API Specification of HSBC ASP Mobile Collection for Retail Payments in India

API Base URL

```
#Production
https://cmb-api.hsbc.com.hk/glcm-mobilecoll-mcin-ea-merchantservices-prod-proxy/v1

#Sandbox
https://devclustercmb.api.p2g.netd2.hsbc.com.hk/glcm-mobilecoll-mcin-ea-merchantservices-cert-proxy/v1
```

Schemes: https

Version: 1.2

Purpose of this document

This document provide the audience with **OpenAPI specification** for describing REST APIs of HSBC ASP Mobile Collection for Retail Payments.

The target audience of this document is the Developer, Business Analyst and other related Project Team Member (who has the basic technical know-how of Web technology such as REST or JSON) of HSBC's client (i.e. the Merchant)

Update Log

- [Oct 28, 2020] **v1.2** Added New API for HSBC UPI Support
- [Aug 10, 2020] **v1.1**
  - Added section Download Swagger
  - Added fields `offers` and `discount` in Redirect, Enquiry and Notification API
- [May 25, 2020] **v1.0** Initial Version

How to Read this Document

This document walks through the API usage and lists the key idea by section like API Usage Flow, API Connectivity and API Operation. There is also a FAQ and Schema Definition that defines API operation.

Channels and Features

HSBC Mobile Collection provides a wide range of online payment solutions which allows e/m-commerce owner to process online payments. The payment platform supports implementation with websites or mobile applications.

Using our APIs services, merchant can accept and manage payments including the following payment channels.

Payment Channels

Payment Channels
Credit and Debit Card
e-Wallet
Internet Banking
UPI (or QR Code)
EMI

Our solution also offer choices between different Payment Gateway Partners depending on merchant's business need. Please contact our team to understand more. To present any proprietary terminology or service provided by one specific Payment Gateway Partner, the content will be highlighted in a coloured `Block Quote` as in the example below:

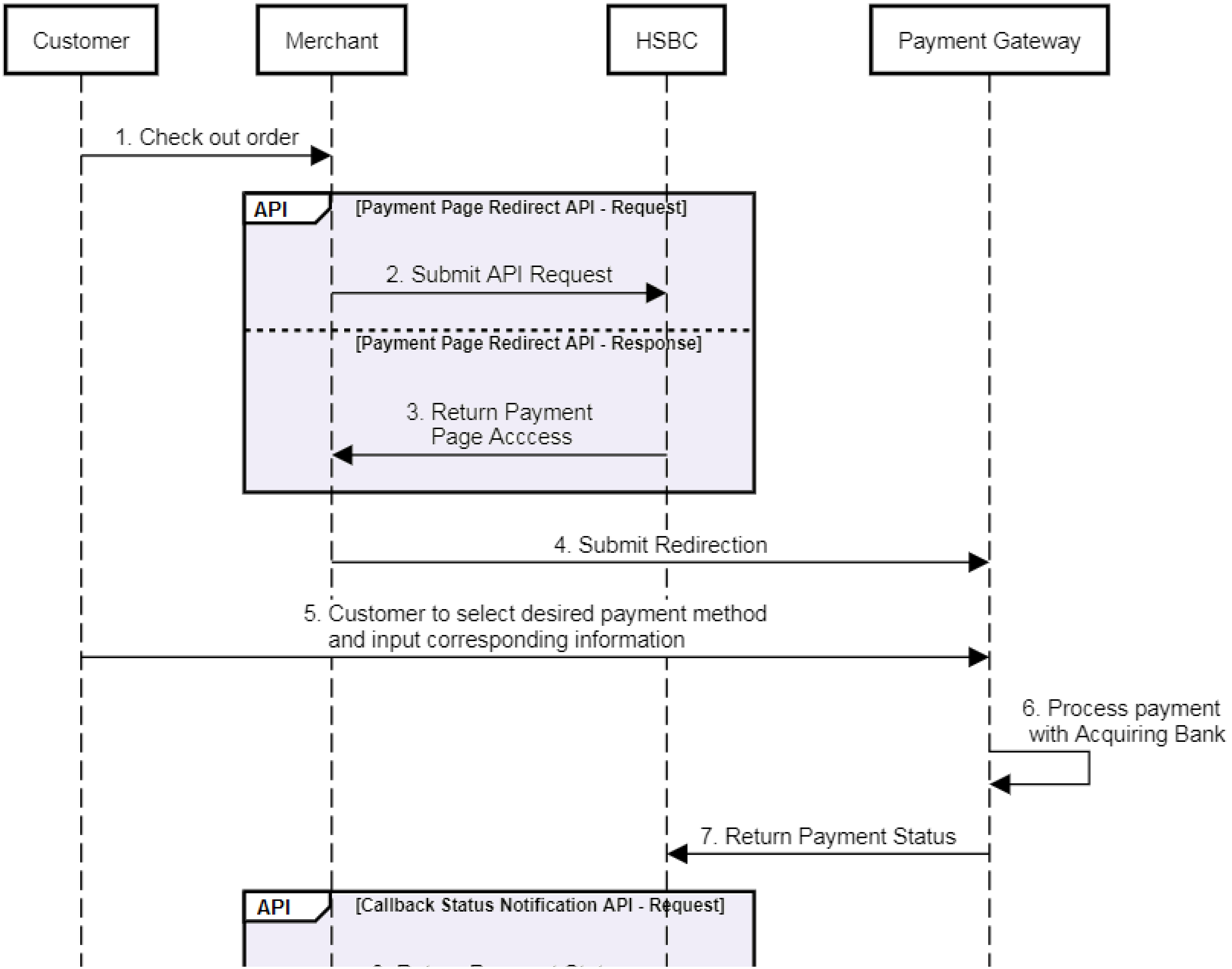
> Gateway 1
>
> **INFORMATION:**
> One difference between two Payment Gateways is the technical design that accesses the Online Payment Page, Payment Gateway #1 offers a HTML Form Submit and requires HTML redirection.

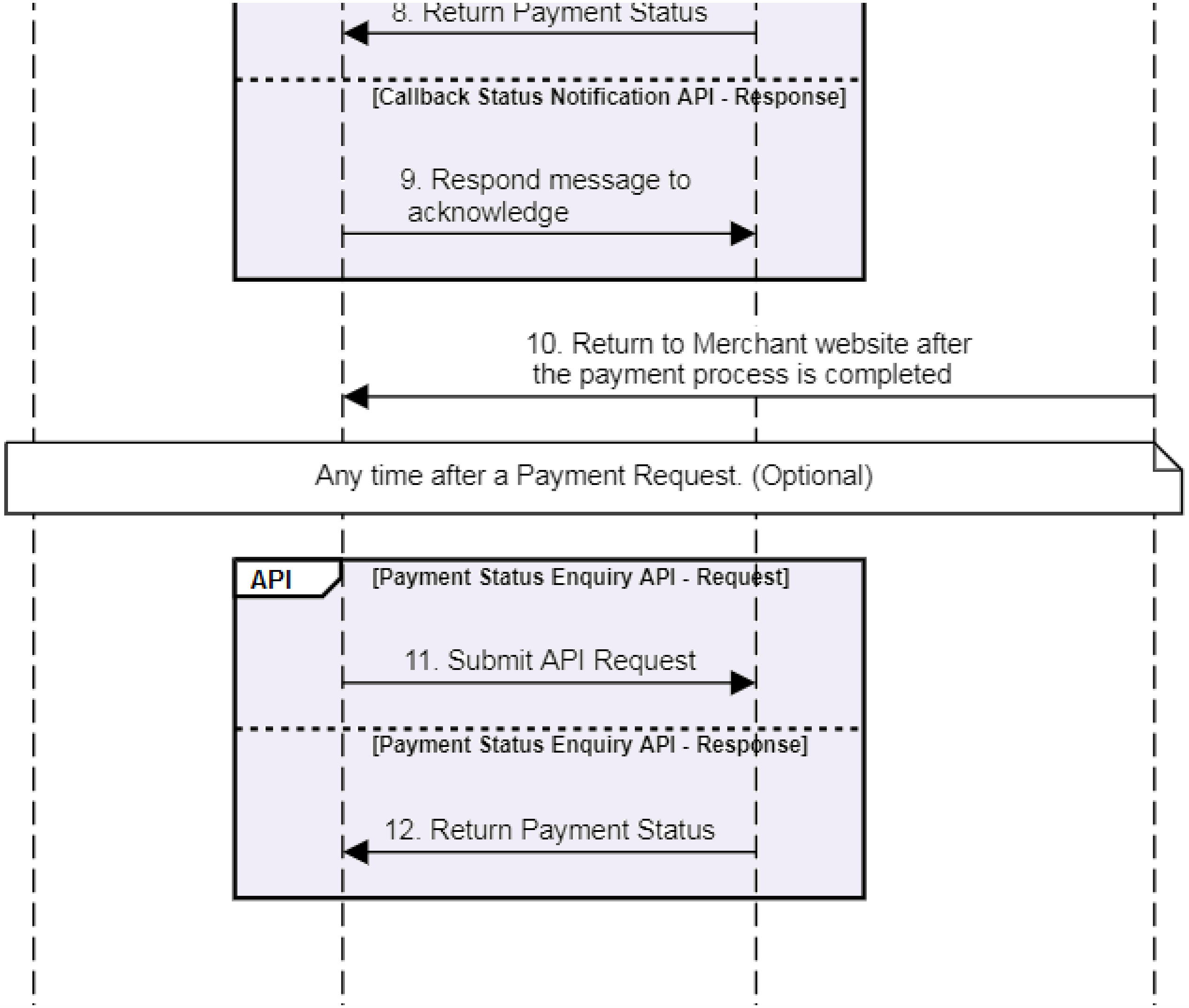
> Gateway 2
>
> **INFORMATION:**
> While Payment Gateway #2 offers either a Javascript Caller Function while the Payment Page will be a overlay on the existing webpage or a static URL link which the payment page will present in a separate page. For more technical details, please see Payment Page Redirect API.

Make Online Payments

Please follow the API use flow in order to implement a complete online payment:

API Use Case


Customer
Merchant
HSBC
Payment Gateway
1. Check out order
API
[Payment Page Redirect API - Request]
2. Submit API Request
[Payment Page Redirect API - Response]
3. Return Payment Page Acccess
4. Submit Redirection
5. Customer to select desired payment method and input corresponding information
6. Process payment with Acquiring Bank
7. Return Payment Status
API
[Callback Status Notification API - Request]


8. Return Payment Status
[Callback Status Notification API - Response]
9. Respond message to acknowledge
10. Return to Merchant website after the payment process is completed
Any time after a Payment Request. (Optional)
API
[Payment Status Enquiry API - Request]
11. Submit API Request
[Payment Status Enquiry API - Response]
12. Return Payment Status

1. Customer conducts checkout process in merchant's website.
2. Merchant submits Payment Page Redirect API request to HSBC.
3. Gateway 1

   NOTICE:

   Payment Page Access will be returned as a HTML submit form where it's contained in response field `redirectLink`.

> Gateway 2
>
> **NOTICE:**
> Payment Page Access will be returned as a Javascript code which it's contained in response field `redirectLink` and a static URL link in field `redirectUrlLink`.

More technical details will be covered in Payment Page Redirect API.

4. Merchant submits page redirection to the Online Payment Page.
5. Customer selects their desired payment channel in the payment page and input corresponding information such as Credit Card details.

> ! **NOTICE:** Some payment channels will lead further webpage redirection such as Internet Banking.

6. Payment page will connect securely to bank and backend systems to process the payment.
7. HSBC will receive payment status once it is updated from backend system.
8. HSBC will then trigger Callback Payment Notification API and send payment status back to Merchant.

> ! **NOTICE:** This server-to-server Notification will only be sent out for a success payment case and Merchant can define their URL endpoint in request field `notificationUrl` in Payment Page Redirect API

9. Merchant responds the API to acknowledge. Fail to return a proper response will trigger Notification resend mechanism.
10. Redirect back to merchant website once the payment process is completed in the Payment Gateway.

> Gateway 1
>
> **NOTICE:**
> Merchant can define this redirect back URL in request fields `redirectSuccessUrl` `redirectFailUrl` and `redirectCancelUrl` in Payment Page Redirect API according to different scenario.

> Gateway 2
>
> **NOTICE:**
> Payment Gateway 2 can only support one redirect back link.

11. Merchant can optionally submit Payment Status Enquiry API at any time after a payment request is submitted. This is useful when Merchant finds no acknowledge message returned after a certain period of time.

12. HSBC will return the latest payment status according to the transaction reference number Merchant provided.

Check Status Feature

Mobile collection offers API to check status of every payment transaction. To implement Check Status, please see the Status Enquiry API.

Cancel & Refund

Merchant can request Order Cancellation & Refund API to either cancel an existing order whose payment transaction is yet to be settled or refund a settled transaction (Settled on both issuing and acquiring bank).

HSBC accepts Full Refund and multiple Partial Refund. Every refund is a new transaction and will be returned in an array object in the Status Enquiry API response message.

Order Confirmation

Regarding to the aforementioned API use case flow, the last step is to redirect the Payment Page back to the Merchant website. Merchant can build a dynamic Order Confirmation Page with payment details where the details can be retrieved from the asynchronous Callback Payment Notification API.

How to Connect

API Connectivity refers to all measures and their components that establishes connection between HSBC, the API Provider and Merchant, the API Consumer.

	Definition	Components
API Authentication	HTTP BASIC Authentication	• Username • Password
	Locate API Gateway Policy of the corresponding user	• Client ID • Client Secret
User Identification	A Merchant Profile	• Merchant ID • Merchant Profile
Connection Security	HTTPS Connection (TLS 1.2) and Network Whitelisting	• SSL Certificate • Network Whitelist
Message Security	Digital Signing and Data Encryption	• A pair of Private Key & Public Key Certificate (PKI Model) • JWS Key ID • JWE Key ID

API Authentication

Username & Password	
Purpose	All APIs are authorized using `Basic Authorization`

Username & Password		
Components	- Username	- Password
Where to get it?	Delivered by HSBC via secure email during onboarding procedure	
Implementation	In HTTP header: `Authorization: Basic [Base64-encoded Credential]`	

Client ID & Client Secret		
Purpose	API Gateway locates the corresponding policy of the specific API consumer	
Components	- Client ID	- Client Secret
Where to get it?	Delivered by HSBC via secure email during onboarding procedure	
Implementation	In HTTP header: `x-hsbc-client-id: [Client ID]`	In HTTP header: `x-hsbc-client-secret: [Client Secret]`

User Identification

Merchant Profile & Merchant ID

Merchant Profile & Merchant ID		
Purpose	- Merchant Profile contains all necessary information from a Merchant in order to enable payment service.	- Merchant ID is used for Merchant identification in each API call.
Components	- Merchant Profile	- Merchant ID
Where to get it?	- Set up by HSBC team after collect information from Merchant	- Delivered by HSBC via secure email during onboarding procedure
Implementation	*nil*	In HTTP header: `x-hsbc-msg-encrypt-id: [Merchant ID]+[JWS ID]+[JWE ID]`

Connection Security

SSL Certificate & Network Whitelist			
Purpose	- Request HSBC API over HTTPS connection (TLS 1.2)	- Accept Callback API request over HTTPS connection (TLS 1.2)	
Components	- Public SSL Certificate issued by HSBC	- Merchant's web server or domain whose HTTPS connection is enabled	- Network Whitelist on HSBC system

SSL Certificate & Network Whitelist			
Where to get it?	- Downloaded automatically by Browsers or API Tools, if any problem found, please contact HSBC	*nil*	*nil*
Implementation	*nil*	*nil*	- Merchant's domain URL will be configured in HSBC's network whitelist by HSBC team

Message Security - Data Encryption and Signing

On top of the Transport Layer Security, HSBC adopts additional security on the message being passed through the connection session. Data Encryption actually serves as a locked briefcase containing the data (the API message) within the HTTPS "tunnel". In other word, the communication has double protection.

> ! **DO YOU KNOW?**
> Javascript Object Signing and Encryption **(JOSE™)**, is a framework intended to provide methods to securely transfer information between parties. The JOSE framework provides a collection of specifications, including JSON Web Signature **(JWS™)** and JSON Web Encryption **(JWE™)**, to serve this purpose.

HSBC uses JWS to sign message payload and JWE to encrypt the signed message while these two objects are created by using a pair of Private Key & Public Key Certificate (PKI Model).

Private Key & Public Key Certificate (PKI Model)		
Purpose	- Digitally sign a API request message - Decrypt a API response message	- Encrypt the signed API request message - Verify a signed API response message

Private Key & Public Key Certificate (PKI Model)		
Components	- Private Key issued by Merchant	- Public Key Certificate issued by HSBC
Where to get it?	- Created by any Public Key Infrastructure (PKI) toolkits, such as Keytool™ and OpenSSL™. Technical detail is in here	- Exchanged with HSBC with the Public Key Certificate issued by Merchant
Implementation	Please see the technical detail in here	

> ! **NOTICE:**
> Technically, X.509 certificate can be served as a SSL Certificate as well as a Public Key Certificate for Data Encryption. However, HSBC recommends Merchant to use a different X.509 Certificate for Data Encryption for segregation of certificate usage.
>
> Moreover, the Public Key Certificate does not have to be CA-signed. However, if Merchant decides to enhance security, a CA-Signed Certificate is always welcome.

keyID of JWS™ & JWE™		
Purpose	- The unique identifier to bind Merchant's Private Key in order to create a JWS object - a signed Message Payload	- The unique identifier to bind HSBC's Public Key Certificate in order to create a JWE object - an encrypted JWS object
Components	- keyID of JWS™	- keyID of JWE™
Where to get it?	- Mutual agreed between Merchant and HSBC	- Mutual agreed between Merchant and HSBC

keyID of JWS™ & JWE™	
Implementation	• Define in program coding, see demo in here, and; • In HTTP header: `x-hsbc-msg-encrypt-id: [Merchant ID]+[JWS ID]+[JWE ID]`

> ! **NOTICE:**
> For security purposes, `HSBC's Public Key Certificate` and its associated `keyID` will be renewed **every** year and a Certificate Renewal process will be triggered. More detail is covered in section Key Renewal

How to Sign and Encrypt Outgoing Message

Every message sent to HSBC must be signed and encrypted. From the point of view of a Merchant, an **Outgoing Message** means:

- the Request Message of a Normal API, or
- the Respond Message of a Callback API.

To help you understand how to construct a Signed and Encrypted Message, let's take the Java program below as an example. Do not worry if you are not familiar with Java, the idea is to let you know the steps and all needed components:

> ! **NOTICE:** These Java codes are for demonstration only and it's not *plug and play*.

```
private JWSObject signMessage(String messagePayload, KeyStore ks, String keyAlias, String keyPw)
  throws UnrecoverableKeyException, KeyStoreException, NoSuchAlgorithmException, JOSEException {
#1  Payload payload = new Payload(messagePayload);
#2  JWSHeader header = new JWSHeader.Builder(JWSAlgorithm.RS256).keyID("0001").build();
#3  JWSObject jwsObject = new JWSObject(header, payload);

#4  PrivateKey privateKey = (PrivateKey) ks.getKey(keyAlias, keyPw.toCharArray());
    JWSSigner signer = new RSASSASigner(privateKey);
#5  jwsObject.sign(signer);

    return jwsObject;
}
```

1. Prepare your **Message Payload**, that is, the plain `json` request message
2. Create **JWS Header** using `RS256` signing algorithm and **JWS keyID**, in this case, `0001`
3. Create **JWS Object** by combining JWS Header and Message Payload
4. Retrieve your **Private Key** as the signer
5. Create **Signed JWS Object** by signing it with the Private Key

Next, you are going to **Encrypt** the Signed JWS Object:

```
private JWEObject getEncryptedJWEObject(JWSObject jwsObject, RSAPublicKey key)
  throws JOSEException {
#1  Payload jwepayload = new Payload(jwsObject.serialize());

#2  JWEHeader jweheader = new JWEHeader.Builder(JWEAlgorithm.RSA_OAEP_256, EncryptionMethod.A128GCM).keyID("0002").build();
#3  JWEObject jweObject = new JWEObject(jweheader, jwepayload);

#4  JWEEncrypter encrypter = new RSAEncrypter(key);
#5  jweObject.encrypt(encrypter);

    return jweObject;
}
```

1. Prepare your **JWE Payload**, that is, the `Signed JWS Object`
2. Create **JWE Header**. The algorithm used to encrypt the message body is `A128GCM` while the algorithm used to encrypt the encryption key is `RSA_OAEP_256`. **JWE keyID** is `0002`.
3. Create **JWE Object** by combining JWE Header and JWE Payload

4. Retrieve **HSBC's Public Key** as the encrypter
5. Create **Encrypted JWE Object** by encrypted it with HSBC's Public Key

Yes, you are now ready to put the Encrypted JWE Object as the message body *(you may need to first serialize it into String format, depends on your program code design)* of any API call.

How to Decrypt Message and Verify Signature of an Incoming Message

Every message sent from HSBC must be decrypted and verified. From the point of view of a Merchant, an **Incoming Message** means:

- the Respond Message of a Normal API, or
- the Request Message of a Callback API.

Let's look into the following example to see how you decrypt a response message from HSBC:

```
private String decryptMessage(String respMsgPayload, KeyStoreFactory keyStore)
  throws KeyStoreException, NoSuchAlgorithmException, CertificateException, IOException,
         java.text.ParseException, UnrecoverableKeyException, JOSEException {
#1  JWEObject jweObject = JWEObject.parse(respMsgPayload);

#2  PrivateKey privateKey = (PrivateKey) keyStore.getPrivateKey("merchant_private_key_alias");

    JWEDecrypter decrypter = new RSADecrypter(privateKey);
#3  jweObject.decrypt(decrypter);

#4  String signedMessage = jweObject.getPayload().toString();
    return signedMessage;
}
```

1. Create **Encrypted JWE Object** by parsing the encrypted response message payload
2. Retrieve **Private Key** as the decrypter
3. Decrypt the JWE Object using your Private Key

4. Get the **Signed Message** from the decrypted JWE Object

You are now able to extract the plain `json` message. Yet, before that, you **must** verify the signature to guarantee data integrity.

```
private String verifySignature(String signedMessage, KeyStore ks, String keyAlias)
  throws KeyStoreException, JOSEException, ParseException {
#1  JWSObject jwsObject = JWSObject.parse(signedMessage);

    Certificate certificate = ks.getCertificate(keyAlias);
#2  JWSVerifier verifier = new RSASSAVerifier((RSAPublicKey) certificate.getPublicKey());

#3  if (!jwsObject.verify(verifier)) {
      throw new ValidationException("Invalid Signature");
    }
#4  return jwsObject.getPayload().toString();
}
```

1. Create **JWS Object** by parsing the `Signed Message`
2. Retrieve **HSBC's Public Key** as the verifier
3. Verify the signed JWS Object. Invoke error handling if invalid signature found *(depends on your code design)*
4. Get the plain `json` message for further actions

Summary

Components \ Steps	Message Signing	Message Encryption	Message Decryption	Verify Signature
JWS Object	Signing Algorithm: `RS256`			
JWE Object		JWE Algorithm: `RSA_OAEP_256` Encryption Method: `A128GCM`		

Components \ Steps	Message Signing	Message Encryption	Message Decryption	Verify Signature
KeyID	`0002`	`0002`		
Merchant's Private Key	Used as `Signer`		Used as `Decrypter`	
HSBC's Public Key		Used as `Encrypter`		Used as `Verifier`

How to Make API Request

API request can be submitted without Message Encryption, in case you want to:

- understand the basic API Call quick;
- test API connectivity before spending substantial development effort on Message Encryption.

However, data encryption is actually a required data security imposed by HSBC standard, Merchant has to invoke the encryption logic before moving to Production and fully tested during testing phase.

Make Your API Request with Plain Messages

> ! **NOTICE:**
> Skipping message encryption is the flexibility provided in Sandbox Environment for testing purpose.

Submit API request using cURL™ as an example

cURL™ is a simple command line tool that enables you to make any HTTP request. Merchant can choose any other GUI tool such as Postman™ and SoapUI™.

Step 1. Run this command in your system platform:

```
curl -X POST "https://devclustercmb.api.p2g.netd2.hsbc.com.hk/glcm-mobilecoll-mcin-ea-merchantservices-cert-proxy/v1/payment/e
 -H "message_encrypt: false"
 -H "Authorization: Basic eW91cl91c2VybmFtZTp5b3VyX3Bhc3N3b3Jk"
 -H "x-HSBC-client-id: 8b915a4f5b5047f091f210e2232b5ced"
 -H "x-HSBC-client-secret: 1bb456a541dc416dB6016B5F9583C606"
 -H "x-HSBC-msg-encrypt-id: 42298549900001+0001+0002"
 -H "Content-Type: application/json"
 -d "{ \"txnRef\": \"PAY-QJZV956664\", \"merId\": \"42298549900001\"}"
```

1. Submit `POST` request to the API URL endpoint
2. Put the secret header `message_encrypt: false` to indicate this API request is without message encryption. This header is only applicable in Sandbox environment.
3. Put the Basic Authorization in HTTP header `Authorization`
4. Put Client ID in HTTP header `x-HSBC-client-id`
5. Put Client Secret in HTTP header `x-HSBC-client-secret`
6. Put Merchant ID, JWS ID and JWE ID in HTTP header `x-HSBC-msg-encrypt-id` respectively
7. Set `Content-Type` to JSON format
8. Plain `json` message payload

Step 2. Receive response message in plain `json` format.

Making API Request with Message Encryption

Step 1. Run this cURL™ command in your system platform:

```
#1 curl -X POST "https://devclustercmb.api.p2g.netd2.hsbc.com.hk/glcm-mobilecoll-mcin-ea-merchantservices-cert-proxy/v1/payment/e
#2  -H "Authorization: Basic eW91cl91c2VybmFtZTp5b3VyX3Bhc3N3b3Jk"
#3  -H "x-HSBC-client-id: 8b915a4f5b5047f091f210e2232b5ced"
#4  -H "x-HSBC-client-secret: 1bb456a541dc416dB6016B5F9583C606"
#5  -H "x-HSBC-msg-encrypt-id: 42298549900001+0001+0002"
#6  -H "Content-Type: application/json"
#7  -d "eyJraWQiOiIwMDAxIiwiZW5jIjoiQTEyOEdDTSIsImFsZyI6IlJTQS1PQUVQLTI1NiJ9.W4nobHoVXUMOXGM5I-WGPZt8sj-hsd_sRujMHFbv80M72K4l0PvW
```

1. Submit `POST` request to the API URL endpoint
2. Put the Basic Authorization in HTTP header `Authorization`
3. Put Client ID in HTTP header `x-HSBC-client-id`
4. Put Client Secret in HTTP header `x-HSBC-client-secret`
5. Put Merchant ID, JWS ID and JWE ID in HTTP header `x-HSBC-msg-encrypt-id` respectively
6. Set `Content-Type` to JSON format
7. Encrypted Message Payload.

> ! **NOTICE:**
> Data Encryption invokes compulsory prerequisites, JOSE library and program coding, please make sure the section Message Security has been gone through thoroughly.

Step 2. For a successful request (HTTP Status Code 200), an encrypted response message will be returned, otherwise, a plain `json` with failure message will be returned.

Data Type Overview

Data Type Control:

Data Type	Allowed Characters	Definition & Important Notice
String *(For general field)*	AlphaNumeric and Symbols	General field means field which is **NOT** a critical field. HSBC system will execute characters checking upon all string fields we received in order to tackle security vulnerability, such as Cross-site Scripting. Yet, we recommend you to try use AlphaNumeric only for most cases.

String *(For critical field)*	`0-9` `a-z`
	`A-Z` `-` `_`
	`.`
		Critical field is used to be either a key or search criteria in HSBC backend system and hence tight restriction is applied to the allowed characters.
		Moreover, the starting and ending space of the string value will be trimmed before stored in HSBC system. For example, string `" example 12 34 "` will be trimmed to `"example 12 34"`.
		List of Critical Fields:
		`txnRef`
		`merId`
		`product_id`
		`rfdRef`

Integer	`0-9`	Instead of having Max Length check for String, integer range will be checked, e.g. `0 ≤ x ≤ 9999`

Field Mandatory Control:

Field Mandatory Type	Definition & Important Notice
Mandatory	Annotated with required tag in field definition section. Field & value must be present in the request with valid `JSON` format.
Optional	Annotated with optional tag in field definition section. If you don't want to pass fields that are optional, your handler should not pass neither empty strings `{"example":""}` nor blank value `{"example":" "}`.

Field Mandatory Type	Definition & Important Notice
Conditional	Annotated with conditional tag in field definition section. Required under a specific condition whose logic is always provided in the field definition if it is a Conditional Field.

Time Zone Control:

Aspect	Format	Definition & Important Notice
In Request Message	`yyyy-MM-dd'T'HH:mm:ssZ`	Time zone is expected to be `GMT+5.5` (India local time). Merchant is required to perform any necessary time zone conversion before submit request if needed.
In Response Message	`yyyy-MM-dd'T'HH:mm:ss±hh:mm`	Timezone returned in `api_gw` object is generated from HSBC API Gateway which located in Cloud and hence is calculated in `GMT+0`. On the other hand, time field in `response` object will be returned together with timezone information. For more details, please read each field definition carefully.

FAQ

SSL Connection Questions

Where can I find HSBC SSL server certificates?

Merchant developer is able to export SSL server certificates that has been installed in your browser. By doing this, visit the **domain** of the corresponding API endpoint in your browser. For example, to get the SSL certificate of sandbox environment, use domain name https://devclustercmb.api.p2g.netd2.hsbc.com.hk/

However, **in production**, we will provide a certificate and require TLS 1.2 implementation.

Message Encryption Questions

What certificates will I need to work for Message Encryption in HSBC's sandbox and production environments?

A self-sign certificate is acceptable. However, If Merchant decides to enhance security, a CA-Signed Certificate is always welcome.

Javascript Object Signing and Encryption (JOSE) Framework Questions

Where can I get more information about JOSE Framework?

If you want to fully understand the framework, you can read here for more details.

Please note the url does not belong to HSBC, use it on your own discretion. By clicking the url or website, it means you accept this terms and conditions.

Where can I download JOSE libraries for development?

For your reference, you may find the following JOSE libraries of different programming languages.

- Ruby
- Python
- PHP

- Java
- Node
- .NET

Please note those urls or websites do not belong to HSBC, use it on your own discretion. By clicking those urls or websites, it means you accept this terms and conditions.

Payments

Contains resource collections for payment page redirection, enquiry, cancel and callback notification.

Payments

Make Payment By UPI

POST `/payment/upi`

DESCRIPTION

Unlike making payment via an Online Payment Page, this API makes a direct UPI payment request.

REQUEST PARAMETERS

Authorization required in header	BASIC [Base64-encoded Credential]
x-hsbc-client-id	[Client ID]

required in header	
x-hsbc-client-secret required in header	[Client Secret]
x-hsbc-msg-encrypt-id optional in header	[Merchant ID]+[JWS ID]+[JWE ID]
Content-Type required in header	application/json

REQUEST BODY

upiReqtModel	*Data Encryption is enforced. API Schema intends to demonstrate the skeleton of the message payload only.*

Request Content-Types: application/json

Request Example

```
{
  "transaction": {
    "txnRef": "ORD-438UL748T6"
  },
  "system": {
    "notificationUrl": "https://www.example.com/notification"
  },
  "payment": {
    "country": "IN",
    "currency": "INR",
    "amount": 10200000,
    "expiry": "2020-01-01T13:02:00+05:30"
  },
  "merchant": {
    "merId": "C0Ds8q"
  },
  "customer": {
    "payer_vpa": "asdfgh@hsbc",
    "customer_firstname": "Ghanshyam",
    "customer_lastname": "Subramaniam"
  },
  "order": {
    "description": "Proceed check out for your order #ORD-438UL748T6",
    "descriptions": [
      {
        "product_name": "Product Item 1",
        "product_id": "PRO-ASDF-1234",
        "unitAmt": 10000,
        "unit": 2,
        "subAmt": 20000
      },
      {
        "product_name": "Product Item 2",
        "product_id": "PRO-JHGF-9876",
        "unitAmt": 50000,
        "unit": 3,
        "subAmt": 150000
      }
    ]
  },
  "other": {
    "udfs": [
      {
        "definition": "Product Image in Base64 format",
        "value": "iVBORw0KGgoAAAANSUhEU..."
      },
      {
        "definition": "Special Notes from Customer",
        "value": "Customer is a non-smoker"
      }
    ]
  }
}
```

RESPONSES

200 OK upiRespModel	Successful operation. *Data Encryption is enforced. API Schema intends to demonstrate the skeleton of the message payload only.*
400 Bad Request commonRespObj	Missing or invalid Parameters.
403 Forbidden	Authorization credentials are missing or invalid.
404 Not Found	Empty resource/resource not found.
500 Internal Server Error	The request failed due to an internal error.

Response Content-Types: application/json

Response Example (200 OK)

```
{
  "api_gw": {
    "messageId": "89817674-da00-4883",
    "returnCode": "200",
    "returnReason": "Successful operation",
    "sentTime": "2016-11-15T10:00:00.000Z",
    "responseTime": "2016-11-15T10:00:00.000Z"
  },
  "response": {
    "system": {
      "sysCode": "000000",
      "sysMsg": "Request Successful"
    },
    "transaction": {
      "txnRef": "PAY-QJZV956664",
      "txnStatus": "Initiated",
      "error_message": "Transaction Initiated"
    },
    "payment": {
      "amount": 10200000,
      "currency": "INR",
      "payment_datetime": "2020-01-01T13:02:00+05:30",
      "payment_option": "UPI"
    },
    "upi": {
      "payer_vpa": "asdfgh@hsbc",
      "payee_vpa": "merchantvpa"
    },
    "other": {
      "udfs": [
        {
          "definition": "Product Image in Base64 format",
          "value": "iVBORw0KGgoAAAANSUhEU..."
        },
        {
          "definition": "Special Notes from Customer",
          "value": "Customer is a non-smoker"
        }
      ]
    }
  }
}
```

Response Example (400 Bad Request)

```
{
  "messageId": "89817674-da00-4883",
  "returnCode": "400",
  "returnReason": "Error Message Here",
  "sentTime": "2016-11-15T10:00:00.000Z",
  "responseTime": "2016-11-15T10:00:00.000Z"
}
```

Payment Page Redirect API

POST `/payment/pageRedirect`

DESCRIPTION

This API returns the access of the Secured Online Payment Page. The access method can be either a `HTML Form Submit`, a `Javascript Event Method` or a `Direct URL Link` depending on which Payment Gateway the merchant subscribes.

Gateway 1

HTML Form Submit

API returns a `HTML FORM POST` with an access token in response field `redirectLink`. Below is a sample, please be noticed any data modification inside the form is not allowed. Otherwise, the data integrity checking will block the connection from accessing the online payment page.

```
<script language="javascript">window.onload=function(){document.pay_form.submit();}</script>
<form id="pay_form" name="pay_form" action="https://test.payu.in/_payment" method="post">
<input name="key" type="hidden" id="key" value="gheewEtg" />
<input name="amount" type="hidden" id="amount" value="1000.00" />
<input name="SALT" type="hidden" id="SALT" value="xxxxxxx" />
/* ...more input fields here... */
</form>
```

Gateway 2

Javascript Event Method

API returns a `Javascript Object` in response field `redirectLink`. Please follow the example to trigger this function. Again, any data modification of the Javascript object is not allowed.

1. Include the script into your HTML page
2. **Optional:** Invoke an event to trigger the caller function. In this example, the trigger point is to click an element whose ID is `pay`.

3. Parse the value of response field `redirectLink` into Javascript object
4. Include this line into your code
5. Include this line into your code
6. Optional code line

Payment Page URL Link

API returns a URL link in response field `redirectUrlLink` where merchant can use it for redirection.

```
#1 <script src="https://checkout.razorpay.com/v1/checkout.js"></script>
   <script>
#2 document.getElementById('pay').onclick = function (e) {
#3   var options = JSON.parse(/* Put the value of redirectLink here */);
#4   var rzp1 = new Razorpay(options);
#5   rzp1.open();
#6   e.preventDefault();
   }
   </script>
```

REQUEST PARAMETERS

Authorization required in header	BASIC [Base64-encoded Credential]
x-hsbc-client-id required in header	[Client ID]
x-hsbc-client-secret required in header	[Client Secret]

x-hsbc-msg-encrypt-id optional in header	[Merchant ID]+[JWS ID]+[JWE ID]
Content-Type required in header	application/json

REQUEST BODY

paymentReqtModel	*Data Encryption is enforced. API Schema intends to demonstrate the skeleton of the message payload only.*

Request Content-Types: application/json

Request Example

```
{
  "transaction": {
    "txnRef": "ORD-438UL748T6"
  },
  "system": {
    "redirectSuccessUrl": "https://www.example.com/successPayment",
    "redirectFailUrl": "https://www.example.com/failPayment",
    "redirectCancelUrl": "https://www.example.com/cancelPayment",
    "notificationUrl": "https://www.example.com/notification"
  },
  "payment": {
    "country": "IN",
    "currency": "INR",
    "amount": 10200000,
    "payment_option": "all",
    "expiry": "2020-01-01T13:02:00+05:30",
    "offers": [
      "offer_#111@222",
      "offer_#333@444"
    ]
  },
  "merchant": {
    "merId": "C0Ds8q"
  },
  "customer": {
    "customer_firstname": "Ghanshyam",
    "customer_lastname": "Subramaniam",
    "customer_email": "customer.name@example.com",
    "customer_phone": "9843176540"
  },
  "order": {
    "description": "Proceed check out for your order #ORD-438UL748T6",
    "descriptions": [
      {
        "product_name": "Product Item 1",
        "product_id": "PRO-ASDF-1234",
        "unitAmt": 10000,
        "unit": 2,
        "subAmt": 20000
      },
      {
        "product_name": "Product Item 2",
        "product_id": "PRO-JHGF-9876",
        "unitAmt": 50000,
        "unit": 3,
        "subAmt": 150000
      }
    ]
  },
  "other": {
    "udfs": [
      {
        "definition": "Product Image in Base64 format",
        "value": "iVBORw0KGgoAAAANSUhEU..."
      },
      {
        "definition": "Special Notes from Customer",
        "value": "Customer is a non-smoker"
      }
    ]
  }
}
```

RESPONSES

200 OK paymentRespModel	Successful operation. *Data Encryption is enforced. API Schema intends to demonstrate the skeleton of the message payload only.*
400 Bad Request commonRespObj	Missing or invalid Parameters.
403 Forbidden	Authorization credentials are missing or invalid.
404 Not Found	Empty resource/resource not found.
500 Internal Server Error	The request failed due to an internal error.

Response Content-Types: application/json

Response Example (200 OK)

```
{
  "api_gw": {
    "messageId": "89817674-da00-4883",
    "returnCode": "200",
    "returnReason": "Successful operation",
    "sentTime": "2016-11-15T10:00:00.000Z",
    "responseTime": "2016-11-15T10:00:00.000Z"
  },
  "response": {
    "transaction": {
      "txnRef": "ORD-438UL748T6"
    },
    "system": {
      "sysCode": "000000",
      "sysMsg": "Request Successful",
      "sysDatetime": "2020-01-01T13:00:00+05:30",
      "redirectLink": "<HTML Form or Javascript Code>",
      "redirectUrlLink": "https://rzp.io/xxxxxxx"
    }
  }
}
```

Response Example (400 Bad Request)

```
{
  "messageId": "89817674-da00-4883",
  "returnCode": "400",
  "returnReason": "Error Message Here",
  "sentTime": "2016-11-15T10:00:00.000Z",
  "responseTime": "2016-11-15T10:00:00.000Z"
}
```

Payments

Payment Status Enquiry API

POST `/payment/enquiry`

DESCRIPTION

Merchant can optionally initiate payment status enquiry at any time after a payment request is submitted. This is used when Merchant wants to check payment status any time after a payment request or find no acknowledge message returned after a certain period of time. HSBC Mobile Collection will return the latest transaction status according to the transaction reference number Merchant provides.

REQUEST PARAMETERS

Authorization required in header	BASIC [Base64-encoded Credential]
x-hsbc-client-id required in header	[Client ID]
x-hsbc-client-secret required in header	[Client Secret]
x-hsbc-msg-encrypt-id optional in header	[Merchant ID]+[JWS ID]+[JWE ID]
Content-Type required in header	application/json

REQUEST BODY

enquiryReqtModel	*Data Encryption is enforced. API Schema intends to demonstrate the skeleton of the message payload only.*

Request Content-Types: application/json

Request Example

```
{
  "transaction": {
    "txnRef": "ORD-438UL748T6"
  },
  "merchant": {
    "merId": "C0Ds8q"
  }
}
```

RESPONSES

200 OK enquiryRespModel	Successful operation. *Data Encryption is enforced. API Schema intends to demonstrate the skeleton of the message payload only.*
400 Bad Request commonRespObj	Missing or invalid Parameters.
403 Forbidden	Authorization credentials are missing or invalid.
404 Not Found	Empty resource/resource not found.
500 Internal Server Error	The request failed due to an internal error.

Response Content-Types: application/json

Response Example (200 OK)

```
{
  "api_gw": {
    "messageId": "89817674-da00-4883",
    "returnCode": "200",
      "returnReason": "Successful operation",
      "sentTime": "2016-11-15T10:00:00.000Z",
      "responseTime": "2016-11-15T10:00:00.000Z"
    },
    "response": {
      "system": {
        "sysCode": "000000",
        "sysMsg": "Request Successful"
      },
      "transaction": {
        "txnRef": "PAY-QJZV956664",
        "txnStatus": "captured",
        "error_code": "E000",
        "error_message": "NO ERROR"
      },
      "payment": {
        "amount": 10200000,
        "discount": 200000,
        "currency": "INR",
        "payment_datetime": "2020-01-01T13:02:00+05:30",
        "payment_option": "CC",
        "bank_ref_num": "3465241441650741",
        "offers": [
          "offer_#111@222",
          "offer_#333@444"
        ]
      },
      "credit_card": {
        "brand": "VISA",
        "mcn": "512345XXXXXX2346"
      },
      "upi": {
        "payer_vpa": "asdfgh@hsbc",
        "payee_vpa": "merchantvpa"
      },
      "other": {
        "udfs": [
          {
            "definition": "Product Image in Base64 format",
            "value": "iVBORw0KGgoAAAANSUhEU..."
          },
          {
            "definition": "Special Notes from Customer",
            "value": "Customer is a non-smoker"
          }
      ]
    },
    "refund": [
      {
        "rfdRef": "RFD-DFCV112233",
        "rfdRequestID": "124748448",
        "rfdStatus": "success",
        "rfdAmount": 1000,
        "rfdDatetime": "2020-01-02T13:00:00+05:30"
      },
      {
        "rfdRef": "RFD-KJDS775511",
        "rfdRequestID": "124749836",
        "rfdStatus": "failure",
        "rfdAmount": 15000,
        "rfdDatetime": "2020-01-03T13:00:00+05:30"
      }
    ]
  }
}
```

Response Example (400 Bad Request)

```
{
  "messageId": "89817674-da00-4883",
  "returnCode": "400",
  "returnReason": "Error Message Here",
  "sentTime": "2016-11-15T10:00:00.000Z",
  "responseTime": "2016-11-15T10:00:00.000Z"
}
```

Payments

Order Cancellation & Refund API

POST `/payment/cancel`

DESCRIPTION

This API can either cancel an unsettled order or send a refund request for a settled transaction. It supports both full and partial refund.

REQUEST PARAMETERS

Authorization required in header	BASIC [Base64-encoded Credential]
x-hsbc-client-id required in header	[Client ID]
x-hsbc-client-secret required in header	[Client Secret]
x-hsbc-msg-encrypt-id optional in header	[Merchant ID]+[JWS ID]+[JWE ID]
Content-Type required in header	application/json

REQUEST BODY

cancelReqtModel	*Data Encryption is enforced. API Schema intends to demonstrate the skeleton of the message payload only.*

Request Content-Types: application/json

Request Example

```
{
  "system": {
    "refundNotificationUrl": "https://www.example.com/refundNotification"
  },
  "transaction": {
    "txnRef": "ORD-438UL748T6",
    "rfdRef": "RFD-DFCV112233",
    "amount": 5000,
    "currency": "INR"
  },
  "merchant": {
    "merId": "C0Ds8q"
  }
}
```

RESPONSES

200 OK cancelRespModel	Successful operation. *Data Encryption is enforced. API Schema intends to demonstrate the skeleton of the message payload only.*
400 Bad Request commonRespObj	Missing or invalid Parameters.
403 Forbidden	Authorization credentials are missing or invalid.

404 Not Found	Empty resource/resource not found.
500 Internal Server Error	The request failed due to an internal error.

Response Content-Types: application/json

Response Example (200 OK)

```
{
  "api_gw": {
    "messageId": "89817674-daOO-4883",
    "returnCode": "200",
    "returnReason": "Successful operation",
    "sentTime": "2016-11-15T10:00:00.000Z",
    "responseTime": "2016-11-15T10:00:00.000Z"
  },
  "response": {
    "system": {
      "sysCode": "000000",
      "sysMsg": "Request Successful"
    },
    "transaction": {
      "txnRef": "ORD-438UL748T6",
      "rfdRef": "RFD-DFCV112233",
      "txnStatus": "success",
      "error_code": "102",
      "error_message": "NO ERROR - Refund Request Queued",
      "rfdRequestID": "124749836",
      "bank_ref_num": "3465241441650741"
    }
  }
}
```

Response Example (400 Bad Request)

```
{
  "messageId": "89817674-da00-4883",
  "returnCode": "400",
  "returnReason": "Error Message Here",
  "sentTime": "2016-11-15T10:00:00.000Z",
  "responseTime": "2016-11-15T10:00:00.000Z"
}
```

Payments

Callback Payment Notification API

POST `/<Callback URL predefined by Merchant>`

DESCRIPTION

Payment status will be returned to Merchant by asynchronous callback once Mobile Collection receives a payment request. After Mobile Collection payment platform completes reconciliation with bank and receives payment result, Mobile Collection will push the result back to Merchant by calling this API.

> ! **Implementation**
> This is a Callback API. HSBC will trigger this API call and defines the interface with OpenAPI standard. Merchant is required to provide implementation.

> ! **Retry Mechanism**
> If no success response is received, up to 3 retries will be triggered in every 3 - 5 minutes. Maximum 4 calls including the 1st attempt.

! **Endpoint Definition**
Field `notificationUrl` from Payment Page Redirect API will be used as URL endpoint of the corresponding transaction.

! **Exception Handling**
Only success case will be returned. Merchant can submit a Payment Status Enquiry API request if found no acknowledge message returned after a certain period of time.

REQUEST PARAMETERS

Content-Type: string required in header	text/plain

REQUEST BODY

callbackPaymentReqtModel	*Data Encryption is enforced. API Schema intends to demonstrate the skeleton of the message payload only.*

Request Content-Types: text/plain

Request Example

```
{
  "transaction": {
    "txnRef": "ORD-438UL748T6",
    "txnStatus": "captured",
    "error_code": "E000",
    "error_message": "NO ERROR"
  },
  "merchant": {
    "merId": "C0Ds8q"
  },
  "order": {
    "amount": 500000,
    "currency": "INR"
  },
  "payment": {
    "amount": 400000,
    "discount": 100000,
    "currency": "INR",
    "payment_datetime": "2020-01-01T13:02:00+05:30",
    "payment_option": "CC",
    "bank_ref_num": "3465241441650741",
    "offers": [
      "offer_#111@222",
      "offer_#333@444"
    ]
  },
  "credit_card": {
    "mcn": "512345XXXXXX2346"
  },
  "upi": {
    "payer_vpa": "asdfgh@hsbc",
    "payee_vpa": "merchantvpa"
  },
  "other": {
    "udfs": [
      {
        "definition": "Product Image in Base64 format",
        "value": "iVBORw0KGgoAAAANSUhEU..."
      },
      {
        "definition": "Special Notes from Customer",
        "value": "Customer is a non-smoker"
      }
    ]
  }
}
```

RESPONSES

200 OK
callbackPaymentRespModel

Successful operation.

Data Encryption is enforced. API Schema intends to demonstrate the skeleton of the message payload only.

Response Content-Types: application/json

Response Example (200 OK)

```
{
  "status": "SUCCESS"
}
```

Payments

Callback Refund Notification API

POST `/<Callback URL predefined by Merchant>`

DESCRIPTION

Refund status will be returned to Merchant by asynchronous callback once Mobile Collection receives a refund request. After Mobile Collection payment platform completes reconciliation with bank and receives refund result, Mobile Collection will push the result back to Merchant by calling this API.

> ! **Implementation**
> This is a Callback API. HSBC will trigger this API call and defines the interface with OpenAPI standard. Merchant is required to provide

implementation.

> ! **Retry Mechanism**
> If no success response is received, up to 3 retries will be triggered in every 3 - 5 minutes. Maximum 4 calls including the 1st attempt.

> ! **Endpoint Definition**
> Field `refundNotificationUrl` from Order Cancellation & Refund API will be used as URL endpoint of the corresponding transaction.

> ! **Exception Handling**
> Only success case will be returned. Merchant can submit a Payment Status Enquiry API request if found no acknowledge message returned after a certain period of time.

REQUEST PARAMETERS

Content-Type: string required in header	text/plain

REQUEST BODY

callbackRefundReqtModel	*Data Encryption is enforced. API Schema intends to demonstrate the skeleton of the message payload only.*

Request Content-Types: text/plain

Request Example

```
{
  "transaction": {
    "txnRef": "ORD-438UL748T6",
    "rfdRef": "RFD-DFCV112233",
    "txnStatus": "success"
  },
  "merchant": {
    "merId": "C0Ds8q"
  },
  "refund": {
    "amount": 300000,
    "currency": "INR",
    "bank_ref_num": "3780984556228904",
    "rfdRequestID": "124749836"
  }
}
```

RESPONSES

200 OK

callbackRefundRespModel

Successful operation.

Data Encryption is enforced. API Schema intends to demonstrate the skeleton of the message payload only.

Response Content-Types: application/json

Response Example (200 OK)

```
{
  "status": "SUCCESS"
}
```

Schema Definitions

commonRespObj: object

PROPERTIES

messageId: string range: (up to 36 chars) required

System generated unique message ID only for HSBC internal reference use

returnCode: string range: (up to 3 chars) required

System Return Code.

- This checking is on API Operational level, in other words, it checks upon Authorization, Connectivity and JSON Message Structure.

Possible Value	Definition
200	Successful operation
400	Bad Request (With detail message in field `returnReason`)
500	Internal Error. **Important Notices:** If any tier comes before the API Cloud Foundry is unavailable, such as the API Gateway, there will be no json respond message returned. Furthermore, the respond message of 500 will be ignored by some common HTTP libraries, in such case, the respond message body can be considered as a hint for troubleshooting during development and testing phase.

returnReason: string range: (up to 200 chars) required

Corresponding Text message of returnCode

Corr. Return Code	Return Message Sample	Definition
200	Successful operation	A successful API operation in terms of Authorization, Connectivity and valid JSON Message Structure. Any checking failure on Business Logic level will be still considered a successful API operation yet the Business Logic checking result will be returned in `response` object.
400	Client ID - Merchant ID mapping is not correct/updated!	The binding of Client ID, Merchant ID and Merchant Public Certificate is incorrect or not up-to-date.
400	object has missing required properties `field name`	Fail to pass JSON Field Mandatory Check.
400	instance type `data type` does not match any allowed primitive type	Fail to pass JSON Field Type Check.
400	string `field value` is too long	Fail to pass JSON Field Max Length Check
400	instance failed to match at least one required schema among `no. of conditional field`	Fail to pass JSON Conditional Field Check.
500	java.net.ConnectException: Connection refused: connect	**Notices:** Message can be varied depended on the downstream systems which return this message. Yet, all reasons can be concluded into Internal Error or System Unavailable.

sentTime: string range: (up to 27 chars) required

Time of request received by HSBC system from client, only for HSBC internal reference use

responseTime: string range: (up to 27 chars) required

Time of HSBC system provides response to client, only for HSBC internal reference use

Example

```
{
  "messageId": "89817674-da00-4883",
  "returnCode": "200",
  "returnReason": "Successful operation",
  "sentTime": "2016-11-15T10:00:00.000Z",
  "responseTime": "2016-11-15T10:00:00.000Z"
}
```

upiReqtModel: object

PROPERTIES

transaction: upi_rqt_txn_Obj required

system: upi_rqt_system_Obj required

payment: upi_rqt_payment_Obj required

merchant: upi_rqt_merchant_Obj required

customer: upi_rqt_customer_Obj required

order: upi_rqt_order_Obj required

other: upi_rqt_other_Obj optional

Example

```
{
  "transaction": {
    "txnRef": "ORD-438UL748T6"
  },
  "system": {
    "notificationUrl": "https://www.example.com/notification"
  },
  "payment": {
    "country": "IN",
    "currency": "INR",
    "amount": 10200000,
    "expiry": "2020-01-01T13:02:00+05:30"
  },
  "merchant": {
    "merId": "C0Ds8q"
  },
  "customer": {
    "payer_vpa": "asdfgh@hsbc",
    "customer_firstname": "Ghanshyam",
    "customer_lastname": "Subramaniam"
  },
  "order": {
    "description": "Proceed check out for your order #ORD-438UL748T6",
    "descriptions": [
      {
        "product_name": "Product Item 1",
        "product_id": "PRO-ASDF-1234",
        "unitAmt": 10000,
        "unit": 2,
        "subAmt": 20000
      },
      {
        "product_name": "Product Item 2",
        "product_id": "PRO-JHGF-9876",
        "unitAmt": 50000,
        "unit": 3,
        "subAmt": 150000
      }
    ]
  },
  "other": {
    "udfs": [
      {
        "definition": "Product Image in Base64 format",
        "value": "iVBORw0KGgoAAAANSUhEU..."
      },
      {
        "definition": "Special Notes from Customer",
        "value": "Customer is a non-smoker"
      }
    ]
  }
}
```

upi_rqt_txn_Obj: object

PROPERTIES

txnRef: string range: (up to 25 chars) required

Unique transaction ID/Reference code assigned by merchant

- No duplicate Transaction Reference is allowed

Example

```
{
  "txnRef": "ORD-438UL748T6"
}
```

upi_rqt_system_Obj: object

PROPERTIES

notificationUrl: string range: (up to 255 chars) required

Define URL endpoint for receiving payment result notification (server-to-server) from HSBC after payment completed

Example

```
{
  "notificationUrl": "https://www.example.com/notification"
}
```

upi_rqt_payment_Obj: object

PROPERTIES

country: string enum: [IN] range: (up to 2 chars) required

Country Code (Format: `ISO alpha-2`)

Possible Value	Definition

Possible Value	Definition
IN	India

currency: string enum: [INR] range: (up to 3 chars) required
Payment Currency (Format: `ISO 4217 Alpha`)

Possible Value	Definition
INR	Indian Rupee

amount: integer range: 1 ≤ x ≤ 9999999999999999 required
Payment Currency (Format: `ISO 4217 Alpha`)

Possible Value	Definition
INR	Indian Rupee

expiry: string range: (up to 25 chars) required
Before a defined date and time, a customer is able to confirm payment on their mobile app. The latest date and time you can define is 45 days (or 64800 minutes) right after the API submission.

- Local time. A `GMT+5:30` timezone information is appended to the end of the timestamp to indicate this time is a India local time. Format: `yyyy-MM-dd'T'HH:mm:ss±hh:mm`

Example

```
{
  "country": "IN",
  "currency": "INR",
  "amount": 10200000,
  "expiry": "2020-01-01T13:02:00+05:30"
}
```

upi_rqt_merchant_Obj: object

PROPERTIES

merId: string range: (up to 50 chars) required

Merchant ID

- Distributed by HSBC for identifying each merchant's identity

Example

```
  {
    "merId": "C0Ds8q"
  }
```

upi_rqt_customer_Obj: object

PROPERTIES

payer_vpa: string range: (up to 255 chars) required

Payer VPA

customer_firstname: string range: (up to 60 chars) optional

Customer's First Name

customer_lastname: string range: (up to 20 chars) optional

Customer's Last Name

Example

```
{
  "payer_vpa": "asdfgh@hsbc",
  "customer_firstname": "Ghanshyam",
  "customer_lastname": "Subramaniam"
}
```

upi_rqt_order_Obj: object

PROPERTIES

description: string range: (up to 100 chars) required

A brief Order Description that will be displayed in the Payment Page

descriptions: Array< descriptionsObj > range: (up to 20 objects) required

Array of Product Descriptions in the basket

Example

```
{
  "description": "Proceed check out for your order #ORD-438UL748T6",
  "descriptions": [
    {
      "product_name": "Product Item 1",
      "product_id": "PRO-ASDF-1234",
      "unitAmt": 10000,
      "unit": 2,
      "subAmt": 20000
    },
    {
      "product_name": "Product Item 2",
      "product_id": "PRO-JHGF-9876",
      "unitAmt": 50000,
      "unit": 3,
      "subAmt": 150000
    }
  ]
}
```

upi_rqt_other_Obj: object

PROPERTIES

udfs: Array< udfsObj > range: (up to 20 objects) optional

Array of User Defined Fields

Example

```
{
  "udfs": [
    {
      "definition": "Product Image in Base64 format",
      "value": "iVBORw0KGgoAAAANSUhEU..."
    },
    {
      "definition": "Special Notes from Customer",
      "value": "Customer is a non-smoker"
    }
  ]
}
```

upiRespModel: object

PROPERTIES

api_gw: commonRespObj required

response: object required

PROPERTIES

system: upi_rpn_sys_Obj required

transaction: upi_rpn_txn_Obj required

payment: upi_rpn_payment_Obj required

upi: upi_rpn_upi_Obj optional

other: upi_rpn_other_Obj optional

Example

```
{
  "api_gw": {
    "messageId": "89817674-da00-4883",
    "returnCode": "200",
    "returnReason": "RETURN_MESSAGE",
    "sentTime": "2016-11-15T10:00:00.000Z",
    "responseTime": "2016-11-15T10:00:00.000Z"
  },
  "response": {
    "system": {
      "sysCode": "000000",
      "sysMsg": "Request Successful"
    },
    "transaction": {
      "txnRef": "PAY-QJZV956664",
      "txnStatus": "Initiated",
      "error_message": "Transaction Initiated"
    },
    "payment": {
      "amount": 10200000,
      "currency": "INR",
      "payment_datetime": "2020-01-01T13:02:00+05:30",
      "payment_option": "UPI"
    },
    "upi": {
      "payer_vpa": "asdfgh@hsbc",
      "payee_vpa": "merchantvpa"
    },
    "other": {
      "udfs": [
        {
          "definition": "Product Image in Base64 format",
          "value": "iVBORw0KGgoAAAANSUhEU..."
        },
        {
          "definition": "Special Notes from Customer",
          "value": "Customer is a non-smoker"
        }
      ]
    }
  }
}
```

upi_rpn_sys_Obj: object

PROPERTIES

sysCode: string range: (up to 6 chars) required

System Return Code

Possible Value	Definition
000000	Request Successful
800030	Invalid VPA Status
800110	Invalid Calculation Found in Product Sub-Amount
800120	Invalid Calculation Found in Order Total Amount
900030	Duplicate Transaction Reference

Possible Value	Definition
999999	System Error

sysMsg: string range: (up to 128 chars) required

System Return Status. This is the corresponding message of System Return Code.

Example

```
{
  "sysCode": "000000",
  "sysMsg": "Request Successful"
}
```

upi_rpn_txn_Obj: object

PROPERTIES

txnRef: string range: (up to 25 chars) required

Returning Transaction Reference

txnStatus: string enum: [Initiated, Failed, Exception] range: (up to 100 chars) required

Transaction Status

error_message: string range: (up to 100 chars) required

Transaction Status Message

Example

```
{
  "txnRef": "PAY-QJZV956664",
  "txnStatus": "Initiated",
  "error_message": "Transaction Initiated"
}
```

upi_rpn_payment_Obj: object

PROPERTIES

amount: integer range: 1 ≤ x ≤ 999999999999999 required

Payment Amount

> ! NOTICE: NO comma or dot. For example: Input `10000` instead of `100.00`

currency: string range: (up to 3 chars) required

Return Payment Currency (Format: `ISO 4217 Alpha`)

payment_datetime: string range: (up to 25 chars) optional

Returning the transaction time of a successful payment

- Bank system local time. A `GMT+5:30` timezone information is appended to the end of the timestamp to indicate this time is a India local time. Format: `yyyy-MM-dd'T'HH:mm:ss±hh:mm`

payment_option: string range: (up to 25 chars) required

Returning Payment Option

Example

```
{
  "amount": 10200000,
  "currency": "INR",
  "payment_datetime": "2020-01-01T13:02:00+05:30",
  "payment_option": "UPI"
}
```

upi_rpn_upi_Obj: object

PROPERTIES

payer_vpa: string range: (up to 255 chars) required

Payer's VPA

payee_vpa: string range: (up to 255 chars) required

Payee's VPA

Example

```
{
  "payer_vpa": "asdfgh@hsbc",
  "payee_vpa": "merchantvpa"
}
```

upi_rpn_other_Obj: object

PROPERTIES

udfs: Array< udfsObj > range: (up to 20 objects) optional

Array of User Defined Fields

Example

```
{
  "udfs": [
    {
      "definition": "Product Image in Base64 format",
      "value": "iVBORw0KGgoAAAANSUhEU..."
    },
    {
      "definition": "Special Notes from Customer",
      "value": "Customer is a non-smoker"
    }
  ]
}
```

paymentReqtModel: object

PROPERTIES

transaction: pay_rqt_txn_Obj required

system: pay_rqt_system_Obj required

payment: pay_rqt_payment_Obj required

merchant: pay_rqt_merchant_Obj required

customer: pay_rqt_customer_Obj optional

order: pay_rqt_order_Obj required

other: pay_rqt_other_Obj optional

Example

```
{
  "transaction": {
    "txnRef": "ORD-438UL748T6"
  },
  "system": {
    "redirectSuccessUrl": "https://www.example.com/successPayment",
    "redirectFailUrl": "https://www.example.com/failPayment",
    "redirectCancelUrl": "https://www.example.com/cancelPayment",
    "notificationUrl": "https://www.example.com/notification"
  },
  "payment": {
    "country": "IN",
    "currency": "INR",
    "amount": 10200000,
      "payment_option": "all",
      "expiry": "2020-01-01T13:02:00+05:30",
      "offers": [
        "offer_#111@222",
        "offer_#333@444"
      ]
    },
    "merchant": {
      "merId": "C0Ds8q"
    },
    "customer": {
      "customer_firstname": "Ghanshyam",
      "customer_lastname": "Subramaniam",
      "customer_email": "customer.name@example.com",
      "customer_phone": "9843176540"
    },
    "order": {
      "description": "Proceed check out for your order #ORD-438UL748T6",
      "descriptions": [
        {
          "product_name": "Product Item 1",
          "product_id": "PRO-ASDF-1234",
          "unitAmt": 10000,
          "unit": 2,
          "subAmt": 20000
        },
        {
          "product_name": "Product Item 2",
          "product_id": "PRO-JHGF-9876",
          "unitAmt": 50000,
          "unit": 3,
          "subAmt": 150000
        }
      ]
    },
    "other": {
      "udfs": [
        {
          "definition": "Product Image in Base64 format",
          "value": "iVBORw0KGgoAAAANSUhEU..."
        },
        {
          "definition": "Special Notes from Customer",
          "value": "Customer is a non-smoker"
        }
    ]
  }
}
```

pay_rqt_txn_Obj: object

PROPERTIES

txnRef: string range: (up to 25 chars) required

Unique transaction ID/Reference code assigned by merchant

- No duplicate Transaction Reference is allowed

Example

```
{
  "txnRef": "ORD-438UL748T6"
}
```

pay_rqt_system_Obj: object

PROPERTIES

redirectSuccessUrl: string range: (up to 255 chars) required

Define URL endpoint for redirecting customer back from Payment Gateway to Merchant website after completing a successful payment

redirectFailUrl: string range: (up to 255 chars) required

Define URL endpoint for redirecting customer back from Payment Gateway to Merchant website after any fail scenario is taken place

redirectCancelUrl: string range: (up to 255 chars) required

Define URL endpoint for redirecting customer back from Payment Gateway to Merchant website after customer cancels the payment

notificationUrl: string range: (up to 255 chars) required

Define URL endpoint for receiving payment result notification (server-to-server) from HSBC after payment completed

Gateway 2

NOTICE:

Regarding to Payment Gateway Option 2, fields `redirectSuccessUrl` `redirectFailUrl` and `redirectCancelUrl` can only support Payment Link URL and the 3 values have to be the same since Payment Link can only redirect to one URL no matter the payment result is.

Example

```
{
  "redirectSuccessUrl": "https://www.example.com/successPayment",
  "redirectFailUrl": "https://www.example.com/failPayment",
  "redirectCancelUrl": "https://www.example.com/cancelPayment",
  "notificationUrl": "https://www.example.com/notification"
}
```

pay_rqt_payment_Obj: object

PROPERTIES

country: string enum: [IN] range: (up to 2 chars) required

Country Code (Format: `ISO alpha-2`)

Possible Value	Definition
IN	India

currency: string enum: [INR] range: (up to 3 chars) required

Payment Currency (Format: `ISO 4217 Alpha`)

Possible Value	Definition
INR	Indian Rupee

amount: integer range: 1 ≤ x ≤ 999999999999999 required

Payment Amount in 2 decimal places regardless whether the currency has decimal places or not

> ! NOTICE: Do not use comma or dot. For example: Input `10000` instead of `100.00`

payment_option: string range: (up to 64 chars) required

To restrict customer payment methods shown in the secured online Payment Page

Definition	Possible Value of Payment Gateway #1	Possible Value of Payment Gateway #2
All Payment Options	all	all

Definition	Possible Value of Payment Gateway #1	Possible Value of Payment Gateway #2
Credit Card	creditcard	card
Debit Card	debitcard	card
Net Banking	netbanking	netbanking
Equated Monthly Installment	emi	emi
Cash Card & eWallet	wallet	wallet
UPI & GPay	upi	upi

expiry: string range: (up to 25 chars) optional

Define the expiry datetime of response field `redirectUrlLink`

- Local time. A `GMT+5:30` timezone information is appended to the end of the timestamp to indicate this time is a India local time. Format: `yyyy-MM-dd'T'HH:mm:ss±hh:mm`

offers: string[] range: (up to 50 chars) optional

The offer key(s) that bind offer(s) created in Merchant Portal

> Gateway 1
>
> **NOTICE:**
> Only accept single key.

> Gateway 2
>
> **NOTICE:**
> Multiple keys are accepted.

Example

```
{
  "country": "IN",
  "currency": "INR",
  "amount": 10200000,
  "payment_option": "all",
  "expiry": "2020-01-01T13:02:00+05:30",
  "offers": [
    "offer_#111@222",
    "offer_#333@444"
  ]
}
```

pay_rqt_merchant_Obj: object

PROPERTIES

merId: string range: (up to 50 chars) required

Merchant ID

- Distributed by HSBC for identifying each merchant's identity

Example

```
{
  "merId": "C0Ds8q"
}
```

pay_rqt_customer_Obj: object

PROPERTIES

customer_firstname: string optional

Customer's First Name

> Gateway 1
>
> **NOTICE:**
>
> String range: (up to 60 chars)

> Gateway 2
>
> **NOTICE:**
>
> String range: (up to 20 chars)

customer_lastname: string range: (up to 20 chars) optional

Customer's Last Name

customer_email: string range: (up to 50 chars) optional

Customer's Email

customer_phone: string range: (up to 50 chars) optional

Customer's Phone

Example

```
{
  "customer_firstname": "Ghanshyam",
  "customer_lastname": "Subramaniam",
  "customer_email": "customer.name@example.com",
  "customer_phone": "9843176540"
}
```

pay_rqt_order_Obj: object

PROPERTIES

description: string range: (up to 100 chars) required

A brief Order Description that will be displayed in the Payment Page

descriptions: Array< descriptionsObj > range: (up to 20 objects) required

Array of Product Descriptions in the basket

Example

```
{
  "description": "Proceed check out for your order #ORD-438UL748T6",
  "descriptions": [
    {
      "product_name": "Product Item 1",
      "product_id": "PRO-ASDF-1234",
      "unitAmt": 10000,
      "unit": 2,
      "subAmt": 20000
    },
    {
      "product_name": "Product Item 2",
      "product_id": "PRO-JHGF-9876",
      "unitAmt": 50000,
      "unit": 3,
      "subAmt": 150000
    }
  ]
}
```

descriptionsObj: object

PROPERTIES

product_name: string range: (up to 200 chars) required

Product Item Name / Description

product_id: string range: (up to 50 chars) required

Product Numner / ID

unitAmt: integer range: 1 ≤ x ≤ 999999999999999 required

Payment Amount in 2 decimal places regardless whether the currency has decimal places or not

> ! NOTICE: Do not use comma or dot. For example: Input `10000` instead of `100.00`

unit: integer range: 1 ≤ x ≤ 99999999 required

No. of Unit

subAmt: integer range: 1 ≤ x ≤ 9999999999999999 required

Payment Amount in 2 decimal places regardless whether the currency has decimal places or not

> ! NOTICE: Do not use comma or dot. For example: Input `10000` instead of `100.00`

Example

```
{
  "product_name": "Product Item 1",
  "product_id": "PRO-ASDF-1234",
  "unitAmt": 1500000,
  "unit": 10,
  "subAmt": 20000
}
```

pay_rqt_other_Obj: object

PROPERTIES

udfs: Array< udfsObj > range: (up to 20 objects) optional

Array of User Defined Fields

Example

```
{
  "udfs": [
    {
      "definition": "Product Image in Base64 format",
      "value": "iVBORw0KGgoAAAANSUhEU..."
    },
    {
      "definition": "Special Notes from Customer",
      "value": "Customer is a non-smoker"
    }
  ]
}
```

udfsObj: object

PROPERTIES

definition: string range: (up to 1024 chars) optional

Merchant Defined Definition

value: string range: (up to 2048 chars) optional

Merchant Defined Value

> ! **NOTICE:** The sequence of this field inside the `udfs` array object you define in the request message of one particular transaction will be maintained the same as it is returned in the response message of other APIs.

Example

```
{
  "definition": "Special Notes from Customer",
  "value": "Customer is a non-smoker"
}
```

paymentRespModel: object

PROPERTIES

api_gw: commonRespObj required

response: object required

PROPERTIES

transaction: pay_rpn_txn_Obj required

system: pay_rpn_system_Obj required

Example

```
{
  "api_gw": {
    "messageId": "89817674-da00-4883",
    "returnCode": "200",
    "returnReason": "Successful operation",
    "sentTime": "2016-11-15T10:00:00.000Z",
    "responseTime": "2016-11-15T10:00:00.000Z"
  },
  "response": {
    "transaction": {
      "txnRef": "ORD-438UL748T6"
    },
    "system": {
      "sysCode": "000000",
      "sysMsg": "Request Successful",
      "sysDatetime": "2020-01-01T13:00:00+05:30",
      "redirectLink": "<HTML Form or Javascript Code>",
      "redirectUrlLink": "https://rzp.io/xxxxxxx"
    }
  }
}
```

pay_rpn_txn_Obj: object

PROPERTIES

txnRef: string range: (up to 64 chars) required

Returning back Transaction Reference

Example

```
{
  "txnRef": "ORD-438UL748T6"
}
```

pay_rpn_system_Obj: object

PROPERTIES

sysCode: string range: (up to 6 chars) required

System Return Code

Possible Value	Definition
000000	Request Successful
800110	Invalid Calculation Found in Product Sub-Amount
800120	Invalid Calculation Found in Order Total Amount
900030	Duplicate Transaction Reference

sysMsg: string range: (up to 128 chars) required

Corresponding Text Message of System Return Code

sysDatetime: string range: (up to 25 chars) optional

Time of sending out this request / response

- Server system time. A `GMT+5:30` timezone information is appended to the end of the timestamp to indicate this time is a Inida local time. Format: `yyyy-MM-dd'T'HH:mm:ss±hh:mm`

redirectLink: string range: (up to 5120 chars) optional

Gateway 1

INFORMATION:

If Payment Gateway option 1 is chosen, this field will return a HTML submit form.

Gateway 2

INFORMATION:

If Payment Gateway option 2 is chosen, this field will return a Javascript code.

redirectURLLink: string range: (up to 1024 chars) optional

Gateway 2

INFORMATION:

Return Payment URL link, only available for Payment Gateway option 2.

Example

```
{
  "sysCode": "000000",
  "sysMsg": "Request Successful",
  "sysDatetime": "2020-01-01T13:00:00+05:30",
  "redirectLink": "<HTML Form or Javascript Code>",
  "redirectUrlLink": "https://rzp.io/xxxxxxx"
}
```

enquiryReqtModel: object

PROPERTIES

transaction: enq_rqt_txn_Obj required

merchant: enq_rqt_merchant_Obj required

Example

```
{
  "transaction": {
    "txnRef": "ORD-438UL748T6"
  },
  "merchant": {
    "merId": "C0Ds8q"
  }
}
```

enq_rqt_txn_Obj: object

PROPERTIES

txnRef: string range: (up to 25 chars) required

Merchant to pass Transaction Reference that refers to one specific transaction

Example

```
{
  "txnRef": "ORD-438UL748T6"
}
```

enq_rqt_merchant_Obj: object

PROPERTIES

merId: string range: (up to 50 chars) required

Merchant ID

- Distributed by HSBC to merchant for identifying each merchant's identity

Example

```
{
  "merId": "C0Ds8q"
}
```

enquiryRespModel: object

PROPERTIES

api_gw: commonRespObj required

response: object required

PROPERTIES

system: enq_rpn_sys_Obj required

transaction: enq_rpn_txn_Obj required

payment: enq_rpn_payment_Obj required

credit_card: enq_rpn_creditcard_Obj optional

upi: enq_rpn_upi_Obj optional

other: enq_rpn_other_Obj optional

refund: Array< enq_rpn_refund_Obj > optional

Returned only if any prior refund request has been made to the transaction

Example

```
{
  "api_gw": {
    "messageId": "89817674-da00-4883",
    "returnCode": "200",
    "returnReason": "Successful operation",
    "sentTime": "2016-11-15T10:00:00.000Z",
    "responseTime": "2016-11-15T10:00:00.000Z"
  },
  "response": {
    "system": {
      "sysCode": "000000",
      "sysMsg": "Request Successful"
    },
    "transaction": {
      "txnRef": "PAY-QJZV956664",
      "txnStatus": "captured",
      "error_code": "E000",
      "error_message": "NO ERROR"
    },
    "payment": {
      "amount": 10200000,
      "discount": 200000,
      "currency": "INR",
      "payment_datetime": "2020-01-01T13:02:00+05:30",
      "payment_option": "CC",
      "bank_ref_num": "3465241441650741",
      "offers": [
        "offer_#111@222",
        "offer_#333@444"
      ]
    },
    "credit_card": {
      "brand": "VISA",
      "mcn": "512345XXXXXX2346"
    },
    "upi": {
      "payer_vpa": "asdfgh@hsbc",
      "payee_vpa": "merchantvpa"
    },
    "other": {
      "udfs": [
        {
          "definition": "Product Image in Base64 format",
          "value": "iVBORw0KGgoAAAANSUhEU..."
        },
        {
          "definition": "Special Notes from Customer",
          "value": "Customer is a non-smoker"
        }
      ]
    },
    "refund": [
      {
        "rfdRef": "RFD-DFCV112233",
        "rfdRequestID": "124748448",
        "rfdStatus": "success",
        "rfdAmount": 1000,
        "rfdDatetime": "2020-01-02T13:00:00+05:30"
      },
      {
        "rfdRef": "RFD-KJDS775511",
        "rfdRequestID": "124749836",
        "rfdStatus": "failure",
        "rfdAmount": 15000,
        "rfdDatetime": "2020-01-03T13:00:00+05:30"
      }
    ]
  }
}
```

enq_rpn_sys_Obj: object

PROPERTIES

sysCode: string range: (up to 6 chars) required

System Return Code

Possible Value	Definition
000000	Request Successful
100010	Transaction is Pending
900010	Transaction Record Not Found
999999	System Error

sysMsg: string range: (up to 128 chars) required

System Return Status. This is the corresponding message of System Return Code.

Example

```
{
  "sysCode": "000000",
  "sysMsg": "Request Successful"
}
```

enq_rpn_txn_Obj: object

PROPERTIES

txnRef: string range: (up to 25 chars) required

Returning Transaction Reference

txnStatus: string range: (up to 100 chars) required

Transaction Status

Definition	Possible Value of Payment Gateway #1	Possible Value of Payment Gateway #2	Payment by HSBC UPI
Transaction is successful	captured	captured	captured

Definition	Possible Value of Payment Gateway #1	Possible Value of Payment Gateway #2	Payment by HSBC UPI
Transaction is failed	failed	failed	failed
Transaction is pending	pending	pending	pending
Cancelled by User	userCancelled	n/a	n/a
Transaction is fully refunded	refunded	refunded	refunded

error_code: string range: (up to 50 chars) required

Transaction Error Code

error_message: string range: (up to 100 chars) required

Transaction Error Message

Example

```
{
  "txnRef": "PAY-QJZV956664",
  "txnStatus": "captured",
  "error_code": "E000",
  "error_message": "NO ERROR"
}
```

enq_rpn_payment_Obj: object

PROPERTIES

amount: integer range: 1 ≤ x ≤ 999999999999999 required

Payment Amount

> ! NOTICE: NO comma or dot. For example: Input `10000` instead of `100.00`

discount: integer range: 1 ≤ x ≤ 999999999999999 optional

Amount of Discount. Returned only if an offer is applied.

> ! NOTICE: NO comma or dot. For example: Input `10000` instead of `100.00`

currency: string range: (up to 3 chars) required

Return Payment Currency (Format: `ISO 4217 Alpha`)

payment_datetime: string range: (up to 25 chars) required

Returning Transaction time for the inward credit payment

- Bank system local time. A `GMT+5:30` timezone information is appended to the end of the timestamp to indicate this time is a India local time. Format: `yyyy-MM-dd'T'HH:mm:ss±hh:mm`

payment_option: string range: (up to 25 chars) required

Returning Payment Option

Definition	Possible Value of Payment Gateway #1	Possible Value of Payment Gateway #2
Credit Card	CC	card
Debit Card	DC	card

Definition	Possible Value of Payment Gateway #1	Possible Value of Payment Gateway #2
Net Banking	NB	netbanking
Equated Monthly Installment	EMI	emi
Cash Card & eWallet	CASH	wallet
UPI & GPay	UPI	upi

bank_ref_num: string range: (up to 25 chars) optional

Returning Bank Reference ID. Only for successful transaction

offers: string[] range: (up to 50 chars) optional

Returning Offer Key(s) applied if any

Example

```
{
  "amount": 10200000,
  "discount": 200000,
  "currency": "INR",
  "payment_datetime": "2020-01-01T13:02:00+05:30",
  "payment_option": "CC",
  "bank_ref_num": "3465241441650741",
  "offers": [
    "offer_#111@222",
    "offer_#333@444"
  ]
}
```

enq_rpn_creditcard_Obj: object

PROPERTIES

brand: string range: (up to 20 chars) optional

Brand Name

mcn: string range: (up to 16 chars) optional

Masked Credit Card Number

- First 6 and last 4 digits of credit card number

Example

```
{
  "brand": "VISA",
  "mcn": "512345XXXXXX2346"
}
```

enq_rpn_upi_Obj: object

PROPERTIES

payer_vpa: string range: (up to 255 chars) required

Payer's VPA

payee_vpa: string range: (up to 255 chars) required

Payee's VPA

Example

```
{
  "payer_vpa": "asdfgh@hsbc",
  "payee_vpa": "merchantvpa"
}
```

enq_rpn_other_Obj: object

PROPERTIES

udfs: Array< udfsObj > range: (up to 20 objects) optional

Array of User Defined Fields

Example

```
{
  "udfs": [
    {
      "definition": "Product Image in Base64 format",
      "value": "iVBORw0KGgoAAAANSUhEU..."
```

```
       },
       {
         "definition": "Special Notes from Customer",
         "value": "Customer is a non-smoker"
       }
     ]
   }
```

enq_rpn_refund_Obj: object

PROPERTIES

rfdRef: string range: (up to 25 chars) required

Unique Refund reference number defined by Merchant

rfdRequestID: string range: (up to 100 chars) required

Returning Refund Request ID

rfdStatus: string range: (up to 100 chars) required

Refund status of the refund transaction

Definition	Possible Value of Payment Gateway #1	Possible Value of Payment Gateway #2	Payment by HSBC UPI
Refund is successfully processed	success	success	success
Refund is pending	pending	pending	n/a
Refund fails	failure	n/a	failure

rfdAmount: integer range: 1 ≤ x ≤ 999999999999999 required

Returning Refund Amount

> ! NOTICE: NO comma or dot. For example: Input `10000` instead of `100.00`

rfdDatetime: string range: (up to 25 chars) required

Time of sending out this request

- Server system time. A `GMT+5:30` timezone information is appended to the end of the timestamp to indicate this time is a India local time. Format: `yyyy-MM-dd'T'HH:mm:ss±hh:mm`

Example

```
{
  "rfdRef": "RFD-PAY-QJZV956664",
  "rfdRequestID": "124748442",
  "rfdStatus": "success",
  "rfdAmount": 5000,
  "rfdDatetime": "2018-12-12T14:10:25+05:30"
}
```

cancelReqtModel: object

PROPERTIES

system: cancel_rqt_sys_Obj required

transaction: cancel_rqt_txn_Obj required

merchant: cancel_rqt_merchant_Obj required

Example

```
{
  "system": {
    "refundNotificationUrl": "https://www.example.com/refundNotification"
  },
  "transaction": {
    "txnRef": "ORD-438UL748T6",
    "rfdRef": "RFD-DFCV112233",
    "amount": 5000,
    "currency": "INR"
  },
  "merchant": {
    "merId": "C0Ds8q"
  }
}
```

cancel_rqt_sys_Obj: object

PROPERTIES

refundNotificationUrl: string range: (up to 255 chars)

Define URL endpoint for receiving refund result notification (server-to-server) from HSBC after refund completed

Example

```
{
  "refundNotificationUrl": "https://www.example.com/refundNotification"
}
```

cancel_rqt_txn_Obj: object

PROPERTIES

txnRef: string range: (up to 25 chars) required

Merchant to pass Transaction Reference that refers to one specific transaction

rfdRef: string range: (up to 25 chars) required

Unique Refund transaction ID assigned by merchant for this refund request

- No duplicate Refund Reference is allowed

amount: integer range: 1 ≤ x ≤ 999999999999999 required

Refund Amount or the Full Amount of a pre-auth transaction

- Refund Amount should not exceed the value of total transaction amount
- Support multiple partial refund
- If the transaction is in pre-auth state currently, then only a full cancellation is allowed. The amount must be same as the auth amount. Partial amount would not be allowed.

> ! NOTICE: NO comma or dot. For example: Input `10000` instead of `100.00`

currency: string enum: [INR] range: (up to 3 chars) required

Payment Currency (Format: `ISO 4217 Alpha`)

Possible Value	Definition
INR	Indian Rupee

Example

```
{
  "txnRef": "ORD-438UL748T6",
  "rfdRef": "RFD-DFCV112233",
  "amount": 5000,
  "currency": "INR"
}
```

cancel_rqt_merchant_Obj: object

PROPERTIES

merId: string range: (up to 50 chars) required

Merchant ID

Example

```
{
  "merId": "C0Ds8q"
}
```

cancelRespModel: object

PROPERTIES

api_gw: commonRespObj required

response: object required

PROPERTIES

system: cancel_rpn_sys_Obj required

transaction: cancel_rpn_txn_Obj required

Example

```
{
  "api_gw": {
    "messageId": "89817674-da00-4883",
    "returnCode": "200",
    "returnReason": "Successful operation",
    "sentTime": "2016-11-15T10:00:00.000Z",
    "responseTime": "2016-11-15T10:00:00.000Z"
  },
  "response": {
    "system": {
      "sysCode": "000000",
      "sysMsg": "Request Successful"
    },
    "transaction": {
      "txnRef": "ORD-438UL748T6",
      "rfdRef": "RFD-DFCV112233",
      "txnStatus": "success",
      "error_code": "102",
      "error_message": "NO ERROR - Refund Request Queued",
      "rfdRequestID": "124749836",
      "bank_ref_num": "3465241441650741"
    }
  }
}
```

cancel_rpn_sys_Obj: object

PROPERTIES

sysCode: string range: (up to 6 chars) required

System Return Code

Possible Value	Definition

Possible Value	Definition
000000	Request Successful
900010	Transaction Record Not Found
900030	Duplicate Refund Transaction Reference
999999	System Error

sysMsg: string range: (up to 128 chars) required

System Return Status

Example

```
{
  "sysCode": "000000",
  "sysMsg": "Request Successful"
}
```

cancel_rpn_txn_Obj: object

PROPERTIES

txnRef: string range: (up to 25 chars) required

Return Transaction Reference

rfdRef: string range: (up to 25 chars) required

Return Refund Transaction Reference

txnStatus: string range: (up to 100 chars) required

Return Status

Definition	Possible Value of Payment Gateway #1	Possible Value of Payment Gateway #2	Payment by HSBC UPI
Refund is successfully processed	success	success	success
Refund is pending	pending	pending	n/a
Refund fails	failure	n/a	failure

error_code: string range: (up to 50 chars) required

Transaction Error Code

error_message: string range: (up to 100 chars) required

Transaction Error Message

rfdRequestID: string range: (up to 25 chars) optional

Return Request ID

bank_ref_num: string range: (up to 25 chars) optional

Returning Bank Reference ID. Only for successful transaction

Example

```
{
  "txnRef": "ORD-438UL748T6",
  "rfdRef": "RFD-DFCV112233",
  "txnStatus": "success",
  "error_code": "102",
  "error_message": "NO ERROR - Refund Request Queued",
  "rfdRequestID": "124749836",
  "bank_ref_num": "3465241441650741"
}
```

callbackPaymentReqtModel: object

PROPERTIES

transaction: notif_rqt_txn_Obj required

merchant: notif_rqt_merchant_Obj required

order: notif_rqt_order_Obj required

payment: notif_rqt_payment_Obj required

credit_card: notif_rqt_cc_Obj optional

upi: notif_rqt_upi_Obj optional

other: notif_rqt_other_Obj optional

Example

```
{
  "transaction": {
    "txnRef": "ORD-438UL748T6",
    "txnStatus": "captured",
    "error_code": "E000",
    "error_message": "NO ERROR"
  },
  "merchant": {
    "merId": "C0Ds8q"
  },
  "order": {
    "amount": 500000,
    "currency": "INR"
  },
  "payment": {
    "amount": 400000,
    "discount": 100000,
    "currency": "INR",
    "payment_datetime": "2020-01-01T13:02:00+05:30",
    "payment_option": "CC",
    "bank_ref_num": "3465241441650741",
    "offers": [
      "offer_#111@222",
      "offer_#333@444"
    ]
  },
  "credit_card": {
    "mcn": "512345XXXXXX2346"
  },
  "upi": {
    "payer_vpa": "asdfgh@hsbc",
    "payee_vpa": "merchantvpa"
  },
  "other": {
    "udfs": [
      {
        "definition": "Product Image in Base64 format",
        "value": "iVBORw0KGgoAAAANSUhEU..."
      },
      {
        "definition": "Special Notes from Customer",
        "value": "Customer is a non-smoker"
      }
    ]
  }
}
```

notif_rqt_txn_Obj: object

PROPERTIES

txnRef: string range: (up to 25 chars) required

Returning Transaction Reference

txnStatus: string range: (up to 100 chars) required

Returning Transaction Status

Definition	Possible Value of Payment Gateway #1	Possible Value of Payment Gateway #2	Payment by HSBC UPI
Transaction is successful	captured	captured	captured
Transaction is failed	failed	failed	failed
Cancelled by User	userCancelled	n/a	n/a

error_code: string range: (up to 50 chars) optional

Transaction Error Code

error_message: string range: (up to 100 chars) optional

Transaction Error Message

Example

```
{
  "txnRef": "ORD-438UL748T6",
  "txnStatus": "captured",
  "error_code": "E000",
  "error_message": "NO ERROR"
}
```

notif_rqt_merchant_Obj: object

PROPERTIES

merId: string range: (up to 50 chars) required

Returning Merchant ID

Example

```
  {
    "merId": "C0Ds8q"
  }
```

notif_rqt_order_Obj: object

PROPERTIES

amount: integer range: 1 ≤ x ≤ 999999999999999 required

Returning Order Amount

currency: string range: (up to 3 chars) required

Order Currency (Format: `ISO 4217 Alpha`)

Example

```
{
  "amount": 500000,
  "currency": "INR"
}
```

notif_rqt_payment_Obj: object

PROPERTIES

amount: integer range: 1 ≤ x ≤ 999999999999999 required

Returning Payment Amount

discount: integer range: 1 ≤ x ≤ 999999999999999 optional

Returning Discount Amount

currency: string enum: [INR] range: (up to 3 chars) required

Payment Currency (Format: `ISO 4217 Alpha`)

Possible Value	Definition
INR	Indian Rupee

payment_datetime: string range: (up to 25 chars) required

Returning Transaction time for the inward credit payment

- Bank system local time. A `GMT+5:30` timezone information is appended to the end of the timestamp to indicate this time is a India local time. Format: `yyyy-MM-dd'T'HH:mm:ss±hh:mm`

payment_option: string range: (up to 10 chars) required

Returning Payment Option

Definition	Possible Value of Payment Gateway #1	Possible Value of Payment Gateway #2
Credit Card	CC	card
Debit Card	DC	card
Net Banking	NB	netbanking
Equated Monthly Installment	EMI	emi
Cash Card & eWallet	CASH	wallet
UPI & GPay	UPI	upi

bank_ref_num: string range: (up to 25 chars) optional

Returning Bank Reference ID. Only for successful transaction

offers: string[] range: (up to 50 chars) optional

Returning offer key(s) applied if any

Example

```
{
  "amount": 400000,
  "discount": 100000,
  "currency": "INR",
  "payment_datetime": "2020-01-01T13:02:00+05:30",
  "payment_option": "CC",
  "bank_ref_num": "3465241441650741",
  "offers": [
    "offer_#111@222",
    "offer_#333@444"
  ]
}
```

notif_rqt_cc_Obj: object

PROPERTIES

mcn: string range: (up to 16 chars) required

Masked Credit Card Number

- First 6 and last 4 digits of credit card number

Example

```
{
  "mcn": "512345XXXXXX2346"
}
```

notif_rqt_upi_Obj: object

PROPERTIES

payer_vpa: string range: (up to 255 chars) required

Payer's VPA

payee_vpa: string range: (up to 255 chars) required

Payee's VPA

Example

```
{
  "payer_vpa": "asdfgh@hsbc",
  "payee_vpa": "merchantvpa"
}
```

notif_rqt_other_Obj: object

PROPERTIES

udfs: Array< udfsObj > range: (up to 20 objects) optional
Array of User Defined Fields

Example

```
{
  "udfs": [
    {
      "definition": "Product Image in Base64 format",
      "value": "iVBORw0KGgoAAAANSUhEU..."
    },
    {
      "definition": "Special Notes from Customer",
      "value": "Customer is a non-smoker"
    }
  ]
}
```

callbackPaymentRespModel: object

PROPERTIES

status: string range: (up to 30 chars) required

Return Message

Example

```
{
  "status": "SUCCESS"
}
```

callbackRefundReqtModel: object

PROPERTIES

transaction: rfd_notif_rqt_txn_Obj required

merchant: rfd_notif_rqt_merchant_Obj required

refund: rfd_notif_rqt_refund_Obj required

Example

```
{
  "transaction": {
    "txnRef": "ORD-438UL748T6",
    "rfdRef": "RFD-DFCV112233",
    "txnStatus": "success"
  },
  "merchant": {
    "merId": "C0Ds8q"
  },
  "refund": {
    "amount": 300000,
    "currency": "INR",
    "bank_ref_num": "3780984556228904",
    "rfdRequestID": "124749836"
  }
}
```

rfd_notif_rqt_txn_Obj: object

PROPERTIES

txnRef: string range: (up to 25 chars) required

Returning Transaction Reference

rfdRef: string range: (up to 25 chars) required

Return Refund Transaction Reference

txnStatus: string range: (up to 100 chars) required

Returning Transaction Status

Definition	Possible Value of Payment Gateway #1	Possible Value of Payment Gateway #2	Payment by HSBC UPI
Refund is successfully processed	success	success	success
Refund is pending	pending	pending	n/a

Definition	Possible Value of Payment Gateway #1	Possible Value of Payment Gateway #2	Payment by HSBC UPI
Refund fails	failure	n/a	failure

Example

```
{
  "txnRef": "ORD-438UL748T6",
  "rfdRef": "RFD-DFCV112233",
  "txnStatus": "success"
}
```

rfd_notif_rqt_merchant_Obj: object

PROPERTIES

merId: string range: (up to 50 chars) required

Returning Merchant ID

Example

```
{
  "merId": "C0Ds8q"
}
```

rfd_notif_rqt_refund_Obj: object

PROPERTIES

amount: integer range: 1 ≤ x ≤ 999999999999999 required

Returning Refund Amount

currency: string enum: [INR] range: (up to 3 chars) required

Payment Currency (Format: `ISO 4217 Alpha`)

Possible Value	Definition
INR	Indian Rupee

bank_ref_num: string range: (up to 25 chars) required

Returning Bank Reference ID. Only for successful transaction

rfdRequestID: string range: (up to 25 chars) required

Return Request ID

Example

```
{
  "amount": 300000,
  "currency": "INR",
  "bank_ref_num": "3780984556228904",
  "rfdRequestID": "124749836"
}
```

callbackRefundRespModel: object

PROPERTIES

status: string range: (up to 30 chars) required

Return Message

Example

```
{
  "status": "SUCCESS"
}
```

Lifecycle of Cryptographic Keys

This section highlights the Lifecycle of cryptographic keys in the following steps:

1. Generate keys pair (Private Key and Public Key Certificate)
2. ***Optional:*** *Export CSR (Certificate Signing Request) and get signed with CA (Certificate Authority)*

> ! **DO YOU KNOW?**
> In public key infrastructure (PKI) systems, a certificate signing request is a message sent from an applicant to a certificate authority in order to apply for a digital identity certificate. It usually contains the public key for which the certificate should be issued.

3. Exchange Certificate with HSBC
4. Key Maintenance
5. Key Renewal Process

Command line tool **Java Keytool™** is used in the demonstration. The tool can generate public key / private key pairs and store them into a Java KeyStore. The Keytool executable is distributed with the **Java SDK (or JRE)™**, so if you have an SDK installed you will also have the Keytool executable. Yet, Merchant is free to choose any other tool to generate and manage keys, such as **OpenSSL™**.

Key Generation and Certificate Exchange with HSBC

1. Create a new keys pair (Private Key and Public Key Certificate) with a new or existing Keystore.

```
keytool -genkey
    -alias merchant_key_pair
    -keyalg RSA
    -keystore merchant_keystore.jks
    -keysize 2048
    -validity 3650
    -storepass <your keystore password>
```

- **-genkey** - command to generate keys pair.
- **-alias** - define the alias name (or unique identifier) of the keys pair stored inside the keystore.
- **-keyalg** - key algorithm, it must be `RSA` regarding to HSBC standard. If `RSA` is taken, the default hashing algorithm will be `SHA-256`.
- **-keystore** - file name of the keystore. If the file already exists in your system location, the key will be created inside your existing keystore, otherwise, a new keystore with the defined name will be created.

> ! **DO YOU KNOW?**
> Keystore is a password-protected repository of keys and certificates. File with extension `jks` means it is a Java Keystore which is originally supported and executable with Java™.
>
> There are several keystore formats in the industry like `PKCS12` with file extension `p12` which is executable with Microsoft Windows™, merchant can always pick the one most fit their application.

- **-keysize** - key size, it must be `2048` regarding to HSBC standard.
- **-validity** - the validity period of the private key and its associated certificate. The unit is `day`, 3650 means 10 years.
- **-storepass** - password of the keystore.

1.1. Provide `Distinguished Name` information after running the command:

```
Information required for CSR generation
--------------------------------------------------------------
What is your first and last name?
  [Unknown]:  MERCHANT INFO
What is the name of your organizational unit?
  [Unknown]:  MERCHANT INFO
What is the name of your organization?
  [Unknown]:  MERCHANT INFO
What is the name of your City or Locality?
  [Unknown]:  HK
What is the name of your State or Province?
  [Unknown]:  HK
What is the two-letter country code for this unit?
  [Unknown]:  HK
Is CN=XXX, OU=XXX, O=XXX, L=HK, ST=HK, C=HK correct? (type "yes" or "no")
  [no]:  yes
```

```
Enter key password for <merchant_key_pair>
        (RETURN if same as keystore password):
Re-enter new password:
```

> ! **NOTICE:** Private Key password and Keystore password can be the same or Merchant can set them differently to be more secure.

2. **Optional:** Export CSR and get signed with CA. This step can be skipped if Merchant decides to work with a Self-Signed Certificate.

```
keytool -certreq
    -alias merchant_key_pair
    -keyalg RSA
    -file merchant_csr.csr
    -keystore merchant_keystore.jks
```

- **-certreq** - command to generate and export CSR.
- **-alias** - the name of the associated keys pair.
- **-keyalg** - key algorithm, it must be `RSA` regarding to HSBC standard.
- **-file** - file name of the CSR. This will be generated at the location where the command is run.
- **-keystore** - specify the keystore which you are working on.

2.1. Select and purchase a plan at Certificate Authority and then submit the CSR accordingly. After a signed Certificate is issued by CA, import the Certificate back to Merchant's keystore.

```
keytool -import
    -alias merchant_signed_cert_0001
    -trustcacerts -file CA_signed_cert.p7b
    -keystore merchant_keystore.jks
```

- **-import** - command to import object into a specific keystore.
- **-alias** - define the alias name (or unique identifier) of the signed Certificate.

- **-trustcacerts -file** - specify the file name of the signed Certificate in Merchant's local file system.

> ! **NOTICE:** `PKCS#7` is one of the common formats that contains certificates and has a file extension of `.p7b` or `.p7c`. The certificate format may be varied depending on the policy of the issuing CA.

- **-keystore** - specify the keystore which you are working on.

3. Export Certificate and send to HSBC for key exchange.

> ! **DO YOU KNOW:**
> A Certificate or Public Key Certificate is an electronic document that contains a public key and additional information that prove the ownership and maintain integrity of the public key. This is essential for the sender to ensure the key is not altered by any chance during delivery.

```
keytool -export
    -alias merchant_key_pair
    -file merchant_cert_0001.cer
    -keystore merchant_keystore.jks
```

- **-export** - command to export object from a specific keystore.
- **-alias** - the name of the associated keys pair.

> ! **NOTICE:** If Merchant associates the original keys pair `merchant_key_pair`, the exported Certificate is without CA-signed, and hence, Self-Signed. However, if Merchant associates the imported Certificate `merchant_signed_cert_0001` mentioned in step #2, the exported Certificate is CA-signed.

- **-file** - specify the file name of the Certificate where the file will be exported to Merchant's local file system.

> ! **NOTICE:** The default Certificate file encoding is binary. HSBC accepts both binary and base64 encoding. To export a printable base64 encoding file, please attach an extra parameter `-rfc` in the command.
> e.g. `-file merchant_cert_0001.crt -rfc`.

- **-keystore** - specify the keystore which you are working on.

4. Import HSBC's Certificate into merchant's Keystore.

```
keytool -import
    -alias hsbc_cert_0002
    -file hsbc_cert_0002.cer
    -keystore merchant_keystore.jks
```

- **-import** - command to import object into a specific keystore.
- **-alias** - define the alias name of HSBC's Certificate in your keystore.
- **-file** - specify the file name of HSBC's Certificate in Merchant's local file system.
- **-keystore** - specify the keystore which you are working on.

5. **Optional:** List keystore objects. Merchant is suggested to verify that all required objects are properly maintained. 2 - 3 entries should be found in your Java Keystore: *(Entries may be varied if other key repository format is used)*

Alias name	Corresponding Object	Remark
merchant_key_pair	• Merchant's Private Key • Merchant's Public Certificate (Self-Signed)	These two objects appear to be one entry in a JAVA Keystore. Merchant can still export them separately into two objects (files) on your local file system depending on your application design.
merchant_signed_cert_0001	• Merchant's Public Certificate (CA-Signed)	Not exist if Merchant skips step #2
hsbc_cert_0002	• HSBC's Public Certificate	

```
keytool -list -v -keystore merchant_keystore.jks

Keystore type: JKS
Keystore provider: SUN

Your keystore contains 3 entries

Alias name: merchant_key_pair
Creation date: Jan 1, 2020
Entry type: PrivateKeyEntry

<Other Information>

*******************************************
*******************************************

Alias name: merchant_signed_cert_0001
Creation date: Jan 1, 2020
Entry type: trustedCertEntry

<Other Information>

*******************************************
*******************************************

Alias name: hsbc_cert_0002
Creation date: Jan 1, 2020
Entry type: trustedCertEntry

<Other Information>

*******************************************
*******************************************
```

Certificates and Keys Maintenance

Here are some recommendations to Merchant of how to properly maintain certificates and keys:

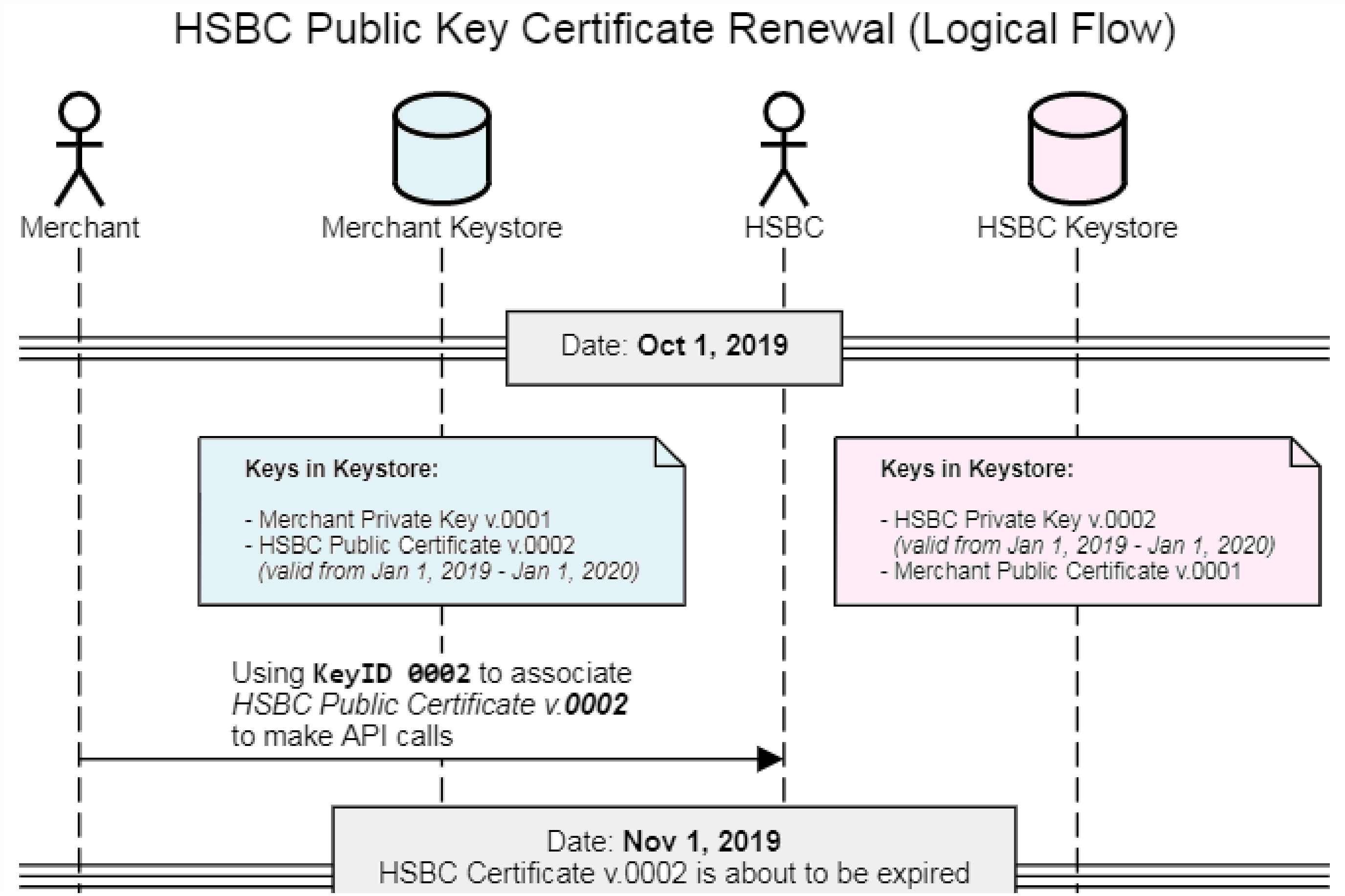
Component	Storage	Validity
Merchant's Private Key	Private Key should be maintained and handled with the most secure approach that a Merchant can apply. The most common and yet secure enough approach is: • **key password** - Do not save the password in plain text or hard-coded in application. Recommend to encrypt it by any Password Encryption Tools • **key storage** - Store inside password-protected key repository, such as `JKS` or `PKCS12` keystore. Keystore password should also be encrypted.	No restriction on the Validity Period. However, if Merchant suspects there is any chance that the key is leaked or for any other security reason, a new Private Key and its associated Public Key Certificate should be generated.
Merchant's Public Key Certificate	Since Public Key Certificate is publicly distributed, a comparative moderate secure storage approach is acceptable. Merchant can store the physical file in any system's file system or store all keys and certificates in one single key repository for a centralised key management.	For a self-signed Certificate, the same condition has been mentioned as above. However, the validity period of a CA-signed Certificate is depended on the purchase plan of the issuing CA. The most common standard is 1 to 2 years.
HSBC's Public Key Certificate	Same as the above	1 Year **NOTICE:** Technically, the validity period is usually 1 Year plus 1 to 2 months more. The spare period is a buffer for a merchant to switch a "to-be-expired" Certificate to the new one during the Certificate Renewal Process. More technical detail will be covered in later section.

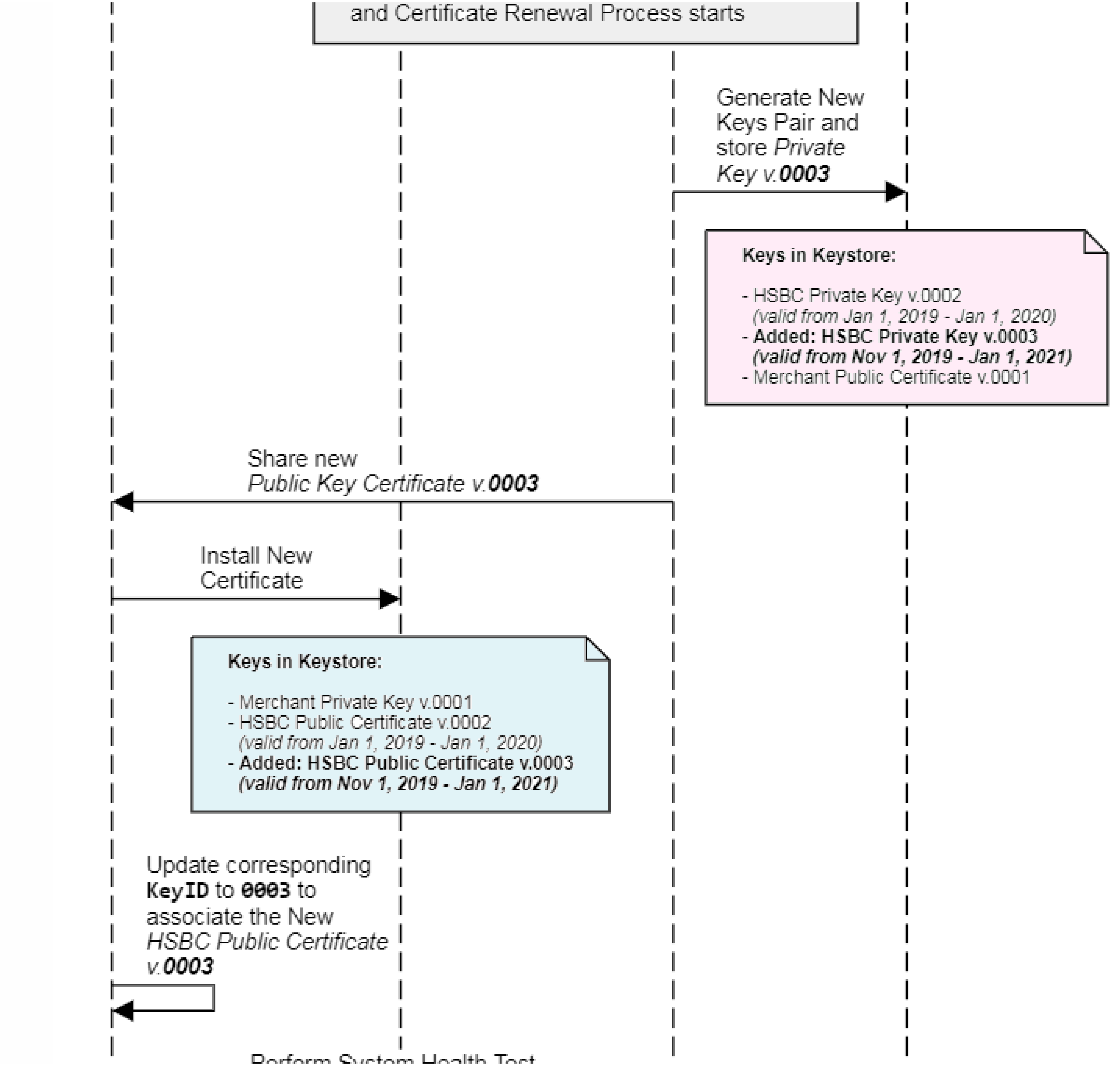
Certificates and Keys Renewal

Every Public Key Certificate has an expiration date and when either Merchant's or HSBC's Certificate is about to expire, a key renewal process will be taken place. Please see the below Key Renewal Process Flow for your reference:

! **SOME RULES YOU SHOULD KNOW:**

- **Keys Repository:** This is a make-up for demonstration purpose only.
- **Keys Name:** Using a `Key Name` `KeyID` naming convention is for a simpler demonstration. The suggested identifier of one key should be the alias name inside a key repository.
- **KeyID Value:** HSBC uses naming convention `0001`, `0002`, `0003`... `n + 1`, when every time HSBC certificate is renewed, the `KeyID` value will be `n + 1`.
- **KeyID Binding:** The binding between `KeyID` and corresponding `Keys Pair` in merchant's system can make use of any key/value logic, such as Database table. In our example below, KeyID `000X` binds to `Private Key v.000X` and `Public Certificate v.000X`, etc.
- **Validity Date:** All dates are make-up for demonstration purpose only.


HSBC Public Key Certificate Renewal (Logical Flow)
Merchant
Merchant Keystore
HSBC
HSBC Keystore
Date: Oct 1, 2019
Keys in Keystore:
- Merchant Private Key v.0001
- HSBC Public Certificate v.0002
(valid from Jan 1, 2019 - Jan 1, 2020)
Keys in Keystore:
- HSBC Private Key v.0002
(valid from Jan 1, 2019 - Jan 1, 2020)
- Merchant Public Certificate v.0001
Using KeyID 0002 to associate
HSBC Public Certificate v.0002
to make API calls
Date: Nov 1, 2019
HSBC Certificate v.0002 is about to be expired

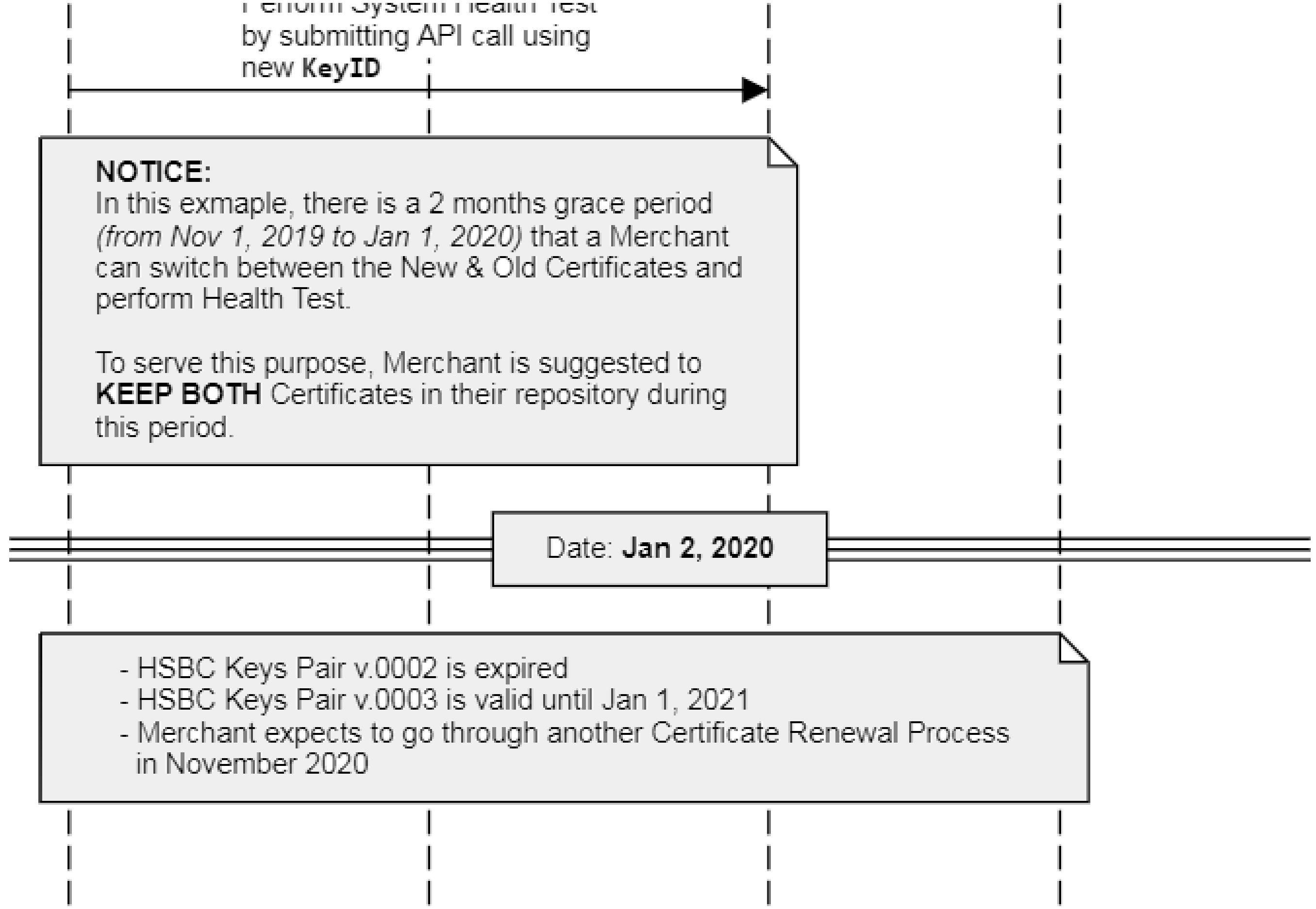

and Certificate Renewal Process starts
Generate New Keys Pair and store *Private Key v.**0003***
Share new *Public Key Certificate v.**0003***
Install New Certificate
Update corresponding **`KeyID`** to **`0003`** to associate the New *HSBC Public Certificate v.**0003***
Perform System Health Test

Keys in Keystore:

- HSBC Private Key v.0002
  (valid from Jan 1, 2019 - Jan 1, 2020)
- **Added: HSBC Private Key v.0003**
  (valid from Nov 1, 2019 - Jan 1, 2021)
- Merchant Public Certificate v.0001

Keys in Keystore:

- Merchant Private Key v.0001
- HSBC Public Certificate v.0002
  (valid from Jan 1, 2019 - Jan 1, 2020)
- **Added: HSBC Public Certificate v.0003**
  (valid from Nov 1, 2019 - Jan 1, 2021)



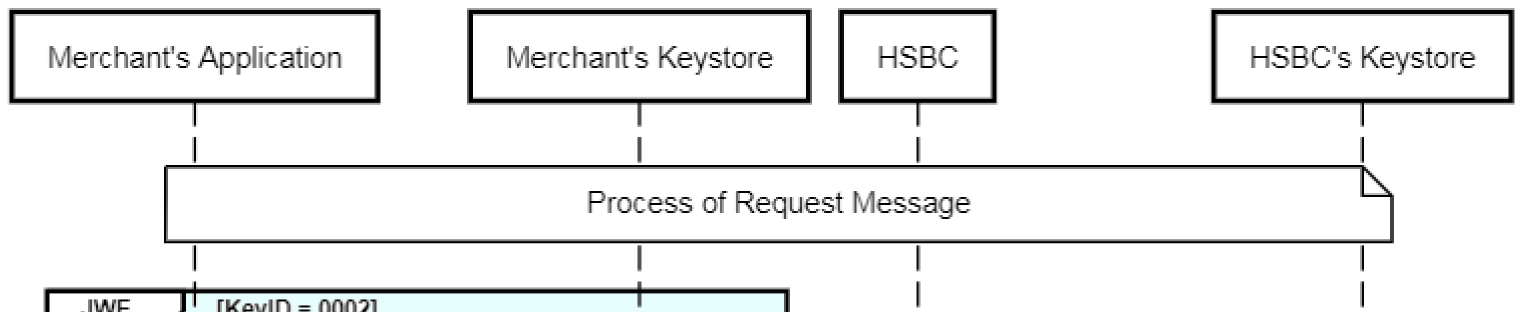
NOTICE:
In this exmaple, there is a 2 months grace period *(from Nov 1, 2019 to Jan 1, 2020)* that a Merchant can switch between the New & Old Certificates and perform Health Test.

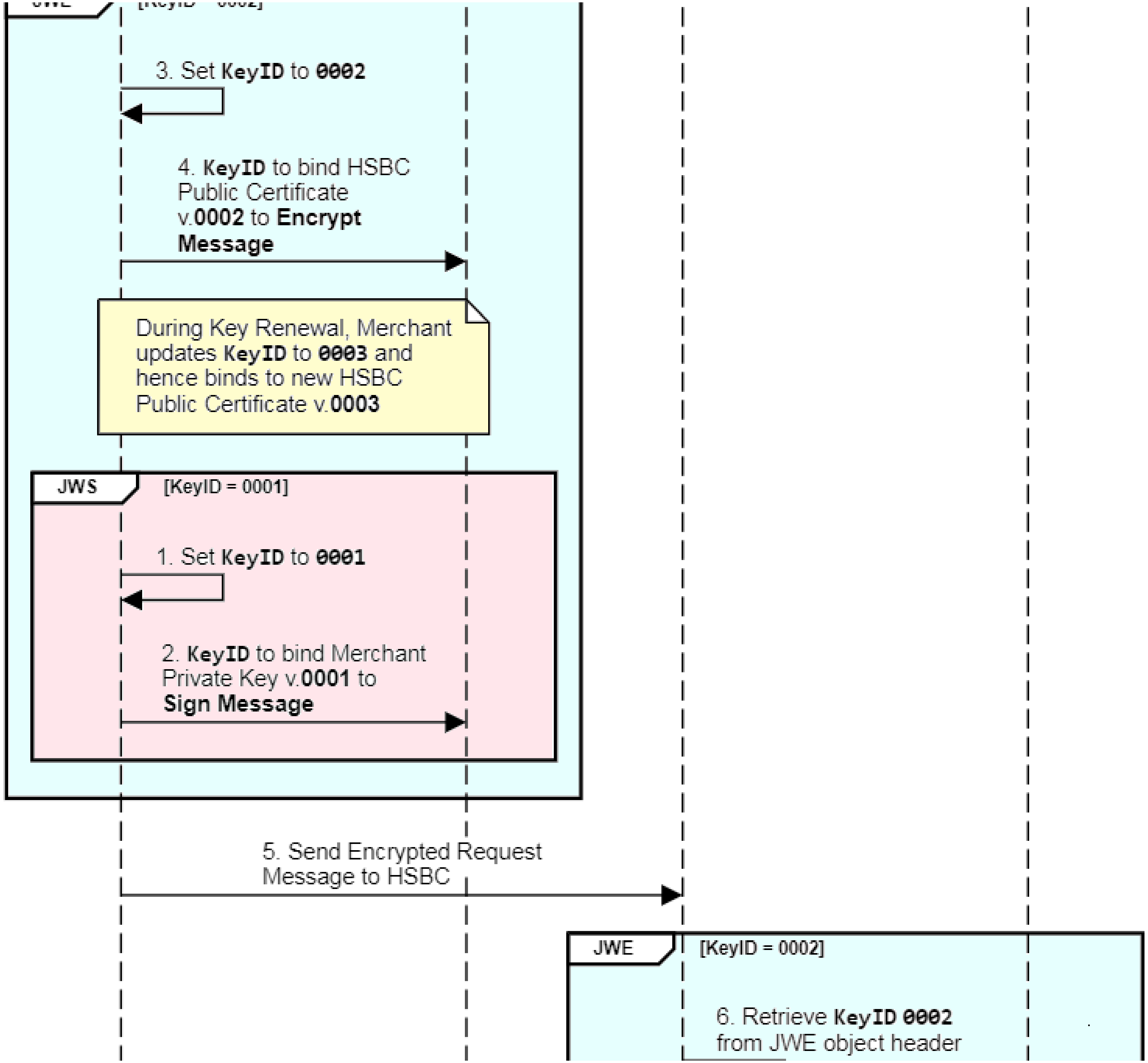
To serve this purpose, Merchant is suggested to **KEEP BOTH** Certificates in their repository during this period.

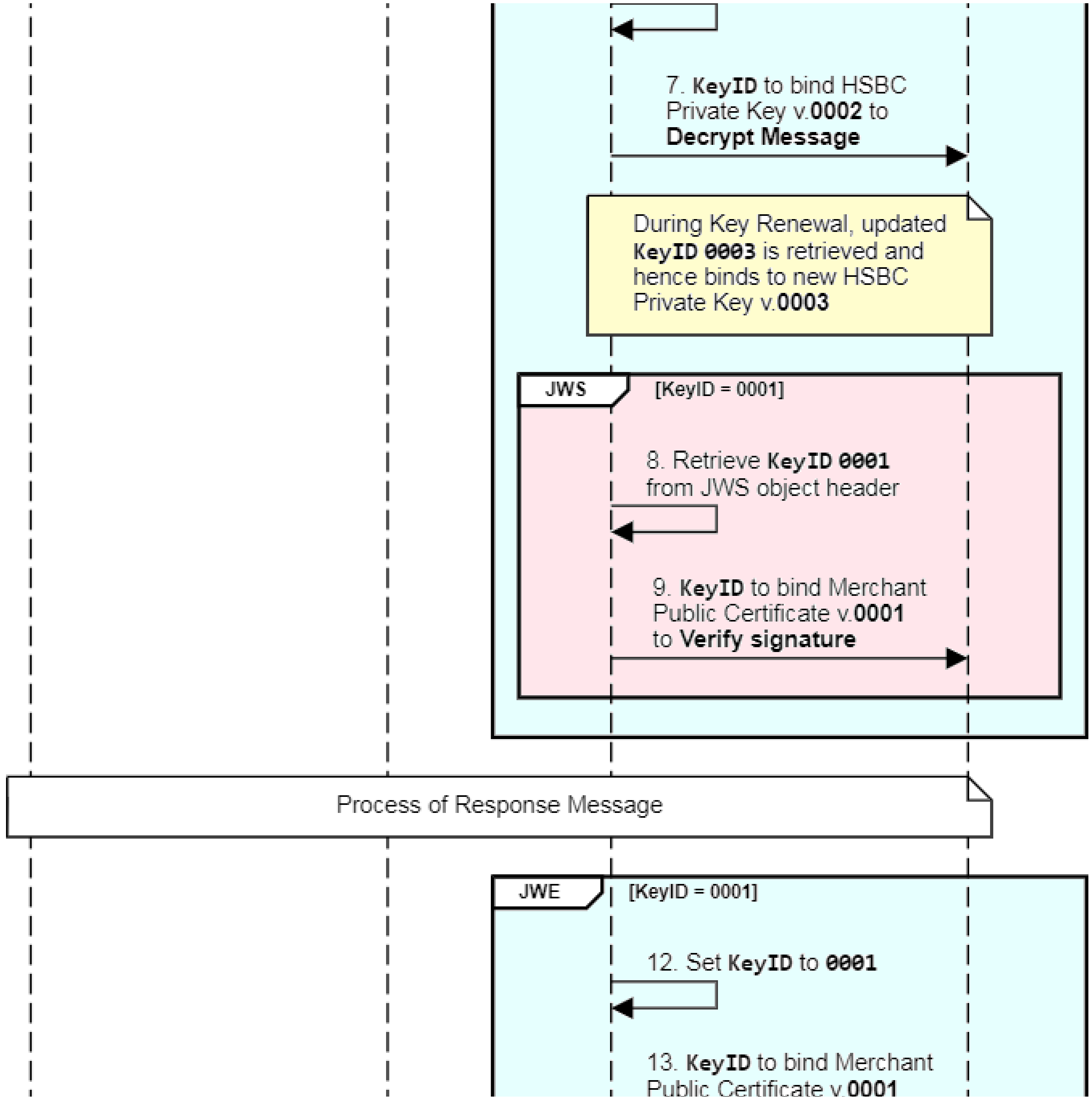
Date: **Jan 2, 2020**

- HSBC Keys Pair v.0002 is expired
- HSBC Keys Pair v.0003 is valid until Jan 1, 2021
- Merchant expects to go through another Certificate Renewal Process in November 2020

Below is the technical flow showing how `Certificates`, `Alias Names` and `KeyIDs` work together during a normal process or a key renewal process:




3. Set KeyID to 0002
4. KeyID to bind HSBC Public Certificate v.0002 to Encrypt Message
During Key Renewal, Merchant updates KeyID to 0003 and hence binds to new HSBC Public Certificate v.0003
JWS
[KeyID = 0001]
1. Set KeyID to 0001
2. KeyID to bind Merchant Private Key v.0001 to Sign Message
5. Send Encrypted Request Message to HSBC
JWE
[KeyID = 0002]
6. Retrieve KeyID 0002 from JWE object header



7. **KeyID** to bind HSBC Private Key v.**0002** to **Decrypt Message**

During Key Renewal, updated **KeyID 0003** is retrieved and hence binds to new HSBC Private Key v.**0003**

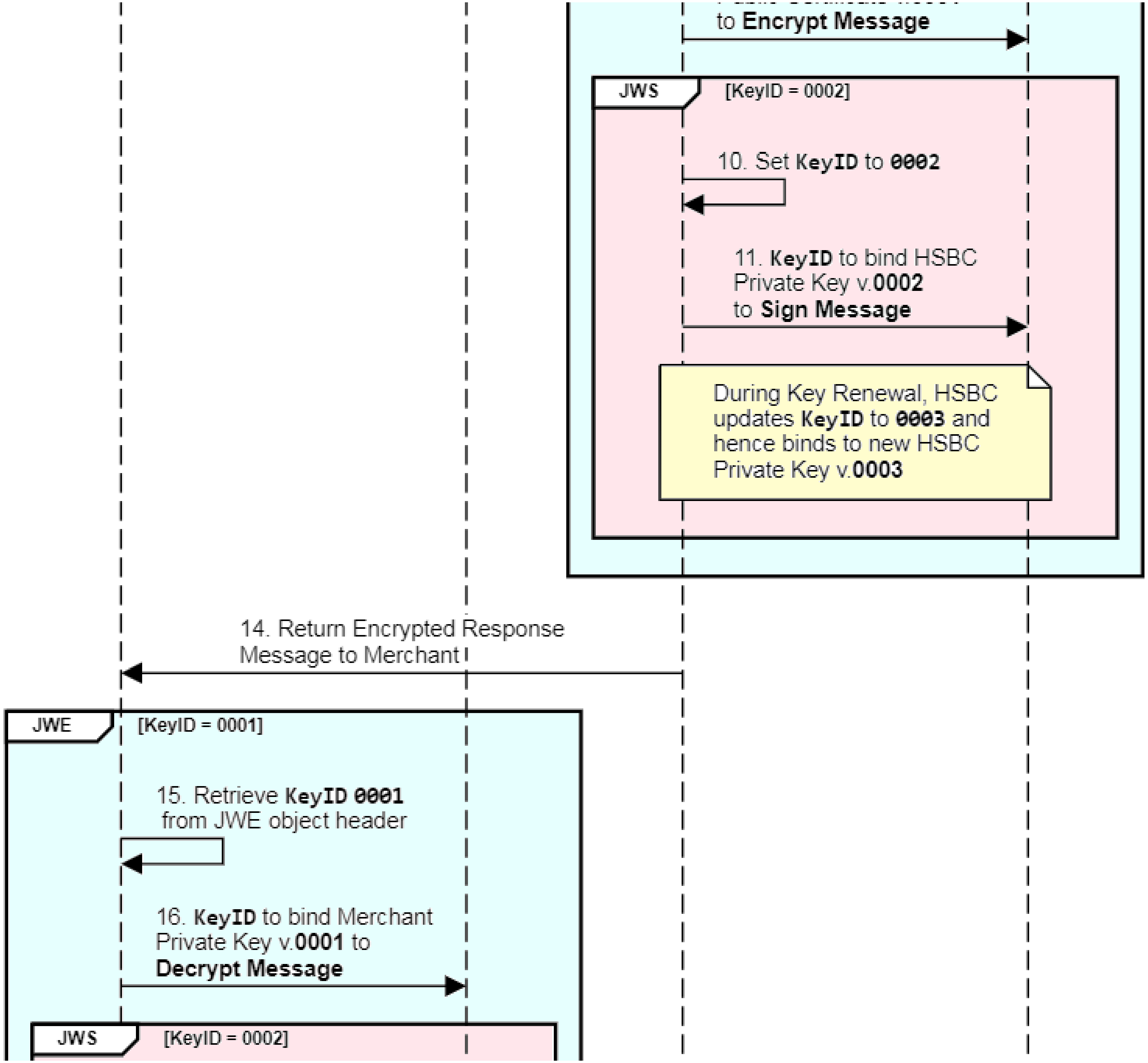
8. Retrieve **KeyID 0001** from JWS object header

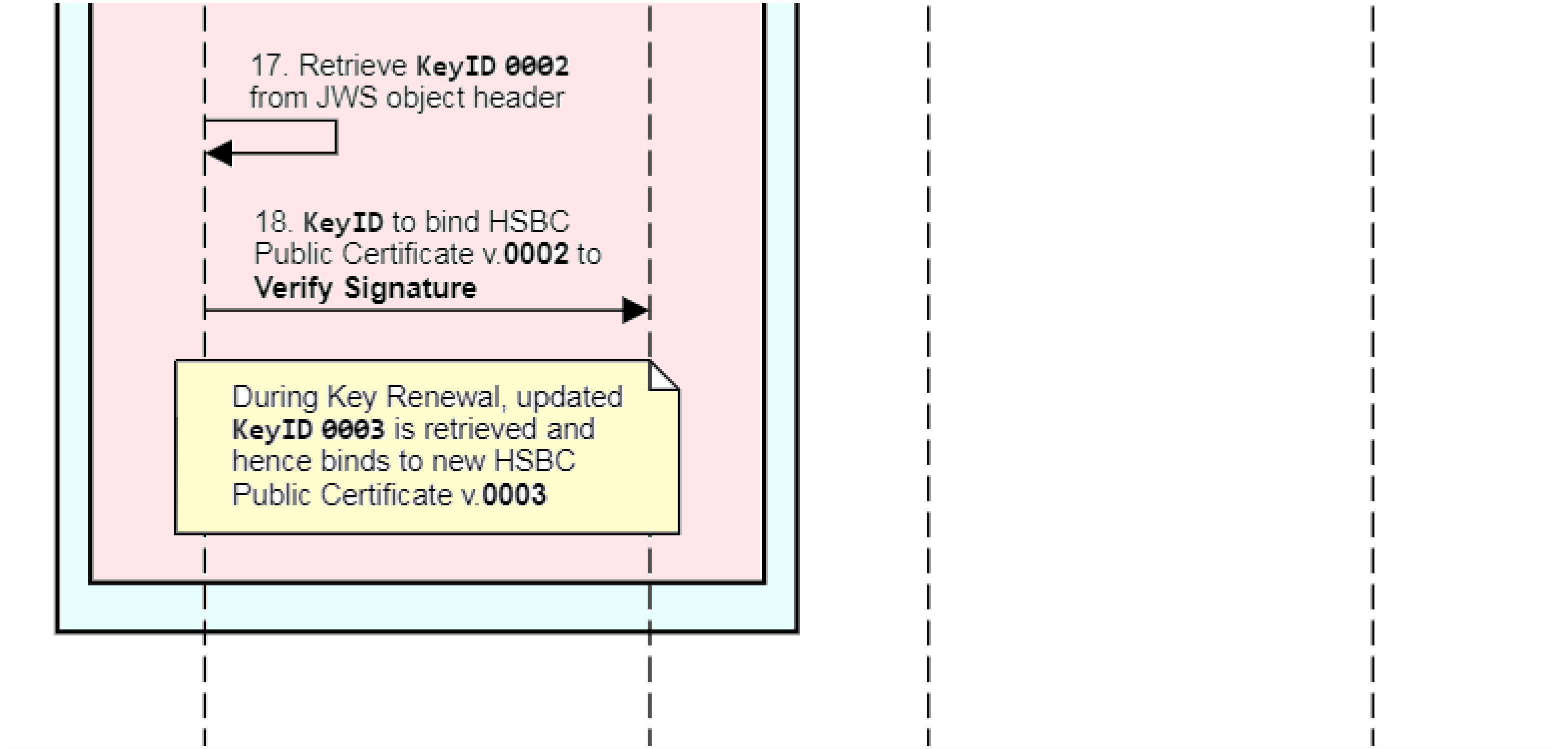
9. **KeyID** to bind Merchant Public Certificate v.**0001** to **Verify signature**

Process of Response Message

12. Set **KeyID** to **0001**

13. **KeyID** to bind Merchant Public Certificate v.**0001**


to Encrypt Message
JWS
[KeyID = 0002]
10. Set KeyID to 0002
11. KeyID to bind HSBC Private Key v.0002 to Sign Message
During Key Renewal, HSBC updates KeyID to 0003 and hence binds to new HSBC Private Key v.0003
14. Return Encrypted Response Message to Merchant
JWE
[KeyID = 0001]
15. Retrieve KeyID 0001 from JWE object header
16. KeyID to bind Merchant Private Key v.0001 to Decrypt Message
JWS
[KeyID = 0002]


17. Retrieve KeyID 0002 from JWS object header
18. KeyID to bind HSBC Public Certificate v.0002 to Verify Signature
During Key Renewal, updated KeyID 0003 is retrieved and hence binds to new HSBC Public Certificate v.0003

! **NOTICE:** All examples above are about the Certificate Renewal of HSBC, whenever Merchant wants to renew their Certificate, please switch your role and steps into HSBC's.

Download Swagger

Click here to download Swagger 2.0 file in YAML format.

Disclaimer

IMPORTANT NOTICE

This document is issued by The Hongkong and Shanghai Banking Corporation Limited, Hong Kong (“HSBC”). HSBC does not warrant that the contents of this document are accurate, sufficient or relevant for the recipient’s purposes and HSBC gives no undertaking and is under no obligation to provide the recipient with access to any additional information or to update all or any part of the contents of this document or to correct any inaccuracies in it which may become apparent. Receipt of this document in whole or in part shall not constitute an offer, invitation or inducement to contract. The recipient is solely responsible for making its own independent appraisal of the products, services and other content referred to in this document. This document should be read in its entirety and should not be photocopied, reproduced, distributed or disclosed in whole or in part to any other person without the prior written consent of the relevant HSBC group member.