Nobill Open API

This document is written to cooperate with merchants to integrate their systems to Nobill services.

Revision History

Version	Author	Date	Memo
0.2	Zenas Song	12/12/2022	1. Add invoiceNo in the Push-Receipt API. 2. Signature generation includes all the request body, not the specified fields.

Prerequisites

The merchant is required to open a merchant account in Nobill. Nobill will issue the corresponding merchant API key and API secret to the merchant after the enrollment is approved.

The merchant then calls the Merchant Admin Open API to create their service API key and API secret.

Nobill provides the following merchant with API key and API secret for the test:

Merchant ID	MC01GM0C0WRQMCXM862ERTX91WCM
Merchant Service ID	MS01GM0C0WRQYAFXPAS76DWE6C8E
API Key	cN0WU6ax8xA1NjSjJ6SKM3wIH7R028Bj
API Secret	KgFQdd8BVHAXAaZtdCSWIUmELd2rIrPY1EIxuOfWu8TmsmqxAIVTFTxEiJZSG8IgUIpIKGHQF6i2hbcoRpHrX69DyaiDXgPdhFPYwGvT4HV8IZhs6cO0Hj4PcGGIVw0T

API Connection Mechanism

The Nobill Open API is a HTTP RESTful API with the JSON protocol guarded by API Key & API Secret signature.

Service Root

The API URL is composed by the Service Root and the endpoint. The Service Root in the pilot environment is:

```
http://ec2-3-28-200-135.me-central-1.compute.amazonaws.com:8080
```

Request Content Type

The API protocol is JSON-based, so the `Content-Type` in the request should be set to `application/json`:

```
Content-Type: application/json
```

Response Structure

The API has a unified structure for the response body, which is a JSON object with three top fields. If the API responds correctly, they are statusCode, code and result, and if not, they are statusCode, code and message. The following is the two examples:

Error Response

The communication error occurs if the “statusCode” is not 200. And the “code” shows the exact reason why the communication failed.
If the “statusCode” is 200, and the “code” is not 0, there is a business exception happened. Please check the Appendix to find all the error codes.

```
{
  "statusCode": 500,
  "code": "1502",
  "message": "Merchant API Key Not Found"
}
```

Success Response

The code is “0” if the API responds correctly and the business logic is running correctly.

```
{
  "statusCode": 200,
  "code": "0",
  "result": {
    "receiptId": "RC01GM3NBYM514WMA0TKC7R3W0JE"
  }
}
```

Security Guard

The Nobill Open API is security-leveled, the restricted APIs are accessed securely with the **HMAC-SHA256** signature based on API Key & API Secret.

Generate Signature

The formula for generating the request signature is:

```
Source = "<API_KEY><API_SECRET><Request Body>"
Secret = "<API_SECRET>"
Sig = HEX(HMAC-SHA256(Source, Secret))
```

The Free HMAC-SHA Online Generator Tool can help to verify the output is correct or not.

If the request has no body, then keep the Request Body to an empty string.

Verify Signature

The request carries the identity and the signature by two extra headers. `X-Nobill-API-KEY` is for the merchant identity. `X-Nobill-API-Signature` is for the signature.

```
X-NoBill-API-KEY: cN0WU6ax8xA1NjSjJ6SKM3wIH7R028Bj
X-NoBill-API-Signature:
fb02bef8bfb1c7686a1579ccc4ecfe1f2dce4616bbec92581a5c43e187783bc1
```

Recap

The Nobill Open API is JSON-based HTTP RESTful web services which has the API Key and the API Signature in the header to identify the merchant and guard the API. Here is a sample for the request:

```
POST {{serviceRoot}}/api/s/receipt/v2/push-receipt
Content-Type: application/json
X-NoBill-API-KEY: cN0WU6ax8xA1NjSjJ6SKM3wIH7R028Bj
X-NoBill-API-Signature:
fb02bef8bfb1c7686a1579ccc4ecfe1f2dce4616bbec92581a5c43e187783bc1

{
  "version": 1,
  "merchantId": "MC01GM0C0WRQMCXM862ERTX91WCM",
  "merchantBranchId": "MBR01GM0C0WRQTAVJG0ZEY5GC9RJQ",
  "merchantReceiptId": "9642b4f0-8163-49ef-a004-0e0413c157fb",
  ...
}
```

API - Push Receipt

Push Receipt API is a restricted API, the API Key and the API Signature is required.

Endpoint

```
/api/s/receipt/v2/push-receipt
```

Method

```
POST
```

Request Body

The request body is a JSON object which includes Meta information, Transaction, Purchases, Payment and Invoice.

Meta Information

The meta-information fields are on the top layer of the object.

Field name	**Required**	**Type**	**Value**	**Description**
version	M	Number	1	the protocol version
merchantId	M	String	string(1-32)	the merchant ID which is generated by *NoBill* when the merchant registers itself on *NoBill*.

merchantBranchId	M	String	string(1-32)	the merchant branch ID which is generated by *NoBill* when the merchant registers itself on *NoBill*.
merchantReceiptId	O	String	string(4-64)	the receipt ID on the merchant system
customerEmail	C	String	string(3-255)	the customer's email to receive the receipt
customerMobile	C	String	string(8-20)	the customer's mobile to receive the receipt
customerId	C	String	string(2-32)	the customer's ID in *NoBill*, the *NoBill* system will resolve it to get an email or a mobile number to send the receipt to.
createTimestamp	M	Number	Unix timestamp (in seconds)	the receipt creating time on the merchant system.
transaction	M	Transaction		transaction object, see it in the following sections
purchase	M	Purchase		purchase object, see it in the following sections
payment	M	Payment		payment object, see it in the following sections

Notice: only one of the customerEmail, customerMobile, customerId is present in the protocol.

Transaction

The transaction field is a JSON object to describe the transaction information.

Field name	Required	Type	Value	Description
txnNo	M	String	string(2-50)	the transaction no in the merchant's system
txnDate	M	String	string(10), yyyy-MM-dd	the transaction date
txnTime	M	String	string(5), hh:mm	the transaction time
cashier	M	String	string(2-50)	the cashier's name

Purchase

The purchase field is a JSON object to describe the purchasing information.

Field name	Required	Type	Value	Description
quantity	M	Number	>0	the quantity of the purchasing items
items	M	Array of PurchaseItem		the list of purchasing items, see it in the following sections

amount	M	PurchaseAmount		the purchasing amount object, see it in the following sections
tax	M	PurchaseTax		the purchasing tax object, see it in the following sections

PurchaseItem

The purchased item is a JSON object to describe the purchased item.

Field name	Required	Type	Value	Description
code	M	String	string(2-20)	the purchase item code in the merchant system
name1	M	String	string(2-50)	the main item name
name2	O	String	string(2-50)	the second name of the item, it can be used to support the second language
quantity	M	Number	unsigned int	the quantity of the item
price	M	Number	float	the price of the item
currency	M	String	string(3)	the currency code of the price
total	M	Number	float	the total amount of the item

taxName	M	String	string(2-10)	the tax name of the item
taxRate	M	Number	float	the tax rate of the item
discountTitle	O	String	string(2-50)	the discount of the item
discountAmount	O	Number	float	the discount amount of the item

PurchaseAmount

Field name	Required	Type	Value	Description
currency	M	String	string(3)	the currency of the receipt
total	M	Number	float	the total amount
discount	M	Number	float	the discount amount

PurchaseTax

Field name	Required	Type	Value	Description
currency	M	String	string(3)	the currency of the tax
exemption	M	Number	float	the amount of the exemption part

exoneration	M	Number	float	the amount of the exoneration part
taxed	M	Number	float	the amount of the taxed part
total	M	Number	float	the final amount of the tax

Payment

The payment is a JSON object to describe how the customer pays for the purchase.

Field name	Required	Type	Value	Description
currency	M	String	string(3)	the currency of the payment
total	M	Number	float	the total amount of the payment
items	M	Array of PaymentItem		the payment method items for the payment, see it in the following sections

PurchaseItem

Field name	Required	Type	Value	Description
paymentMethod	M	String	string(2-50)	the payment method, e.g. "VISA", "MASTER", etc.

currency	M	String	string(3)	the currency of the payment method
paymentAmount	M	Number	float	the amount of using the payment method
changes	M	Number	float	the changes, if there is no changes, it should be set to 0

Invoice

The invoice field is to describe the invoice number generated by the merchant system.

Field name	**Required**	**Type**	**Value**	**Description**
invoiceNo	M	String	string(2-50)	Invoice number generated the merchant system

Request Sample

```
POST {{serviceRoot}}/api/s/receipt/v2/push-receipt
Content-Type: application/json
X-NoBill-API-KEY: cN0WU6ax8xA1NjSjJ6SKM3wIH7R028Bj
X-NoBill-API-Signature:
fb02bef8bfb1c7686a1579ccc4ecfe1f2dce4616bbec92581a5c43e187783bc1

{
  "version": 1,
  "merchantId": "MC01GM0C0WRQMCXM862ERTX91WCM",
  "merchantBranchId": "MBR01GM0C0WRQTAVJG0ZEY5GC9RJQ",
  "merchantReceiptId": "9642b4f0-8163-49ef-a004-0e0413c157fb",
  "customerEmail": "songzenghui@gmail.com",
  "#customerMobile": "+971585930810",
  "#customerId": "xxx",
  "createTimestamp": 1667356613,
  "transaction": {
    "txnNo": "101000392399323499",
    "txnDate": "2022-10-31",
    "txnTime": "10:24",
    "cashier": "John"
  },
  "purchase": {
    "quantity": 1,
    "items": [
      {
        "code": "0103823",
        "name1": "Lache, Dos Pinos",
        "name2": "Milk, Two Pine Trees",
        "quantity": 1,
        "price": 24.50,
        "currency": "CRC",
        "total": 24.50,
        "taxName": "VAT",
        "taxRate": 0.13,
        "discountTitle": "Summer Sale 40% Off",
        "discountAmount": 10.00
      }
    ],
    "amount": {
      "currency": "CRC",
      "total": 24.50,
      "discount": 0
    },
    "tax": {
      "currency": "CRC",
      "exemption": 0,
      "taxed": 21.68,
      "exoneration": 0,
      "total": 2.82
    }
  },
  "payment": {
    "currency": "CRC",
    "total": 24.50,
    "items": [
      {
        "paymentMethod": "VISA Credit Card",
        "currency": "AED",
        "paymentAmount": 24.50,
        "changes": 0
      }
    ]
  },
  "invoice": {
    "invoiceNo": "1234_5678"
  }
}
```

Appendix

Error Codes

The following error codes are written in Rust, and will be polished.

```rust
pub const SUCCESS_CODE: &'static str = "0";
pub const UNKNOWN_ERROR: &'static str = "-1";
pub const GATEWAY_DOWNSTREAM_FAILED: &'static str = "1000";
pub const GATEWAY_DOWNSTREAM_NOT_FOUND: &'static str = "1001";
pub const PROXY_DATA_STORE_ERROR: &'static str = "1500";
pub const PROXY_DATA_CACHE_ERROR: &'static str = "1501";
pub const PROXY_SECURITY_ERROR: &'static str = "1502";
pub const PROXY_DOWNSTREAM_ERROR: &'static str = "1503";
pub const COMM_MALFORMED_PROTOCOL: &'static str = "2000";
pub const COMM_LACK_OF_API_KEY: &'static str = "2001";
pub const COMM_UNKNOWN_API_KEY: &'static str = "2004";
pub const COMM_LACK_OF_MERCHANT_ID: &'static str = "2005";
pub const COMM_UNKNOWN_MERCHANT_ID: &'static str = "2006";
pub const COMM_LACK_OF_APP_NAME: &'static str = "2007";
pub const COMM_UNKNOWN_APP_NAME: &'static str = "2008";
pub const COMM_LACK_OF_API_SIGNATURE: &'static str = "2009";
pub const COMM_BAD_API_SIGNATURE: &'static str = "2010";
pub const COMM_LACK_OF_API_SECRET: &'static str = "2011";
pub const COMM_STORE_ERROR: &'static str = "2020";
pub const COMM_LACK_OF_PRIVILEGES: &'static str = "2021";
pub const COMM_DUPLICATED_DATA: &'static str = "2022";
pub const COMM_DIRTY_DATA: &'static str = "2023";
pub const COMM_DATA_STORE_ERROR: &'static str = "2024";
pub const COMM_SEC_ERROR: &'static str = "2025";
pub const COMM_RES_NOT_FOUND: &'static str = "2026";
pub const COMM_RES_OPEN_ERROR: &'static str = "2027";
pub const COMM_RES_READ_ERROR: &'static str = "2028";
pub const NOTIFY_CHANNEL_ERROR: &'static str = "30000";
pub const NOTIFY_TEMPLATE_ERROR: &'static str = "30001";
pub const NOTIFY_DATA_STORE_ERROR: &'static str = "30002";
pub const CUSTOMER_DATA_STORE_ERROR: &'static str = "40000";
pub const CUSTOMER_NOT_FOUND: &'static str = "40001";
pub const CUSTOMER_CREATION_NO_ARGS: &'static str = "40002";
pub const MERCHANT_NOT_FOUND: &'static str = "70000";
pub const MERCHANT_APP_NOT_FOUND: &'static str = "70001";
pub const MERCHANT_INACTIVE: &'static str = "70002";
pub const MERCHANT_FUND_NOT_FOUND: &'static str = "70003";
pub const MERCHANT_BRANCH_INACTIVE: &'static str = "70004";
pub const RECEIPT_VALIDATION_ERROR: &'static str = "80000";
pub const RECEIPT_NOT_FOUND: &'static str = "80001";
pub const RECEIPT_PURCHASE_NOT_FOUND: &'static str = "80002";
pub const RECEIPT_INVOICE_NOT_FOUND: &'static str = "80003";
pub const RECEIPT_MERCHANT_BIZ_NOT_FOUND: &'static str = "80004";
pub const RECEIPT_MERCHANT_BRANCH_NOT_FOUND: &'static str = "80005";
pub const RECEIPT_TRANSACTION_NOT_FOUND: &'static str = "80006";
pub const RECEIPT_PAYMENT_NOT_FOUND: &'static str = "80007";
```