retrieve image file from this URL.</td></tr>
<tr><td>ImageUrlSmall</td><td>ans{200}</td><td>M</td><td>ImageUrlSmall refers to a URL to retrieve the small QR image.

*Perform HTTP Request (HTTP GET) to retrieve the image file from this URL.</td></tr>
<tr><td>customImageURL</td><td>ans{200}</td><td>O</td><td>customImageURL refers to URL to retrieve the custom QR image

*Perform HTTP Request (HTTP GET) to retrieve the image file from this URL.</td></tr>
<tr><td>extraInfo</td><td>object</td><td>O</td><td>extraInfo contains the additional information of the payment transaction.</td></tr>
<tr><td>statusCode</td><td>n{2}</td><td>M</td><td>Status Code refers to the status indicator for payment transactions. (refer to Status Code)</td></tr>
<tr><td>errorCode</td><td>n{4}</td><td>C</td><td>Error Code refers to details. (refer to Error Code)

(if success will return empty value for errorCode)</td></tr>
<tr><td>transactionDateTime</td><td>ans{19}</td><td>M</td><td>transaction date time refers to the transaction date time at PG.

Format : yyyy-MM-ddTHH:mm:ss</td></tr>
<tr><td>hashType</td><td>ans{3..11}</td><td>O</td><td>Hash Type is the hashing algorithm used to generate signatures. Left empty will default as md5 (in v1 only).

<table><tr><th>Value</th><th>Description</th></tr><tr><td>md5</td><td>MD5</td></tr><tr><td>hmac-sha256</td><td>HMAC-SHA256</td></tr></table></td></tr>
<tr><td>signature</td><td>ans{32}</td><td>M</td><td>All parameters required for signature creation (refer to Generate Signature)</td></tr>
<tr><td>metadata</td><td>text</td><td>O</td><td>metadata refers to the additional transaction information generated by the merchant (specifically for v4). This data should be passed in a JSON-encoded format.</td></tr>
</table>

Example

HTTP Method	**POST /precreate.php**
Request Parameters in HTTP Body (*x-www-form-urlencoded format*)	**amount**=10.00&**applicationCode**=3f2504e04f8911d39a0c0305e82c3301&**businessDate**=2016-08-01&**currencyCode**=MYR&**description**=Fish_pasar&**paymentReferenceId**=2016072010291101&**referenceId**=2016072010291102&**version**=V1&**signature**=de3e87068a930f816b0be312f5019643 (not a real and valid signature)

Response (*JSON format*)	**200 OK** (refer to HTTP Status Code)
	{
	"**amount**": 10.00,
	"**applicationCode**": "3f2504e04f8911d39a0c0305e82c3301",
	"**baseAmount**": 16.64,
	"**baseCurrencyCode**": "CNY",
	"**channelId**": "16",
	"**currencyCode**": "MYR",
	"**exchangeRate**": 1.66,
	"**molTransactionId**": “152688225",
	"**payerId**": "*kev12@*.com",
	"**referenceId**": "2016072010291102,
	"**paymentReferenceId**": "2016072010291102,
	"**statusCode**": "00",
	"**transactionDateTime**": "2016-07-20T10:29:15",
	"**version**": "V1",
	"**signature**" : "761922c12d2415cbed81a745de0d959e"
	}

5.7 Payment Notification

Upon successful payment, a notification will be triggered to the merchant server. To receive the notification, merchant has to set up endpoint and configure notification URL at RMS Merchant Portal: https://portal.fiuu.com

Parameter Name	Data Type (Size)	M/O/C	Description
applicationCode	ans{1..32}	M	***Application Code*** is a unique identifier provided by PG. A merchant can have many stores and each store will have its own unique applicationCode.
version	ans{1..3}	M	***Version*** of Offline Payment API starts with prefix "v" followed by the version number. When a merchant makes an HTTP request to PG, the version must be specified in the parameter. **Refer to section Version**
referenceId	ans{1..40}	M	***Reference Id*** is a unique identifier of payment provider specified for the payment.
authorizationCodeType	n{1}	M	***authorizationCodeType*** is to specify the type of the given authorization code. The supporting types depend on the channel passed.
authorizationCode	n{50}	M	***authorizationCode*** is captured by scanning the barcode generated from user eWallet.
currencyCode	a{3}	M	***Currency Code*** refers to currency of the transacted amount. Have to be the same as the original transaction.
channelId	n{2}	O	***channelId*** is a unique identifier of the payment provider specified for the payment.
amount	ns{10,2}	M	***Amount*** is the payment amount to collect from the user. **Format** : Positive number. **Always 2 decimal places with "." as a decimal point.**
molTransactionId	n{10}	M	***PG Transaction id*** is an unique identifier given by payment gateway for transaction reference purpose.
payerId	ans{100}	O	***Payer Id*** is a piece of user data returned from a

			payment provider which may be used to print on receipt. (only available if transaction is success)
exchangeRate	ns{10,4}	C	***exchangeRate*** refers to the rate of conversion from the given currency to the currency in the user eWallet. (only available if transaction is success)
baseCurrencyCode	a{3}	C	***baseCurrencyCode*** is the currency of the user eWallet. (only available if transaction is success)
baseAmount	ns{10,2}	C	***baseAmount*** is the payment amount in user eWallet currency. (only available if transaction is success) **Format** : Positive number. **Always 2 decimal places with "." as a decimal point.**
extraInfo	object	O	**extraInfo** contains the additional information of the payment transaction.
statusCode	n{2}	M	***Status Code*** refers to the status indicator for payment transactions. (refer to Status Code)
errorCode	n{4}	C	***Error Code*** refers to details. (refer to Error Code) (if success will return empty value for errorCode)
transactionDateTime	ans{19}	M	***transaction date time*** refers to the transaction date time at PG. **Format** : yyyy-MM-dd**T**HH:mm:ss
hashType	ans{3..11}	O	***Hash Type*** is the hashing algorithm used to generate signatures. Left empty will default as md5 (in v1 only). **Value** / **Description** **md5** / MD5 **hmac-sha256** / HMAC-SHA256

signature	ans{32}	M	All parameters required for signature creation (*refer to* Generate Signature)

6. e-Voucher

Merchants are able to create marketing campaigns through RMS merchant portal (https://portal.fiuu.com/) and call this function to redeem the specific voucher in the campaign.

Environment	Service URL
Sandbox	~~https://**sandbox-api.fiuu.com/RMS**/API/MOLOPA/evoucher.php~~
Production	https://**api.fiuu.com/RMS**/API/MOLOPA/evoucher.php

Request Header	
HTTP Method	POST
Content Type	x-www-form-urlencoded

Request Body Message

Parameter Name	Data Type (Size)	M/O/C	Description
applicationCode	ans{1..32}	M	***Application Code*** is a unique identifier provided by PG. A merchant can have many stores and each store will have its own unique applicationCode.
version	ans{1..3}	M	***Version*** of Offline Payment API starts with prefix "v" followed by the version number. When a merchant makes a HTTP request to PG, the version must be specified in the parameter. **Current version**: v1
promoVoucher	ans{18..32}	M	***promoVoucher*** is the e-voucher to be redeemed for promotion or campaign.
terminalId	ans{1..40}	M	***Terminal Id*** *is a unique identifier provided by a merchant for each distinct terminal.*

storeId	ans{3..11}	M	***Store Id*** is a unique identifier provided by a merchant for each distinct store.
hashType	ans{3..11}	O	***Hash Type*** is the hashing algorithm used to generate signatures. Left empty will default as md5 (in v1 only). **Value** / **Description** **md5** / MD5 **hmac-sha256** / HMAC-SHA256
signature	ans{32}	M	All parameters required for signature creation (*refer to* Generate Signature)

Response Body Message

** Original parameter value passed from merchant, merely for reference purpose

Parameter Name	Data Type (Size)	M/O/C	Description
applicationCode	ans{1..32}	M	***Application Code*** is a unique identifier provided by PG. A merchant can have many stores and each store will have its own unique applicationCode.
version	ans{1..3}	M	****Version*** of Offline Payment API starts with prefix "v" followed by the version number. When a merchant makes an HTTP request to PG, the version must be specified in the parameter. **Current version**: v1
statusCode	n{2}	M	***Status Code*** refers to the status indicator for payment transactions. (refer to Status Code)
errorCode	n{4}	C	***Error Code*** refers to details. (refer to Error Code) (if success will return empty value for errorCode)
transactionDateTime	ans{19}	M	***transaction date time*** refers to the transaction date time at PG.

			Format : yyyy-MM-dd**T**HH:mm:ss
hashType	ans{3..11}	O	***Hash Type*** is the hashing algorithm used to generate signatures. Left empty will default as md5 (in v1 only). **Value** / **Description** **md5** / MD5 **hmac-sha256** / HMAC-SHA256
signature	ans{32}	M	All parameters required for signature creation (*refer to* Generate Signature)

Example

HTTP Method	**POST /evoucher.php**
Request Parameters in HTTP Body (*x-www-form-urlencoded format*)	**version**=v1&**applicationCode**=4fcb8da2b039001c30f6378bc3f78a79&**promoVoucher**=UFHChWwZcPgjNL6ifU&**terminalId**=terminal1&**storeId**=store1&**signature**=2759440ee3fb3499d07be9b759293b03 (not a real and valid signature)

Response (*JSON format*)	**200 OK** (refer to HTTP Status Code)
	
	{
	"applicationCode": "4fcb8da2b039001c30f6378bc3f78a79",
	"version": "v1",
	"statusCode": "00",
	"errorCode": "",
	"promoVoucher": "UFHChWwZcPgjNL6ifU",
	"terminalId": "terminal1",
	"storeId": "store1",
	"transactionDateTime": "2019-12-03T19:32:48",
	"signature": "511b62b79e05931bd88cd44b08fa92cd"
	}

7. Signature

7.1 Generate Signature

- A Signature is a **MD5 or HMAC-SHA256** (based on hashType parameter) hash string combination of a sequence of parameters and a ***Secret Key***.
- **MD5** is obsoleted in v2, merchant is recommended to use **HMAC-SHA256**
- Secret Key is a server-side shared secret, this key is assigned to merchants by RMS.
- All parameters used in the message exchange will form a part of the signature hash ***Except*** :
    - Empty parameter value (NOT zero)
    - Signature parameter itself.
- **All** parameter values that form a part of the signature hash must **sort alphabetically** based on parameter name.
- All parameters that form a part of the signature hash must be in their original form **(not URL encoded)**.
- All parameters that form a part of the signature hash **ARE** case sensitive.
- All strings will have leading and trailing whitespace stripped off.

Example using MD5

The following example explains how to generate signature for parameters with **non-empty** values:

Secret Key: **Ziu61T9xY227aazS530Pk8C5424y663r**

Parameter Name	Value
applicationCode	3f2504e04f8911d39a0c0305e82c3301
referenceId	TRX1708901
authorizationCode	123456789123456789
authorizationCodeType	1
channelId	16

currencyCode	MYR
description	Sample
amount	10.00
storeId	17001
terminalId	17001001
version	v1

1. Sort parameter values ordered by parameter name alphabetically.

{ amount } + { applicationCode } + { authorizationCode } + { authorizationCodeType } + {channelId} + { currencyCode } + { description } + { referenceId } + { storeId } + {terminalId } + {version} + **{ secretKey }**

2. Concatenate/combine the actual parameter's value.

10.003f2504e04f8911d39a0c0305e82c3301123456789123456789116MYRSampleTRX1708901170011700100 1v1**Ziu61T9xY227aazS530Pk8C5424y663r**

3. Hash concatenated string using MD5 algorithm.

MD5(10.003f2504e04f8911d39a0c0305e82c3301123456789123456789 1MYRTRX1708901170011700100 1v1**Ziu61T9xY227aazS530Pk8C5424y663r**) = **bee92e0042f51e9f3d626fe8b2b47069**

4. Use hashed value generated from above step as Signature parameter.

applicationCode=3f2504e04f8911d39a0c0305e82c3301&referenceId=TRX1708901&authorizationCode=123456789123456789&authorizationCodeType=1&channelId=16¤cyCode=MYR&description=Sample&amount=10.00&storeId=17001&terminalId=17001001&version=v1&**signature=bee92e0042f51e9f3d626fe8b2b47069**

Example using HMAC-SHA256

The following example explains how to generate signature for parameters with **non-empty** values:

Secret Key: **Ziu61T9xY227aazS530Pk8C5424y663r**

Parameter Name	Value
applicationCode	3f2504e04f8911d39a0c0305e82c3301
referenceId	TRX1708901
authorizationCode	123456789123456789
authorizationCodeType	1
channelId	16
currencyCode	MYR
description	Sample
amount	10.00
storeId	17001
terminalId	17001001
version	v1
hashType	hmac-sha256

1. Sort parameter values ordered by parameter name alphabetically.

```
{ amount } + { applicationCode } + { authorizationCode } + { authorizationCodeType } +
{channelId} + { currencyCode } + { description } + {hashType} + { referenceId } + { storeId } +
{terminalId } + {version}
```

2. Concatenate/combine the actual parameter's value.

```
10.003f2504e04f8911d39a0c0305e82c3301123456789123456789116MYRSamplehmac-sha256TRX17089011700117001001v1
```

3. Hash concatenated string using HMAC-SHA256 algorithm with the **secret key**.

```
HMAC-SHA256(10.003f2504e04f8911d39a0c0305e82c330112345678912345678911 6
```

MYRS
amplehmac-sha256TRX17089011700117001001v1,
Ziu61T9xY227aazS530Pk8C5424y663r) =
db0624605d8a8b9c40b3eeb97f906a454195f1b35d1a2f9b75700e1e8cc942ba

4. Use hashed value generated from above step as Signature parameter.

applicationCode=3f2504e04f8911d39a0c0305e82c3301&referenceId=TRX1708901
&authorizationCode=123456789123456789&authorizationCodeType=1&channelId=
16¤cyCode=MYR&description=Sample&amount=10.00&storeId=17001
&terminalId=17001001&version=v1&hashType=hmacsha256&**signature=db0624605d8a
8b9c40b3eeb97f906a454195f1b35d1a2f9b75700e1e8cc942ba**

Example For v4 API

The following example explains how to generate signature for parameters with non-empty values:
Secret Key: Ziu61T9xY227aazS530Pk8C5424y663r

Parameter Name	Value
applicationCode	3f2504e04f8911d39a0c0305e82c3301
referenceId	TRX1708901
authorizationCode	123456789123456789
authorizationCodeType	1
channelId	16
currencyCode	MYR
description	Sample
amount	10.00
storeId	17001
terminalId	17001001
version	v4
hashType	hmac-sha256
metadata	{“tranID”:”123456”}

5. Sort parameter values ordered by parameter name alphabetically.

```
{ amount } + { applicationCode } + { authorizationCode } + { authorizationCodeType } +
{channelId} + { currencyCode } + { description } + {hashType} + {metadata} + {
referenceId } + { storeId } +
{terminalId } + {version}
```

6. Concatenate/combine the actual parameter's value.

```
10.003f2504e04f8911d39a0c0305e82c3301123456789123456789116MYRSamplehmac
-sha256{"tranID":"123456"}TRX17089011700117001001v1
```

7. Hash concatenated string using HMAC-SHA256 algorithm with the secret key.

```
HMAC-SHA256(10.003f2504e04f8911d39a0c0305e82c3301123456789123456789116M
YRSamplehmac-sha256{"tranID":"123456"}TRX17089011700117001001v1,
Ziu61T9xY227aazS530Pk8C5424y663r) =
d28aec422b283cbc8bc1dbfeb7ba0d03fc8b0b7a42fce827019d5540063063ad
```

8. Use hashed value generated from above step as Signature parameter.

```
applicationCode=3f2504e04f8911d39a0c0305e82c3301&referenceId=TRX1708901
&authorizationCode=12345678912345678 9&authorizationCodeType=1&channelId=
16&currencyCode=MYR&description=Sample&amount=10.00&storeId=17001
&terminalId=17001001&metadata={"tranID":"123456"}&version=v1&hashType=hmacsha
256&signature=d28aec422b283cbc8bc1dbfeb7ba0d03fc8b0b7a42fce827019d55400630
63ad
```

7.2 Validate Signature

All service request and response messages must have a Signature parameter and will be validated by payment gateway to prevent data tampering. If the signature is invalid then the payment gateway will return HTTP Status 401.

It's highly **RECOMMENDED** for merchants to perform similar validation to ensure data validity against the origin source. Repeat the same steps from 1 - 4 described in generate signature and compare with the signature received from the payment gateway.

8. Error Response

Whenever an API returns an HTTP Status Code **other than 200**, indicates that the request has failed to be accepted. Same time, different response body messages consisting of the error details will be returned.

Response Body Message (Error Response)

Parameter	Data Type (Size)	M/O/C	Description
message	ans{1..255}	M	Readable message regarding the error.

9. HTTP Status Code

Following ***HTTP Status Code***s applicable to message response from payment gateway.

Status Code	Description
200	OK - Successful response for HTTP requests.
400	Bad Request – PG server rejects request from merchant due to : syntax error or insufficient request information (missing parameters) transaction amount less than minimum amount Invalid API version
401	Unauthorized – merchant request does not pass the PG authentication. Example scenarios such as an unregistered merchant server's IP address trying to make a request to the PG server. Invalid application code. Duplicate referenceID.
404	Not found – Application code or transaction not found
500	Internal Server Error – Error occurred due to PG internal processing.
502	Bad Gateway – Error occurred on channel. Merchant required to inquiry to check/reversal.
504	Gateway timeout – timeout between PG server and channel. Merchant required inquiry to check/reversal.

10. Status Code

Code	Description
00	**Success** The transaction was completed successfully.
01	**Pending** Unknown transaction status. **Merchant is required to inquire about the transaction to further confirm transaction status.** Please refer to the case study in the "Pending Authorize" section.
11	**Pending authorize** Transaction is pending for the user to authorize. Typically the user will prompt to enter a password or payment PIN on a wallet app on mobile. Merchant is required to trigger a status inquiry to further confirm the transaction status. Best practise is to send inquiry every 10 seconds for at least 30 seconds, if still not able to get approval status, void the payment. Please refer to the case study in the "Pending Authorize" section.
99	**Failed** The transaction failed.

11. Error Code

Code	Description
1000	**Client version not matched** User eWallet client version is not up-to-date. Please ask the user to update to the latest version before retry payment.
1001	**Invalid authorization code** The authorization code captured is invalid or expired. Please rescan and retry payment.
1002	**Insufficient balance** User eWallet does not have sufficient balance to proceed payment.
1003	**Exceed transaction limit** Transaction had exceeded the limitation set in eWallet payment provider.
1004	**Forbidden word** There is a forbidden word in the description that does not pass China authority.
1005	**Payer account not exists** Payer account does not exist.
1006	**Forbidden payer account** Payer account is disabled or current status does not allow it to perform payment.
1007	**Payer disabled payment option** Payer has disabled or does not enable related payment options to allow payment.
1008	**Refund amount exceeded** Refund amount had exceeded the original amount or the remaining amount.
1009	**Unable to reverse or refund** Transactions had been reversed or refunded.
1010	**Trade Closed** Transaction had expired.

1011	**Merchant account unauthorized.** Merchant account not found or invalid merchant account info at channel side.
1012	**Invalid channel coupon / voucher.** Applied channel coupon / voucher unable to use.
1013	**Invalid forex rate, currency info, or country info.** System unable to detect forex rate, currency info, or country info.
1014	**Unable to reverse or refund** Payment made with a voucher unable to reverse or refund.
1015	**Transaction refunded or voided due to success payment exceeding payment window** Payment was refunded or voided due to the payment window had closed.
9999	**Other error**

Code	Description
40000	**Bad Request**
40001	**Invalid AuthorizationCode**
40002	**Invalid API version**
40003	**Invalid CurrencyCode**
40004	**Promo code fully redeemed**
40005	**Invalid ChannelId**
40006	**This channel does not support the following API**
40007	**Image size value must be in WIDTHxHEIGHT format**
40008	**Exceed authorized amount**
40009	**Duplicate Reference Id**
40010	**Duplicate request is not allowed**
40100	**Unauthorized**
40101	**Invalid ApplicationCode**
40102	**Invalid Hash Type**

40103	**Invalid Signature**
40104	**Channel not enabled or account inactive**
40105	**Minimum amount is @currency 0.10**
40106	**Image format not supported**
40107	**Image resolution not supported**
40108	**Trade closed**
40109	**Invalid promo Voucher** Voucher/Coupon used does not meet the terms and conditions stated.
40110	**Transaction not allowed to reverse or refund**
40111	**Invalid reconciliation request info**
40400	**Payment Not Found**
40401	**@parameter is required**
40402	**Merchant account not found**
40403	**Refund record not found**
50000	**Internal server error**
50030	**System is busy now or temporarily out of service. Please try again later**
50200	**Bad Gateway**

12. Reconciliation File

Reconciliation files all the successful transaction including
payment, reversal, refund made in T-1 day. Merchant system can use this reconciliation file to perform reconciliation.. There will be three types of reconciliation files,

- Transaction Reconciliation File
- Summary Reconciliation File (****Will be added in the near future**)
- Store Summary Reconciliation File (****Will be added in the near future)**

Environment	Service URL
Sandbox	https://sandbox.fiuu.com/RMS/API/MOLOPA/reconciliation.php
Production	https://**api**.fiuu.com/RMS/API/MOLOPA/reconciliation.php

Request Header	
HTTP Method	GET
Content Type	x-www-form-urlencoded

Request Body Message

Parameter Name	Data Type (Size)	M/O/C	Description
applicationCode	ans{1..32}	M	***Application Code*** is uniquely identifying merchant applications which integrate with Offline Payment API. A merchant company could have up to 1 merchant account and multiple application accounts.
version	ans{1..3}	M	***Version*** of Offline Payment API starts with prefix "v" followed by the version number. When a merchant makes an HTTP request to PG, the version must be specified in the parameter. **Refer to section Version**

<table>
<tr><td>businessDate</td><td>ans{10}</td><td>M</td><td>Business date is a business date on the merchant side to which the transaction belongs. This is to ease the reconciliation when merchants have different business cutoff time with PG. Leave empty will follow the transaction date at the PG system.

Format: yyyy-MM-dd</td></tr>
<tr><td>hashType</td><td>ans{3..11}</td><td>M</td><td>Hash Type is the hashing algorithm used to generate signatures. Left empty will default as md5 (only for v1).

<table><tr><th>Value</th><th>Description</th></tr><tr><td>md5</td><td>MD5</td></tr><tr><td>hmac-sha256</td><td>HMAC-SHA256</td></tr></table></td></tr>
<tr><td>type</td><td>a{3}</td><td>M</td><td>Type refers to the report that requires to be generated

<table><tr><th>Value</th><th>Description</th></tr><tr><td>txn</td><td>Transaction recon file</td></tr></table></td></tr>
<tr><td>download</td><td>a{3}</td><td>O</td><td>Download refers to the type of file extension that will be exported after successfully generating the list of transactions

<table><tr><th>Value</th><th>Description</th></tr><tr><td>txt</td><td>.txt</td></tr><tr><td>csv</td><td>.csv</td></tr></table></td></tr>
<tr><td>signature</td><td>ans{32}</td><td>M</td><td>All parameters required for signature creation (refer to Generate Signature)</td></tr>
</table>

12.1 Transaction Reconciliation File

File Format

The file name format is according to transaction date, transaction_YYYYMMDD.txt. The file consists of 2 parts,

First two rows are file header and subsequence rows are record details.

File Header

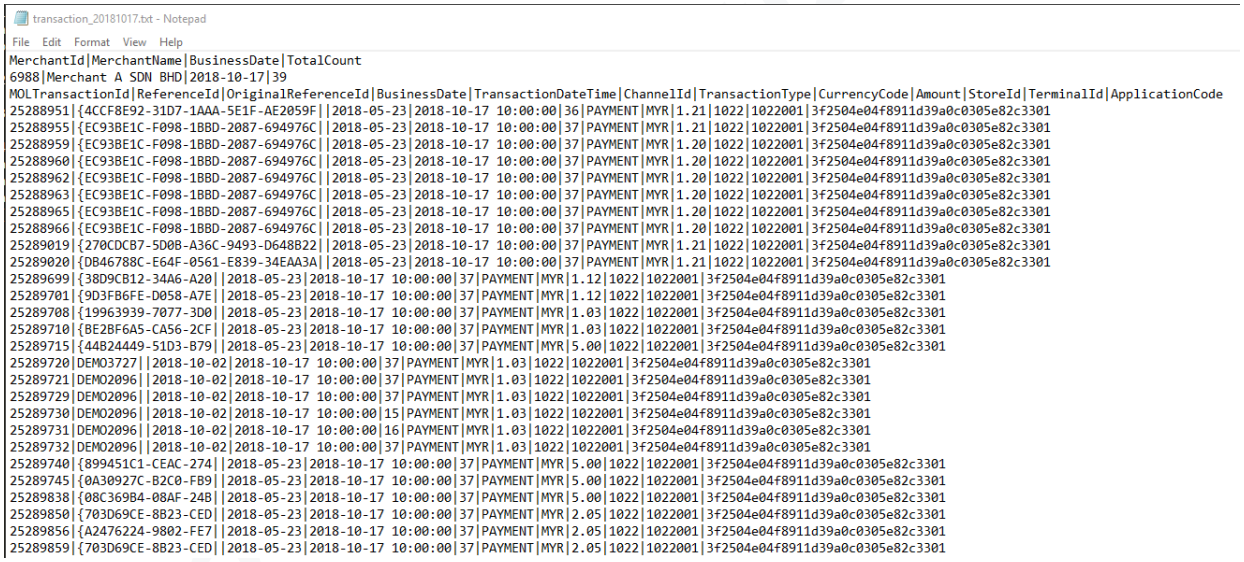
Field	Type	Description	Sequence
MerchantId	Number(9)	Merchant's unique identifier	1
MerchantName	String(100)	Merchant name	2
BusinessDate	String(10)	Format YYYY-MM-DD	3
TotalCount	Number(9)	Total number of records in this file.	4

Record Detail

Field	Type	Description	Sequence
MOLTransactionId	Number(10)	PG transaction identifier	1
ReferenceId	String(40)	Merchant transaction identifier	2
OriginalReferenceId	Number(40)	Original Merchant's payment transaction identifier for Reversal and Refund. Same as ReferenceId if this is Payment transaction.	3
BusinessDate	String(10)	Format YYYY-MM-DD	4

TransactionDateTime	String(19)	Format YYYY-MM-DD HH:MM:SS	5
ChannelId	Number(9)	Payment Channel	6
TransactionType	String(10)	PAYMENT;REFUND	7
CurrencyCode	String(3)	Currency Code	8
Amount	Decimal(9,2)	Transacted amount	9
StoreId	String(20)	Store Id	10
TerminalId	String(20)	Terminal Id	11
ApplicationCode	String(32)	Application's Code	12

Sample transaction reconciliation file:



transaction_20181017.txt - Notepad

File Edit Format View Help

```
MerchantId|MerchantName|BusinessDate|TotalCount
6988|Merchant A SDN BHD|2018-10-17|39
MOLTransactionId|ReferenceId|OriginalReferenceId|BusinessDate|TransactionDateTime|ChannelId|TransactionType|CurrencyCode|Amount|StoreId|TerminalId|ApplicationCode
25288951|{4CCF8E92-31D7-1AAA-5E1F-AE2059F||2018-05-23|2018-10-17 10:00:00|36|PAYMENT|MYR|1.21|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25288955|{EC93BE1C-F098-1BBD-2087-694976C||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|1.21|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25288959|{EC93BE1C-F098-1BBD-2087-694976C||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|1.20|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25288960|{EC93BE1C-F098-1BBD-2087-694976C||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|1.20|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25288962|{EC93BE1C-F098-1BBD-2087-694976C||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|1.20|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25288963|{EC93BE1C-F098-1BBD-2087-694976C||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|1.20|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25288965|{EC93BE1C-F098-1BBD-2087-694976C||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|1.20|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25288966|{EC93BE1C-F098-1BBD-2087-694976C||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|1.20|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289019|{270CDCB7-5D0B-A36C-9493-D648B22||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|1.21|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289020|{DB46788C-E64F-0561-E839-34EAA3A||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|1.21|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289699|{38D9CB12-34A6-A20||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|1.12|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289701|{9D3FB6FE-D058-A7E||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|1.12|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289708|{19963939-7077-3D0||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|1.03|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289710|{BE2BF6A5-CA56-2CF||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|1.03|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289715|{44B24449-51D3-B79||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|5.00|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289720|DEMO3727||2018-10-02|2018-10-17 10:00:00|37|PAYMENT|MYR|1.03|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289721|DEMO2096||2018-10-02|2018-10-17 10:00:00|37|PAYMENT|MYR|1.03|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289729|DEMO2096||2018-10-02|2018-10-17 10:00:00|37|PAYMENT|MYR|1.03|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289730|DEMO2096||2018-10-02|2018-10-17 10:00:00|15|PAYMENT|MYR|1.03|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289731|DEMO2096||2018-10-02|2018-10-17 10:00:00|16|PAYMENT|MYR|1.03|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289732|DEMO2096||2018-10-02|2018-10-17 10:00:00|37|PAYMENT|MYR|1.03|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289740|{899451C1-CEAC-274||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|5.00|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289745|{0A30927C-B2C0-FB9||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|5.00|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289838|{08C369B4-08AF-24B||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|5.00|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289850|{703D69CE-8B23-CED||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|2.05|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289856|{A2476224-9802-FE7||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|2.05|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
25289859|{703D69CE-8B23-CED||2018-05-23|2018-10-17 10:00:00|37|PAYMENT|MYR|2.05|1022|1022001|3f2504e04f8911d39a0c0305e82c3301
```

12.2 Summary Reconciliation File

File Format

The file name format is according to transaction date, summary_YYYYMMDD.txt. The file consists of 2 parts,

First two rows are file header and subsequence rows are record details.

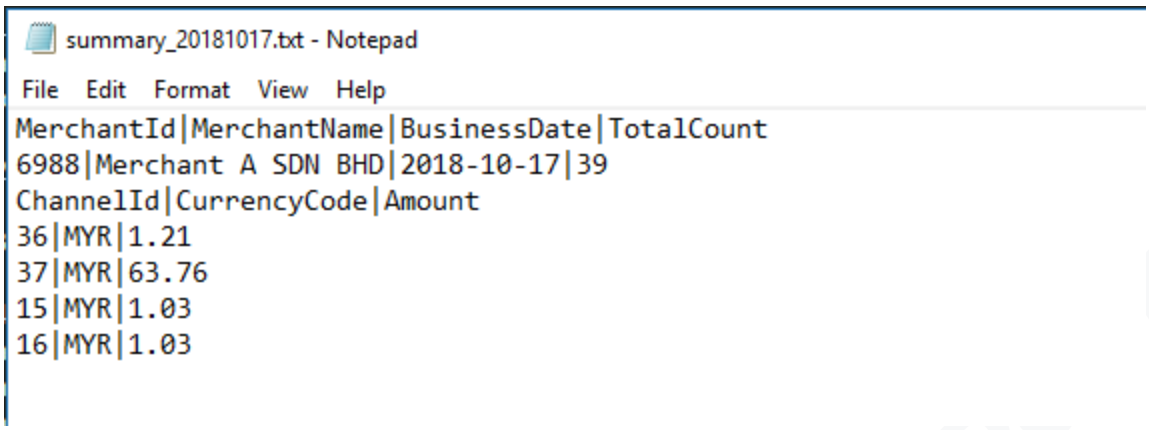
File Header

Field	Type	Description	Sequence
MerchantId	Number(9)	Merchant's unique identifier	1
MerchantName	String(100)	Merchant name	2
BusinessDate	String(10)	Format YYYY-MM-DD	3
TotalCount	Number(9)	Total number of records in this file.	4

Record Detail

Field	Type	Description	Sequence
ChannelId	Number(9)	Payment Channel	1
CurrencyCode	String(3)	Currency Code	2
Amount	Decimal(9,2)	Transacted amount	3

Sample summary reconciliation file:


summary_20181017.txt - Notepad

File Edit Format View Help

```
MerchantId|MerchantName|BusinessDate|TotalCount
6988|Merchant A SDN BHD|2018-10-17|39
ChannelId|CurrencyCode|Amount
36|MYR|1.21
37|MYR|63.76
15|MYR|1.03
16|MYR|1.03
```

12.3 Store Summary Reconciliation File

File Format

The file name format is according to transaction date, store_summary_YYYYMMDD.txt. The file consists of 2 parts,

First two rows are file header and subsequence rows are record details.

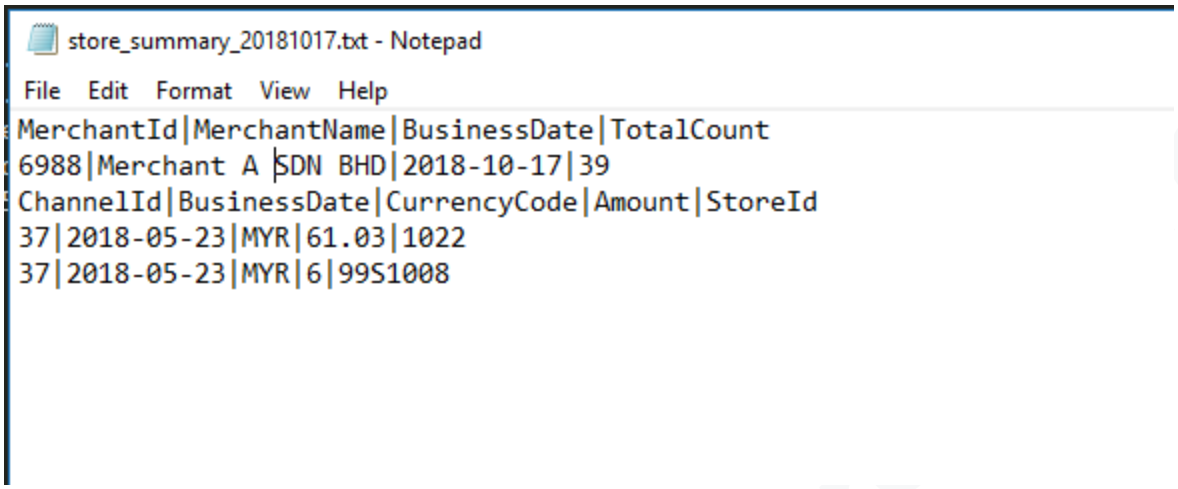
File Header

Field	Type	Description	Sequence
MerchantId	Number(9)	Merchant's unique identifier	1
MerchantName	String(100)	Merchant name	2
BusinessDate	String(10)	Format YYYY-MM-DD	3
TotalCount	Number(9)	Total number of records in this file.	4

Record Detail

Field	Type	Description	Sequence
ChannelId	Number(9)	Payment Channel	1
BusinessDate	String(10)	Format YYYY-MM-DD	2
CurrencyCode	String(3)	Currency Code	3
Amount	Decimal(9,2)	Transacted amount	4
StoreId	String(20)	Store Id	5

Sample transaction reconciliation file:



13. Receipt Requirements

Below is the required information to be printed in the customer's receipt.

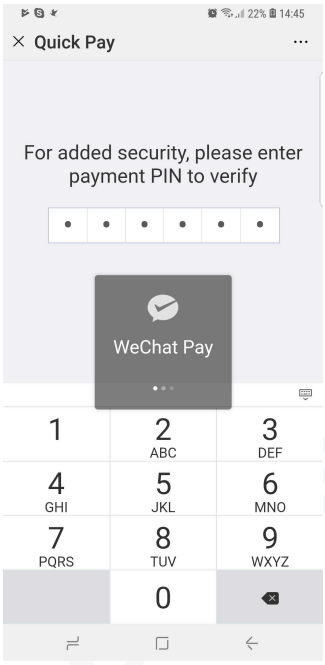
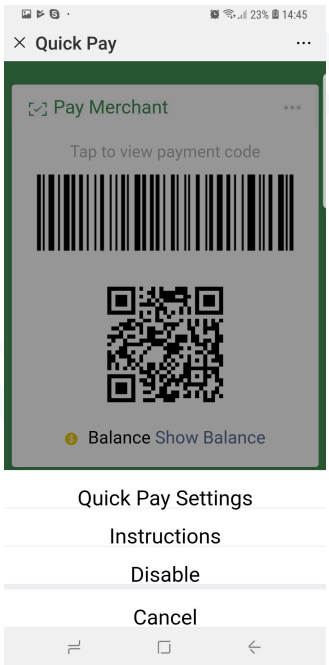
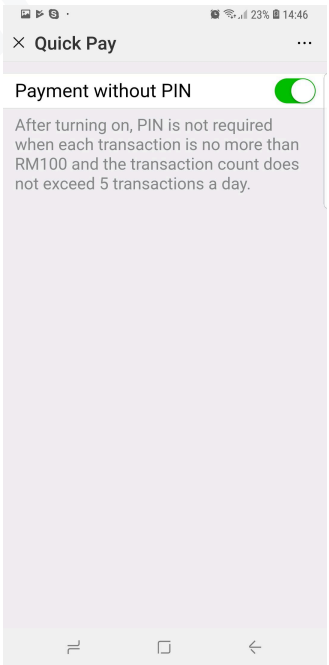
- Merchant name or DBA
- Store ID/name
- Address or contact method
- Terminal ID
- Cashier ID
- Merchant Reference ID
- PG Transaction ID
- Transaction Date time
- Transaction amount in local currency
- Transaction amount in wallet currency
- Forex rate (if any)

14. Pending Authorize (MUST READ)

This is a payment flow that requires the buyer to interact during the payment process. Once a wallet issuer or operator detects any high-risk transaction, manual payment authorization will be prompted in the APP and internet access is required to complete the authorization process in some cases.

For ***Wechat Pay Malaysia***, whenever the transaction amount is greater than MYR100.00, or conducts more than 5 times payment a day, payment PIN will be required to approve the transaction.

For ***Alipay*** & ***Touch 'n Go eWallet***, any unusual activity might trigger a request for Payment PIN or OTP via SMS, in order to approve the transaction.

		
Buyer will be requested to set payment PIN for security reason	Buyer can actually disable the payment PIN for small transaction amount in "Quick Pay Settings"	Payment without PIN: After turning on, PIN is not required when each transaction is no more than RM100 and the transaction count does not exceed 5 transactions a day.

In case the payment flow needs approval from the buyer, POS or terminal will need to send status inquiry every 10 seconds for **at least 60 seconds** or 6 times, and display proper message on the screen that “waiting for buyer approval” or “waiting for buyer to enter payment PIN”.If still not able to get the approval status, sending a **VOID (REVERSAL)** request to cancel the payment regardless of the status.

15. extraInfo

It contains additional information provided by the payment processor.

Parameter	Description
channelReferenceId	Reference ID returned from channel
GrabPay	
paymentMethod	Payment option that the user chose during the payment, e.g. GPWALLET, POSTPAID, INSTALMENT_4.
DuitNow QR	
DbtrAgt	User bank/wallet account issuer during the payment, e.g. MBBEMYKL
DbtrAcct_Type	User account type, e.g. SVGS, CCRD, CACC, WALL.
TxnType	User account origin and payment location, e.g DOMESTIC, CROSSBORDER.
UnionPay	
discountDescription	Describe the discount information.
discountAmount	Total discount amount applied in the payment.
netAmount	Total net amount paid by the user after deducting the discount amount.
baseDiscountAmount	Total discount amount in the user wallet currency.
baseNetAmount	Total net amount paid in the user wallet currency.
Alipay+	
walletIssuer	Payment providers like digital wallet, bank app, and etc.

16. Channel Validity Duration

Channel Name	Customer Presented (Payment API)	Merchant Presented (Precreate API)	Minimum Due Time (Seconds)	Maximum Due Time (Seconds)
RazerPay	❌	❌	❌	❌
Alipay	❌	❌	❌	❌
TNG-D	❌	❌	❌	❌
Alipay Pre-Auth	❌	❌	❌	❌
Boost	❌	❌	❌	❌
MAE by Maybank2u ** Only one option is allowed	❌	❌	❌	❌
GrabPay	❌	❌	❌	❌
UnionPay	❌	❌	❌	❌
ShopeePay	✔️	✔️	60	120
DuitNow QR	❌	✔️	60	180
Alipay+	❌	❌	❌	❌
Atome	❌	❌	❌	❌
WeChatPay (CN)	❌	❌	❌	❌
WeChatPay (MY)	❌	❌	❌	❌
PayNow	❌	✔️	120	600
KBank QR	❌	❌	❌	❌
QRPH	❌	✔️	❌	1800

✔️ - Available
❌ - Not available

_The_End_