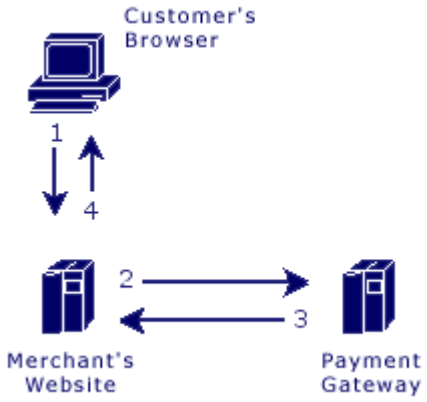
Methodology

Payment API

Transactions



Steps:

1. The customer sends their payment information to the merchant's web site.
2. The merchant web site posts the payment data to the Payment Gateway.
3. The Payment Gateway responds immediately with the results of the transactions.
4. The merchant web site displays the appropriate message to the customer.

The communication method used to send messages to the Payment Gateway's server is the standard HTTP protocol over an SSL connection.

In the Payment API method, the communications with the cardholder (Steps 1 and 4) are developed completely by the merchant and therefore are not defined by the Payment Gateway. Step 1 should simply collect the payment data from the cardholder and Step 4 should display the appropriate transaction receipt or declined message.

In Step 2, transaction details should be delivered to the Payment Gateway using the POST method with the appropriate variables defined below posted along with the request.

In Step 3, the transaction responses are returned in the body of the HTTP response in a query string name/value format delimited by ampersands. For example:
variable1=value1&variable2=value2&variable3=value3

Customer Vault

The Customer Vault was designed specifically for businesses of any size to address concerns about handling customer payment information. Visa and MasterCard have instituted the Payment Card Industry (PCI) Data Security to protect cardholder data, wherever it resides, ensuring that members, merchants, and service providers maintain the highest information security standards.

These associations have also deemed that merchants will be held liable for any breach of cardholder data. This has become a major concern for merchants who handle credit card or electronic check payments. The

Customer Vault is designed for these merchants who desire to avoid the tremendous costs and resources involved in becoming PCI compliant under these circumstances.

The Customer Vault does this by allowing merchants to transmit their payment information through a Secure Sockets Layer (SSL) connection for storage in our Level 1 PCI certified data facility. Once the customer record has been securely transmitted to the Customer Vault, the merchant can then initiate transactions remotely without having to access cardholder information directly. This process is accomplished without the merchant storing the customer's payment information in their local database or payment application.

Transaction Types

Payment API

Sale (sale)

Transaction sales are submitted and immediately flagged for settlement.

Authorization (auth)

Transaction authorizations are authorized immediately but are not flagged for settlement. These transactions must be flagged for settlement using the capture transaction type.

Capture (capture)

Transaction captures flag existing authorizations for settlement. Only authorizations can be captured. Captures can be submitted for an amount equal to or less than the original authorization.

Void (void)

Transaction voids will cancel an existing sale or captured authorization. In addition, non-captured authorizations can be voided to prevent any future capture. Voids can only occur if the transaction has not been settled.

Refund (refund)

Transaction refunds will reverse a previously settled or pending settlement transaction. If the transaction has not been settled, a transaction void can also reverse it.

Credit (credit)

Transaction credits apply an amount to the cardholder's card that was not originally processed through the Gateway. In most situations credits are disabled as transaction refunds should be used instead.

Validate (validate)

This action is used for doing an "Account Verification" on the cardholder's credit card without actually doing an authorization.

Update (update)

Transaction updates can be used to update previous transactions with specific order information, such as a tracking number and shipping carrier.

Transaction Variables

Payment API

POST URL

POST URL:	https://secure.nmi.com/api/transact.php

Sale/Authorization/Credit/Validate/Offline

Variable Name	Description
type*	The type of transaction to be processed. Values: 'sale', 'auth', 'credit', 'validate', or 'offline'
security_key*	API Security Key assigned to a merchant account. New keys can be generated from the merchant control panel in Settings > Security Keys
payment_token	The tokenized version of the customer's card or check information. This will be generated by Collect.js and is usable only once.
transaction_session_id‡‡‡‡	A single use session ID used by Kount to link the transaction and Data Collector information together. This ID should be generated every time a payment form is loaded by the cardholder, and be random/unpredictable (do not use sequential IDs). This ID should not be reused within a 30 day period. This can be used with Collect.js or the Payment API when using the Kount DDC with Gateway.js. Format: alphanumeric, 32 characters required
googlepay_payment_data	The encrypted token created when integration directly to the Google Pay SDK.
ccnumber**	Credit card number.
ccexp**	Credit card expiration date. Format: MMYY
cvv	The card security code. While this is not required, it is strongly recommended.
checkname***	The name on the customer's ACH account.
checkaba***	The customer's bank routing number.
checkaccount***	The customer's bank account number.
account_holder_type	The type of ACH account the customer has. Values: 'business' or 'personal'
account_type	The ACH account entity of the customer. Values: 'checking' or 'savings'
sec_code	The Standard Entry Class code of the ACH transaction. Values: 'PPD', 'WEB', 'TEL', or 'CCD'
amount	Total amount to be charged. For validate, the amount must be omitted or set to 0.00. Format: x.xx
surcharge	Surcharge amount. Format: x.xx

cash_discount	How much less a customer paid due to a cash discount. Format: x.xx, only applicable to cash and check transactions
tip	The final tip amount, included in the transaction, associated with the purchase Format: x.xx
currency	The transaction currency. Format: ISO 4217
payment***	The type of payment. Default: 'creditcard' Values: 'creditcard', 'check', or 'cash'
processor_id	If using Multiple MIDs, route to this processor (processor_id is obtained under Settings ? Transaction Routing in the Control Panel).
authorization_code‡	Specify authorization code. For use with "offline" action only.
dup_seconds	Sets the time in seconds for duplicate transaction checking on supported processors. Set to 0 to disable duplicate checking. This value should not exceed 7862400.
descriptor	Set payment descriptor on supported processors.
descriptor_phone	Set payment descriptor phone on supported processors.
descriptor_address	Set payment descriptor address on supported processors.
descriptor_city	Set payment descriptor city on supported processors.
descriptor_state	Set payment descriptor state on supported processors.
descriptor_postal	Set payment descriptor postal code on supported processors.
descriptor_country	Set payment descriptor country on supported processors.
descriptor_mcc	Set payment descriptor mcc on supported processors.
descriptor_merchant_id	Set payment descriptor merchant id on supported processors.
descriptor_url	Set payment descriptor url on supported processors.
billing_method	Should be set to 'recurring' to mark payment as a recurring transaction or 'installment' to mark payment as an installment transaction. Values: 'recurring', 'installment'
billing_number	Specify installment billing number, on supported processors. For use when "billing_method" is set to installment. Values: 0-99
billing_total	Specify installment billing total on supported processors. For use when "billing_method" is set to installment.
order_template	Order template ID.
order_description	Order description. Legacy variable includes: orderdescription
orderid	Order Id
ipaddress	IP address of cardholder, this field is recommended. Format: xxx.xxx.xxx.xxx
tax****	The sales tax included in the transaction amount associated with the purchase. Setting tax equal to any negative value indicates an order that is exempt from sales tax. Default: '0.00' Format: x.xx
shipping****	Total shipping amount.

ponumber****	Original purchase order.
first_name	Cardholder's first name. Legacy variable includes: firstname
last_name	Cardholder's last name Legacy variable includes: lastname
company	Cardholder's company
address1	Card billing address
address2	Card billing address, line 2
city	Card billing city
state	Card billing state. Format: CC
zip	Card billing zip code
country	Card billing country. Country codes are as shown in ISO 3166. Format: CC
phone	Billing phone number
fax	Billing fax number
email	Billing email address
social_security_number	Customer's social security number, checked against bad check writers database if check verification is enabled.
drivers_license_number	Driver's license number.
drivers_license_dob	Driver's license date of birth.
drivers_license_state	The state that issued the customer's driver's license.
shipping_firstname	Shipping first name
shipping_lastname	Shipping last name
shipping_company	Shipping company
shipping_address1	Shipping address
shipping_address2	Shipping address, line 2
shipping_city	Shipping city
shipping_state	Shipping state Format: CC
shipping_zip	Shipping zip code
shipping_country	Shipping country Country codes are as shown in ISO 3166. Format: CC
shipping_email	Shipping email address
merchant_defined_field_#	You can pass custom information in up to 20 fields. Format: merchant_defined_field_1=Value
customer_receipt	If set to true, when the customer is charged, they will be sent a transaction receipt. Values: 'true' or 'false'
signature_image	Cardholder signature image. For use with "sale" and "auth" actions only. Format: base64 encoded raw PNG image. (16kiB maximum)
cardholder_auth‡‡	Set 3D Secure condition. Value used to determine E-commerce indicator (ECI). Values: 'verified' or 'attempted'

cavv‡‡	Cardholder authentication verification value. Format: base64 encoded
xid‡‡	Cardholder authentication transaction id. Format: base64 encoded
three_ds_version‡‡	3DSecure version. Examples: "2.0.0" or "2.2.0"
directory_server_id	Directory Server Transaction ID. May be provided as part of 3DSecure 2.0 authentication. Format: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
source_transaction_id	Specifies a payment gateway transaction id in order to associate payment information with a Subscription or Customer Vault record. Must be set with a 'recurring' or 'customer_vault' action.
pinless_debit_override	Set to 'Y' if you have Pinless Debit Conversion enabled but want to opt out for this transaction. Feature applies to selected processors only.
Recurring specific fields	
recurring	Recurring action to be processed. Values: add_subscription
plan_id	Create a subscription tied to a Plan ID if the sale/auth transaction is successful.
plan_payments	The number of payments before the recurring plan is complete. Note: Use '0' for 'until canceled'
plan_amount	The plan amount to be charged each billing cycle. Format: x.xx
day_frequency	How often, in days, to charge the customer. Cannot be set with 'month_frequency' or 'day_of_month'.
month_frequency	How often, in months, to charge the customer. Cannot be set with 'day_frequency'. Must be set with 'day_of_month'. Values: 1 through 24
day_of_month	The day that the customer will be charged. Cannot be set with 'day_frequency'. Must be set with 'month_frequency'. Values: 1 through 31 - for months without 29, 30, or 31 days, the charge will be on the last day
start_date	The first day that the customer will be charged. Format: YYYYMMDD
Customer Vault specific fields	
customer_vault	Associate payment information with a Customer Vault record if the transaction is successful. Values: 'add_customer' or 'update_customer'
customer_vault_id	Specifies a Customer Vault id. If not set, the payment gateway will randomly generate a Customer Vault id.
Stored Credentials (CIT/MIT)	
initiated_by	Who initiated the transaction. Values: 'customer' or 'merchant'
initial_transaction_id	Original payment gateway transaction id.

stored_credential_indicator	The indicator of the stored credential. Values: 'stored' or 'used' Use **'stored'** when processing the initial transaction in which you are storing a customer's payment details (customer credentials) in the Customer Vault or other third-party payment storage system. Use **'used'** when processing a subsequent or follow-up transaction using the customer payment details (customer credentials) you have already stored to the Customer Vault or third-party payment storage method.
Level III specific order fields	
shipping†	Freight or shipping amount included in the transaction amount. Default: '0.00' Format: x.xx
tax†	The sales tax, included in the transaction amount, associated with the purchase. Setting tax equal to any negative value indicates an order that is exempt from sales tax. Default: '0.00' Format: x.xx
ponumber†	Purchase order number supplied by cardholder
orderid†	Identifier assigned by the merchant. This defaults to gateway transaction id.
shipping_country†	Shipping country (e.g. US) Format: CC
shipping_postal†	Postal/ZIP code of the address where purchased goods will be delivered. This field can be identical to the 'ship_from_postal' if the customer is present and takes immediate possession of the goods.
ship_from_postal†	Postal/ZIP code of the address from where purchased goods are being shipped, defaults to merchant profile postal code.
summary_commodity_code†	4 character international description code of the overall goods or services being supplied. The acquirer or processor will provide a list of current codes.
duty_amount	Amount included in the transaction amount associated with the import of purchased goods. Default: '0.00' Format: x.xx
discount_amount	Amount included in the transaction amount of any discount applied to complete order by the merchant. Default: '0.00' Format: x.xx
national_tax_amount	The national tax amount included in the transaction amount. Default: '0.00' Format: x.xx
alternate_tax_amount	Second tax amount included in the transaction amount in countries where more than one type of tax can be applied to the purchases. Default: '0.00' Format: x.xx
alternate_tax_id	Tax identification number of the merchant that reported the alternate tax amount.

vat_tax_amount	Contains the amount of any value added taxes which can be associated with the purchased item. Default: '0.00' Format: x.xx
vat_tax_rate	Contains the tax rate used to calculate the sales tax amount appearing. Can contain up to 2 decimal places, e.g. 1% = 1.00. Default: '0.00' Format: x.xx
vat_invoice_reference_number	Invoice number that is associated with the VAT invoice.
customer_vat_registration	Value added tax registration number supplied by the cardholder.
merchant_vat_registration	Government assigned tax identification number of the merchant for whom the goods or services were purchased from.
order_date	Purchase order date, defaults to the date of the transaction. Format: YYMMDD
Level III specific line item detail fields	
item_product_code_#†	Merchant defined description code of the item being purchased.
item_description_#†	Description of the item(s) being supplied.
item_commodity_code_#†	International description code of the individual good or service being supplied. The acquirer or processor will provide a list of current codes.
item_unit_of_measure_#†	Code for units of measurement as used in international trade. Default: 'EACH'
item_unit_cost_#†	Unit cost of item purchased, may contain up to 4 decimal places.
item_quantity_#†	Quantity of the item(s) being purchased. Default: '1'
item_total_amount_#†	Purchase amount associated with the item. Defaults to: 'item_unit_cost_#' x 'item_quantity_#' rounded to the nearest penny.
item_tax_amount_#†	Amount of sales tax on specific item. Amount should not be included in 'total_amount_#'. Default: '0.00' Format: x.xx
item_tax_rate_#†	Percentage representing the value-added tax applied. Default: '0.00'
item_discount_amount_#	Discount amount which can have been applied by the merchant on the sale of the specific item. Amount should not be included in 'total_amount_#'.
item_discount_rate_#	Discount rate for the line item. 1% = 1.00. Default: '0.00'
item_tax_type_#	Type of value-added taxes that are being used.
item_alternate_tax_id_#	Tax identification number of the merchant that reported the alternate tax amount.
Payment Facilitator Specific Fields	
payment_facilitator_id‡‡‡	Payment Facilitator/Aggregator/ISO's ID Number
submerchant_id‡‡‡	Sub-merchant Account ID
submerchant_name‡‡‡	Sub-merchant's Name

submerchant_address‡‡‡	Sub-merchant's Address
submerchant_city‡‡‡	Sub-merchant's City
submerchant_state‡‡‡	Sub-merchant's State
submerchant_postal‡‡‡	Sub-merchant's Zip/Postal Code
submerchant_country‡‡‡	Sub-merchant's Country
submerchant_phone‡‡‡	Sub-merchant's Phone Number
submerchant_email‡‡‡	Sub-merchant's Email Address

* Always required

** Required for credit card transactions

*** Required for ACH transactions

**** Required for Level 2 transactions

† Required for Level 3 transactions

‡ Required for offline transactions

‡‡ Required for 3D Secure transactions

‡‡‡ Required fields for Payment Facilitator enabled transactions vary by card brand

‡‡‡‡ Required for API transactions using the Kount Service

Notes:

- Level II fields are required for Level II processing.
- Level II and Level III fields are required for Level III processing.
- You can pass only credit card **or** e-check transaction variables in a request, not both in the same request.
- Certain banks may require some optional fields.
- Some characters output from base64 encoding can not be passed directly into the API (i.e. "+") so ensure these fields are also properly URL encoded.

Capture

Variable Name	Description
type*	Type of transaction. Values: 'capture'
security_key*	API Security Key assigned to a merchant account. New keys can be generated from the merchant control panel in Settings > Security Keys
transactionid*	Original payment gateway transaction id
amount*	Total amount to be settled. This amount must be equal to or less than the original authorized amount. Format: x.xx
tracking_number	Shipping tracking number
shipping_carrier	Shipping carrier. Values: 'ups', 'fedex', 'dhl', or 'usps'
orderid	Order id.
signature_image	Cardholder signature image. Format: base64 encoded raw PNG image. (16kiB maximum)

* Always required

Void

Variable Name	Description
type*	Type of transaction. Values: 'void'
security_key*	API Security Key assigned to a merchant account. New keys can be generated from the merchant control panel in Settings > Security Keys
transactionid*	Original payment gateway transaction id
void_reason**	Reason the EMV transaction is being voided. Values: 'fraud', 'user_cancel', 'icc_rejected', 'icc_card_removed', 'icc_no_confirmation', or 'pos_timeout'
payment***	The type of payment. Default: 'creditcard' Values: 'creditcard' or 'check'

* Always required

** Conditionally required for EMV transactions

*** Required for ACH transactions

Refund

Variable Name	Description
type*	Type of transaction. Values: 'refund'
security_key*	API Security Key assigned to a merchant account. New keys can be generated from the merchant control panel in Settings > Security Keys
transactionid*	Original payment gateway transaction id
amount	Total amount to be refunded. This amount may be equal to or less than the settled amount. Setting the amount to 0.00 will refund the entire amount. Format: x.xx
payment**	The type of payment. Default: 'creditcard' Values: 'creditcard' or 'check'

* Always required

** Required for ACH transactions

Update

Variable Name	Description
type*	Type of transactions. Values: 'update'
security_key*	API Security Key assigned to a merchant account. New keys can be generated from the merchant control panel in Settings > Security Keys
transactionid*	Original payment gateway transaction id
payment**	The type of payment. Default: 'creditcard' Values: 'creditcard' or 'check'
tracking_number	Shipping tracking number
shipping	Total shipping amount. Format: x.xx

shipping_postal	Postal/ZIP code of the address where purchased goods will be delivered. This field can be identical to the 'ship_from_postal' if the customer is present and takes immediate possession of the goods.
ship_from_postal	Postal/ZIP code of the address from where purchased goods are being shipped, defaults to merchant profile postal code.
shipping_country	Shipping Country Code.
shipping_carrier	Shipping carrier. Values: 'ups', 'fedex', 'dhl', or 'usps'
shipping_date	Shipping date. Format: YYYYMMDD
order_description	Order Description. Legacy variable includes: orderdescription
order_date	Order date. Format: YYYYMMDD
customer_receipt	If set to true, when the customer is charged, they will be sent a transaction receipt. Values: 'true' or 'false'
signature_image	Cardholder signature image. Format: base64 encoded raw PNG image. (16kiB maximum)
ponumber	Cardholder's purchase order number.
summary_commodity_code	4 character international description code of the overall goods or services being supplied. The acquirer or processor will provide a list of current codes.
duty_amount	Amount included in the transaction amount associated with the import of purchased goods. Format: x.xx
discount_amount	Amount included in the transaction amount of any discount applied to complete order by the merchant. Format: x.xx
tax	The sales tax, included in the transaction amount, associated with the purchase. Setting tax equal to any negative value indicates an order that is exempt from sales tax. Default: '0.00' Format: x.xx
national_tax_amount	The national tax amount included in the transaction amount. Format: x.xx
alternate_tax_amount	Second tax amount included in the transaction amount in countries where more than one type of tax can be applied to the purchases. Format: x.xx
alternate_tax_id	Tax identification number of the merchant that reported the alternate tax amount.
vat_tax_amount	Contains the amount of any value added taxes which can be associated with the purchased item.
vat_tax_rate	Contains the tax rate used to calculate the sales tax amount appearing. Can contain up to 2 decimal places, e.g. 1% = 1.00.
vat_invoice_reference_number	Invoice number that is associated with the VAT invoice.
customer_vat_registration	Value added tax registration number supplied by the cardholder.

merchant_vat_registration	Government assigned tax identification number of the merchant for whom the goods or services were purchased from.
merchant_defined_field_#	Merchant Defined Fields. Format: merchant_defined_field_1=Value

* Always required

** Required for ACH transactions

Invoicing Variables

Payment API

POST URL

POST URL:	https://secure.nmi.com/api/transact.php

Create Invoice

Variable Name	Description
invoicing*	Create a new invoice and email it to the customer. Values: 'add_invoice'
security_key*	API Security Key assigned to a merchant account. New keys can be generated from the merchant control panel in Settings > Security Keys
amount*	Total amount to be invoiced. Must be greater than 0.00. Format: x.xx
email*	Billing email address An invoice will be sent to this address when it is created.
payment_terms	When the invoice should be paid Default: 'upon_receipt' Values: 'upon_receipt', or integers from 0-999.
payment_methods_allowed	What payment methods a customer may use when paying invoice. Defaults to all available payment methods available in your merchant account Values: 'cc', 'ck', and 'cs'. Multiple payment types can be selected by comma-separating values.
processor_id	If using Multiple MIDs, route to this processor (processor_id is obtained under Settings ? Transaction Routing in the Control Panel). If allowing multiple payment types, one processor_id can be selected per payment type by submitting comma-separated values.
currency	The transaction currency. Format: ISO 4217
order_description	Order description. Legacy variable includes: orderdescription
orderid	Order ID.
customer_id	Customer ID.
customer_tax_id	Customer Tax ID.
tax	Total sales tax amount.
shipping	Total shipping amount.
ponumber	Original purchase order.
first_name	Cardholder's first name. Legacy variable includes: firstname
last_name	Cardholder's last name. Legacy variable includes: lastname
company	Cardholder's company.

address1	Card billing address.
address2	Card billing address, line 2.
city	Card billing city.
state	Card billing state. Format: CC
zip	Card billing zip code.
country	Card billing country. Country codes are as shown in ISO 3166. Format: CC
phone	Billing phone number.
fax	Billing fax number.
website	Customer website.
shipping_firstname	Shipping first name.
shipping_lastname	Shipping last name.
shipping_company	Shipping company.
shipping_address1	Shipping address.
shipping_address2	Shipping address, line 2.
shipping_city	Shipping city.
shipping_state	Shipping state. Format: CC
shipping_zip	Shipping zip code.
shipping_country	Shipping country. Country codes are as shown in ISO 3166. Format: CC
shipping_email	Shipping email address.
merchant_defined_field_#	You can pass custom information in up to 20 fields. Format: merchant_defined_field_1=Value
Product Information	
item_product_code_#	Merchant defined description code of the item being purchased.
item_description_#	Description of the item(s) being supplied.
item_commodity_code_#	International description code of the individual good or service being supplied. The acquirer or processor will provide a list of current codes.
item_unit_of_measure_#	Code for units of measurement as used in international trade. Default: 'EACH'
item_unit_cost_#	Unit cost of item purchased, may contain up to 4 decimal places.
item_quantity_#	Quantity of the item(s) being purchased. Default: '1'
item_total_amount_#	Purchase amount associated with the item. Defaults to: 'item_unit_cost_#' x 'item_quantity_#' rounded to the nearest penny.
item_tax_amount_#	Amount of sales tax on specific item. Amount should not be included in 'total_amount_#'. Default: '0.00' Format: x.xx

item_tax_rate_#	Percentage representing the value-added tax applied. Default: '0.00'
item_discount_amount_#	Discount amount which can have been applied by the merchant on the sale of the specific item. Amount should not be included in 'total_amount_#'.
item_discount_rate_#	Discount rate for the line item. 1% = 1.00. Default: '0.00'
item_tax_type_#	Type of value-added taxes that are being used.
item_alternate_tax_id_#	Tax identification number of the merchant that reported the alternate tax amount.

* Always required

Update Invoice

Variable Name	Description
invoicing*	Update an existing invoice. Values: 'update_invoice'
security_key*	API Security Key assigned to a merchant account. New keys can be generated from the merchant control panel in Settings > Security Keys
invoice_id*	The invoice ID to be updated.

* Always required

Notes:

All variables (besides currency) on an invoice may be updated. Updating an invoice will not result in a new invoice being sent to the customer. To send the invoice after updating an invoice, use the send_invoice request after making changes.

Send Invoice

Variable Name	Description
invoicing*	Send an existing invoice to a customer. Values: 'send_invoice'
security_key*	API Security Key assigned to a merchant account. New keys can be generated from the merchant control panel in Settings > Security Keys
invoice_id*	The invoice ID to be emailed.

* Always required

Notes:

The invoice will be sent to the billing email address assigned to the invoice.

Close Invoice

Variable Name	Description
invoicing*	The invoice to be closed. Values: 'close_invoice'
security_key*	API Security Key assigned to a merchant account. New keys can be generated from the merchant control panel in Settings > Security Keys
invoice_id*	The invoice ID to be closed.

* Always required

Retail Data

Payment API

Passing Unencrypted Retail Magnetic Stripe Data

Variable Name	Description
track_1	Raw Magnetic Stripe Data
track_2	Raw Magnetic Stripe Data
track_3	Raw Magnetic Stripe Data

Passing MagTek Magensa Encrypted Magnetic Stripe Data

Variable Name	Description
magnesafe_track_1	Raw MagTek Magensa Data
magnesafe_track_2	Raw MagTek Magensa Data
magnesafe_magneprint	Raw MagTek Magensa Data
magnesafe_ksn	Raw MagTek Magensa Data
magnesafe_magneprint_status	Raw MagTek Magensa Data

Passing IDTech M130 Encrypted Swipe Data

Variable Name	Description
encrypted_track_1	Raw encrypted data
encrypted_track_2	Raw encrypted data
encrypted_track_3	Raw encrypted data
encrypted_ksn	Raw encrypted data

Passing IDTech M130 Encrypted Keyed Data

Variable Name	Description
encrypted_data	Raw encrypted data

Passing Ingenico Telium 2 Chip Card Data

Variable Name	Description
entry_mode	The type of transaction data to be processed. Value: 'emv_icc'
emv_auth_request_data	EMV Data for the transaction as received from the EMV Chip Card SDK.
emv_device	The EMV - capable card reader. Value: 'ingenico_rba'
verification_method	Method used to verify the EMV transaction. Values: 'signature', 'offline_pin', 'offline_pin_signature', or 'none'
encrypted_ksn	Raw encrypted data
encrypted_track_2	Raw encrypted data

Passing Ingenico Telium 2 Swipe Data

Variable Name	Description
entry_mode	The type of transaction data to be processed. Values: 'swiped' or 'swiped_emv_fallback'
emv_device	The EMV - capable card reader. Value: 'ingenico_rba'
encrypted_ksn	Raw encrypted data
encrypted_track_1	Raw encrypted data
encrypted_track_2	Raw encrypted data

Passing Ingenico Telium 2 NFC Data

Variable Name	Description
entry_mode	The type of transaction data to be processed. Value: 'nfc_msd'
emv_device	The EMV - capable card reader. Value: 'ingenico_rba'
encrypted_ksn	Raw encrypted data
encrypted_track_2	Raw encrypted data

Passing Ingenico Telium 2 Keyed Data

Variable Name	Description
entry_mode	The type of transaction data to be processed. Value: 'keyed'
emv_device	The EMV - capable card reader. Value: 'ingenico_rba'
encrypted_ksn	Raw encrypted data
encrypted_track_2	Raw encrypted data

Apple Pay

Payment API

This documentation is intended for Apple Pay in **iOS apps**. For information about using Apple Pay on the web see Apple Pay with Collect.js

Supported Processors

Apple Pay supports Global Payments East - EMV, First Data Nashville, Chase Paymentech Salem, Chase Paymentech Tampa, EPX, Vantiv Now Worldpay eCommerce - Host Capture (Litle & Co), Global Payments Canada, First Data Nashville North, Vantiv Now Worldpay Core - Terminal Capture, Paymentech Salem Dev, Vantiv Now Worldpay eCommerce - Terminal Capture (Litle & Co.), First Data Nashville North V2, FACe - Vantiv Pre-Live, FACe - Vantiv, First Data Compass, TSYS - EMV, FACe - Vantiv (Next Day Funding), Elavon viaConex, Elavon EISOP UK/EU - EMV, American Express Direct UK/EU - EMV, Credorax ePower EU - EMV, Worldpay APACS UK/EU - EMV, First Data APACS UK/EU - EMV, Lloyds Cardnet APACS UK/EU - EMV, Barclaycard HISO UK/EU - EMV, AIBMS APACS UK/EU - EMV, Global Payments APACS UK/EU - EMV, Checkout.com Unified Payments, eMerchantPay Genesis and NMI Payments processors configured for e-commerce.

Configuring Apple Pay

Creating an Apple Merchant ID

First, you must obtain an Apple Merchant ID before you can generate the Certificate Signing Request that Apple requires. You will need to set up an Apple Merchant ID in your iOS Developer Account. Follow these steps to complete the setup:

1. Go to Apple's Developer Portal and log in to the Member Center to create a new Merchant ID.
2. Navigate to the Certificates, Identifiers, and Profiles area of the Member Center, and then begin the Register Merchant ID process.
3. You must then set the Apple Merchant ID within your gateway Control Panel under Settings -> Apple Pay.

Generating the Certificate Signing Request

Next, you will need to associate a Certificate with the Merchant ID in Apple's Developer Portal. After downloading the Certificate Signing Request from the gateway's options page, follow these steps.

1. In Apple's Developer Portal, click on the Merchant ID and then click "Edit".
2. Click "Create Certificate".
3. You are obtaining a CSR file from a Payment Provider so you will not have to create one. Click "Continue" to proceed to the upload page.
4. Click "Choose File..." and select the Gateway.certSigningRequest file you downloaded from the gateway's options page.

How to Obtain Apple Pay Payment Data

PassKit provides the payment data in the *(PKPayment *)payment* that is returned to your app's *paymentAuthorizationViewController:didAuthorizePayment:completion* method. The Apple Pay encrypted payment data is found in *payment.token.paymentData*.

payment.token.paymentData is a binary (NSData) object, so you must encode it as a hexadecimal string before

it can be passed to the Gateway.

Passing Apple Pay Payment Data

To submit a payment with Apple Pay, send the encrypted token data into the applepay_payment_data variable. There is no need to decrypt the data in your app. Only the Gateway will have access to the private key that can decrypt the token.

Notes

When passing in applepay_payment_data, you should not include the variables ccnumber or ccexp; they are extracted from the token data.

Important Note: The authorization amount must match the amount the customer approves in the app. If you pass in a currency, that must also match the currency approved in the app. If omitted, the currency from the app is used.

For working example code, including how to obtain the PKPayment object and how to pass a simple transaction to the Gateway, download the sample project.

Variables

Variable Name	Description
applepay_payment_data	The encrypted Apple Pay payment data (payment.token.paymentData) from PassKit encoded as a hexadecimal string

Troubleshooting

If you receive the error "Failed to decrypt Apple Pay data. Ensure that the Apple Pay Merchant ID is correct in the Gateway Settings and that the certificate was generated from a Gateway Certificate Signing Request.", try these steps:

1. Verify that the Merchant ID that Apple has in the developer portal exactly matches the Merchant ID in the Gateway's settings.
2. Verify that your app's PKPaymentRequest's merchantIdentifier exactly matches the Merchant ID in the Gateway's settings.
3. Ensure that the correct Merchant ID is checked in the Apple Pay section of the Capabilities tab in your project's target settings.
4. Try creating a new Merchant ID. Reusing an existing Merchant ID with a new certificate may sometimes cause issues with encryption.

Google Pay

This documentation is intended for Google Pay in **Android apps**. For information about using Google Pay on the web see Google Pay with Collect.js

Google Pay allows your customers to submit payment data via a payment method they trust. This also enables you to collect their payment information in a tokenized form so that the plain text credit card information never touches your environment.

Set Up

Google Pay documentation can be found here. Those documents are maintained by Google and will be kept up to date with any changes and enhancements to the Google Pay SDK.

When setting up your merchant account in the Google Pay Business Console, Google will ask if you are doing a "Direct" or "Gateway" integration; you should select "Gateway." The SDK will ask you to provide "gateway" and "gatewayMerchantId" values.

gateway	"gatewayservices"
gatewayMerchantId	Your "Gateway ID" as listed in the Account Information settings page of the merchant portal.

Overview

While Google's documentation will have specific details for using the Google Pay SDK, this is an overview of what the integration process and user flow should look like.

1. Your application will render the Google Pay button using libraries provided by Google Mobile Services
2. If the customer taps on the Google Pay button, the Google Pay SDK will open an interface for the customer to select their payment information.
3. The user confirms their payment information, which closes the interface and returns the user to your app.
4. Your app will receive a tokenized version of the customer's payment information from the Google Pay SDK.
5. That tokenized payload should be passed to the gateway for decryption as a part of your Payment API request.

Variable Name	Description
googlepay_payment_data	The encrypted token created by the Google Pay SDK.
Example Input	

{"signature":"MEYCIQCkKULRjzfFR/WlREB8ZoFRFCpkhWBHqQy+LIHvICls0QIhAOi09iSGudeQslZCQ8qM26rzIZ1BJvfr1T+wXnhF/85S","protocolVersion":"ECv1","signedMessage":"{\"encryptedMessage\":\"uUh4kCxpIGivV4U3nXb5uxfFXy7GxYNHkdj2KZCtMridD2TKzdhSbllUrDuQuo4jxsqA4kQDMBBkVKA8yYrdD5/3pQnaQubXLBBVGlOS40uhc2YSTHnTFAbVVk5IyM1FVQocBMXpIDhFgtFwo4TYlT1x8HjNzscOF/A1o6WXkslRhbfX/z9EyaIWpz2jeghM4fM4Exd8Re3b6ZPsjrkF5PQSE2opEqppvkaoah7qRdfg0YeaAjbFpsTJ+D0GTEWirll/T+cqBJT86iL6L6RZkEYkVX5IF3pmgWLqYFUY4g9Oe4rWrRqT9/0sRFcBe7ClR1ojYIZCuz50ISpBG5bCMfo/jFX2br32JEo1Qz/xEZrzNG05gbalQx6daNYiW7kcqcMwD6PDcxrHPe1nNV88ByY5zjC6wIwuP0kDdvMEHSb8qrpoBI/K0u6ICuo\\u003d\",\"ephemeralPublicKey\":\"BFC+ibo/f3HUmz8t3OBPuOUF17WOb3/aUgw4iAQFlPU2ovLtJijwJ68lHbhXsoKt3NKCVS5qRqcSzM0uyDCr0fY\\u003d\",\"tag\":\"1R4JBFAxVfw9PNTgI5FXEWFXbSXEdPWIEYkqg/AnyQk\\u003d\"}"}

In the event that there is data in the Payment API request and the Google Pay token, such as the customer's zip code, the gateway will prefer the value in the Google Pay token, as that is what the customer explicitly provided.

Recurring Variables

Payment API

POST URL

POST URL:	https://secure.nmi.com/api/transact.php

Add a Plan

Variable Name	Description
recurring*	Add a recurring plan that subscriptions can be added to in the future. Value: 'add_plan'
plan_payments*	The number of payments before the recurring plan is complete. Notes: '0' for until canceled
plan_amount*	The plan amount to be charged each billing cycle. Format: x.xx
plan_name*	The display name of the plan.
plan_id*	The unique plan ID that references only this recurring plan.
day_frequency**	How often, in days, to charge the customer. Cannot be set with 'month_frequency' or 'day_of_month'.
month_frequency***	How often, in months, to charge the customer. Cannot be set with 'day_frequency'. Must be set with 'day_of_month'. Values: 1 through 24
day_of_month***	The day that the customer will be charged. Cannot be set with 'day_frequency'. Must be set with 'month_frequency'. Values: 1 through 31 - for months without 29, 30, or 31 days, the charge will be on the last day

* Always required

** Required unless 'month_frequency' and 'day_of_month' is set.

*** Required unless 'day_frequency' is set.

Edit a Plan

Variable Name	Description
recurring*	Edit an existing recurring plan. Value: 'edit_plan' - Be careful when editing an existing plan, as all customers signed up for this plan will have their billing changed based on your edits.
current_plan_id*	Only relevant for editing an existing plan, the value will be the 'plan_id' that will be edited in this request.
plan_payments	The number of payments before the recurring plan is complete. Notes: '0' for until canceled
plan_amount	The plan amount to be charged each billing cycle. Format: x.xx
plan_name	The display name of the plan.
plan_id	The unique plan ID that references only this recurring plan.
day_frequency**	How often, in days, to charge the customer. Cannot be set with 'month_frequency' or 'day_of_month'.

month_frequency***	How often, in months, to charge the customer. Cannot be set with 'day_frequency'. Must be set with 'day_of_month'. Values: 1 through 24
day_of_month***	The day that the customer will be charged. Cannot be set with 'day_frequency'. Must be set with 'month_frequency'. Values: 1 through 31 - for months without 29, 30, or 31 days, the charge will be on the last day

* Always required

** Required unless 'month_frequency' and 'day_of_month' is set.

*** Required unless 'day_frequency' is set.

Add a Subscription to an Existing Plan

Variable Name	Description
recurring*	Associate payment information with a recurring plan. Value: add_subscription
plan_id*	The plan ID of the plan that the subscription will be associated with.
start_date	The first day that the customer will be charged. Format: YYYYMMDD
payment_token	The tokenized version of the customer's card or check information. This will be generated by Collect.js and is usable only once.
googlepay_payment_data	The encrypted token created when integration directly to the Google Pay SDK.
ccnumber**	Credit card number.
ccexp**	Credit card expiration. Format: MMYY
payment***	The type of payment. Default: 'creditcard' Values: 'creditcard' or 'check'
checkname***	The name on the customer's ACH account.
checkaccount***	The customer's bank account number.
checkaba***	The customer's bank routing number.
account_type	The customer's ACH account type. Values: 'checking' or 'savings'
currency	Set transaction currency.
account_holder_type	The customer's ACH account entity. Values: 'personal' or 'business'
sec_code	ACH standard entry class codes. Values: 'PPD', 'WEB', 'TEL', or 'CCD'
first_name	Cardholder's first name. Legacy variable includes: firstname
last_name	Cardholder's last name. Legacy variable includes: lastname
address1	Card billing address.
city	Card billing city
state	Card billing state.
zip	Card billing postal code.

country	Card billing country code.
phone	Billing phone number.
email	Billing email address.
company	Cardholder's company.
address2	Card billing address, line 2.
fax	Billing fax number.
orderid	Order ID
order_description	Order Description
merchant_defined_field_#	Can be set up in merchant control panel under 'Settings'->'Merchant Defined Fields'.
ponumber	Cardholder's purchase order number.
processor_id	If using Multiple MIDs, route to this processor (processor_id is obtained under Settings->Transaction Routing in the Control Panel).
customer_receipt	If set to true, when the customer is charged, they will be sent a transaction receipt. Values: 'true' or 'false'
source_transaction_id	Specifies a payment gateway transaction id in order to associate payment information with a Subscription record.
acu_enabled	If set to true, credit card will be evaluated and sent based upon Automatic Card Updater settings. If set to false, credit card will not be submitted for updates when Automatic Card Updater runs. Default: 'true' Values: 'true' or 'false'

* Always required

** Required for credit card transactions

*** Required for ACH transactions

Adding a Custom Subscription

Variable Name	Description
recurring*	Add a custom recurring subscription that is NOT associated with an existing plan Value: 'add_subscription'
plan_payments*	The number of payments before the recurring plan is complete. Notes: '0' for until canceled
plan_amount*	The plan amount to be charged each billing cycle. Format: x.xx
day_frequency**	How often, in days, to charge the customer. Cannot be set with 'month_frequency' or 'day_of_month'.
month_frequency***	How often, in months, to charge the customer. Cannot be set with 'day_frequency'. Must be set with 'day_of_month'. Values: 1 through 24
day_of_month***	The day that the customer will be charged. Cannot be set with 'day_frequency'. Must be set with 'month_frequency'. Values: 1 through 31 - for months without 29, 30, or 31 days, the charge will be on the last day

start_date	The first day that the customer will be charged. Format: YYYYMMDD
payment_token	The tokenized version of the customer's card or check information. This will be generated by Collect.js and is usable only once.
googlepay_payment_data	The encrypted token created when integration directly to the Google Pay SDK.
ccnumber****	Credit card number.
ccexp****	Credit card expiration. Format: MMYY
payment†	The type of payment. Default: 'creditcard' Values: 'creditcard' or 'check'
checkname†	The name on the customer's ACH account.
checkaccount†	The customer's bank account number.
checkaba†	The customer's bank routing number.
account_type	The customer's ACH account type. Values: 'checking' or 'savings'
account_holder_type	The customer's ACH account entity. Values: 'personal' or 'business'
sec_code	ACH standard entry class codes. Values: 'PPD', 'WEB', 'TEL', or 'CCD'
first_name	Cardholder's first name. Legacy variable includes: firstname
last_name	Cardholder's last name. Legacy variable includes: lastname
address1	Card billing address.
city	Card billing city
state	Card billing state.
zip	Card billing postal code.
country	Card billing country code.
phone	Billing phone number.
email	Billing email address.
company	Cardholder's company.
address2	Card billing address, line 2.
fax	Billing fax number.
orderid	Order ID
order_description	Order Description Legacy variable includes: orderdescription
merchant_defined_field_#	Can be set up in merchant control panel under 'Settings'->'Merchant Defined Fields'.
ponumber	Cardholder's purchase order number.
processor_id	If using Multiple MIDs, route to this processor (processor_id is obtained under Settings->Transaction Routing in the Control Panel).

customer_receipt	If set to true, when the customer is charged, they will be sent a transaction receipt. Values: 'true' or 'false'
source_transaction_id	Specifies a payment gateway transaction id in order to associate payment information with a Subscription record.
acu_enabled	If set to true, credit card will be evaluated and sent based upon Automatic Card Updater settings. If set to false, credit card will not be submitted for updates when Automatic Card Updater runs. Default: 'true' Values: 'true' or 'false'
paused_subscription	If set to true, the subscription will be paused. Values: 'true' or 'false'

* Always required

** Required unless 'month_frequency' and 'day_of_month' is set.

*** Required unless 'day_frequency' is set.

**** Required for credit card transactions

† Required for ACH transactions

Update a Custom Subscription's Plan Details

Variable Name	Description
recurring*	Update the subscription's billing information. Value: 'update_subscription'
subscription_id*	The subscription ID that will be updated.
plan_payments	The number of payments before the recurring plan is complete. Notes: '0' for until canceled
plan_amount	The plan amount to be charged each billing cycle. Format: x.xx
day_frequency	How often, in days, to charge the customer. Cannot be set with 'month_frequency' or 'day_of_month'.
month_frequency	How often, in months, to charge the customer. Cannot be set with 'day_frequency'. Must be set with 'day_of_month'. Values: 1 through 24
day_of_month	The day that the customer will be charged. Cannot be set with 'day_frequency'. Must be set with 'month_frequency'. Values: 1 through 31 - for months without 29, 30, or 31 days, the charge will be on the last day
* Always required	

Update Subscription

Variable Name	Description
recurring*	Update the subscription's billing information. Value: 'update_subscription'
subscription_id*	The subscription ID that will be updated.
plan_payments	The number of payments before the recurring plan is complete. Notes: '0' for until canceled
plan_amount	The plan amount to be charged each billing cycle. Format: x.xx

day_frequency	How often, in days, to charge the customer. Cannot be set with 'month_frequency' or 'day_of_month'.
month_frequency	How often, in months, to charge the customer. Cannot be set with 'day_frequency'. Must be set with 'day_of_month'. Values: 1 through 24
day_of_month	The day that the customer will be charged. Cannot be set with 'day_frequency'. Must be set with 'month_frequency'. Values: 1 through 31 - for months without 29, 30, or 31 days, the charge will be on the last day
start_date	The first day that the customer will be charged. Format: YYYYMMDD
payment_token	The tokenized version of the customer's card or check information. This will be generated by Collect.js and is usable only once.
googlepay_payment_data	The encrypted token created when integration directly to the Google Pay SDK.
ccnumber	Credit card number.
ccexp	Credit card expiration. Format: MMYY
checkname	The name on the customer's ACH account.
checkaccount	The customer's bank account number.
checkaba	The customer's bank routing number.
account_type	The customer's ACH account type. Values: 'checking' or 'savings'
account_holder_type	The customer's ACH account entity. Values: 'personal' or 'business'
sec_code	ACH standard entry class codes. Values: 'PPD', 'WEB', 'TEL', or 'CCD'
first_name	Cardholder's first name. Legacy variable includes: firstname
last_name	Cardholder's last name. Legacy variable includes: lastname
address1	Card billing address.
city	Card billing city
state	Card billing state.
zip	Card billing postal code.
country	Card billing country code.
phone	Billing phone number.
email	Billing email address.
company	Cardholder's company.
address2	Card billing address, line 2.
fax	Billing fax number.
orderid	Order ID
order_description	Order Description Legacy variable includes: orderdescription

merchant_defined_field_#	Can be set up in merchant control panel under 'Settings'->'Merchant Defined Fields'.
ponumber	Cardholder's purchase order number.
processor_id	If using Multiple MIDs, route to this processor (processor_id is obtained under Settings->Transaction Routing in the Control Panel).
customer_receipt	If set to true, when the customer is charged, they will be sent a transaction receipt. Values: 'true' or 'false'
source_transaction_id	Specifies a payment gateway transaction id in order to associate payment information with a Subscription record.
acu_enabled	If set to true, credit card will be evaluated and sent based upon Automatic Card Updater settings. If set to false, credit card will not be submitted for updates when Automatic Card Updater runs. Default: 'true' Values: 'true' or 'false'
paused_subscription	If set to true, the subscription will be paused. Values: 'true' or 'false'

* Always required

Delete a Subscription

Variable Name	Description
recurring*	Delete the subscription. Customer will no longer be charged. Value: 'delete_subscription'
subscription_id*	The subscription ID that will be deleted.

* Always required

Customer Vault Variables

Payment API

POST URL

POST URL:	https://secure.nmi.com/api/transact.php

Add/Update Customer Record

Variables	Description
customer_vault*	Add/Update a secure Customer Vault record. Values: 'add_customer' or 'update_customer'
customer_vault_id	Specifies a Customer Vault id. If not set, the payment gateway will randomly generate a Customer Vault id.
billing_id	Billing id to be assigned or updated. If none is provided, one will be created or the billing id with priority '1' will be updated.
security_key*	API Security Key assigned to a merchant account. New keys can be generated from the merchant control panel in Settings > Security Keys
payment_token	The tokenized version of the customer's card or check information. This will be generated by Collect.js and is usable only once.
googlepay_payment_data	The encrypted token created when integration directly to the Google Pay SDK.
ccnumber**	Credit card number.
ccexp**	Credit card expiration. Format: MMYY
checkname***	The name on the customer's ACH account.
checkaba***	The customer's bank routing number.
checkaccount***	The customer's bank account number.
account_holder_type	The customer's ACH account entity. Values: 'personal' or 'business'
account_type	The customer's ACH account type. Values: 'checking' or 'savings'
sec_code	ACH standard entry class codes. Values: 'PPD', 'WEB', 'TEL', or 'CCD'
currency	Set transaction currency.
payment	Set payment type to ACH or credit card. Values: 'creditcard' or 'check'
orderid	Order id
order_description	Order Description Legacy variable includes: orderdescription
merchant_defined_field_#	Can be set up in merchant control panel under 'Settings'->'Merchant Defined Fields'. Format: merchant_defined_field_1=Value
first_name	Cardholder's first name. Legacy variable includes: firstname

last_name	Cardholder's last name. Legacy variable includes: lastname
address1	Card billing address.
city	Card billing city
state	Card billing state.
zip	Card billing postal code.
country	Card billing country code.
phone	Billing phone number.
email	Billing email address.
company	Cardholder's company.
address2	Card billing address, line 2.
fax	Billing fax number.
shipping_id	Shipping entry id. If none is provided, one will be created or the billing id with priority '1' will be updated.
shipping_firstname	Shipping first name.
shipping_lastname	Shipping last name.
shipping_company	Shipping company.
shipping_address1	Shipping address.
shipping_address2	Shipping address, line 2.
shipping_city	Shipping city
shipping_state	Shipping state.
shipping_zip	Shipping postal code.
shipping_country	Shipping country code.
shipping_phone	Shipping phone number.
shipping_fax	Shipping fax number.
shipping_email	Shipping email address.
source_transaction_id	Specifies a payment gateway transaction id in order to associate payment information with a Customer Vault record.
acu_enabled	If set to true, credit card will be evaluated and sent based upon Automatic Card Updater settings. If set to false, credit card will not be submitted for updates when Automatic Card Updater runs. Default: 'true' Values: 'true' or 'false'

* Always required

** Required for credit card transactions

*** Required for ACH transactions

Customer Vault initiated Sale/Auth/Credit/Offline

Variable	Description
security_key*	API Security Key assigned to a merchant account. New keys can be generated from the merchant control panel in Settings > Security Keys

customer_vault_id*	Specifies a Customer Vault id.
amount	Total amount to be charged. For validate, the amount must be omitted or set to 0.00. Format: x.xx
currency	The transaction currency. Format: ISO 4217
processor_id	If using Multiple MIDs, route to this processor (processor_id is obtained under Settings->Transaction Routing in the Control Panel).
descriptor	Set payment descriptor on supported processors.
descriptor_phone	Set payment descriptor phone on supported processors.
order_description	Order description. Legacy variable includes: orderdescription
orderid	Order ID
Stored Credentials (CIT/MIT)	
initiated_by	Who initiated the transaction. Values: 'customer' or 'merchant'
initial_transaction_id	Original payment gateway transaction id.
stored_credential_indicator	The indicator of the stored credential. Values: 'stored' or 'used' Use **'stored'** when processing the initial transaction in which you are storing a customer's payment details (customer credentials) in the Customer Vault or other third-party payment storage system. Use **'used'** when processing a subsequent or follow-up transaction using the customer payment details (customer credentials) you have already stored to the Customer Vault or third-party payment storage method.

* Always required

Delete Customer Record

Variable	Description
customer_vault*	Deletes a secure Customer Vault record. Values: 'delete_customer'
customer_vault_id*	Specifies a Customer Vault id.
security_key*	API Security Key assigned to a merchant account. New keys can be generated from the merchant control panel in Settings > Security Keys

* Always required

Notes:

- If you do not pass a customer_vault_id, our system will randomly generate one. If you include a customer_id and customer_vault_id, they must match.
- You can only pass Credit Card **or** Electronic Check transaction variables.

Product Manager Variables

Payment API

POST URL

POST URL:	https://secure.nmi.com/api/transact.php

Add a Product

Variables	Description
products*	Add a product to the Product Manager. Value: “add_product“
product_sku*	The unique SKU for this product. An error will be returned if the SKU is already in use by another product. Examples: “000001” or “324-6323-0005“
product_description*	The user-facing name of the product. Examples: “1 Gallon Milk” or “Phone Case”
product_cost*	The cost of the product before any tax or discounts. Must be greater than 0.00.
product_currency*	The currency for the product’s price. Examples: “USD” or “EUR”
product_commodity_code	The commodity code for the product.
product_unit_of_measure	The unit of measure for the product. Defaults to “NAR” (number of articles). Examples: “TDK” or “MTQ”
product_tax_amount	The tax that should be added to the product cost. This is a fixed amount, not a percentage. Example: “1.54”
product_discount_amount	The discount that will subtracted from the cost of the product.
product_image_data**	The Base64-encoded version of the image for the product. The format must be JPG, PNG, or GIF, and be 2MB or smaller.
product_image_name**	The file name of the image being added with product_image_data. Examples: “product.png” or “sku-1234.jpg”

* Always required

** Required if adding an image to the product

Update a Product

Variables	Description
products*	Update a product already in the Product Manager. Value: “update_product”
product_id*	The automatically generated ID for the product. This was returned in the add_product API response, or can be found in the UI under the Product Details page in the Product Manager. Example: “5538585252”

product_sku	The unique SKU for this product. An error will be returned if the SKU is already in use by another product. Examples: “000001” or “324-6323-0005”
product_description	The user-facing name of the product. Examples: “1 Gallon Milk” or “Phone Case”
product_cost	The cost of the product before any tax or discounts. Must be greater than 0.00.
product_currency	The currency for the product’s price. Examples: “USD” or “EUR”
product_commodity_code	The commodity code for the product.
product_unit_of_measure	The unit of measure for the product. Examples: “TDK” or “MTQ”
product_tax_amount	The tax that should be added to the product cost. This is a fixed amount, not a percentage. Example: “1.54”
product_discount_amount	The discount that will subtracted from the cost of the product.
product_image_data**	The Base64-encoded version of the image for the product. The format must be JPG, PNG, or GIF, and be 2MB or smaller.
product_image_name**	The file name of the image being added with product_image_data. Examples: “product.png” or “sku-1234.jpg”

* Always required

** Required if adding an image to the product

Delete a Product

Variables	Description
products*	Delete a product to the Product Manager. This action can not be undone. Value: “delete_product”
product_id*	The automatically generated ID for the product. This was returned in the add_product API response, or can be found in the UI under the Product Details page in the Product Manager. Example: “5538585252”

* Always required

Partial Payment Information

Payment API

Request Details

Variable	Description
partial_payment_id	Unique identifier returned when making the original transaction. This should only be used for secondary transactions.
partial_payments	This variable allows the following two values to be passed to it:
	settle_partial: Settles any amount of tender collected (captured partial auth's and approved partial sales) at cut off.
	payment_in_full: Required that any split tendered transaction is collected in-full before settlement gets initiated.
type	This variable can be passed the value 'complete_partial_payment' which will complete a payment_in_full transaction that has not been collected in full. This allows industries that require payment_in_full but subsequently decide to still settle the transaction even though it has not been collected in full.

Response Details

Variable	Description
partial_payment_id	A numeric identifier which is used when submitting subsequent transactions.
partial_payment_balance	Returns the payment's remaining balance.
amount_authorized	Provides the amount that was authorized.

Examples

Example 1: In this request, if nothing more was done, a transaction for 30.00 would settle at the next cut-off.

Request	...type=sale&partial_payments=settle_partial&ccnumber=4111111111111111&ccexp=1016&amount=100.00...
Response	...response=1&partial_payment_id=123456789&partial_payment_balance=70.00&amount_authorized=30.00...

Example 2: In this request, payment_in_full was required and two transaction were collected - this transaction would settle at the next cut-off.

Request 1	...type=sale&partial_payments=payment_in_full&ccnumber=4111111111111111&ccexp=1016&amount=100.00
Response 1	...response=1&partial_payment_id=123456789&partial_payment_balance=70.00&amount_authorized=30.00...
Request 2	...type=sale&partial_payment_id=123456789&partial_payments=payment_in_full&ccnumber=40000000000000

Response 2	...response=1& partial_payment_id=123456789&partial_payment _balance=0.00&amount_authorized=70.00...

Example 3: In this example, payment_in_full was required and two transactions were attempted, but only one collected. The merchant decided to force it out anyways - this transaction would settle at the next cut-off.

Request 1	...type=sale&partial_payments=payment_in_full&ccnumber=4111111111111111&ccexp=1016&amount=100.00
Response 1	...response=1&partial_payment_id=123456789&partial_payment_balance=70.00&amount_authorized=30.00...
Request 2	...type=sale&partial_payment_id=123456789&partial_payments=payment_in_full&ccnumber=40000000000000
Response 2	...response=2&partial_payment_id=123456789&partial_payment_balance=70.00&amount_authorized=70.00...
Request 3	...type=complete_partial_payment& partial_payment_id=123456789&partial_payments=payment_in_full&amou
Response 3	...response=1& partial_payment_id=123456789&partial_payment_balance=0.00&amount_authorized=70.00...

Credential on File Information

Payment API

Please note the below is meant to be a guide for how the platform supports CIT and MIT use cases. This is not meant to be an exhaustive list of items needed in order to be compliant. For more information on CIT/MIT compliance, please consult your processor.

Credential on File regulations apply any time data is stored to process future purchases for a cardholder.

Customer vs Merchant Initiated

When a customer is actively engaged in checkout - either physically present in a store, or checking out online in their browser, that is a **Customer Initiated Transaction** (CIT).

When the customer isn't actively engaged, but has given permission for their card to be charged, that is a **Merchant Initiated Transaction** (MIT). In order for a merchant to submit a Merchant Initiated Transaction, a Customer Initiated transaction is required first.

Overview

A cardholder's consent is required for the initial storage of credentials. When a card is stored, an initial transaction should be submitted (Validate, Sale, or Auth) with the **correct credential-on-file type**. The transaction must be approved (not declined or encounter an error.) Then, store the transaction ID of the initial customer initiated transaction. The transaction ID must then be submitted with any follow up transactions (MIT or CIT.)

Credential on File types include Recurring, Installment, and Unscheduled types.

For simplicity - we are using the Payment API variables. These match the names of the Batch Upload, Collect.js, Browser Redirect, or the Customer-Present Cloud APIs. The Three-Step API follows the same pattern, and the variables should be submitted on Step 1.

Request Details

Variable	Description
initiated_by	Who initiated the transaction. Values: 'customer' or 'merchant'
initial_transaction_id	Original payment gateway transaction id.
stored_credential_indicator	The indicator of the stored credential. Values: 'stored' or 'used' Use **'stored'** when processing the initial transaction in which you are storing a customer's payment details (customer credentials) in the Customer Vault or other third-party payment storage system. Use **'used'** when processing a subsequent or follow-up transaction using the customer payment details (customer credentials) you have already stored to the Customer Vault or third-party payment storage method.

Response Details

Variable	Description
cof_supported	Credential on File support indicator specific to the transaction. Values: 'stored' or 'used' Value will be **'stored'** if CIT/MIT transaction was sent to a processor that supports the feature. Value will be **'used'** if CIT/MIT transaction was sent to a processor that does not support the feature or if a merchant-initiated transaction cannot occur due to Cross-Processor limitations.

Please Note: For Three-Step Redirect transactions, the request details must be sent in Step 1 and the 'cof-supported' element will be returned in the response of Step 3.

Referencing the Initial Transaction:

When doing a credential-on-file type transaction, we will reject any follow up transactions that pass in a card number that does not match the card brand used in the initial transaction. For example, using a Mastercard when the original transaction uses Visa will result in the transaction getting rejected. The card brands each have independent systems for tracking card-on-file transactions, so an initial transaction ID cannot be reused between them. We reject this type of incorrect reuse at the time of the request because it can result in settlement failures, downgrades, etc. later.

If a customer changes their card on file, a good practice is to first store it as a new initial transaction, and reference that initial transaction ID for future payments on the new card.

Recurring:

A transaction in a series of transactions that uses a stored credential and are processed at fixed, regular intervals (not to exceed one year between transactions), and represents cardholder agreement for the merchant to initiate future transactions for the purchase of goods or services provided at regular intervals.

If a customer is signing up for a **recurring** subscription, the merchant is expected to send "an initial recurring transaction" every time the customer signs up for a new recurring subscription.

For an initial transaction:

- For a free trial, the initial transaction will be a validate transaction type (or auth if validate is not supported.)
- If the customer is being charged immediately for a product, the initial transaction will be a sale or an authorization for the correct amount.

Either transaction MUST INCLUDE three items:

- billing_method=recurring
- initiated_by=customer
- stored_credential_indicator=stored

Examples

Example 1: In this request, an initial recurring sale is sent and an approved transaction is returned in the response. *Store* this transaction for the follow up request.

Request	...type=sale&billing_method=recurring&initiated_by=customer&stored_credential_indicator=stored...
Response	...response=1&responsetext=Approved&transactionid=1234567890...

The transaction ID would be stored and submitted on follow up transactions. The follow up transaction(s) would include:

- billing_method=recurring
- initiated_by=merchant
- stored_credential_indicator=used
- initial_transaction_id=XXXXXXXXXX

Example 2: In this request, the subsequent merchant initiated sale is processed using the stored transaction from Example 1.

Request	...type=sale&billing_method=recurring&initiated_by=merchant&stored_credential_indicator=used&initial_transa
Response	...response=1&responsetext=Approved&transactionid=1234567891...

Please Note: This transaction ID cannot be used for "unscheduled" or "installment" credential-on-file transactions.

Installment:

An “installment” transaction is a series of transactions that uses a stored credential and represents cardholder agreement with the merchant to initiate one or more future transactions over a period of time for a single purchase of goods or services.

Installment transactions work just like Recurring in that you need a customer initiated transaction for a subsequent installment transaction. The difference is the billing_method will be “installment”.

The customer initiated transaction MUST INCLUDE at least three items (* recommended to send, if available):

- billing_method=installment
- initiated_by=customer
- stored_credential_indicator=stored
- * billing_total
- * billing_number (Values: 0-99)

Examples

Example 3: In this request, an initial installment sale is sent and an approved transaction is returned in the response. Store this transaction for the follow up request.

Request	...type=sale&billing_method=installment&initiated_by=customer&stored_credential_indicator=stored&billing_tot =25.00...
Response	...response=1&responsetext=Approved&transactionid=1234567890…

The transaction ID would be stored and submitted on follow up transactions. The follow up transaction(s) would include (* recommended to send, if available):

- billing_method=installment
- initiated_by=merchant
- stored_credential_indicator=used
- initial_transaction_id=XXXXXXXXXX
- * billing_total
- * billing_number

Example 4: In this request, the subsequent merchant initiated sale is processed using the stored transaction from Example 3.

Request	...type=sale&billing_method=installment&initiated_by=merchant&stored_credential_indicator=used&initial_trans 100.00&billing_number=1&amount=25.00...
Response	...response=1&responsetext=Approved&transactionid=1234567891…

Please Note: *This transaction ID cannot be used for "unscheduled" or "recurring" card on file transactions.*

Unscheduled Credential On File:

For payments that aren't recurring or installment - there are unscheduled options as well.

The first customer initiated transaction will include these two items (no billing method):

- initiated_by=customer
- stored_credential_indicator=stored

Examples

Example 5: In this request, an initial unscheduled sale is sent and an approved transaction is returned in the response. Store this transaction for the follow up request.

Request	...type=sale&initiated_by=customer&stored_credential_indicator=stored...
Response	...response=1&responsetext=Approved&transactionid=1234567890...

The transaction ID can be used, without a billing method, for a customer initiated or merchant initiated transaction.

Please Note: The transaction ID cannot be used for a “recurring” or “installment” transaction.

Unscheduled, Customer Initiated: A card-absent transaction initiated by the cardholder where the cardholder does not need to enter their card details as the merchant uses the payment credential previously stored by the cardholder to perform the transaction. Examples include a transaction using customer’s merchant profile or digital wallet.

This is your typical shopping cart scenario where the customer checks out without having to re-enter their card details.

The follow up transaction(s) would include:

- initiated_by=customer
- stored_credential_indicator=used

Example 6: In this request, a subsequent unscheduled sale is sent and an approved transaction is returned in the response.

Request	...type=sale&initiated_by=customer&stored_credential_indicator=used...
Response	...response=1&responsetext=Approved&transactionid=1234567891...

Unscheduled, Merchant Initiated: A transaction using a stored credential for a fixed or variable amount that does not occur on a scheduled or regularly occurring transaction date, where the cardholder has provided consent for the merchant to initiate one or more future transactions. An example of this transaction is an account auto-top up transaction.

An example of an account auto-top up would be a customer with an account with a balance. When that balance gets low, the customer's card is charged automatically, without the customer's involvement.

The follow up transaction(s) would include:

- initiated_by=merchant
- stored_credential_indicator=used
- initial_transaction_id=XXXXXXXXXX

Example 7: In this request, a subsequent unscheduled sale is sent and an approved transaction is returned in the response.

Request	...type=sale&initiated_by=merchant&stored_credential_indicator=used&initial_transaction_id=1234567890...
Response	...response=1&responsetext=Approved&transactionid=1234567892...

Appendix 1: Recommend Further Reading:

If there is any question where a transaction type falls, we recommend reviewing the official card brand documentation. Visa's guidelines are the most stringent, and generally if you follow those guidelines, you'll also be compliant for MasterCard, American Express and Discover.

Visa:
https://usa.visa.com/dam/VCOM/global/support-legal/documents/stored-credential-transaction-framework-vbs-10-may-17.pdf

MasterCard:
https://www.mastercard.us/content/dam/public/mastercardcom/na/us/en/banks-and-credit-unions/other/credential-on-file-the-digital-commerce-growth-engine.pdf

Transaction Response Variables

Payment API

Standard Response

Variable Name	Description
response	1 = Transaction Approved 2 = Transaction Declined 3 = Error in transaction data or system error
responsetext	Textual response
authcode	Transaction authorization code.
transactionid	Payment gateway transaction id.
avsresponse	AVS response code (See AVS Response Codes).
cvvresponse	CVV response code (See See CVV Response Codes).
orderid	The original order id passed in the transaction request.
response_code	Numeric mapping of processor responses (See See Result Code Table).
emv_auth_response_data	This will optionally come back when any chip card data is provided on the authorization. This data needs to be sent back to the SDK after an authorization.

Conditional Response

Variable Name	Description
customer_vault_id	The original customer_vault_id passed in the transaction request or the resulting customer_vault_id created on an approved transaction. **Note:** Only returned when the "Customer Vault" service is active.
kount_score	The Kount "Omniscore" indicating the level of risk on a given transaction. The higher the score, the lower the risk. **Note:** Only returned when the "Kount" service is active.
merchant_advice_code	Mastercard's Merchant Advice Code (MAC) is returned in response if one is provided by the processor. **Note:** Only returned if API configuration is set to return this value.

Testing Information

Payment API

Transaction Testing Methods

Method 1: Put your account in test mode

Transactions can be submitted to any merchant account that is in test mode. Keep in mind that if an account is in test mode, all valid credit cards will be approved but **no charges will actually be processed and nothing will be sent to the credit card or ACH processor.**

Method 2: Send in a one-off test transaction

One-off test transactions can be processed using the below `test_mode` variable. This will process this singular transaction in test mode, but it will not impact anything else on the account. An example use case would be running test transactions in a developent environment while your website is actively processing real transactions from customers.

test_mode:	If set to "enabled" *and* providing one of the test credit card numbers listed below with "1025" as the expiration date, the single transaction will process in test mode. To see this transaction in reporting, you will need to toggle your account to test mode, but the Payment API testing can be done without doing this.

Method 3: Dedicated test account

The Payment Gateway Demo Account can be used for testing at any time. Please use the below security key for testing with this account. This account is always available and allows testing in a completely sandboxed environment. Like all testing methods, no card or check data will ever be sent for actual processing.

security_key:	6457Thfj624V5r7WUwc5v6a68Zsd6YEm

Transaction POST URL

Transaction details should be POST'ed to the following URL:

POST URL:	https://secure.nmi.com/api/transact.php

Test Data

Transactions can be submitted using the following information:

Visa:	4111111111111111
MasterCard:	5431111111111111
Discover:	6011000991300009

American Express:	341111111111111
Diner's Club:	30205252489926
JCB:	3541963594572595
Maestro:	6799990100000000019
Credit Card Expiration:	10/25
account (ACH):	24413815
routing (ACH):	490000018

Triggering Errors in Test Mode

- To cause a declined message, pass an amount less than 1.00.
- To trigger a fatal error message, pass an invalid card number.
- To simulate an AVS match, pass 888 in the address1 field, 77777 for zip.
- To simulate a CVV match, pass 999 in the cvv field.

AVS Response Codes

Payment API

AVS Response Codes

X	Exact match, 9-character numeric ZIP
Y	Exact match, 5-character numeric ZIP
D	Exact match, 5-character numeric ZIP
M	Exact match, 5-character numeric ZIP
2	Exact match, 5-character numeric ZIP, customer name
6	Exact match, 5-character numeric ZIP, customer name
A	Address match only
B	Address match only
3	Address, customer name match only
7	Address, customer name match only
W	9-character numeric ZIP match only
Z	5-character ZIP match only
P	5-character ZIP match only
L	5-character ZIP match only
1	5-character ZIP, customer name match only
5	5-character ZIP, customer name match only
N	No address or ZIP match only
C	No address or ZIP match only
4	No address or ZIP or customer name match only
8	No address or ZIP or customer name match only
U	Address unavailable
G	Non-U.S. issuer does not participate
I	Non-U.S. issuer does not participate
R	Issuer system unavailable
E	Not a mail/phone order
S	Service not supported
0	AVS not available
O	AVS not available
B	AVS not available

CVV Response Codes

Payment API

CVV Response Codes

M	CVV2/CVC2 match
N	CVV2/CVC2 no match
P	Not processed
S	Merchant has indicated that CVV2/CVC2 is not present on card
U	Issuer is not certified and/or has not provided Visa encryption keys

Result Code Table

Payment API

Result Code Table

100	Transaction was approved.
200	Transaction was declined by processor.
201	Do not honor.
202	Insufficient funds.
203	Over limit.
204	Transaction not allowed.
220	Incorrect payment information.
221	No such card issuer.
222	No card number on file with issuer.
223	Expired card.
224	Invalid expiration date.
225	Invalid card security code.
226	Invalid PIN.
240	Call issuer for further information.
250	Pick up card.
251	Lost card.
252	Stolen card.
253	Fraudulent card.
260	Declined with further instructions available. (See response text)
261	Declined-Stop all recurring payments.
262	Declined-Stop this recurring program.
263	Declined-Update cardholder data available.
264	Declined-Retry in a few days.
300	Transaction was rejected by gateway.
400	Transaction error returned by processor.
410	Invalid merchant configuration.
411	Merchant account is inactive.
420	Communication error.
421	Communication error with issuer.
430	Duplicate transaction at processor.
440	Processor format error.
441	Invalid transaction information.
460	Processor feature not available.
461	Unsupported card type.

Rate Limits

Payment API

Rate Limits

In order to ensure the platform is available for everyone, in very rare cases, some users may encounter rate limits. If your request rate exceeds the limit, you may receive one of two responses: the Payment API-specific Rate Limit response, or the System-Wide Rate Limit response.

Payment API-Specific Rate Limit

If you exceed the Payment API rate limit, you will receive a response with the following fields:

response	3
responsetext	Rate limit exceeded
response_code	301

Example

Response	response=3&responsetext=Rate limit exceeded&authcode=&transactionid=&avsresponse=&cvvresponse=&orderid=&type=&response_code=301

System-Wide Rate Limit

You may encounter the system-wide rate limit if you using other services, for example if you are making requests to both the Payment API and the Query API, or if there are too many concurrent connections from your IP. In this case, your request will receive HTTP 429: Too Many Requests

Handling Rate Limit Responses

Wait before retrying

If you receive a response_code of 301 or an HTTP 429 response, do not immediately retry the request. Immediately retrying may increase the delay before transactions are allowed again.

Use fewer simultaneous connections

Reduce the number of threads or processes sending connections if you regularly encounter rate limits.

Check your credentials

Repeated authentication failures from your IP may result in temporarily lowered rate limits.

Contact support

If you've tried the above fixes and still need additional help, contact Customer Support for other options.