**[MustangPay] Merchant API (Web Payment) Interface Manual V1**

**Version Record**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Writer** | **Description** |
| 1.O | 2024/09/10 | Hyssop / Leyvi | initialization |

Catalog

**[Version Record](#_Toc181123142)** [1](#_Toc181123142)

[1.](#_Toc181123143) **[Access Instructions](#_Toc181123143)** [2](#_Toc181123143)

[1.1](#_Toc181123144) **[Communication Method](#_Toc181123144)** [2](#_Toc181123144)

[1.2](#_Toc181123145) **[Character Encoding](#_Toc181123145)** [2](#_Toc181123145)

[1.3](#_Toc181123146) **[Synchronous Interface](#_Toc181123146)** [2](#_Toc181123146)

[1.4](#_Toc181123147) **[Asynchronous Interface](#_Toc181123147)** [2](#_Toc181123147)

[1.5](#_Toc181123148) **[IP whitelist](#_Toc181123148)** [2](#_Toc181123148)

[1.6](#_Toc181123149) **[Request (Response) Message Format Description (JSON)](#_Toc181123149)** [3](#_Toc181123149)

[1.6.1](#_Toc181123150) **[Message Field Type Description](#_Toc181123150)** [3](#_Toc181123150)

[1.6.2](#_Toc181123151) **[All Message Parameters（JSON）](#_Toc181123151)** [3](#_Toc181123151)

[1.6.3](#_Toc181123152) **[Request Message Header Format (JSON)](#_Toc181123152)** [4](#_Toc181123152)

[1.7](#_Toc181123153) **[Signature Mechanism](#_Toc181123153)** [5](#_Toc181123153)

[1.7.1](#_Toc181123154) **[AES Encryption and Decryption](#_Toc181123154)** [5](#_Toc181123154)

[1.7.2](#_Toc181123155) **[RSA Signature](#_Toc181123155)** [5](#_Toc181123155)

[1.7.3](#_Toc181123156) **[Encryption and Signature Sequence (Requesting Party)](#_Toc181123156)** [5](#_Toc181123156)

[1.7.4](#_Toc181123157) **[Signature Verification and Decryption Sequence (Receiving Party)](#_Toc181123157)** [6](#_Toc181123157)

[2.](#_Toc181123158) **[API Instructions (Message Body)](#_Toc181123158)** [6](#_Toc181123158)

[2.1](#_Toc181123159) **[Pre-order API (Synchronous Interface)](#_Toc181123159)** [7](#_Toc181123159)

[2.2](#_Toc181123160) **[Merchant Order Query API (Synchronous Interface)](#_Toc181123160)** [11](#_Toc181123160)

[2.3](#_Toc181123161) **[Asynchronous Response Message Format](#_Toc181123161)** [13](#_Toc181123161)

[3.](#_Toc181123162) **[Reference](#_Toc181123162)** [14](#_Toc181123162)

[3.1](#_Toc181123163) **[Interface Return Code List](#_Toc181123163)** [14](#_Toc181123163)

[3.2](#_Toc181123164) **[payMethod](#_Toc181123164)** [15](#_Toc181123164)

[3.3](#_Toc181123165) **[orderStatus](#_Toc181123165)** [15](#_Toc181123165)

[3.4](#_Toc181123166) **[currency](#_Toc181123166)** [15](#_Toc181123166)

[3.5](#_Toc181123167) **[Country](#_Toc181123167)** [16](#_Toc181123167)

[4.](#_Toc181123168) **[Important Notes](#_Toc181123168)** [16](#_Toc181123168)

[4.1](#_Toc181123169) **[Integration Testing Notes](#_Toc181123169)** [16](#_Toc181123169)

[4.2](#_Toc181123170) **[Production Deployment Notes](#_Toc181123170)** [16](#_Toc181123170)

[5.](#_Toc181123171) **[POSTMAN](#_Toc181123171)** [16](#_Toc181123171)

[6.](#_Toc181123172) **[Integration Code (Java)](#_Toc181123172)** [16](#_Toc181123172)

[6.1](#_Toc181123173) **[Jar Package](#_Toc181123173)** [16](#_Toc181123173)

[6.2](#_Toc181123174)  **[Test Environment Public and Private Keys](#_Toc181123174)** [17](#_Toc181123174)

[6.2.1](#_Toc181123175) **[MustangPay Public Key](#_Toc181123175)** [17](#_Toc181123175)

[6.2.2](#_Toc181123176) **[Online Public and Private Key Exchange](#_Toc181123176)** [17](#_Toc181123176)

[7.](#_Toc181123177) **[Environment Information](#_Toc181123177)** [17](#_Toc181123177)

1. **Access Instructions**

1.1 **Communication Method**

All message fields are sent in **JSON** string format via **HTTPS POST** requests.

The platform's interface adopts two acceptance mechanisms:

* synchronous interface
* asynchronous interface (synchronous request acceptance with asynchronous notification of the final result).

1.2 **Character Encoding**

All encoding required for API transmission and encryption uses the **UTF-8** character set.

1.3 **Synchronous Interface**

The merchant sends an API request to MustangPay, and MustangPay synchronously responds with the parameter data defined by the interface.

There are no further processes, and the transaction is completed in one step.

1.4 **Asynchronous Interface**

When the merchant receives the string 'SUCCESS' from the asynchronous notification interface, it indicates that the notification has been successfully received.

At this point, the merchant needs to actively call the query interface or manually verify the transaction status.

1.5 **IP whitelist**

Both parties must configure the IP whitelist for API requests in their respective systems.

When sending requests, IP addresses are checked against the whitelist, and only whitelisted IPs are allowed. If the IP is not whitelisted, an IP restriction error will be returned.

IP binding can only be done manually and requires contacting MustangPay support to request the binding.。

1.6 **Request (Response) Message Format Description (JSON)**

1.6.1 **Message Field Type Description**

|  |  |
| --- | --- |
| **Fields** | **Instructions** |
| country | Only Support（ZAF） |
| currency | Only Support（ZAR） |
| amount | AmountType  {  "value":Long  "currency":String  } |

1.6.2 **All Message Parameters（JSON）**

*All interface requests and responses are in JSON format and contain the following fields, for reference:*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Fields** | **Instructions** | **Type** | **Mandatory** | **Remark** |
| merchantId | Merchant Order Number | String | Y | The unique merchant number provided by MustangPay when the merchant registers with the platform. |
| head | Message Header | Text | Y | SON format, refer to the "Request/Response Message Header Format." |
| encryptData | Encrypted Text | FixStrin（1024） | Y | Refer to the signature mechanism. |
| encryptKey | Encrypted Text | Text | Y | Refer to the signature mechanism. |

{

"merchantId": "240717251000200170",

"encryptData": "",

"encryptKey": "gtHWDQdKvgPGEheZWaZzfoAiYp40jqtkisDHFhvD0jee1gdJUZN0TS7Cl4WRncVocpA6PU3wkdvoBaq\_IsM1VukiV2MnEkmzNFgfWGTVg0EP9x2XRCCisdLzfNxBs6DaXlCASBj-BvQA4uaMfxoRbw22MNsy-88WSPI5BxoPSaI"

}

1.6.3 **Request Message Header Format (JSON)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Fields** | **Instructions** | **Type** | **Mandatory** | **Remark** |
| version | Interface Version Number | Fix String（3） | Y | 1.0 |
| reqTime | Timestamp | Datetime | Y | Timestamp of the sent message  yyyy-MM-dd'T'HH:mm:ssXXX |

{"version":"1.0","reqTime":"2020-07-24T11:32:40+08:00"}

1.7 **Signature Mechanism**

To ensure the authenticity, integrity, and non-repudiation of data during transmission, we need to apply a digital signature to the data. After receiving the signed data, a signature verification will be performed. The signing algorithm used is RSA. The signature prevents the request and notification data submitted from the merchant's system from being tampered with illegally.

1.7.1 **AES Encryption and Decryption**

AES encryption uses the GCM mode of the AES algorithm. The data is padded using the PKCS#7 padding scheme.

1.7.2 **RSA Signature**

Both the merchant and the platform generate their own pair of RSA key files (public.key for the public key and private.key for the private key), with a key size of 1024 bits and in PKCS#8 format. Both parties exchange public key files (public.key).

The signature method uses the algorithm: Sha256WithRSA.

1.7.3 **Encryption and Signature Sequence (Requesting Party)**

*This process is suitable for the requesting party to organize request data signatures and for the responding party to sign response data.*

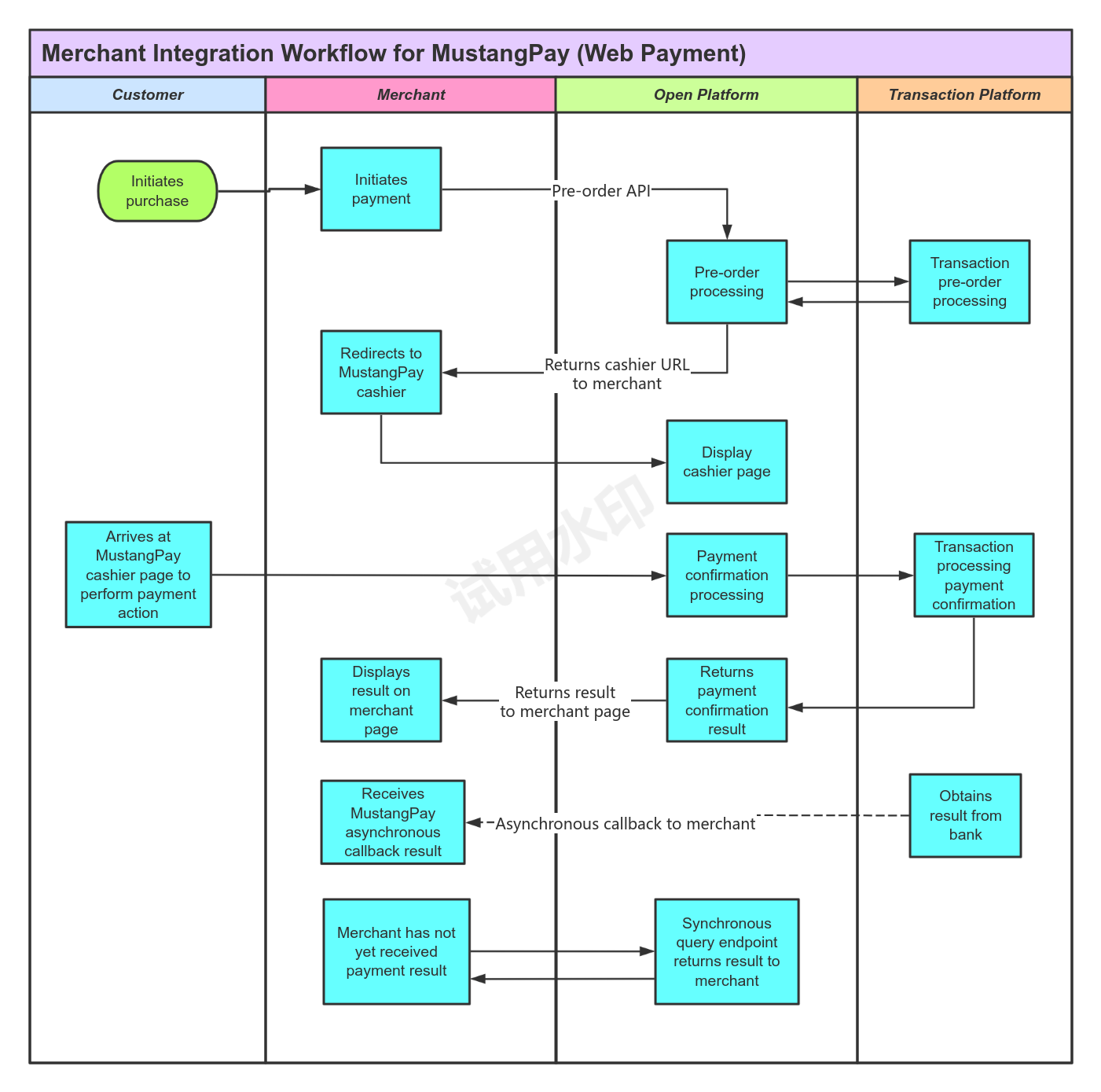
1. Generate the original JSON format for the Message Header: srcHead and the message body: srcBody according to the requirements of each interface.
2. Wrap srcBody as a JSON string.
3. Sort the JSON string using SerializerFeature.MapSortField.\*(this is fastjson’s method,you can using other method with the same function)
4. Use the merchant's private key and RSA toolkit to sign, and place the signature value into the JSON object from step 1 (Field name: sign).
5. Once again, sort the JSON using the SerializerFeature.MapSortField method.
6. Generate a random aesKey.
7. Encrypt the JSON string from step 4 with the random aesKey.
8. Encrypt the aesKey using MustangPay's public key.
9. Package the encrypted AES key and data into a JSON object with the following three fields to send to MustangPay:
10. Place the encrypted AES key value into the field encryptKey.
11. Place the encrypted data into the field encryptData.
12. Include the merchant ID in the field merchantId.

1.7.4 **Signature Verification and Decryption Sequence (Receiving Party)**

*This process is suitable for the receiving party to verify signatures and for verifying the response data received by the requesting party.*

1. After receiving the data from MustangPay, encryptData is the encrypted message, and encryptKey is the encrypted key.
2. Use the public key provided by MustangPay to verify the signature.
3. Extract the content of encryptKey and encryptData:
4. Use the merchant's private key to decrypt encryptKey and obtain the aesKey used to encrypt the message.
5. Use the AES algorithm with the decrypted aesKey to decrypt encryptData and obtain the originalData.
6. Parse originalData into a FastJSON object: originalDataObj.
7. Extract the sign field from originalDataObj and remove sign from the originalDataObj.
8. With the sign field, MustangPay's public key, and the originalDataObj, use the RSA algorithm to verify whether the signature is valid.。

2. **API Instructions (Message Body)**



2.1 **Pre-order API (Synchronous Interface)**

**Request URL:** /openApi/merchant\_direct/cashier/preorder (see domain in section 7)

**Request Method:** POST

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Request Body** | | | | | |
| **Fields** | **Type** | **Mandatory** | **Length** | **Description** | **Reference** |
| merchantId | String | Yes | 32 | Merchant ID | 240717251000200170 |
| reference | String | Yes | 64 | Customer Order No | merchantOrderNo\_1dd99038590 |
| amount | Amount | Yes |  | Order Amount | {"value": 10, "currency": "ZAR"} |
| -value | Long |  |  | Total Amount (unit: Cents) | 10 |
| -currency | String |  | 8 | Currency: ZAR | ZAR |
| payMethods | List<String> | No |  | Payment Methods | ["CardPayment", "InstantEFT"] |
| currency | String | Yes | 8 | Currency | ZAR |
| country | String | Yes | 8 | Country | ZAF |
| businessType | String | Yes | 64 | Business Type | MerchantAcquiring |
| remark | String | No | 255 | Remark | remark\_83c200fa64ff |
| callbackUrl | String | Yes | 255 | Merchant Callback URL (Required) | callbackUrl |
| returnUrl | String | No | 255 | Merchant Return URL (Optional) | returnUrl |
| cancelUrl | String | No | 255 | Merchant Cancel Return URL (Optional) | cancelUrl |
| ip | String | No | 64 | User IP Address | 127.0.0.1 |
| expireAt | Integer | No |  | Order Expiry Time (in minutes) | 30 minutes |
| vatNumber | String | No | 64 | VAT Number | 797879798 |
| vat | Amount | No |  | Tax | {"value": 10, "currency": "ZAR"} |
| -value | Long |  |  | Total Amount (unit: Cents) | 10 |
| -currency | String |  | 8 | Currency: ZAR | ZAR |
| product | Product | No |  | Product Information |  |
| -name | String | Yes | 128 | Product Name | Women's T-Shirt |
| -shortName | String | No | 64 | Product Short Name | T-Shirt |
| -description | String | No | 255 | Product Description | Women's Summer Cartoon Dog T-Shirt |
| metadata | Map | No |  | Additional Fields | Custom Fields |

**Example of a request message:**

{

"merchantId": "240717251000200170",

"reference": "merchantOrderNo\_14dd90e38590", //Merchant Order Number

"amount": {"value": 10, "currency": "ZAR"},

"currency": "ZAR",

"country": "ZAF",

"businessType": "MerchantAcquiring",

"remark": "remark\_83c200fa64ff",

"callbackUrl": "https://mustangpay-test-demo-dev.mstpay-inc.com/callback",

"returnUrl": "https://mustangpay-test-demo-dev.mstpay-inc.com",

"cancelUrl": "",

"payMethods": [], // Place data according to 3.2 payMethod catalog.

"ip": "127.0.0.1",

"expireAt": 30,

"vat": 10,

"vatNumber": "1234",

"product": {

"name": "productname",

"shortName": "short",

"description": "productDesc"

}

}

|  |  |  |  |
| --- | --- | --- | --- |
| **Response Body** | | | |
| **Fields** | **Type** | **Description** | **Reference** |
| code | String | Six digits indicate success, others indicate failure | 000000 |
| message | String |  | ok |
| data | Object |  | {...} |
| - reference | String | Customer Order No | merchantOrderNo\_1dd990e38590 |
| - orderNo | String | Order No | 240719101001007914 |
| - orderStatus | String | Order Status | Initial |
| - amount | Amount | Order Amount | {"value":100000,"currency":"ZAR"} |
| - vat | Amount | VAT Amount | {"value":100000,"currency":"ZAR"} |
| errorCode | String | Error Code |  |
| errorMessage | String | Error Description |  |
| cashierUrl | String | Redirect Payment URL | Note: this is the mustangPay redirect URL |

**Example of a return message：**

{

"code": "000000",

"message": "ok",

"data": {

"reference": "merchantOrderNo\_14dd90e38590",

"orderNo": "240719101001007914",

"orderStatus": "Initial",

"amount": null,

"vat": null,

"errorCode": null,

"errorMessage": null,

"merchantId": "240717251000200170",

"redirectPayUrl": "https://cashier-front.mustangpay.co.za/cashier?orderNo=o7J6eVzXMq0rIh8ogXO0olYY1-WPnlRyPZy6ocY0-wM="

}

}

2.2 **Merchant Order Query API (Synchronous Interface)**

**Request URL:** /openApi/merchant\_direct/cashier/getOrderStatusByMerchantOrderNo (see domain in section 7)

**Request Method:** POST

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Request Body** | | | | | |
| **Fields** | **Type** | **Mandatory** | **Length** | **Description** | **Remark** |
| merchantId | String | Yes | 32 | Merchant ID | MustangPay Branch |
| reference | String | Yes | 64 | Merchant Order Number | Customer Order Number |

**Example of a request message:**

{

"merchantId": "240717251000200170",

"reference": "merchantOrderNo\_14dd90e38590"

}

|  |  |  |  |
| --- | --- | --- | --- |
| **Response Body** | | | |
| **Fields** | **Type** | **Description** | **Reference** |
| code | String | Six digits indicate success, others indicate failure | 000000 |
| message | String |  | ok |
| data | Object |  | {...} |
| -merchantId | String | Merchant ID | 123456789 |
| -merchantOrderNo | String | Merchant Order Number | merchantOrderNo\_123456 |
| -merchantName | String | Merchant Name | merchantName |
| -orderNo | String | Mustangpay Order No | 240909011013192628533424 |
| -orderStatus | String | Order Status | Success |
| -amount | Amount | Order Amount | {"value":100000,"currency":"ZAR"} |
| -errorCode | String | Error Code | 999999 |
| -errorMessage | String | Error Description | System exception |

**Example of a response message:**

{

"merchantId": "123456789",

"merchantOrderNo": "BUF123456",

"orderNo": "240909011103192628533242",

"orderStatus": "Success",

"merchantName": "Example Merchant",

"vatNumber": "VAT123456",

"errorCode": "",

"errorMessage": "",

"amount": {

"value": 100.00,

"currency": "ZAR"

}

}

2.3 **Asynchronous Response Message Format**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Request body** | | | | |
| **Fields** | **Type** | **Length** | **Mandatory** | **Remark** |
| orderNo | String | 32 | Yes | Order number |
| merchantId | String | 32 | Yes | Merchant ID |
| merchantOrderNo | String | 64 | Yes | Merchant Order Number |
| orderStatus | String | 32 | Yes | Refer to Section 3.3 |
| errorCode | String | 32 | No | Error code |
| errorMessage | String | 255 | No | Error message |
| amount | Amount |  | Yes | Amount |
| -value | long |  | Yes | Amount in Cents |
| -currency | String | 8 | Yes | Currency - ZAR |

**Example of a notification message:**

{

"orderNo": "240909011103192628533242",

"merchantId": "123456789",

"merchantOrderNo": "BUF123567",

"orderStatus": "Success",

"merchantName": "Example Merchant",

"vatNumber": "VAT123456",

"errorCode": "",

"errorMessage": "",

"amount": {

"value": 100.00,

"currency": "ZAR"

}

}

3. **Reference**

3.1 **Interface Return Code List**

***Note: The meaning of error codes may vary depending on the environment. Verify the context to ensure accuracy before implementation.***

|  |  |  |
| --- | --- | --- |
| **Code** | **Description** | **Remark** |
| **000000** | Successful transaction | The transaction has been processed successfully. During the processing, no errors were detected, the transaction does not violate any rules, and no suspicious activity was detected that would trigger a security alert. Therefore, the transaction has been processed as legitimate and successful. |
| 100022 | Invalid payment method | Check payment settings. |
| 100018 | Unsupported payment method | Verify if the method is still supported. |
| 100002 | Transaction failed | Retry or check payment method. |
| 030380 | Transaction failed | Retry or check payment method. |
| 03015 | IP address blocked | Contact network administrator. |
| 06001 | Transaction is being processed | Pending |
| 030362 | Merchant ID does not exist | Check the merchant ID. |
| 030383 | Merchant account has been closed | Merchant account is no longer active with MustangPay. |

3.2 **payMethod**

|  |  |  |
| --- | --- | --- |
| **Code** | **Description** | **Remark** |
| CardPayment | Card Payment | Code must be in uppercase |
| InstantEFT | Instant Electronic Fund Transfer (EFT) | Code must be in uppercase |

3.3 **orderStatus**

|  |  |  |
| --- | --- | --- |
| **Code** | **Description** | **Remark** |
| Initial | Order initiated (pre-order) |  |
| Pending | Processing |  |
| Success | Successful transaction |  |
| Fail | Transaction failed |  |
| Return | Ticket returned |  |
| Close | Order closed due to timeout |  |

3.4 **currency**

|  |  |  |
| --- | --- | --- |
| **Code** | **Description** | **Remark** |
| ZAR | South Africa Currency | Currency |

3.5 **Country**

|  |  |  |
| --- | --- | --- |
| **Code** | **Description** | **Remark** |
| ZAF | South Africa |  |

4. **Important Notes**

4.1 **Integration Testing Notes**

1. Before integrating the withdrawal interface, the user signing interface must be integrated; otherwise, the withdrawal will fail.
2. The integration testing environment shares the same setup as the testing environment, so the integration environment does not involve real payments.
3. The payment methods (payMethod) supported by the merchant are agreed upon during the signing process and maintained by the development team in the test data.

4.2 **Production Deployment Notes**

Use the method provided by the MustangPay API to generate the public and private keys. Send the public key to the staff's email: hyssop.zhang@mustangcash.com.

Keep the private key securely stored and do not disclose it. MustangPay reserves the right to pursue compensation for any financial loss caused by the disclosure of the private key on the merchant's side.

5. **POSTMAN**

Please follow the documentation to change the postman address. 

6. **Integration Code (Java)**

6.1 **Jar Package**

https://github.com/mustangpay/mustangpay\_v1

6.2  **Test Environment Public and Private Keys**

6.2.1 **MustangPay Public Key& Private Key**



in the test environment, mustangpay and merchant use the same public key and private key for debugging interface.

6.2.2 **Online Public and Private Key Exchange**

Use the MerchantRsaKeyCreateTest in the demo code to generate the public and private keys.

The private key is used by the merchant to decrypt the encrypted data returned by MustangPay: for signature verification and decryption.!!!

Send the public key to the email: hyssop.zhang@mustangcash.com.

7. **Environment Information**

|  |  |
| --- | --- |
| **Sandbox Environment URL** | https://openapi-sandbox.mstpay-inc.com |
| **Production Environment URL** | https://openapi.mustangpay.co.za |
| **Sandbox Merchant Id** | 4449999220 |