Payment API

Version 2.15

August 10, 2023

Table of Contents

Revision History

v2.15 - August 10, 2023

- Added payment.initiator field
- Added new field values to payment.detailed_payment_type

v2.14 - June 15, 2023

- Added recipient object to payment request fields
- Added an example merchant funding transaction

v2.13 - March 17, 2023

- Update AVS result codes

v2.12 - March 3, 2023

- New PAD reason codes

v2.11 - October 25, 2022

- Payment API supports Stored Credential in Token object - single_use_token

v2.10 - October 3, 2022

- Add surcharge details

v2.9 - June 8, 2022

- Updated response text for reason code 201008 from DECLINED (CONTRA) to CARD NOT ALLOWED

v2.8 - May 6, 2022

- add User Pay details
- add cheque_reprocess transaction

v2.7 - December 3, 2021

- Add new token error
- Add PAD Token operation

v2.6 - September 27, 2021

- Updated Visa response codes

v2.5 - March 15, 2021

- Changed card payment testing transaction to be triggered on amount instead of expiry date
- Added information on response codes in declined card transactions
- Added list of MasterCard and Visa response codes

v2.4 - June 8, 2020

- Add Authentication source code examples
- Add PAD Notifications section
- Add PAD pad_account_verify transaction
- Add PAD Response Fields section
- Add PAD eft_payment example
- Add field types to response fields
- Add Card Payment risk reason codes

v2.3 - January 24, 2020

- Removed EFT Reversal

v2.2 - March 1, 2019

- Added card payment processing details

Introduction

The Payment API is a RESTful web service that supports PAD and card payment processing for Payroc partners and customers.

All messages are HTTPS POST with JSON format being used for data exchange.

Process Overview

Customers can request an account setup for Payment API by contacting Payroc Merchant Services.

> Tel: 647-258-3708
>
> Toll Free: 1-855-812-5191
>
> Email: canada-support@payroc.com

After receiving their account credentials, customers can construct and send request messages to Payroc's API server

Payroc's API server will return a response specifying whether the PAD request has been captured for submission to the bank or rejected due to validation errors.

Authentication

Authentication of requests is performed based on following values included in HTTP header:

X-User-ID

> X-User-ID value must be base64 encoded API User ID assigned by Payroc.
>
> API User ID will be an ASCII string up to 32 bytes in length that can contain: A-Z,a-z,0-9,-,_

X-Message-Hash

> X-Message-Hash value must be base64 encoded result of hash function.
>
> Message hash is calculated using the API Key provided by Payroc and the entire JSON string included in the request.
>
> API Key will be an alphanumeric ASCII string up to 64 bytes in length.
>
> Hash is to be generated using HMAC-SHA256 algorithm - see RFC 2104 for details.

HTTP header example:

```
Content-Type: application/json
X-User-ID: Qmx1ZVBheUNhbmFkYS0wMDAx
X-Message-Hash: kCaHLZMAF+3Ov923dfPF+AkauzmSclAn1vnFaoTu/rk=
```

Response returned for failed authentication:

```
HTTP Code: 401

{ "message":"", "details":{} }
```

Source Code Examples

Below are some basic source code examples in various languages to show how to generate the message hash.

Example: C#

```
using System;
using System.Security.Cryptography;
using System.Text;

namespace PaymentAPIHash
{
    class MainClass
    {
    public static Encoding utf8 = Encoding.UTF8;

    public static void Main(string[] args)
    {
        string uid = "api-user-id";
        string key = "api-secret-key";
        string jsonstr = "{ ... }";

        byte[] hmac = HashHMAC(key, jsonstr);

        string uid_b64 = Convert.ToBase64String(utf8.GetBytes(uid));
        string sig_b64 = Convert.ToBase64String(hmac);

        Console.WriteLine("uid_b64: {0}", uid_b64);
        Console.WriteLine("sig_b64: {0}", sig_b64);
    }

    private static byte[] HashHMAC(string key, string message)
    {
        var hash = new HMACSHA256(utf8.GetBytes(key));
        byte[] hmac = hash.ComputeHash(utf8.GetBytes(message));
        return hmac;
    }
    }
}
```


Example: Java

```
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.util.Base64;

public class Hmac256 {

    public static void main ( String[] args ) {
    final String uid = "api-user-id";
    final String key = "api-secret-key";
    final String jsonstr = "{ ... }";

    String hmac = genHmac ( key, jsonstr );

    System.out.println("uid_b64: " + Base64.getEncoder().encodeToString(uid.getBytes()));
    System.out.println("sig_b64: " + hmac);
    }

    private static String genHmac( String key, String data ) {
    try {
        Mac sha256_HMAC = Mac.getInstance("HmacSHA256");
        SecretKeySpec secret_key = new SecretKeySpec(key.getBytes("UTF-8"), key);
        sha256_HMAC.init(secret_key);

        return Base64.getEncoder().encodeToString(sha256_HMAC.doFinal(data.getBytes("UTF-8")));

    } catch (java.security.NoSuchAlgorithmException e) {
        e.printStackTrace();
    } catch (java.security.InvalidKeyException e) {
        e.printStackTrace();
    } catch (java.io.UnsupportedEncodingException e) {
        e.printStackTrace();
    }

    return "";
    }
}
```

Example: Perl

```
#!/usr/bin/perl

use strict;
use Digest::SHA 'hmac_sha256';
use MIME::Base64 'encode_base64';

my $uid = 'api-user-id';
my $key = 'api-secret-key';
my $jsonstr = '{ ... }';

my $sig = hmac_sha256 $jsonstr, $key;

my $uid_b64 = encode_base64 $uid, '';
my $sig_b64 = encode_base64 $sig, '';

printf "uid_b64: %s\n", $uid_b64;
printf "sig_b64: %s\n", $sig_b64;
```


Example: Python

```
#!/usr/bin/env python3

import hmac
import hashlib
import base64

def main ():
    uid = "api-user-id"
    key = b"api-secret-key"
    jsonstr = b"{ ... }"

    hm = hmac.new(key, digestmod=hashlib.sha256)
    hm.update(jsonstr)

    dig = hm.digest();

    uidb = bytes(uid, 'utf-8')
    uidb64 = base64.b64encode(uidb)

    print ( uidb64 )
    print ( base64.b64encode( dig ) )

if __name__ == "__main__":
    main()
```


Pre-Authorized Debit (PAD) Processing

Transaction Types

PAD Debit

Funds are collected via debit to the client's bank account and credit is posted to the merchant's bank account.

PAD Refund

Funds are returned to the client's bank account to correct a PAD Debit previously processed.

EFT Payment

Merchant can issue a credit via EFT transfer in order to make a payment to a vendor/supplier.

PAD Debit/Refund Void

Void can be used to cancel a PAD debit or PAD refund transaction if it has not already been submitted to the bank.

PAD Debit Reprocess

Reprocess a PAD Debit transaction previously declined due to merchant thresholds.

Cheque Reprocess

Reprocess a Cheque transaction previously declined due to merchant thresholds.

PAD Account Verification

A low value transaction ("penny transaction") used to validate the client's bank account information. Funds are sent via a credit to the client's bank account and a debit is posted to the merchant's bank account.

PAD Token Add

Create a token for either the client or merchant bank information.

PAD Token Update

Update an existing token.

PAD Token Deactivate

Deactivate an existing token.

PAD Token Reactivate

Reactivate an existing token.

PAD Token Operations

Tokens for merchant and client banking information can be created and maintained using PAD Token transaction types. Token IDs can be used in place of banking information for other transaction types - see Request Fields and Request Examples sections for details.

Request Fields

The table below details the fields that can be included in transaction request messages.

Field	Description
terminal_id Type: String Length: 8	**TERMINAL ID** Unique identifier provided by Payroc for a merchant/location. Required for all requests.
transaction_type	**TRANSACTION TYPE** Values: pad_debit pad_debit_void pad_debit_reprocess cheque_reprocess pad_refund pad_refund_void pad_account_verify eft_payment token_add token_update token_deactivate token_reactivate Required for all requests.
reference_number Type: String Length: 15	**REFERENCE NUMBER** Unique transaction reference number. Required for all requests.
amount Type: Numeric Length: 12	**AMOUNT** Amount of funds to be transferred. Required for all requests.
client_id Type: String Length: 29	**CLIENT ID** Unique value representing the client/consumer. Required for pad_debit pad_refund pad_account_verify eft_payment

... continued on next page

Field	Description
charge_description Type: String Length: 30	**CHARGE DESCRIPTION** Description that is to appear on the client statement; typically the business name of the merchant. Required for pad_debit pad_refund pad_account_verify eft_payment
bank_number Type: Numeric Length: 3	**CLIENT BANK ID** 3 digit bank number Required if token is not being used pad_debit pad_account_verify eft_payment Not required if token is being used
branch_number Type: Numeric Length: 5	**CLIENT BRANCH/TRANSIT NUMBER** 5 digit branch or transit number Required if token is not being used pad_debit pad_account_verify eft_payment Not required if token is being used
account_number Type: Numeric Length: 12	**CLIENT BANK ACCOUNT NUMBER** Up to 12 digit bank account number Required if token is not being used pad_debit pad_account_verify eft_payment Not required if token is being used
token Type: String Length: 30	**CLIENT BANK INFORMATION TOKEN** A unique ID that can be used in place of client bank information. Optional for pad_debit pad_account_verify eft_payment

... continued on next page

Field	Description
merchant_bank_number Type: Numeric Length: 3	**MERCHANT BANK ID** 3 digit bank number Required if merchant_token is not being used pad_debit pad_account_verify eft_payment Not required if merchant_token is being used
merchant_branch_number Type: Numeric Length: 5	**MERCHANT BRANCH/TRANSIT NUMBER** 5 digit branch or transit number Required if merchant_token is not being used pad_debit pad_account_verify eft_payment Not required if merchant_token is being used
merchant_account_number Type: Numeric Length: 12	**MERCHANT BANK ACCOUNT NUMBER** Up to 12 digit bank account number Required if merchant_token is not being used pad_debit pad_account_verify eft_payment Not required if merchant_token is being used
merchant_token Type: String Length: 30	**MERCHANT BANK INFORMATION TOKEN** A unique ID that can be used in place of merchant bank information. Optional for pad_debit pad_account_verify eft_payment
effective_date Type: String Length: 10	**EFFECTIVE DATE** Can be used to specify the date when the debit/payment transaction is to be processed. Not applicable for refund transactions. Only dates up to 30 days in future are allowed. If not specified then the transaction will be included in next processing run. Format: YYYY-MM-DD Optional for pad_debit

... continued on next page

Field	Description
threshold_override Type: String Length: 1	**THRESHOLD OVERRIDE** Values: Y N Value of Y indicates that merchant thresholds are to be overridden when re-processing PAD Debit or Cheque transaction. Threshold override option is only allowed for approved Payment Facilitators partners. This field is only applicable when the original PAD Debit request was declined due to merchant thresholds. Required for pad_debit_reprocess cheque_reprocess

Bank Information Object

Field		Description
bank_information		
	bank_number Type: Numeric Length: 3	**Bank Number** 3 digit bank number Required for token_add Optional for token_update
	branch_number Type: Numeric Length: 5	**BRANCH/TRANSIT NUMBER** 5 digit branch or transit number Required for token_add Optional for token_update
	account_number Type: Numeric Length: 12	**BANK ACCOUNT NUMBER** Up to 12 digit bank account number Required for token_add Optional for token_update

Token Object

Field		Description
token		
	token Type: String Length: 30	**TOKEN ID** A unique ID that can be created and used in place of bank information for future payment processing. For *Token Operations*, the token ID value can be specified by the merchant and/or automatically generated by Payroc. Submitting a token ID value containing a "?" specifies that the token ID value is to be generated by Payroc. Token IDs will be generated by Payroc based on configuration parameters chosen by the merchant. 12 characters is the minimum length for auto-generated Token IDs Characters supported: 0-9 A-Z : @ | - + / _ , Required for token_add token_update token_deactivate token_reactivate
	client_id Type: String Length: 30	**TOKEN CLIENT ID** This field can be used to include additional reference data with a token such as a client identifier. Characters supported: A-Z a-z 0-9 - _ . @ / Optional for token_add token_update
	start_date Type: String Length: 8	**TOKEN START DATE** The effective start date for the token. This token will not be accepted for payments before this date. Format: YYYYMMDD Optional for token_add token_update

	end_date Type: String Length: 8	**TOKEN END DATE** The effective end date for the token. This token will not be accepted for payments after this date. Format: YYYYMMDD Optional for token_add token_update
	reference Type: String Length: 30	**TOKEN REFERENCE DATA** Reference data to associate with a token. Characters supported: A-Z a-z 0-9 - / \ Optional for token_add token_update

Response Fields

Response Field	Description
message Type: String Length: 128	**MESSAGE** This is the response text of the message. For successful transactions this field will be blank. For errors and decline responses this field contains the error message or decline message.
reason_code Type: String Length: 6	**REASON CODE** 6-digit reason code will be returned code if the transaction was not successfully processed.

Responses

For successful requests, HTTP 202 will be returned with the following object:

```
{
    "message": "",
    "details":{}
}
```

For failed requests, HTTP 4xx code is returned. For some HTTP codes, such as 405, 415, 404, an empty body will be returned.

For validation errors, HTTP 400 will be returned with the following object:

```
{
    "message": "",
    "details":{
        "reason_code":""
    }
}
```

Reason Codes

These are the reason code values returned for non-successful or negative-type transaction requests.

Reason Code	Response Text and Explanation
101003	Invalid Terminal ID
101004	Invalid Merchant Bank ID
101005	Invalid Merchant Bank Transit Number
101006	Invalid Merchant Bank Account Number
101007	Merchant Bank Information Mismatch
101010	Invalid Charge Description
102001	Invalid Amount
102002	Invalid Client Bank ID

... continued on next page

Reason Code	Response Text and Explanation
102003	Invalid Client Bank Transit Number
102004	Invalid Client Bank Account Number
102005	Invalid Reference Number
102006	Duplicate Reference Number
102007	Invalid Client ID
102008	Invalid Effective Date
102009	Refund No Match
102010	Amount Exceeds Risk Threshold
102011	Invalid Transaction Type
102012	Void No Match
102013	Transaction Already Processed
102014	Threshold Override No Match
102015	Invalid Threshold Override Flag
102016	MTD Count Exceeds Limit
102017	MTD Amount Exceeds Limit

Request Examples

PAD Debit

An example PAD Debit transaction for $150.00:

```
{
    "terminal_id":"EXAMPLE1",
    "client_id":"client-id",
    "transaction_type":"pad_debit",
    "charge_description":"DEBIT-DESC",
    "reference_number":"DEBIT-1234",
    "amount":15000,
    "bank_number":"001",
    "branch_number":"12345",
    "account_number":"1234567",
    "merchant_bank_number":"001",
    "merchant_branch_number":"23456",
    "merchant_account_number":"2345678"
}
```

Merchant banking information will be validated based on the Payroc merchant setup for authentication purposes.

PAD Debit with Token

An example PAD Debit transaction for $150.00 using a token for the client bank information and merchant bank information:

```
{
    "terminal_id":"EXAMPLE1",
    "client_id":"client-id",
    "transaction_type":"pad_debit",
    "charge_description":"DEBIT-DESC",
    "reference_number":"DEBIT-1234",
    "amount":15000,
    "token":"EXAMPLETOKEN1",
    "merchant_token":"EXAMPLETOKEN2"
}
```

Tokens can be used for the client bank information and the merchant bank information.

PAD Refund

An example PAD Refund transaction for $150.00:

```
{
    "terminal_id":"EXAMPLE1",
    "client_id":"client-id",
    "transaction_type":"pad_refund",
    "charge_description":"REFUND-DESC",
```



```
    "reference_number":"DEBIT-1234",
    "amount":15000
}
```

Banking information is not required for refunds as the Terminal ID, Reference Number, Amount and Charge Description will be used to retrieve the information from the original debit/payment.

PAD Debit Void

An example PAD Debit Void transaction:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"pad_debit_void",
    "reference_number":"DEBIT-1234",
    "amount":15000
}
```

The reference number must match a PAD Debit transaction that has not been submitted to the bank.

PAD Debit Reprocess

An example PAD Debit Reprocess transaction:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"pad_debit_reprocess",
    "reference_number":"DEBIT-1234",
    "amount":15000,
    "threshold_override":"Y"
}
```

The reference number must match a PAD Debit transaction that was declined due to merchant thresholds.

Cheque Reprocess

An example Cheque Reprocess transaction:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"cheque_reprocess",
    "reference_number":"CHQ-1234",
    "amount":15000,
    "threshold_override":"Y"
}
```

The reference number must match a Cheque transaction that was declined due to merchant thresholds.

PAD Account Verification

An example PAD Account Verification transaction for $0.01:

```
{
    "terminal_id":"EXAMPLE1",
    "client_id":"client-id",
    "transaction_type":"pad_account_verify",
    "charge_description":"ACCTVER-DESC",
    "reference_number":"ACCTVER-1234",
    "amount":1,
    "bank_number":"001",
    "branch_number":"12345",
    "account_number":"1234567",
    "merchant_bank_number":"001",
    "merchant_branch_number":"23456",
    "merchant_account_number":"2345678"
}
```

Merchant banking information will be validated based on the Payroc merchant setup for authentication purposes.

EFT Payment

An example EFT Payment transaction for $12.00:

```
{
    "terminal_id":"EXAMPLE1",
    "client_id":"client-id",
    "transaction_type":"eft_payment",
    "charge_description":"EFTPAY-DESC",
    "reference_number":"EFTPAY-1234",
    "amount":1200,
    "bank_number":"001",
    "branch_number":"12345",
    "account_number":"1234567",
    "merchant_bank_number":"001",
    "merchant_branch_number":"23456",
    "merchant_account_number":"2345678"
}
```

Merchant banking information will be validated based on the Payroc merchant setup for authentication purposes.

PAD Token Add

In this example, we will associate a token with bank account information.

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"token_add",
    "bank_information":{
```



```
            "bank_number":"001",
            "branch_number":"12345",
            "account_number":"1234567"
        },
        "token":{
            "token":"EXAMPLETOKEN1",
            "reference":"JSMITH-BANKINFO-REF"
        }
    }
```

PAD Token Update

In this example, we will be updating the token to change the reference.

```
    {
        "terminal_id":"EXAMPLE1",
        "transaction_type":"token_update",
        "token":{
            "token":"EXAMPLETOKEN1",
            "reference":"TEST-CHANGE-1"
        }
    }
```

PAD Token Deactivate

In this example, we will de-activating the token.

```
    {
        "terminal_id":"EXAMPLE1",
        "transaction_type":"token_deactivate",
        "token":{
            "token":"EXAMPLETOKEN1",
        }
    }
```

PAD Token Reactivate

In this example, we will re-activate the token. All data is preserved while the token is deactivated, but the token cannot be used.

```
    {
        "terminal_id":"EXAMPLE1",
        "transaction_type":"token_reactivate",
        "token":{
            "token":"EXAMPLETOKEN1",
        }
    }
```


Testing Transactions

Validation errors can be generated for testing purposes by including a specific amount in the request:

Amount	Reason Code	Response Text
1010	102001	Invalid Amount
1020	102002	Invalid Client Bank ID
1030	102003	Invalid Client Bank Transit Number
1040	102004	Invalid Client Bank Account Number
1050	102005	Invalid Reference Number
1060	102006	Duplicate Reference Number
1070	102007	Invalid Client ID
1080	102008	Invalid Effective Date
1090	102010	Amount Exceeds Risk Threshold
1110	102009	Refund No Match
1120	102016	MTD Count Exceeds Limit
1130	102017	MTD Amount Exceeds Limit

PAD Notifications

Process Overview

API User setup includes an option to enable PAD Notifications by providing a PAD notification URL.

When the state of a PAD transaction changes, a message will be sent to the provided PAD notification URL.

Message Delivery

Notification Request

A notification URL must be provided to enable PAD notifications. The notifcation URL is required to be SSL encrypted (HTTPS) with a valid SSL certificate.

The notification message will be a HTTP POST with a JSON payload in the body.

HTTP Headers will include:

Content-Type

> application/json

X-User-ID

> ASCII string up to 32 bytes in length and can contain: A-Z,a-z,0-9,-,_
>
> X-User-ID will be base64 encoded.

X-Message-Hash

> X-Message-Hash will be calculated using the secret key from the API user specified by X-User-ID and the JSON payload in the notification.
>
> The hash will be generated using HMAC-SHA256 algorithm - see RFC 2104 for details.
>
> X-Message-Hash will be base64 encoded.

Notification Response

A HTTP response status code 200 is required to acknowledge receipt of the notification.

If 200 is not returned then delivery of the notification will be retried. Other pending notifications will not be attempted until the failed notification message has been successfully delivered.

PAD Transaction States

Notification messages will be issued for following transaction state updates:

Cancelled

PAD debit transaction is cancelled prior to submission to the bank.

Pending

PAD debit transaction is updated to Pending status after previous cancellation.

Rejected

EFT debit record is rejected by bank at time of submission.

Paid

Merchant credit is issued.

Returned

Rejected/returned item is received from the bank.

Message Fields

The table below details the fields that can be included in the notification messages.

Field	Description
terminal_id Type: String Length: 8	**TERMINAL ID** Unique identifier provided in the original transaction.
transaction_type	**TRANSACTION TYPE** The transaction type provided in the original transaction. Values: pad_debit pad_debit_reprocess pad_refund pad_account_verify
reference_number Type: String Length: 15	**REFERENCE NUMBER** Reference number provided in the original transaction.
amount Type: Numeric Length: 12	**AMOUNT** Amount provided in the original transaction.
client_id Type: String Length: 29	**CLIENT ID** Unique value representing the client/consumer provided in the original transaction. This field is conditional, and will exists only if a client id was provided in the original transaction.
charge_description Type: String Length: 30	**CHARGE DESCRIPTION** Description from the original transaction if provided. This field is conditional, and will exists only if a charge description was provided in the original transaction.
effective_date Type: String Length: 10	**EFFECTIVE DATE** The effective date from the original transaction if provided. Format: YYYY-MM-DD This field is conditional, and will exists only if an effective date was provided in the original transaction.

... continued on next page

Field	Description
transaction_state	**TRANSACTION STATE** Values: cancelled pending rejected returned paid
transaction_state_epoch Type: Numeric Length: 12	**TRANSACTION STATE EPOCH** Date/time when the notification was created. This is a standard Unix epoch value.
reason_code Type: String Length: 5	**REASON CODE** The reason code provided by the bank in the EFT reject/return.
reason_text Type: String Length: 128	**REASON TEXT** The reject/return message associated with the EFT reject/return reason code.

Examples

Paid Example

```
{
    "terminal_id":"EXAMPLE1",
    "client_id":"TEST-EXAMPLE1",
    "transaction_type":"pad_debit",
    "reference_number":"REF12345",
    "amount":1000,
    "transaction_state":"paid",
    "transaction_state_epoch":1547138041
}
```

Reject/Returned Example

```
{
    "terminal_id":"EXAMPLE1",
    "client_id":"TEST-EXAMPLE1",
    "transaction_type":"pad_debit",
    "reference_number":"REF12345",
    "amount":1000,
    "transaction_state":"rejected",
    "transaction_state_epoch":1547138041,
    "reason_code":"900",
    "reason_text":"08-ACCOUNT NO. INVALID"
}
```


Card Payment Processing

How Credit Card Processing Works

The credit card charging process consists of three stages: authorization, settlement, and deposit.

Authorization is where Payroc asks the card issuer to tell us if there is enough open-to-buy credit available on the card to make the purchase. A successful authorization lowers the card's open-to-buy limit, but not the card's available credit. This open-to-buy reduction will be restored if there is no subsequent deposit or reversal for that authorization. The time it takes for the open-to-buy to be automatically restored is usually around five days, but can be shorter or longer depending on the card issuer.

Settlement is the process where all of the transactions that will cause funds to move are gathered up for inclusion in the day's deposit files. This is also known as a batch close. Depending on your setup, this can be time-based (automatic), or merchant-initiated.

Deposit is where we collect all of the transaction batches marked for deposit and send them in deposit files to the merchant's bank. Once at the bank, the funds are placed into the merchant's account. The charged credit cards' available credit is affected at this point. Payroc will deposit all settled transactions daily.

Types of Credit Card Processing

Payroc supports the following types of card payment processing:

Standard card payment processing is what most merchants think of when they are considering the acceptance of card payments where no fees are passed to the consumer.

Surcharge, also known as Checkout fee, where the merchant can charge a fee to the consumer to cover their processing costs for eligible credit cards.

UserPay, also known as a Convenience fee or Service fee, where merchants in specific industries can charge a fee to consumers for card payments.

Tokenization

What is Tokenization?

Tokenization is the process of exchanging credit card account data for a token which "stands in" for the credit card data for subsequent transactions sent to Payroc.

The main benefit of tokenization is that the merchant no longer has to store their own credit card data; they store a token instead. Payroc will retrieve the card data associated with the token in order to perform transactions against the card. The credit card account data never again needs to be sent by the merchant once a token is acquired. The card data is stored in Payroc's secure credit card vault, which is an encrypted, protected database designed to be PCI compliant.

Benefits Of Using Tokenization

Tokenization offers many benefits for the merchant:

- In case of a merchant data security breach or information leak, no card numbers can be stolen.

- The merchant does not have to deal with encryption and key management, including periodic PCI-mandated key changes.
- Recurring payments are simplified due to the fact that account numbers can be used as tokens in place of credit card data.
- Payment applications can process transactions such as Completions and Returns without card numbers, greatly simplifying PA-DSS compliance.
- Third party developers can work on merchants' e-commerce applications without the risk of them gaining access to sensitive customer credit card data.
- The risk to your company's reputation is greatly diminished in the event of a merchant data security breach.
- The merchant will be able to make use of future developments in Payroc's tokenization solutions, including tokenization of other sensitive data.
- PCI compliance is made much easier and less costly for the merchant using tokenization since the card storage zone is now in Payroc's hands.

How Payroc's Tokenization Solution Works

Associating Tokens With Cards

When a merchant has collected credit card information and wishes to convert it into a token, they will send a transaction to Payroc containing the credit card number and expiry date. The merchant can specify the token that they would like to use to refer to the credit card account or the token can be automatically generated by Payroc. The card information will be encrypted by Payroc and stored in our secure token vault. This token will now be available to the merchant for charging of the customer's credit card. At Payroc, each token is associated with a particular company or merchant.

Charging Cards Using Tokens

Once a token is in Payroc's secure credit card vault it can be used to charge the associated card. At this point, transactions being sent to Payroc's authorization gateway no longer require a card number and expiry date; a token is sent in their place. The token can be used for all types of transactions, including Sales, Returns, Voids, Pre-Authorizations, and Completions.

Storage of the token at the merchant site is much safer than storing the real card data, since the token can only be used to initiate charges between Payroc's merchant and the stored card number. A hacker or thief cannot directly profit from the tokenized card data. Even if the merchant's financial system is compromised, any transactions performed by intruders using that system can be corrected upon detection by the merchant and Payroc. The cardholder's data is always safe.

Modifying Token Data

From time to time, certain data associated with a token may need to be updated. A customer's card may expire, or the merchant may want to change the card number on file. Payroc provides a transaction to make either of these modifications. The token name and the new credit card information are sent to Payroc and we update the information in our vault immediately. Tokens can also be deactivated and reactivated in the same manner.

Example Of Tokenization Use

Here is a very simple example of the use of tokenization for payment processing:

Imagine that you run a small telephone company. In order to keep things simple, your customer's account number in your billing system is the same as their phone number. Bob Smith has the phone number 416-555-1212, so this is his account number in your company's system.

When Bob signs up for service with your phone company and tells you he would like to pay by credit card, your billing system sends Payroc a transaction requesting that his credit card information be tokenized. The tokenization transaction to Payroc contains the credit card number and the expiry date, as well as Bob's account number "416-555-1212".

Payroc's card processing system sends a reply back to your company's billing system confirming that we now have the customer's payment information. At this point, the phone company's system can discard the credit card data. The token "416-555-1212" will now be sufficient information to charge Bob Smith's card.

A few months later, Bob's card is about to expire. Updating the expiry date on file at Payroc is as simple as sending the token with a new expiry date. The merchant does not need to collect the credit card number again.

Auto-Generated Token IDs

Merchants can specify the Token ID values to be associated with their customers' card information or unique Token ID values can be automatically generated by Payroc.

A combination of Payroc auto-generated Token IDs and merchant assigned Token ID values can be used if there are multiple processes where Tokenization is being implemented.

If the merchant chooses to implement auto-generated Token IDs then they can select the maximum length and format of the Token IDs to be created. Length and format to be used for auto-generated Token IDs will be setup as parameters when Tokenization is enabled for the merchant.

Options available for the format of auto-generated token IDs are:

- Merchant can specify a prefix to be included in the Token ID value
- Card type can be included in the Token ID suffix
- Last 4 digits of the card number can be included in the Token ID suffix

Auto-generated Token IDs are created by including a "?" in the TOKEN field when submitting a Token Add request. See Token Transactions in the Request Examples section for more details on the formatting of auto-generated Token IDs.

Card Authentication

Card Security Code (CSC) & AVS authentication can be performed by including additional fields when submitting a Token Add or Token Update request.

If the CSC and/or AVS fields are included in a request then the values will be sent to the card issuer for authentication prior to adding or updating a token. See the Address Verification Service (AVS) and Card Security Code (CSC) Authentication sections for additional details.

If either the CSC or AVS authentication fails then the token operation (Add or Update) will not be executed. If the authentication is successful then the requested token operation will be processed.

Payment Scheduling

A payment schedule can be created for a Token where the transactions will be automatically processed by Payroc based on the frequency specified by the merchant.

See Card Payment Transaction Types, Card Payment Request Fields, and Payment Schedule Transactions in the Request Examples section for more information regarding Payment Scheduling requests.

Enhanced Data (Level 2/3) Processing

Enhanced Data, which is also known as Level 2/3 Data, can be included when processing payments for commercial cards in order to provide the cardholder with invoice details.

Addtitional information including fields required for Visa, MasterCard, and American Express is available in **Enhanced Data (Level 2/3) Processing** documentation.

See Card Payment Request Fields and Level 2/Level 3 Transaction in the Request Examples section for more information regarding Level 2/3 processing.

Address Verification Service (AVS)

Address Verification Service (AVS) allows cardholder address information to be included with a credit card transaction or token operation for comparison with the address that the card issuer has on file.

AVS is currently available for Visa, MasterCard, American Express, and Discover.

The following rules apply:

- Maximum AVS data length of 29 characters including letters, numbers, and hyphen (-)
- Address information can include street address or PO box, and ZIP/postal code
- Street address can include street number and street name or only street number
- If the ZIP/postal code is included then it must be placed in the last 5-9 characters
- Numbered street names cannot be spelled out, e.g.: Third Street must be submitted as 3RDSTREET
- If a PO box is being submitted then a hyphen (-) must be placed between the PO box and the ZIP code/postal code, e.g.: POBOX1234-M9M9M9, POBOX1234-99999

AVS Examples

Address of 27 Mill St East, Toronto, ON, J8Z 3N5 can be submitted as follows:

27MILLJ8Z3N5
27J8Z3N5
J8Z3N5

AVS Result Codes

Result of the address verification will be returned in the AVS response field.

List of possible AVS result codes are provided in the tables below. Separate tables have been provided based on type of result.

AVS Result Codes - Match

All of the result codes listed below specify a case where the postal/zip code matched the issuer's records.

Result Code	Description
W	Postal code match, street address not provided in the request or does not match
X	Postal code match, street address match or not provided in the request
Y	Postal code match, street address match or not provided in the request
Z	Postal code match, street address not provided in the request or does not match

AVS Result Codes - Address Match Only

Result codes listed below indicate that only the street address matched the issuer's records. If only postal/zip code was included in the request then these result codes should be considered as no AVS match.

Result Code	Description
A	Street address match only

AVS Result Codes - No Match

Result codes listed below indicate that the both the street address and postal/zip code do not match the issuer's records. Some issuers may return the same result code when only street address or postal/zip code was included in the request.

Result Code	Description
N	Street address and ZIP code/postal code do not match

AVS Result Codes - Not Processed

All of the result codes included in this table indicate that the issuer did not verify the address information.

Result Code	Description
G	Address information not verified
R	Issuer/service not available
S	AVS service not supported by issuer
U	No AVS response received from issuer

AVS Testing

For AVS testing, the AVS result code returned will be based on the first character of the AVS data included in the request.

The first character of the AVS data can be set to either a number or letter to perform testing for different formats of postal/zip codes - see the 1st Digit and 1st Letter columns included in the tables provided below.

AVS Result Codes - Match

1st Digit	1st Letter	Result Code	Description
4	E	W	Postal code match, street address not provided in the request or does not match
5	F	X	Postal code match, street address match or not provided in the request

... continued on next page

1st Digit	1st Letter	Result Code	Description
6	G	Y	Postal code match, street address match or not provided in the request
7	H	Z	Postal code match, street address not provided in the request or does not match

AVS Result Codes - Address Match Only

1st Digit	1st Letter	Result Code	Description
8	I	A	Street address match only

AVS Result Codes - No Match

1st Digit	1st Letter	Result Code	Description
9	K	N	Street address and ZIP code/postal code do not match

AVS Result Codes - Not Processed

1st Digit	1st Letter	Result Code	Description
	M	G	Address information not verified
	P	R	Issuer/service not available
0	Q	S	AVS service not supported by issuer
	R	U	No AVS response received from issuer

Card Security Code/CVV Authentication

Card Security Code (CSC) authentication allows the security code printed on a card to be included with a credit or Token transaction for comparison with the value that the card issuer has on file.

Card Security Code is commonly referred to as CVV within the payment industry.

CSC authentication is currently available for Visa, MasterCard, American Express, and Discover.

CSC Result Codes

The following table provides a list of possible result codes.

Result Code	Description
Y	Match (American Express cards)
M	Match (Visa, MasterCard and Discover cards)
N	No Match
P	Not Processed

... continued on next page

Result Code	Description
S	Unexpected Issuer Response
U	Issuer does not participate in CSC service
X	Unexpected Issuer Response

CSC Testing

For CSC testing, the CSC result code will depend on the first digit of the submitted CSC data.

1st Digit	Result Code	Description
0		Reserved (No CSC result code will be returned)
1		Reserved (No CSC result code will be returned)
2		Reserved (No CSC result code will be returned)
3	Y	Match (American Express cards)
4	M	Match (Visa, MasterCard and Discover cards)
5	N	No Match
6	P	Not Processed
7	S	Unexpected Issuer Response
8	U	Issuer does not participate in CSC service
9	X	Unexpected Issuer Response

Test transactions are approved or declined based on the CSC included in the request. See the Card Payments Testing Transactions section for additional details.

Magnetic Card Track Data Field Formats

When processing transactions where the card information is read by a magnetic swipe reader, there is no need to submit the card number or expiry date fields, since these are embedded in the card swipe data.

Credit cards typically contain two magnetic "tracks" of data: track 1 and track 2. It is possible to submit either track 1 or track 2 data for authorizations. Only one track should be submitted per transaction.

Sending Track 1

Track 1 data is variable length, with a maximum length of 79 characters. In order to submit this track, you must first remove the framing characters (start/end sentinels and LRC character).

The separators should be converted to "^" (hex 5E) or the ASCII Unit Separator character (hex 1F) prior to sending, but most card readers will do this automatically.

Example - Track 1 Contents:

```
B4000********7659^PRESLEY/GREGORY^03081010000000010001006400000000
```


Sending Track 2

Track 2 data is variable length, with a maximum of 37 characters. In order to submit this track, you must first remove the framing characters (start/end sentinels and LRC character). The separators must be converted to "=" (hex 3D) or "D" (hex 44) prior to sending, but most card readers will do this automatically.

Example - Track 2 Contents:

```
4000********7659=03081010000064010001
```

User Pay

User Pay Introduction

Eligible merchants are permitted to assess a web payment fee to accept credit and debit cards as a form of payment. Cardholders must be notified of the fee at the time of payment and be given the opportunity to opt out of the sale.

The following functionality is supported for the User Pay:

- Web payment fees can be assessed by establishing different rates by card type or card product
- Separate charges will be processed for the fee and payment amounts
- Transactions can be processed using either tokens or card numbers
- CVV & AVS authentication can be included when authorizing the amounts to be charged to the cardholder

To minimize customer service inquires, the web payment fee charged must be processed as a separate and unique transaction and cannot be included in the total amount for the product or service paid for.

User Pay Processing Flow

The User Pay process flow is a two-step process where the fee amount is calculated first followed by the processing of the payment.

Example of processing flow:

- Invoice/bill amount to be paid is determined by the merchant application
- Merchant sends a calculate request to determine the amount of the fees to be paid by the cardholder
- The fee amount is calculated by Payroc and the response is returned to the Merchant’s server
- The fee amount and total to be paid is presented to the cardholder for their approval
- Cardholder enters their card information or a token can be selected which is to be used for payment
- The payment request is sent to Payroc for authorization
- The payment response is returned to the Merchant’s server indicating whether the transaction(s) has been approved or declined

The processing flow may vary based on the functionality being implemented or to simplify integration with the merchant’s existing processes.

Transaction Types

Account Status Inquiry

Check the status of a card

Authorization Reversal

Adjust the authorization amount after the transaction has taken place

Authorize Only

Request authorization only

Commercial Card Check

Check to see if a particular card is a commercial card

Completion

Process a charge associated with a previously approved Pre-Authorization transaction

Contra Add

Add a card number to a list of cards that you do not want to accept

Contra Delete

Remove a card number from the list of cards that you do not want to accept

Contra Query

Check to see if a particular card number is present in the Terminal ID's Contra table

Force Post

Transaction for settlement for which you have previously obtained an authorization code

Level 3 Details

Submit Level 3 detail line items for a previously authorized transaction containing Level 2 data

Pre-Authorization

Obtain approval when an order is placed

Return

Issue a refund to the customer's card if goods are returned

Return Void

Cancel a previously submitted Return transaction

Sale

Charge the specified amount to the credit card (if approved) and marks the transaction for deposit during the next settlement period

Settlement

Mark the batch for deposit and reset the totals for the terminal ID and operator ID pair

Void

Cancel or remove a previously authorized Sale, Completion, or Force Post transaction

Card Token Add

Create a token

Card Token Deactivate

Deactivate an existing token

Card Token Reactivate

Reactivate an existing token

Card Token Update

Update an existing token

Payment Schedule Add

Create a payment schedule

Payment Schedule Deactivate

Deactivate an existing payment schedule

Payment Schedule Reactivate

Reactivate an existing payment schedule

Payment Schedule Update

Update an existing payment schedule

Fee Calculation

Calculate the fee for given transaction amount based on merchant setup

Fee Payment

Charge the specified transaction amount and fee amount to the card (if approved) and marks the transactions for deposit during the next settlement period

Fee Void

Cancel or remove the previously authorized fee payment transactions

Request Fields

The table below details the fields that can be included in transaction request messages.

Field	Description
terminal_id Type: String Length: 8	**TERMINAL ID** Unique identifier provided by Payroc for a merchant/location.
transaction_type	**TRANSACTION TYPE** Values: card_account_status_inquiry card_authorization_only card_authorization_reversal card_commercial_card_check card_completion card_contra_add card_contra_delete card_contra_query card_force_post card_invoice_line_item card_preauthorization card_return card_return_void card_sale card_settlement card_void schedule_add schedule_deactivate schedule_reactivate schedule_update token_add token_deactivate token_reactivate token_update fee_calculate fee_payment fee_void

card_product Type: String Length: 2	**CARD PRODUCT** This field is only used for fee_calculate and fee_payment transactions. Used to indicate the card type or card product to be used for payment. This field can be used if a fee is to be calculated based on card type or product, i.e., optional field to be used in cases where the Card or Token information has not yet been captured. Values to be used for card type: VC - Visa MC - MasterCard Values to be used for card product: VC - Visa credit VD - Visa debit VB - Visa business MC - MasterCard credit MD - MasterCard debit MB - MasterCard business
device_id Type: Integer Length: 8	**DEVICE ID** Device ID is required for Retail transactions to identify the POS Terminal used to process the payment
echo_data Type: String Length: 60	**ECHO DATA** Data that is submitted in this field is returned in the reply. This can be useful for transaction tracking in single-process or batch applications. This field's submitted data is returned in the echo_data field of the transaction response.
fee_amount Type: Integer Length: 8	**FEE AMOUNT** This field is only used for fee_payment and fee_void transactions. Fee amount as received in the fee calculate response. If fee_calc_override=Y this field will specify the fee amount to be charged.
fee_calc_override Type: String Length: 1	**FEE CALCULATION OVERRIDE** This field is only used for fee_payment transaction. Use of fee_calc_override must be approved to ensure adherence with the associated card brand rules. Indicates whether to override the fee calculate validation for the payment transaction. If Y is specified then a fee_calculate transaction is not required. Value: Y

fee_reference Type: String Length: 30	**FEE REFERENCE NUMBER** This field is only used for fee_calculate, fee_payment and fee_void transactions. Unique reference number associated with the fee calculate request. If this reference number does not match the fee calculate request then the payment request will be rejected. Characters supported: A-Z a-z 0-9 - / \
operator_id Type: String Length: 3	**OPERATOR ID** The operator ID is a container within the terminal ID. Usually "001" or unspecified (empty). Defaults to "001" if unspecified. The operator ID is 3 characters, alphanumeric. An operator ID is created by its first use. Sending a transaction with operator "01A" will cause that operator to be created. If the operator ID is omitted, the default is "001". For terminal IDs which Payroc automatically settles nightly, operator rotation is performed. Operator rotation is where Payroc automatically switches the operator ID for each day of the week. For Sunday, operator "001" is used, Monday is "002", and so on. Using this system, transaction processing days can be isolated for reporting and corrections. Since operator "003" is only used for Wednesday, problems with Wednesday's transactions will not interfere with Thursday's processing, which will automatically occur on operator "004". For most Payroc merchants, this field is not sent with the transaction.
password Type: String Length: 16	**PASSWORD** For added security, your terminal ID can be set up with a password. This password, if enabled, is required for all transactions to this terminal ID. PASSWORD should not be provided if this feature is not enabled on your terminal ID. If this feature is enabled, you must provide your password for every transaction. All terminal IDs must either come from a fixed IP address or use a password.
reference Type: String Length: 60	**REFERENCE NUMBER** This is a reference number associated with a transaction. The reference number should be unique in order to facilitate transaction tracking, Voids, and searches. This value MUST be unique if you are sending any Level 3 data. The reference number can be up to 60 characters (alphanumeric with certain other characters allowed) and may not contain spaces. The reference number may contain hyphens ("-") and slashes ("/"). Certain restrictions may be placed on the reference number depending on your individual situation. Important: If you are going to be processing Completion transactions without supplying the credit card number or a token, the reference number MUST be unique within the past 30 days. Besides card number or token, reference number is the only key with which Payroc can match a Completion back to its corresponding Pre-Authorization. Note: this is not the reference number which appears on the cardholder's statement. Payroc has no control over the statement reference number.

resend Type: String Length: 1	**RESEND** Values: Y N Default: N This field indicates whether a transaction has been previously sent, and is used to retrieve the original response. You can set this field to "Y" (uppercase) in order to receive the response to an already submitted transaction within the past 48 hours. If operator rotation is in use you can only receive responses for transactions submitted since midnight on the current day. If Payroc did not receive the original transaction, it will be processed as a new transaction. You may resubmit the transaction as many times as necessary in order to receive your response. Please note that if you have to do this often you may have network problems. Please note that if the original transaction is not yet finished, the RESEND operation will not work since transactions are only available for the RESEND operation after the response has been sent. This means that you cannot use this to prevent customers from being charged twice when they double-click the SUBMIT button on an e-commerce site.
show_duplicate_status Type: String Length: 1	**SHOW DUPLICATE STATUS** Values: Y N Default: N This is the show duplicate status flag. If you submit "Y", your transaction response will contain a flag indicating whether this transaction is a duplicate or not. This must be used in conjunction with resend flag above. The field name returned is called *duplicate_transaction* The returned response will contain "Y" if the transaction was a duplicate, or "N" if not. "Y" means that you are being shown a previous transaction response. The transaction is not processed for a second time in either case.
total_amount Type: Integer Length: 10	**TOTAL AMOUNT** This field is only used for fee_payment and fee_void transactions. The total amount to be charged to the card.
unique_id Type: String Length: 7	**LEVEL 2/3 UNIQUE IDENTIFIER** This is a unique identifier returned in the response when you submit a Level 2 enhanced data transaction with a commercial card. You must include this field when submitting details (Level 3 Details). This field is used to match the Level 2 and Level 3 data at the credit card issuing institution.

Payment Object

Field		Description
payment		
	amount Type: Integer Length: 12	**AMOUNT** The transaction amount for the transaction with no currency signs or decimal. Example "100" would be one dollar on a Canadian funds account, and would be one Euro on a Euro account
	authorization_code Type: String Length: 6	**AUTHORIZATION NUMBER** This field is used for Force Post transactions only. When you already have an authorization number for a particular transaction, there is no need to obtain another. A Force Post transaction will allow you to deposit transaction charges for which you already have an authorization number. In order to do this, you will need to provide the original authorization number in this field. The authorization number can be from 2 to 6 characters long, alphanumeric.
	detailed_payment_type Type: String Length: 1	**DETAILED PAYMENT TYPE** This field is required for PreAuthorization and Sale transactions processed using a Stored Credential/Token. This field is used to indicate whether a transaction using a stored credential/token is a recurring payment, an unscheduled/ad hoc transaction, a partial shipment, or a resubmission. Partial Shipment: when an agreed amount of goods is ordered and authorized using multiple transactions then the requests need to be flagged as a partial shipment. Example: Merchant checks availability after receiving an order for 5 items and only 3 items are currently available. Merchant submits an authorization request for the 3 available items with detailed_payment_type = P (Partial Shipment). Authorization request is submitted with detailed_payment_type = P when the remaining items become available. Resubmission: A previous attempt to obtain authorization for a transaction has been declined but the issuer's response does not prohibit the merchant from trying again later. Example: Insufficient Funds response is received for a transaction where authorization retry is allowed. Merchant retries the authorization request with detailed_payment_type = X (Resubmission). Values: R P U X 'R' - recurring payment transaction 'P' - partial shipment 'U' - unscheduled/ad hoc transaction 'X' - resubmission Default: U If this field is omitted then default value of 'U' will be used if a Stored Credential/Token is being used for transaction processing.

	dynamic_dba Type: String Length: 25	**DYNAMIC MERCHANT DESCRIPTION** This field contains a custom merchant description that will appear on the cardholder's statement. This will override your default merchant description. You must be set up ahead of time in order to use this feature. Limits on length vary by card type: Visa: 25 characters MasterCard: 22 characters American Express: 20 characters This feature may not be available on all merchant setups.
	initiator Type: String Length: 1	**PAYMENT INITIATOR** This field is required for PreAuthorization and Sale transactions processed using a Stored Credential/Token. This field is used to indicate whether a transaction using a stored credential/token is a cardholder-initiated transaction or a merchant-initiated transaction. A cardholder-initiated transaction is any transaction where the cardholder is actively participating in the transaction. A merchant-initiated transaction is any transaction where the cardholder is not actively participating in the transaction. Values: C M 'C' - cardholder-initiated transaction 'M' - merchant-initiated transaction Default: C If this field is omitted then default value of 'C' will be used if a Stored Credential/Token is being used for transaction processing.
	surcharge_amount Type: Integer Length: 8	**SURCHARGE AMOUNT** This field is only support by the Worldnet gateway. Please contact Payroc Merchant Services department for assistance. If surcharge applies to a payment then the surcharge amount must be included in the transaction amount, e.g., if the payment amount is $100.00 and surcharge amount is $2.00 then the payment.amount would be 10200 and the payment.surcharge_amount would be 200. The surcharge amount for the transaction with no currency signs or decimal. Example "100" would be one dollar
	recipient	**FUNDING TRANSACTION RECIPIENT DATA** Object that contains the name of the funding recipient. Send this object when you need to provide the name of the recipient, for example, when you send funds by wire transfer.

		name_first Type: String Length: 35	**FIRST NAME** Recipient's first name.
		name_middle Type: String Length: 1	**MIDDLE NAME** First letter of the recipient's middle name.
		name_last Type: String Length: 35	**LAST NAME** Recipient's last name.

Card Information Object

Field		Description
card_information		
	avs_data Type: String Length: 25	**ADDRESS VERFICATION SERVICE DATA** Address information is submitted for verification by including this field in a credit card request message or token operation.
	card_number Type: String Length: 19	**CREDIT CARD NUMBER** The credit card number or PAN to be charged. Digits only. No spaces, dashes, hyphens or any other punctuation are allowed. Can be omitted if a Token is being used to process a payment transaction. Required if creating a token. Optional if updating a token.
	cardholder_name Type: String Length: 40	**CARDHOLDER NAME** The cardholder name can be submitted for authentication as embossed on the face of the card. Cardholder name authentication is currently only supported for Discover Characters supported: A-Z 0-9 . ' - _ / \ and space

	csc Type: Integer Length: 4	**CARD SECURITY CODE** This 3 or 4 digit number is used to ensure the physical presence of a card in an environment where the cardholder is not in front of the merchant at the time of the purchase, such as during an Internet or phone transaction. This value appears as additional numbers following the credit card number which is printed within the signature panel on the back of the card. For some card types, the security code is on the front of the card near the card number. This field is used for: VISA: CVV2 MasterCard: CVC2 American Express: CID Discover: CID
	expiry_year Type: Integer Length: 4	**Expiry Year** The expiry year on the card. The expiry year must be specified in the form YYYY, and must not contain any characters other than the digits. Can be omitted if a Token is being used to process a payment transaction. Can be omitted for Completion, Void and Authorization Reversal transactions without card numbers or token data. Required if creating a token. Optional if updating a token. Format: YYYY
	expiry_month Type: Integer Length: 2	**Expiry Month** Values: 1-12 The expiry month on the card. Can be omitted if a Token is being used to process a payment transaction. Can be omitted for Completion, Void and Authorization Reversal transactions without card numbers or token data. Required if creating a token. Optional if updating a token.
	track_data Type: String Length: 79	**TRACK 1 OR 2 DATA** If you are not sure which magnetic track you are reading from the card, you may send it in this field. Our system will identify the correct track type and use it. *card_number*, *expiry_year*, *expiry_month* and *token* fields should be omitted if track data is sent.
	track1_data Type: String Length: 79	**TRACK 1 DATA** Unaltered track 1 data from the card magnetic stripe. *card_number*, *expiry_year*, *expiry_month* and *token* fields should be omitted if track 1 data is sent.
	track2_data Type: String Length: 37	**TRACK 2 DATA** Unaltered track 2 data from the card magnetic stripe. *card_number*, *expiry_year*, *expiry_month* and *token* fields should be omitted if track 2 data is sent.

Token Object

Field		Description
token		
	token Type: String Length: 30	**TOKEN ID** A unique ID that can be created and used in place of a credit card for future payment processing. For *Token Operations*, the token ID value can be specified by the merchant and/or automatically generated by Payroc. Submitting a token ID value containing a "?" specifies that the token ID value is to be generated by Payroc. Token IDs will be generated by Payroc based on configuration parameters chosen by the merchant. 12 characters is the minimum length for auto-generated Token IDs A token ID value must be specified for Payment Scheduling transactions. Characters supported: 0-9 A-Z : @ | - + / _ ,
	client_id Type: String Length: 30	**TOKEN CLIENT ID** This field can be used to include additional reference data with a token such as a client identifier. Characters supported: A-Z a-z 0-9 - _ . @ /
	start_date Type: String Length: 8	**TOKEN START DATE** The effective start date for the token. This token will not be accepted for payments before this date. Format: YYYYMMDD
	end_date Type: String Length: 8	**TOKEN END DATE** The effective end date for the token. This token will not be accepted for payments after this date. Format: YYYYMMDD
	reference Type: String Length: 30	**TOKEN REFERENCE DATA** Reference data to associate with a token. Characters supported: A-Z a-z 0-9 - / \

	single_use_token Type: String Length: 1	**TOKEN USAGE** To identify the token is used only once. Values: Y N 'Y' - specifies that the third party Token was created with usage restricted to a single payment 'N' - specifies that a third party Token was created as a Stored Credential
	token_card_check Type: String Length: 1	**TOKEN CARD CHECK** To check the card with the token. Values: Y N 'Y' - card check 'N' - no card check

Schedule Object

Field		Description
schedule		
	type	**SCHEDULE TYPE** Values: WEEKLY MONTHLY Specifies the interval for the processing of the card payment transactions. Required if a payment schedule is to be created. Required if updating a schedule and the start date or number of payments are being modified.
	frequency Type: Integer Length: 2	**SCHEDULE FREQUENCY** Values: 1-12 Default: 1 The frequency for the processing of the card payment transactions is set based on the schedule type and the value provided for this field.
	number_of_payments Type: Integer Length: 3	**NUMBER OF PAYMENTS** Specifies the number of card payment transactions to be processed. Required if a payment schedule is to be created Required if updating a schedule and the schedule start date or schedule type are changed.

	start_date Type: Integer Length: 8	**SCHEDULE START DATE** Specifies the date when the first card payment transaction is to be processed. The schedule start date must be a future date, i.e., payments cannot be scheduled to start on the day when the request is submitted. Required if a payment schedule is to be created. Required if updating and number of payments or type are changed. Format: YYYYMMDD
	reference Type: String Length: 50	**SCHEDULE REFERENCE DATA** Reference data to associate with a payment schedule. Characters supported: A-Z a-z 0-9 - / \

Invoice Object (Level 2 Data)

Invoice Object must be included for Sale, Completion and Return transactions if Level 2 Data is to be provided to the cardholder.

Field		Description
invoice		
	version Type: String Length: 6	**LEVEL 2 DATA VERSION** Value: PC3.02
	city_tax_amount Type: Integer Length: 8	**CITY TAX AMOUNT** City tax amount. Two implied decimal places, no currency sign or decimal point to be included. Example: Value of "100" represents $1.00. Amex Only used for U.S. merchant transactions
	city_tax_rate Type: Integer Length: 4	**CITY TAX RATE** City tax rate expressed as a percentage. Two implied decimal places, no decimal point to be included. Example: Value of "700" represents 7%. Amex Only used for U.S. merchant transactions

	county_tax_amount Type: Integer Length: 8	**COUNTY TAX AMOUNT** County tax amount. Two implied decimal places, no currency sign or decimal point to be included. Example: Value of "100" represents $1.00. Amex Only used for U.S. merchant transactions
	county_tax_rate Type: Integer Length: 4	**COUNTY TAX RATE** County tax rate expressed as a percentage. Two implied decimal places, no decimal point to be included. Example: Value of "700" represents 7%. Amex Only used for U.S. merchant transactions
	customer_gst_registration Type: String Length: 15	**CUSTOMER GST/HST REGISTRATION NUMBER** Customer's GST or HST registration number. Visa & Amex Required if registration number is provided
	customer_name Type: String Length: 40	**CUSTOMER NAME** Name of the customer who placed the order. Amex Used for cardholder's name.
	customer_po_number Type: String Length: 22	**CUSTOMER PURCHASE ORDER NUMBER** Identifier provided by the customer such as purchase order number or job number. Required if the customer has provided a PO Number or reference number.
	customer_reference Type: String Length: 17	**CUSTOMER REFERENCE** Cardholder's reference data. Amex Can include client accounting information

	duty_amount Type: Integer Length: 10	**DUTY AMOUNT** Amount of duty to be paid for the invoice. Two implied decimal places, no currency sign or decimal point to be included. Value of "0" can be used. Example: Value of "100" represents $1.00. Visa Not supported for Canada or U.S. Can be included as Invoice Line Item
	gst_amount Type: Integer Length: 10	**GST/HST AMOUNT** Total GST or HST for the transaction. Two implied decimal places, no currency sign or decimal point to be included. Value of "0" is to be used for tax exempt transactions. Example: Value of "100" represents $1.00.
	gst_rate Type: Integer Length: 4	**GST/HST RATE** The GST or HST rate expressed as a percentage. Two implied decimal places, no decimal point to be included. Value of "0" is to be used for tax exempt transactions. Example: Value of "700" represents 7%.
	invoice_number Type: String Length: 22	**Invoice Number** Original record of charge of invoice. Amex Uses invoice_reference if left blank
	invoice_reference Type: String Length: 15	**INVOICE REFERENCE NUMBER** External system transaction ID assigned by the merchant, typically an order or invoice number. Should be unique. Recommended that a value be provided for Visa & MasterCard. Amex Optional invoice_number takes precedence
	invoice_total Type: Integer Length: 10	**INVOICE TOTAL** Total amount of the order including all taxes, shipping & handling, and discount. Recommended that a value be included in order to help troubleshoot any invoice balancing issues.

	line_item_count Type: Integer Length: 3	**LINE ITEM COUNT** Number of invoice line items included in the transaction (1 to 999). Number of line items supported by some card associations and financial institutions may be less than 999. Recommended that a value be included in order to help troubleshoot any invoice balancing issues.
	merchant_gst_registration Type: String Length: 30	**MERCHANT GST/HST REGISTRATION NUMBER** Merchant's GST or HST registration number. GST/HST registration number included in Payroc merchant setup will be used as a default if no value is provided for this field.
	merchant_pst_registration Type: String Length: 30	**MERCHANT PST/QST REGISTRATION NUMBER** Merchant's Provincial Sales Tax registration number. Visa & Amex Required if the customer has provided a registration number
	order_date Type: Integer Length: 8	**ORDER DATE** Date when the order was placed. Format: YYYYMMDD Visa Required if included on the invoice. Must be earlier or same date as the transaction date.
	discount_amount Type: Integer Length: 10	**ORDER DISCOUNT AMOUNT** Total order discount amount. Two implied decimal places, no currency sign or decimal point to be included. Value of "0" can be used. Example: Value of "100" represents $1.00. Visa Required if discount has been applied at invoice level. Value must be zero if the discount is applied at the Invoice Line Item Level.
	pst_amount Type: Integer Length: 10	**PST/QST AMOUNT** Total PST, QST, or U.S. Local Tax for the transaction. Two implied decimal places, no currency sign or decimal point to be included. Required if PST/QST taxes apply. Example: Value of "100" represents $1.00.

	pst_rate Type: Integer Length: 4	**PST/QST RATE** The PST, QST, or U.S. Local Tax rate expressed as a percentage. Two implied decimal places, no decimal point to be included. Required if PST/QST taxes apply. Example: Value of "700" represents 7%.
	requester_buyer_name Type: String Length: 40	**REQUESTER/BUYER NAME** The name of the individual/buyer (not the company) requesting the goods or services. Recommended that a value be provided for all transactions. Visa Required if invoice amount > $150.00
	shipping_amount Type: Integer Length: 10	**SHIPPING & HANDLING AMOUNT** Total shipping & handling for the order. Two implied decimal places, no currency sign or decimal point to be included. Value of "0" can be used. Example: Value of "100" represents $1.00. Visa If charged separately then the amount must be specified at invoice level or as an invoice line. Visa recommends that Shipping & Handling be included as a line item.
	ship_to_address_line_1 Type: String Length: 40	**SHIP TO ADDRESS LINE 1** Amex Required if ship_to_name exists
	ship_to_address_line_2 Type: String Length: 40	**SHIP TO ADDRESS LINE 2**
	ship_to_city Type: String Length: 30	**SHIP TO CITY** Amex Required if ship_to_name exists

	ship_to_country_code Type: String Length: 3	**SHIP TO COUNTRY** Visa Required for international shipments. Defaults to Canada Visa & MasterCard 2 char ISO alpha country code is required Amex Only supports Canada or U.S. Following country codes are supported: 124 or CAN for Canada 840 or USA for U.S.
	ship_to_name Type: String Length: 40	**SHIP TO NAME**
	ship_to_postal_code Type: String Length: 15	**SHIP TO POSTAL CODE/ZIP CODE** Amex Required if ship_to_name exists
	ship_to_prov_state Type: String Length: 2	**SHIP TO PROVINCE/STATE** Amex Required if ship_to_name exists
	state_tax_amount Type: Integer Length: 8	**STATE TAX AMOUNT** State tax amount. Two implied decimal places, no currency sign or decimal point to be included. Example: Value of "100" represents $1.00. Amex Only used for U.S. merchant transactions
	state_tax_rate Type: Integer Length: 4	**STATE TAX RATE** State tax rate expressed as a percentage. Two implied decimal places, no decimal point to be included. Example: Value of "700" represents 7%. Amex Only used for U.S. merchant transactions

	supplementary_data Type: String Length: 163	**SUPPLEMENTARY DATA** For Amex transactions, the Supplementary Data field can be used to pass up to 4 lines of free form text (also known as 4x40). Maximum of 40 characters per text line can be included with each line separated by a semicolon. For MasterCard transactions, optional Merchant Reference Number can be included. Example: Text line 1;Text line 2;Text line 3;Text line 4 Characters supported: 0-9 A-Z a-z ; - spaces

Invoice Line Item Object (Level 3 Data)

Invoice Line Item Object is to be included for Level 3 Details transactions.

Field		Description
invoice_line_item		
	version	**LEVEL 3 DATA VERSION** Value: PC3.03
	sequence_number Type: Integer Length: 3	**LEVEL 3 SEQUENCE NUMBER** This sequence number is used to preserve the correct ordering of Level 3 invoice line item details. Each Level 2 item can have up to 999 detail (Level 3) entries. Some deposit institutions limit this to 99 entries; please check with Payroc Merchant Services if you are unsure. Typically, this sequence number would start at "1". If the sequence number is not specified, Level 3 data could appear in an incorrect order on the customer's statement.
	city_tax_amount Type: Integer Length: 8	**CITY TAX AMOUNT** City tax amount on the line item. Two implied decimal places, no currency sign or decimal point to be included. Example: Value of "100" represents $1.00. Amex Only used for U.S. merchant transactions

	city_tax_rate Type: Integer Length: 4	**CITY TAX RATE** City tax rate expressed as a percentage. Two implied decimal places, no decimal point to be included. Example: Value of "700" represents 7%. Amex Only used for U.S. merchant transactions
	county_tax_amount Type: Integer Length: 8	**COUNTY TAX AMOUNT** County tax amount on the line item. Two implied decimal places, no currency sign or decimal point to be included. Example: Value of "100" represents $1.00. Amex Only used for U.S. merchant transactions
	county_tax_rate Type: Integer Length: 4	**COUNTY TAX RATE** County tax rate expressed as a percentage. Two implied decimal places, no decimal point to be included. Example: Value of "700" represents 7%. Amex Only used for U.S. merchant transactions
	customer_po Type: String Length: 22	**CUSTOMER PO** Customer purchase order number for specific item.
	customer_sku Type: String Length: 30	**CUSTOMER SKU NUMBER** Customer's stock number for item being sold/returned.
	discount_amount Type: Integer Length: 10	**LINE ITEM DISCOUNT AMOUNT** Two implied decimal places, no currency sign or decimal point to be included. decimal point to be included. Example: Value of "100" represents $1.00. Visa Required if discount has been applied at Line Item level

	division_number Type: String Length: 40	**DIVISION NUMBER** This field may contain the division number of the card member who purchased the item or service.
	extended_item_amount Type: Integer Length: 10	**EXTENDED ITEM AMOUNT** Two implied decimal places, no currency sign or decimal point to be included. Value of "0" can be used for zero cost line items. Example: Value of "100" represents $1.00. Visa & MasterCard calculated as: Unit Cost * Quantity - Line Item Discount Amex calculated as: Unit Cost * Quantity
	gl_account_number Type: String Length: 40	**GL ACCOUNT NUMBER** This field may contain the general ledger account number for the item/service purchased.
	gst_amount Type: Integer Length: 10	**GST/HST AMOUNT** Total GST or HST for the line item. Two implied decimal places, no currency sign or decimal point to be included. Value of "0" is to be used. Example: Value of "100" represents $1.00. Visa Required if taxes are calculated at line item level
	gst_rate Type: Integer Length: 4	**GST/HST RATE** The GST or HST rate expressed as a percentage. Two implied decimal places, no decimal point to be included. Value of "0" is to be used for tax exempt or zero cost line items. Example: Value of "700" represents 7%. Visa Required even if taxes are calculated at invoice level

	item_commodity_code Type: String Length: 12	**ITEM COMMODITY CODE** Item commodity code which identifies the type of purchase. Visa Value to be provided if required by the merchant's business
	item_description Type: String Length: 26	**ITEM DESCRIPTION** Merchant's description of the item being sold/returned.
	manufacturer_sku Type: String Length: 30	**MANUFACTURER SKU NUMBER** Manufacturer's stock number for item being sold/returned. sold/returned.
	po_line_number Type: Integer Length: 5	**PO LINE NUMBER** This field may contain the PO line number.
	product_code Type: String Length: 12	**PRODUCT CODE** Merchant's SKU number for the item being sold/returned.
	pst_amount Type: Integer Length: 10	**PST/QST AMOUNT** Total PST, QST, or U.S. Local Tax for the line item. Two implied decimal places, no currency sign or decimal point to be included. Value of "0" can be used. Example: Value of "100" represents $1.00.
	pst_rate Type: Integer Length: 4	**PST/QST RATE** The PST, QST, or U.S. Local Tax rate expressed as a percentage. Two implied decimal places, no decimal point to be included. Value of "0" is to be used for tax exempt or zero cost line items. Example: Value of "700" represents 7%.

	quantity Type: Integer Length: 10	**QUANTITY** Number of units sold or returned. Two implied decimal places, no decimal point to be included. Example: Value of "100" represents 1 unit. Visa "0" can be used for zero cost line items MasterCard & Amex "0" cannot be used
	state_tax_amount Type: Intgers Length: 8	**STATE TAX AMOUNT** State tax amount on the line item. Two implied decimal places, no currency sign or decimal point to be included. Example: Value of "100" represents $1.00. Amex Only used for U.S. merchant transactions
	state_tax_rate Type: Integer Length: 4	**STATE TAX RATE** State tax rate expressed as a percentage. Two implied decimal places, no decimal point to be included. Example: Value of "700" represents 7%. Amex Only used for U.S. merchant transactions
	supplementary_data Type: String Length: 40	**SUPPLEMENTARY DATA/ORDER NUMBER** This field may contain a free form product or service description for the line item.
	merchant_reference Type: String Length: 15	**MERCHANT REFERENCE NUMBER** Unique transaction ID assigned by the merchant.
	unit_cost Type: Integer Length: 10	**UNIT COST** Cost per unit of measure. Two implied decimal places, no currency sign or decimal point to be included. Example: Value of "100" represents $1.00.

	unit_of_measure Type: String Length: 12	**UNIT OF MEASURE** Unit of measure indicating how the item is sold. Example: case, each, bucket, drum Amex Unit of measure code must be expressed as 2 char code defined in ASC X12 (EDI) Standard. Examples of ASC X12 codes: EA=Each BX=Box PK=Pack PC=Piece E5=Inches Value of "EA" will be used if valid code is not provided

Response Fields

All transactions return a message field. Other response fields may be returned depending on the type of transaction submitted.

The reason code field will contain a 6-digit reason code for transactions that were not successful. Response for some non-card-charging transactions may return reason code for some conditions that do not necessarily indicate failure.

See section Card Payment Reason Codes for a list of values generated by Payroc.

Response Field	Description
message Type: String Length: 128	**MESSAGE** This is the response text of the message. For successful transactions this field will be blank. For errors and decline responses this field contains the error message or decline message. The message may include a two character response code. See section Card Payment Response Codes for a list of values returned from MasterCard or Visa
reason_code Type: String Length: 6	**REASON CODE** 6-digit reason code will be returned code if the transaction was not successfully processed. Reason code may indicate that an authorization request was declined, card status is bad, or a transaction was rejected due to an error condition.
authorization_code Type: String Length: 6	**AUTHORIZATION NUMBER** This field provides the authorization number for transaction types such as Sale, Pre-Authorization, Completion, Force Post, Authorization Only.

... continued on next page

Response Field	Description
unique_id Type: String Length: 7	**COMMERCIAL CARD TRANSACTION UNIQUE IDENTIFIER** When a financial transaction with Level 2 data is sent and the card presented is a commercial card, a unique identifier will be returned in this field. You must include this ID in any subsequent details (Level 3 Detail Submission) to be delivered for this transaction.
avs_result Type: String Length: 1	**ADDRESS VERIFICATION RESULT CODE** For transactions where AVS data was submitted, this field returns the AVS result code. 1-character code indicates if the address was a match, partial match, or no match See the AVS Result Codes section for values.
echo_data Type: String Length: 60	**ECHO DATA** If you submitted a transaction with the ECHO field, this field will be present in the reply and will contain the exact data you placed in the ECHO field of the transaction.
duplicate_transaction Type: String Length: 1	**DUPLICATE TRANSACTION INDICATOR** If you submitted a transaction with "show_duplicate_status=Y", you will receive this field in your response. If you submitted "resend=Y" with the transaction and it was a duplicate, you will receive "duplicate_transaction=Y". This means that the transaction response from a previous transaction is being sent to you. The transaction is not processed twice; you are re-receiving your original transaction response. "duplicate_transaction=N" in the reply means that this is the first time Payroc has seen your transaction.
token Type: String Length: 30	**TOKEN** If a Token is generated by Payroc then the Token ID value will be returned in this field.
card_type Type: String Length: 4	**CARD TYPE** Card type may be returned in this field if the card number passed validation rules. Values: AMEX - American Express DISC - Discover JCB MCRD - MasterCard VISA
card_product Type: String Length: 2	**CARD PRODUCT** Card product may be returned. Currently only available for Visa and MasterCard. Values: VC - Visa Credit VD - Visa Debit VB - Visa Business/Commercial MC - MasterCard Credit MD - MasterCard Debit MB - MasterCard Business/Commercial

... continued on next page

Response Field	Description
card_country_of_origin Type: String Length: 3	**CARD COUNTRY OF ORIGIN** Card country of origin may be returned. Currently only available for Visa and MasterCard only. Numeric country code will be returned as defined in ISO 3166-1 indicating the country where the card was issued.
csc_result Type: String Length: 1	**CARD SECURITY CODE (CSC) RESULT** For transactions where CVV2 value was included, this field returns the CSC result code. CSC result code is a 1-letter code indicating the result of the CSC authentication request. See the CSC Result Codes section in Card Security Code (CSC) Authentication section for values.
response_type Type: String Length: 1	**RESPONSE TYPE** Values: D - Decline N - Network Failure E - Error/Reject Response Type will not be returned if the transaction is approved.
card_last_four_digits Type: String Length: 4	**LAST 4 DIGITS OF CARD NUMBER** The last 4 digits of the card number may be returned in the response.
expiry_date Type: String Length: 4	**EXPIRY DATE** The card expiry date may be returned in this field.
settlement_total Type: Numeric Length: 12	**SETTLEMENT TOTAL** Total amount will be returned for successful Settlement transactions.
transaction_id Type: String Length: 16	**TRANSACTION IDENTIFIER** A unique id representing the transaction.
fee_reference Type: String Length: 30	**FEE REFERENCE NUMBER** This field is only returned for fee_calculate, fee_payment and fee_void transactions. Unique reference number associated with the fee calculate request.
fee_amount Type: Integer Length: 8	**FEE AMOUNT** This field is only returned for fee_calculate, fee_payment and fee_void transactions. The fee amount calculated based on the requested card type/product and payment amount.

... continued on next page

Response Field	Description
total_amount Type: Integer Length: 10	**TOTAL AMOUNT** This field is only returned for fee_calculate, fee_payment and fee_void transactions. The total amount to be charged to the card.
fee_response Type: String Length: 64	**FEE TRANSACTION RESPONSE** This field is only used for fee_payment and fee_void transactions. This field contains the authorization code, decline message or void confirmation for the fee amount. Authorization code will be provided if the fee amount was approved. Decline message will be provided if the fee amount was declined. This field may be blank for the following conditions: payment request was rejected due to validation errors, or payment amount was declined
payment_response Type: String Length: 64	**PAYMENT TRANSACTION RESPONSE** This field is only used for fee_payment and fee_void transactions. This field contains the authorization code, decline message, or void confirmation for the payment amount. Authorization code will be provided if the payment amount was approved. Decline message will be provided if the payment amount was declined. This field will be blank if the payment request was rejected due to validation errors.

Reason Codes

These are the reason code values returned by Payroc for non-successful or negative-type transaction requests.

Reason Code	Response Text and Explanation
200001 - 201299	Decline messages from various financial institutions. Text may not be constant for a particular reason code. Hard-coding of actions based on any of these values is not recommended.
201000	**MALFORMED TRANS** Some data in the transaction was missing or invalid, or the reference number was too long.
201001	**ACCESS DENIED** or **TOTL DENIED** Your terminal ID is not activated or you do not have permission use it from this IP address.
201002	**UNSUPPORTED TRANS** The transaction type you submitted is not supported, or is not supported for your merchant setup.
201003	**DECLINED (CV)** Transaction declined by Payroc card velocity tracking system. This card has been used more than the preset limit for this terminal ID allows.
201004	**DECLINED (MV)** Transaction declined by Payroc merchant velocity system. This merchant has exceeded their preset transaction volume limits.
201005	**DECLINED (CL)** Transaction declined by Payroc card limiting system. The amount of money charged to this card has exceeded preset limits.

... continued on next page

Reason Code	Response Text and Explanation
201006	**DECLINED (ML)** Transaction declined by Payroc merchant limiting system. The amount of money charged on this terminal ID has exceeded preset limits.
201007	**DECLINED (DUP)** This terminal ID is using Payroc's duplicate elimination system and the transaction has already been processed within the preset time.
201008	**CARD NOT ALLOWED** This terminal ID is using the Payroc contra checking system and the card is in the Contra table. (unwelcome)
201010	**ILLEGAL CHAR** The transaction request message contained one or more illegal characters.
201011	**ILLEGAL AMOUNT** The amount you entered is not valid.
201012	**NOT PRESENT** The card you are querying is not in the Contra table. (Contra Query transaction only)
201014	**NOT FOUND** The card you are attempting to delete from the Contra table is not present in the table. (Contra Query transaction only)
201015	**MRV NO MATCH** There is no matching Return for the Return Void you are attempting.
201016	**COMPLETION NO MATCH** There is no matching Pre-Authorization for the Completion you are attempting.
201017	**NO MATCH** There is no matching Sale, Completion, or Force Post for the Void you are attempting.
201018	**CARD TYPE INVALID** You are trying to process a card type which you are not set up for, or the card you are trying to process is not a valid type.
201019	**TRAN TYPE INVALID** You are trying to use a transaction type which you are not set up for, or you are trying to use a transaction type which does not exist.
201020	**CARD NUMBER INVALID** The card number did not pass check-digit test for that card type.
201021	**EOT RECEIVED** An upstream processor has disconnected. Retrying the transaction is recommended.
201022	**LEVEL 2 LENGTH ERR** You have sent too much Level 2 data. Resend your transaction with less than 600 characters of Level 2 data.
201023 201024 201025	**NETWORK FAILURE** These error codes indicate a communications failure in the authorization network. The transaction was neither approved nor declined.
201026	**AVS DATA INVALID** The AVS data you entered contained invalid or unrecognized data.
201028	**OPERATOR INVALID** The Operator ID you submitted contained something other than letters and numbers. (Operator IDs are 3 digits, alphanumeric)
201029	**AUTH NUMBER MISSING** You have submitted a Force Post transaction without an authorization number.
201030	**CARD LENGTH ERR** The card number you submitted did not contain the correct number of digits for that card type.

... continued on next page

Reason Code	Response Text and Explanation
201031	**VISA DEBIT FORBIDDEN** Visa Debit is not permitted on this Terminal ID.
201032	**INVALID TOKEN EXPIRY** The expiry date has either expired or is not in the correct format.
201035	**NO SURCHARGE SETUP** Terminal ID is not setup to allow surcharge activity. Please contact Payroc Merchant Services department for assistance.
201036	**INVALID SURCHARGE AMOUNT** The surcharge amount you entered is not valid.
201040	**AVS NO MATCH** The AVS dota does not match the cardholder's information.
201041	**CVV2 NO MATCH** The CSC/CVV does not match the card information.
201050	**UID MISSING** No Unique Identifier (UID) was sent with a Level 3 Detail Submission transaction. Details not captured.
201051	**NO MATCH** Unique Identifier (UID) was not found in UID database. The Level 2 transaction must be sent first, and UID must be included with a Level 3 Detail Submission transaction.
201052	**LEVEL 2 FORBIDDEN** This terminal ID is not set up to handle Level 2 enhanced data transactions. Please contact our Merchant Services department for assistance.
201053	**LEVEL 2 FORMAT ERR** or **L2 ERR FLDS: *x,y,z*** You have sent a Level 2 transaction with an incorrect number of fields (FORMAT ERR) or incorrect field data (L2 ERR FLDS). x,y, and z in the above message are replaced with field numbers where errors have been detected. Up to 3 errors per Level 2 data submission will be returned at a time. Authorization will not be performed if you send invalid Level 2 data.
201054	**LEVEL 3 FORMAT ERR** You have sent a Level 3 Detail Submission transaction with an incorrect number of fields.
201055	**MAX AMOUNT EXCEEDED** Your terminal ID has a maximum amount per transaction limit set, and this transaction exceeds that amount.
201056	**LEVEL 3 MISSING** You are attempting to send a Level 3 Detail Submission transaction without any Level 3 field data.
201066	**DECLINE - ACQUIRER** Acquirer rules forbid us from capturing this transaction.
201080	**NON-COMMERCIAL** The card number submitted is NOT a commercial card (Commercial Card Check transaction type only)
201081	**MONTH AMT EXCEEDED** Your merchant ID has a maxiumum monthly volume limit set, and this transaction exceeds that volume.
201082	**MONTH COUNT EXCEEDED** Your merchant ID has a maxiumum number of monthly transactions and this transaction exceeds that volume.
201083	**MONTH AMT EXCEEDED** Your terminal ID has a maxiumum monthly volume limit set, and this transaction exceeds that volume.

... continued on next page

Reason Code	Response Text and Explanation
201084	**MONTH COUNT EXCEEDED** Your terminal ID has a maxiumum number of monthly transactions and this transaction exceeds that volume.
201088	**TRANS ABORTED** The authorization network has disconnected.
201099	**NETWORK FAILURE** A network failure has occurred while communicating with the card issuer.
201100	**INVALID TOKEN ACTION** An invalid token action was specified.
201101	**TOKEN NOT FOUND** The token sent with the transaction was not found in our token vault.
201102	**TOKEN ALREADY EXISTS** The token sent is already in use.
201103	**TOKEN NOT ACTIVE** The token is in the token vault, but is marked as inactive.
201104	**TOKEN SUSPENDED** The token is in the token vault, but it is marked as suspended.
201105	**TOKEN SETUP ERROR** There is a problem with your terminal ID's tokenization setup. Please contact Payroc Merchant Services.
201106	**BAD END DATE** A token was sent with an end date that was not in the correct format, or was invalid.
201107	**BAD START DATE** A token was sent with a start date that was not in the correct format, or was invalid.
201108	**END DATE PAST** A token was sent with an end date that has already passed.
201109	**DATE SEQ ERROR** A token was sent with an end date that occurs before the start date.
201110	**PAN UPDATE FORBIDDEN** Card number cannot be updated for auto-generated Token IDs that include the card type and/or last four digits of the card number.
201111	**TOKEN SETUP ERR** Terminal ID is not setup to use tokens or the setup is incomplete. Please contact Payroc Merchant Services department for assistance.
201112	**TOKEN NO CONFIG** Tokenization services are currently not available. Please contact Payroc Merchant Services department if this condition persists.
201113	**TOKEN CONFIG ERROR** Tokenization services are currently not available. Please contact Payroc Merchant Services department if this condition persists.
201114	**AUTO TOKEN FMT ERR** Auto-generated Token ID creation failed as the length is invalid.
201115	**AUTO TOKEN FORBIDDEN** Merchant and associated Terminal IDs are not configured to allow the use of auto-generated Token IDs. Please contact Payroc Merchant Services department for assistance.
201116	**TOKEN NETWORK ERROR** Tokenization services are currently not available. Please contact Payroc Merchant Services department if this condition persists.

... continued on next page

Reason Code	Response Text and Explanation
201118	**TOKEN PENDING** The Token is not yet available for use as the start date is setup with a future date.
201119	**TOKEN EXPIRED** The Token is no longer available for use as the end date has been passed.
201120	**CARD NO IN TOKEN** Card number cannot exist in the token, token reference, or token client id fields.
201121	**TOKEN USAGE ERROR** SINGLE_USE_TOKEN or TOKEN_CARD_CHECK value is not 'Y','N' or token action error occurred
300002	**INVALID TERMINAL ID** The terminal id provided in the fee calculate request is invalid or the terminal id is not setup for User Pay processing.
300003	**INVALID REFERENCE NUMBER** The reference number provided in the fee calculate request is not valid.
300004	**INVALID CARD PRODUCT** The card product provided in the fee calculate request is not valid.
300005	**INVALID AMOUNT** The amount provided in the fee calculate request is not a valid amount.
300006	**ONLY CARD OR TOKEN ALLOWED** The fee calculate request can only have a card number and expiry date or a token not both.
300007	**INVALID CARD NUMBER** The card number provided in the fee calculate request is not valid.
300008	**INVALID EXPIRY DATE** The expiry date provided in the fee calculate request is not valid.
300009	**INVALID TOKEN** The token provided in the fee calculate request is not valid.
300102	**INVALID TERMINAL ID** The terminal id provided in the fee payment or fee void request is invalid or the terminal id is not setup for User Pay processing.
300103	**INVALID REFERENCE NUMBER** The reference number provided in the fee payment or fee void request is not valid.
300104	**INVALID CARD PRODUCT** The card product provided in the fee payment request is not valid.
300105	**INVALID AMOUNT** The amount provided in the fee payment or fee void request is not a valid amount.
300106	**ONLY CARD OR TOKEN ALLOWED** The fee payment request can only have a card number and expiry or a token not both.
300107	**INVALID FEE AMOUNT** The fee amount provided in the fee payment or fee void request is not valid.
300108	**INVALID TOTAL AMOUNT** The total amount provided in the fee payment or fee void request is not valid.
300109	**INVALID TOKEN** The token provided in the fee payment request is not valid.
300110	**INVALID RECURRING PAYMENT FLAG** The recurring payment flag provided in the fee payment request is not valid.
300111	**CARD PRODUCT MISMATCH** The card product provided in the fee payment request and the card product of the card number/token do not match.

... continued on next page

Reason Code	Response Text and Explanation
300112	**AMOUNT MISMATCH** The amount provided in the fee_calculate transaction does not match the amount provided in the fee_payment transaction. The amount provided in the fee_payment transaction does not match the amount provided in the fee_void transaction.
300113	**FEE AMOUNT MISMATCH** The fee amount returned in the fee_calculate response does not match the fee amount provided in the fee_payment transaction. The fee amount provided in the fee_payment transaction does not match the fee amount provided in the fee_void transaction.
300114	**PAYMENT TOTAL MISMATCH** The total amount returned in the fee_calculate response does not match the total amount provided in the fee_payment transaction. The total fee amount provided in the fee_payment transaction does not match the fee amount provided in the fee_void transaction.
300115	**VOID UNSUCCESSFUL** The void of the payment transaction was not successful.
300116	**INVALID CARD NUMBER** The card number provided in the fee payment request is not valid.
300117	**INVALID EXPIRY DATE** The expiry date provided in the fee payment request is not valid.
300118	**INVALID POSTAL CODE** The postal code provided in the fee payment request is not valid.
300119	**INVALID CARD SECURITY CODE** The card security code (CSC) provided in the fee payment request is not valid.

Response Codes

Response code values returned from MasterCard or Visa for non-successful or negative-type transaction requests are only supported for Payroc Canada acquired merchants.

MasterCard

Response Code	Description
01	Refer to card issuer
03	Invalid merchant
04	Capture card
05	Do not honor
08	Honor with ID
10	Partial Approval
12	Invalid transaction
13	Invalid amount
14	Invalid card number
15	Invalid issuer
30	Format error
41	Lost card
43	Stolen card

... continued on next page

Response Code	Description
51	Insufficient funds/overcredit limit
54	Expired card
55	Invalid PIN
57	Transaction not permitted to issuer/cardholder
58	Transaction not permitted to acquirer/terminal
61	Exceeds withdrawal amount limit
62	Restricted card
63	Security violation
65	Exceeds withdrawal count limit OR Identity CheckSoft-Decline of EMV 3DS Authentication (merchantshould resubmit authentication with3DSv1)
70	Contact Card Issuer
71	PIN Not Changed
75	Allowable number of PIN tries exceeded
76	Invalid/nonexistent “ToAccount” specified
77	Invalid/nonexistent “FromAccount” specified
78	Invalid/nonexistent account specified (general)
81	Domestic Debit Transaction Not Allowed (Regional use only)
84	Invalid Authorization Life Cycle
85	Not declined valid for all zero amount transactions
86	PIN Validation not possible
87	Purchase Amount Only, No Cash Back Allowed
88	Cryptographic failure
89	Unacceptable PIN—Transaction Declined—Retry
91	Authorization System or issuer system inoperative
92	Unable to route transaction
94	Duplicate transmission detected
96	System error

Visa

Category 1 (Issuer will not approve)

Merchants are not permitted to reattempt a transaction if the response code is in this category.

Response Code	Description
04	Pick up card (no fraud)
07	Pickup card, special condition (fraud account)
12	Invalid transaction
14	Invalid account number (no such number) [1]
15	No such issuer (first 8 digits of account number do not relate to an issuing identifier)
41	Lost card, pick up (fraud account)
43	Stolen card, pick up (fraud account)
46	Closed account
57	Transaction not permitted to cardholder

... continued on next page

Response Code	Description
R0	Stop payment order
R1	Revocation of authorization order
R3	Revocation of all authorizations order

Category 2 (Issuer cannot approve at this time)

Merchants are permitted to reattempt a transaction if the response code is in this category however reattempts are limited to 15 attempts in 30 days.

Response Code	Description
03	Invalid merchant
19	Re-enter transaction
51	Not sufficient funds
59	Suspected fraud
61	Exceeds approval amount limits
62	Restricted card (card invalid in region or country)
65	Exceeds withdrawal frequency limit
75	Allowable number of PIN-entry tries exceeded
78	Blocked, first used or special condition (account is temporarily blocked)
86	Cannot verify PIN
91	Issuer switch inoperative; STIP not applicable or not available for this transaction
93	Transaction cannot be completed-violation of law
96	System malfunction
N3	Cash service not available
N4	Cash request exceeds issuer or approved limit

Category 3 (Data quality issues)

Merchants are permitted to reattempt a transaction if the response code is in this category however reattempts are limited to 15 attempts in 30 days.

Response Code	Description
14	Invalid account number (no such number) [1]
54	Expired card or expiration date missing
55	PIN incorrect or missing
70	PIN data required (Europe Region only)
82	Negative online CAM, dCVV, iCVV, CVV, CAVV, dCVV2, TAVV, or DTVV results Or Offline PIN authentication interrupted
1A	Additional customer authentication required (Europe Region only)
6P	Verification data failed
N7	Decline for CVV2 failure

Category 4 (Generic response codes)

Merchants are permitted to reattempt a transaction if the response code is in this category however reattempts are limited to 15 attempts in 30 days.

Response Code	Description
01	Refer to card issuer
02	Refer to card issuer, special condition
05	Do not honor
06	Error
13	Invalid amount or currency conversion field overflow
21	No action taken
39	No credit account
52	No checking account
53	No savings account
58	Transaction not allowed at terminal
64	Transaction does not fulfill AML requirement
74	Different value than that used for PIN encryption errors
76	Unsolicited reversal-reversl no no original transaction in history. V.I.P. unable to match reversal request to an original message
79	Reversed (by switch)
80	No financial impact (used in reversal responses to decline originals)
81	Cryptographic error found in PIN (used for cryptographic error condition found by security module during PIN deccryption)
85	No reason to decline request for address verification, CVV2 verification, or credit voucher or merchandise return
92	Financial institution or intermediate network facility cannot be found for routing (receiving institution ID invalid)
94	Duplicate transmission. Transaction submitted containing values in the tracing data fileds that duplicate values in a previously submitted transaction
B1	Surchage amount not permitted on Visa card or EBT food stamps (U.S. acquirers only)
N0	Force STIP
N8	Transaction amount exceeds preauthorized approval amount
P5	Denied PIN unblock-PIN change or unblock request declined by issuer
P6	Denied PIN change-request PIN unsafe
Q1	Card authentication failed or Offline PIN authentication interrupted
R2	Transaction does not qualify for Visa PIN
Z1	Offline-declined
Z3	Unable to go online; offline-declined

[1] Reattempt with the same account number is not permitted for this response code. If the account number was entered incorrectly then it can be fixed and retried.

Request Examples

The following are some examples of various transaction scenarios. Terminal ID "EXAMPLE1" was used for these transactions.

Sale Transaction

Sale transaction examples are provided for approved and rejected transactions as the response fields returned would vary based on the validation/authorization results.

The Sale transaction is the most commonly used transaction for charging credit cards.

Approved Sale

In this example we process a Sale for $49.95, which is authorized and captured for settlement. We also add a unique custom description to appear on the cardholder's statement.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_sale",
    "reference":"SALE-EXAMPLE-APP",
    "payment":{
        "amount":4995,
        "dynamic_dba":"ABC CO 6598574588 56"
    },
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3
    }
}
```

Response:

```
{
    "message": "",
    "details": {
        "authorization_code": "T03637",
        "card_last_four_digits": "1111",
        "card_type": "VISA",
        "expiry_date": "0319",
        "transaction_id": "LTT2-256903637"
    }
}
```

Approved Sale with Surcharge

In this example we process a Sale with a surcharge amount, which is authorized and captured for settlement. The Sale amount of $102.00 includes the base transaction amount of $100, plus surcharge amount of $2.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_sale",
    "reference":"SALE-SURCHARGE-EXAMPLE-APP",
    "payment":{
        "amount":10200,
```



```
        "surcharge_amount":200
    },
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3
    }
}
```

Response:

```
{
    "message": "",
    "details": {
        "authorization_code": "T03637",
        "card_last_four_digits": "1111",
        "card_type": "VISA",
        "expiry_date": "0319",
        "transaction_id": "LTT2-256903637"
    }
}
```

Approved Sale for Funding Transaction

In this example we process a Sale for a funding transaction with recipient data, which is authorized and captured for settlement.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_sale",
    "reference":"SALE-FUNDING-EXAMPLE-APP",
    "payment":{
        "amount":10200
    },
    "recipient": {
        "name_first": "John",
        "name_middle": "Q",
        "name_last": "Doe"
    },
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3
    }
}
```

Response:

```
{
    "message": "",
    "details": {
        "authorization_code": "T03637",
```



```
            "card_last_four_digits": "1111",
            "card_type": "VISA",
            "expiry_date": "0319",
            "transaction_id": "LTT2-256903637"
        }
    }
```

Rejected Sale - Decline

In this example we process a Sale for $12.75, which is declined.

Request:

```
    {
        "terminal_id":"EXAMPLE1",
        "transaction_type":"card_sale",
        "reference":"SALE-EXAMPLE-REJ-DECLINE",
        "payment":{
            "amount":1275
        },
        "card_information":{
            "card_number":"4111111111111111",
            "expiry_year":2022,
            "expiry_month":4
        }
    }
```

Response:

```
    {
        "message":"05 DECLINE",
        "details":{
            "card_last_four_digits":"1111",
            "card_type":"VISA",
            "expiry_date":"0422",
            "reason_code":"201205",
            "response_type":"D",
            "transaction_id":"LTT2-256903638"
        }
    }
```

Rejected Sale - Invalid Card

In this example we process a Sale for $1.25 using an invalid card number, which is rejected.

Request:

```
    {
        "terminal_id":"EXAMPLE1",
        "transaction_type":"card_sale",
        "reference":"SALE-EXAMPLE-REJ-INVL-CARD",
        "payment":{
            "amount":125
        },
```



```
        "card_information":{
            "card_number":"1111111111111111",
            "expiry_year":2019,
            "expiry_month":3
        }
    }
```

Response:

```
    {
        "message":"CARD NUMBER INVALID",
        "details":{
            "card_last_four_digits":"1111",
            "expiry_date":"0319",
            "reason_code":"201020",
            "response_type":"E",
            "transaction_id":"LTT2-256903641"
        }
    }
```

Settlement Transaction

In this example we process a Settlement transaction, to close this batch and let Payroc know that we would like to deposit the money into our merchant account.

Request:

```
    {
        "terminal_id":"EXAMPLE1",
        "transaction_type":"card_settlement"
    }
```

Response:

```
    {
        "message":"",
        "details":{
            "settlement_total":14995,
            "transaction_id":"LTT2-256903644"
        }
    }
```

The settlement results indicate that $149.95 will be deposited into the merchant's account for this batch.

Authorization Only Transaction

Authorization Only Transaction For $10.00

In this example we process a Authorization Only for $10.00, which is approved.

Request:


```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_authorization_only",
    "reference":"AUTHONLY-EXAMPLE",
    "payment":{
        "amount":1000
    },
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "authorization_code":"T03645",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0319",
        "transaction_id":"LTT2-256903645"
    }
}
```

Authorization Only Transaction Using The Resend Flag

For this example, suppose that the response was not received for above transaction, and we would like to see the transaction results. This is where the resend flag is used. The original transaction results will be re-sent if the transaction has already been processed at Payroc. If the transaction has not been processed, it will now be processed.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_authorization_only",
    "reference":"AUTHONLY-EXAMPLE",
    "resend":"Y",
    "payment":{
        "amount":1000
    },
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3
    }
}
```

Response:

```
{
```



```
    "message":"",
    "details":{
        "authorization_code":"T03645",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0319",
        "transaction_id":"LTT2-256903647"
    }
}
```

Authorization Only Transaction Using An Expired Card

For this example, an authorization attempt is going to be processed on an expired card. Note the decline response.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_authorization_only",
    "reference":"AUTHONLY-EXAMPLE-EXPIRED1",
    "payment":{
        "amount":1000
    },
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2022,
        "expiry_month":4
    }
}
```

Response:

```
{
    "message":"54 DECLINE",
    "details":{
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0422",
        "reason_code":"201205",
        "response_type":"D",
        "transaction_id":"LTT2-256903648"
    }
}
```

Force Post Transaction

In this example we will Force Post $10.00 using the authorization number received from the Authorization Only transaction.

Request:

```
{
```



```
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_force_post",
    "reference":"FORCEPOST-EXAMPLE",
    "payment":{
        "amount":1000,
        "authorization_code":"T03645"
    },
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "authorization_code":"T03645",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0319",
        "transaction_id":"LTT2-256903651"
    }
}
```

Pre-Authorization Transaction

In this example, we will pre-authorize a charge of $50.00. We also add echo data to the request, which will be included in the response.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_preauthorization",
    "reference":"PREAUTH-EXAMPLE",
    "echo_data":"MY ECHO-DATA 1",
    "payment":{
        "amount":5000
    },
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3
    }
}
```

Response:

```
{
    "message":"",
```



```
    "details":{
        "authorization_code":"T03652",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "echo_data":"MY ECHO-DATA 1",
        "expiry_date":"0319",
        "transaction_id":"LTT2-256903652"
    }
}
```

Completion Transaction

Completion Without Card

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_completion",
    "reference":"PREAUTH-EXAMPLE",
    "payment":{
        "amount":5000
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "authorization_code":"T03652",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0319",
        "transaction_id":"LTT2-256903653"
    }
}
```

Note that the card data and authorization number in the Completion response was the same as the Pre-Authorization. Payroc maintains your Pre-Authorization information until you complete the transaction, or until the card authorization lifetime expires

Completion Transaction For $50.00 That Has Already Been Completed

A subsequent Completion sent for the previous Pre-Authorization will be rejected.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_completion",
    "reference":"PREAUTH-EXAMPLE",
    "payment":{
```



```
            "amount":5000
        }
    }
```

Response:

```
    {
        "message":"COMPLETION NO MATCH",
        "details":{
            "reason_code":"201016",
            "transaction_id":"LTT2-256903654"
        }
    }
```

Multiple Completion Transactions

Multiple Completions can be processed for the same Pre-Authorization transaction as long as the dollar amount of the original Pre-Authorization is not exceeded.

Pre-Authorization for $100.00

Request:

```
    {
        "terminal_id":"EXAMPLE1",
        "transaction_type":"card_preauthorization",
        "reference":"MULTI-COMPLETION-PREAUTH",
        "payment":{
            "amount":10000
        },
        "card_information":{
            "card_number":"4111111111111111",
            "expiry_year":2019,
            "expiry_month":3
        }
    }
```

Response:

```
    {
        "message":"",
        "details":{
            "authorization_code":"T03657",
            "card_last_four_digits":"1111",
            "card_type":"VISA",
            "expiry_date":"0319",
            "transaction_id":"LTT2-256903657"
        }
    }
```

First Completion Transaction For $50.00 Without The Credit Card Number

A Completion attempt using the same reference number as the original Pre-Authorization and an amount less that the original dollar amount will succeed.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_completion",
    "reference":"MULTI-COMPLETION-PREAUTH",
    "payment":{
        "amount":5000
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "authorization_code":"T03657",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0319",
        "transaction_id":"LTT2-256903658"
    }
}
```

Second Completion Transaction For $100.00 Without The Credit Card Number

A Completion attempt using the same reference number will fail when the dollar amount of the original Pre-Authorization amount is exceeded.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_completion",
    "reference":"MULTI-COMPLETION-PREAUTH",
    "payment":{
        "amount":10000
    }
}
```

Response:

```
{
    "message":"COMPLETION NO MATCH",
    "details":{
        "reason_code":"201016",
        "transaction_id":"LTT2-256903659",
    }
}
```

Third Completion Transaction For $50.00 Without The Credit Card Number

Request:


```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_completion",
    "reference":"MULTI-COMPLETION-PREAUTH",
    "payment":{
        "amount":5000
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "authorization_code":"T03657",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0319",
        "transaction_id":"LTT2-256903660"
    }
}
```

Fourth Completion Transaction For $0.01 Without The Credit Card Number

Any additional Completion attempts using the same reference number will fail because the total Completion amount cannot exceed the original Pre-Authorization amount.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_completion",
    "reference":"MULTI-COMPLETION-PREAUTH",
    "payment":{
        "amount":1
    }
}
```

Response:

```
{
    "message":"COMPLETION NO MATCH",
    "details":{
        "reason_code":"201016",
        "transaction_id":"LTT2-256903661"
    }
}
```

Void Transaction

Void Transaction For $50.00 Without The Credit Card Number

In this example we will Void the third Completion transaction for $50.00 from the preceding example. We will not use the credit card number, since it is not required.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_void",
    "reference":"MULTI-COMPLETION-PREAUTH",
    "payment":{
        "amount":5000
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "transaction_id":"LTT2-256903662"
    }
}
```

Second Attempt To Void Completion Transaction For $50.00 Without The Credit Card Number

In this example we will attempt to *Void* the second Completion transaction for $50.00 from the preceding example without the credit card number. Since we have already voided this transaction, the attempt will fail.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_void",
    "reference":"MULTI-COMPLETION-PREAUTH",
    "payment":{
        "amount":5000
    }
}
```

Response:

```
{
    "message":"NO MATCH",
    "details":{
        "reason_code":"201017",
        "transaction_id":"LTT2-256903664"
    }
}
```

No current matching voidable transaction was found.

Return Transaction

Approved Return

In this example we will process a refund of $25.00 to the credit card.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_return",
    "reference":"RETURN-EXAMPLE",
    "payment":{
        "amount":2500
    },
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "authorization_code": "T03665",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0319",
        "transaction_id":"LTT2-256903665"
    }
}
```

Approved Return with Surcharge

In this example we will process a Return with a surcharge amount. The Return amount of $20.40 includes the base transaction amount of $20, plus surcharge amount of $0.40.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_return",
    "reference":"RETURN-SURCHARGE-EXAMPLE",
    "payment":{
        "amount":2040,
        "surcharge_amount":40
    },
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3
```



```
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "authorization_code": "T03665",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0319",
        "transaction_id":"LTT2-256903665"
    }
}
```

Return Void Transaction

Void A Return Transaction For $25.00 Without The Credit Card Number

In this example, we process a Return Void on the Return transaction from the preceding example. The card number is not required, since the reference number and amount are used for the matching.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_return_void",
    "reference":"RETURN-EXAMPLE",
    "payment":{
        "amount":2500
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "transaction_id":"LTT2-256903666"
    }
}
```

Second Request To Void A Return Transaction For $25.00 Without The Credit Card Number

In this example, we will attempt to perform a Return Void on the previous *Return* transaction which has already has a Return Void performed on it. The attempt will fail.

Request:


```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_return_void",
    "reference":"RETURN-EXAMPLE",
    "payment":{
        "amount":2500
    }
}
```

Response:

```
{
    "message":"MRV NO MATCH",
    "details":{
        "reason_code":"201015",
        "transaction_id":"LTT2-256903667"
    }
}
```

No current matching transaction eligible for Return Void was found.

Contra Transactions

In this set of examples, we will show how Contra checking is used.

Add A Credit Card Number To The Contra List

Adding a card number to the Contra list will block its acceptance on the associated terminal ID.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_contra_add",
    "card_information":{
        "card_number":"4111111111111111"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "transaction_id":"LTT2-256903668"
    }
}
```

Contra Decline

Future authorization attempts on this card will be blocked until it is removed from the Contra list. Here is an attempt to perform a Sale on a blocked card.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_sale",
    "reference":"CONTRA-DECLINE",
    "payment":{
        "amount":1000,
    },
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3
    }
}
```

Response:

```
{
    "message":"CARD NOT ALLOWED",
    "details":{
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0319",
        "reason_code":"201008",
        "response_type":"E",
        "transaction_id":"LTT2-256903669"
    }
}
```

Contra Delete

In this example we will remove the credit card from the Contra table

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_contra_delete",
    "card_information":{
        "card_number":"4111111111111111"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "card_last_four_digits":"1111",
```



```
            "card_type":"VISA",
            "transaction_id":"LTT2-256903670"
        }
    }
```

Contra Query - Found

In this example we will query the Contra table for a card number and find it.

Request:

```
    {
        "terminal_id":"EXAMPLE1",
        "transaction_type":"card_contra_query",
        "card_information":{
            "card_number":"4111111111111111"
        }
    }
```

Response:

```
    {
        "message":"",
        "details":{
            "card_last_four_digits":"1111",
            "card_type":"VISA",
            "transaction_id":"LTT2-256903670"
        }
    }
```

Contra Query - Not Found

In this example we will query the Contra table for a card number and not find it.

Request:

```
    {
        "terminal_id":"EXAMPLE1",
        "transaction_type":"card_contra_query",
        "card_information":{
            "card_number":"4111111111111111"
        }
    }
```

Response:

```
    {
        "message":"NOT PRESENT",
        "details":{
            "card_last_four_digits":"1111",
            "card_type":"VISA",
            "reason_code":"201012",
            "transaction_id":"LTT2-256903674"
        }
    }
```


Token Transactions

This set of examples will show the use of tokens in transaction processing.

Creating A Token

In this example, we will associate a token with a credit card and expiry date. We will also assign some token reference data that can be used for Token Search via Payroc's Dashboard web application.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"token_add",
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3
    },
    "token":{
        "token":"EXAMPLETOKEN1",
        "reference":"JSMITH-VISA"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0319",
        "transaction_id":"LTT2-256903675"
    }
}
```

Creating Auto-Generated Tokens

In the following examples, we will associate a credit card and expiry date with a token where the Token ID value is automatically generated by Payroc.

Examples are based on configuration for a merchant where the maximum length for Token IDs has been setup as 16 characters. TOKEN field containing a "?" is included in the request for all examples indicating that the Token ID is to be generated by Payroc.

In this example, the options for card type and last four digits of card number suffix are not turned on so a 16 digit unique Token ID value is generated.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"token_add",
    "card_information":{
```



```
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3
    },
    "token":{
        "token":"?",
        "reference":"JSMITH-VISA"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0319",
        "token":"20000002569036777",
        "transaction_id":"LTT2-256903677"
    }
}
```

Auto-generated Token ID options for card type and last four digits of card number have been turned on for this example. The Token ID is generated to include a unique 11 digit value along with a suffix specifying the card type and last four digits of the card number.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"token_add",
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3
    },
    "token":{
        "token":"?",
        "reference":"JSMITH-VISA"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0319",
        "token":"202569036777V1111",
        "transaction_id":"LTT2-256903677"
    }
}
```


Auto-generated Token ID formatting options turned on for this example are use of customer prefix along with suffix containing card type and last four digits of the card number.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"token_add",
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3
    },
    "token":{
        "token":"31121345?",
        "reference":"JSMITH-VISA"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0319",
        "token":"31121345V1111",
        "transaction_id":"LTT2-256903677"
    }
}
```

Token Update

In this example, we will be updating the token to change the expiry date.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"token_update",
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":4
    },
    "token":{
        "token":"EXAMPLETOKEN1"
    }
}
```

Response:

```
{
```



```
    "message":"",
    "details":{
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0419",
        "transaction_id":"LTT2-256903681"
    }
}
```

Token Operation with Card Authentication

In these examples, we will request card authentication for a Token Add operation.

Values for CSC (CVV2 field) and AVS included in the first example will simulate successful authentication when submitted with Visa, MasterCard or Discover card along with a test Terminal ID. CSC value starting with a '3' will need to be used to simulate successful authentication for an American Express card.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"token_add",
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3,
        "avs_data":"12345",
        "csc":"400"
    },
    "token":{
        "token":"EXAMPLETOKEN2",
        "reference":"JSMITH-VISA"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "avs_result":"D",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "csc_result":"M",
        "expiry_date":"0319",
        "transaction_id":"LTT2-256903683"
    }
}
```

Values for CSC (CVV2 field) and AVS included in this example will simulate authentication failure when submitted with a test Terminal ID. The Token operation (ADD or UPDATE) is not executed since the authentication result was a failure.

Request:


```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"token_add",
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3,
        "avs_data":"99999",
        "csc":500
    },
    "token":{
        "token":"EXAMPLETOKEN2",
        "reference":"JSMITH-VISA"
    }
}
```

Response:

```
{
    "message":"82 CVV2 MISMATCH",
    "details":{
        "avs_result":"N",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "csc_result":"N",
        "expiry_date":"0319",
        "reason_code":"201013",
        "response_type":"D",
        "transaction_id":"LTT2-256903685"
    }
}
```

Token Deactivate

In this example, we will de-activating the token.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"token_deactivate",
    "token":{
        "token":"EXAMPLETOKEN1"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "transaction_id":"LTT2-256903687"
    }
}
```


Token Reactivate

In this example, we will re-activate the token. All data is preserved while the token is deactivated, but the token cannot be used.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"token_reactivate",
    "token":{
        "token":"EXAMPLETOKEN1"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "transaction_id":"LTT2-256903688"
    }
}
```

Sale Transaction Using A Token

In this example, we will perform a Sale transaction on the token from the previous examples. The card associated with the token will be charged.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_sale",
    "reference":"SALE-TOKEN-EXAMPLE",
    "payment":{
        "amount":345
    },
    "token":{
        "token":"EXAMPLETOKEN1"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "authorization_code":"T03689",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0419",
        "transaction_id":"LTT2-256903689"
    }
}
```


Pre-Authorization Transaction Using A Token

In this example, we will process a Pre-Authorization transaction using the token from the preceding examples while also performing an address verification check.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_preauthorization",
    "reference":"PREAUTH-TOKEN-EXAMPLE",
    "payment":{
        "amount":456
    },
    "token":{
        "token":"EXAMPLETOKEN1"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "authorization_code":"T03690",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0419",
        "transaction_id":"LTT2-256903690"
    }
}
```

Payment Schedule Transactions

Creating a Payment Schedule for an existing Token

In this example, we will create a Payment Schedule for an existing Token using a Payment Scheduling transaction type.

The Payment Schedule will be setup for 6 bi-monthly payments starting on date of 20150401 (YYYYMMDD).

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"schedule_add",
    "payment":{
        "amount":500
    },
    "token":{
        "token":"EXAMPLETOKEN1"
    },
    "schedule":{
        "type":"MONTHLY",
```



```
            "frequency":2,
            "number_of_payments":6,
            "start_date":20150401
        }
    }
```

Response:

```
    {
        "message":"",
        "details":{
            "card_last_four_digits":"1111",
            "expiry_date":"0419",
            "transaction_id":"LTT2-256903692"
        }
    }
```

Updating a Payment Schedule

In this example, we will update a Payment Schedule using a Payment Scheduling transaction type.

The Payment Schedule will be changed to 10 weekly payments starting on date of 20150801 (YYYYMMDD).

Request:

```
    {
        "terminal_id":"EXAMPLE1",
        "transaction_type":"schedule_update",
        "payment":{
            "amount":500
        },
        "token":{
            "token":"EXAMPLETOKEN1"
        },
        "schedule":{
            "type":"WEEKLY",
            "frequency":1,
            "number_of_payments":10,
            "start_date":20150801
        }
    }
```

Response:

```
    {
        "message":"",
        "details":{
            "card_last_four_digits":"1111",
            "expiry_date":"0419",
            "transaction_id":"LTT2-256903693"
        }
    }
```

Updating a Payment Schedule Amount

In this example, we will update the amount setup for a Payment Schedule using a Payment Scheduling transaction type.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"schedule_update",
    "payment":{
        "amount":505
    },
    "token":{
        "token":"EXAMPLETOKEN1"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "card_last_four_digits":"1111",
        "expiry_date":"0419",
        "transaction_id":"LTT2-256903694"
    }
}
```

Deactivating a Payment Schedule

In this example, we will deactivate a Payment Schedule using a Payment Scheduling transaction type.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"schedule_deactivate",
    "token":{
        "token":"EXAMPLETOKEN1"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "card_last_four_digits":"1111",
        "expiry_date":"0419",
        "transaction_id":"LTT2-256903695"
    }
}
```

Reactivating a Payment Schedule

In this example, we will reactivate a Payment Schedule using a Payment Scheduling transaction type.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"schedule_reactivate",
    "token":{
        "token":"EXAMPLETOKEN1"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "card_last_four_digits":"1111",
        "expiry_date":"0419",
        "transaction_id":"LTT2-256903696"
    }
}
```

Full Authorization Reversal

In this example, we will demonstrate the use of Authorization Reversal transactions to perform full reversals.

Pre-Authorization Transaction For $100.00

To begin, we will Pre-Authorize a transaction using the token from the previous examples in the Token Transactions examples.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_preauthorization",
    "reference":"REV-FULL-PREATUH-EXAMPLE",
    "payment":{
        "amount":10000
    },
    "token":{
        "token":"EXAMPLETOKEN1"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "authorization_code":"T03698",
        "card_last_four_digits":"1111",
```



```
            "card_type":"VISA",
            "expiry_date":"0419",
            "transaction_id":"LTT2-256903698"
        }
    }
```

Pre-Authorization Reversal

In this example, we will perform a reversal for the full amount of $100.00.

An amount of "0" is used as the replacement amount, meaning that we would like the actual authorization to be for zero dollars. This effectively removes the authorization, freeing the card's open-to-buy limit of the $100.00 charge.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_authorization_reversal",
    "reference":"REV-FULL-PREATUH-EXAMPLE",
    "payment":{
        "amount":0
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "authorization_code":"T03698",
        "expiry_date":"0419",
        "transaction_id":"LTT2-256903699"
    }
}
```

Partial Authorization Reversal

In this example, we will demonstrate the use of Authorization Reversal transactions to perform partial charge reversals.

Pre-Authorization Transaction For $375.25

To begin, we will Pre-Authorize a transaction using the token from the previous examples in the Token Transactions examples.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_preauthorization",
    "reference":"REV-PARTIAL-PREATUH-EXAMPLE",
    "payment":{
        "amount":37525
```



```
    },
    "token":{
        "token":"EXAMPLETOKEN1"
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "authorization_code":"T03700",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "expiry_date":"0419",
        "transaction_id":"LTT2-256903700"
    }
}
```

Replacing The Pre-Authorized Amount

In this example we will be replacing the original Pre-Authorization amount of $375.25 with a new amount of $255.59. The balance of the original Pre-Authorization will be reversed. The card's open-to-buy will now be reduced by $255.59 instead of $375.25.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_authorization_reversal",
    "reference":"REV-PARTIAL-PREATUH-EXAMPLE",
    "payment":{
        "amount":25559
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "authorization_code":"T03700",
        "expiry_date":"0419",
        "transaction_id":"LTT2-256903701"
    }
}
```

Account Status Inquiry

In this example, we will demonstrate the use of Account Status Inquiry transactions to check the status of card and perform CSC and AVS authentication.

Account Status Inquiry - Positive Card Status

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_account_status_inquiry",
    "card_information":{
        "card_number":"4111111111111111",
        "expiry_year":2019,
        "expiry_month":3,
        "avs_data":"45678",
        "csc":456
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "authorization_code":"T03702",
        "avs_result":"W",
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "csc_result":"M",
        "expiry_date":"0319",
        "transaction_id":"LTT2-256903702"
    }
}
```

An authorization code may not be included in responses for live transactions as Account Status Inquiry transactions do not include an amount to be approved by the card issuer.

Sale with Third Party Token Usage

In this example, it demonstrates using SINGLE_USE_ONLY field to identify payment transactions where a third party Stored Credential (Token) is being used.

Sale with Third Party Token Usage

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_sale",
    "reference":"Sale-3rd-Party-Token",

    "payment": {
        "amount":355
     },

    "card_information":{
        "card_number":"5191111111111111",
        "csc":"123",
```



```
            "expiry_year":2022,
            "expiry_month":8
        },

        "token":{
            "token":"",
            "single_use_token":"N",
            "start_date":"",
            "end_date":"",
            "reference":"",
            "token_card_check":""
        }
    }
```

Response:

```
    {
        "details":{
            "expiry_date":"0822",
            "authorization_code":"T54664",
            "card_last_four_digits":"1111",
            "transaction_id":"LTD1-554664",
            "card_type":"MCRD",
            "card_product":"MC",
            "card_country_of_origin":"484"
        },
        "message":""
    }
```

Commercial Card Check

In this set of examples, we will demonstration how to check whether a card is a commercial card.

Commercial Card Check - Commercial Card

In this example, we process a commercial card check example on a commercial card.

Request:

```
    {
        "terminal_id":"EXAMPLE1",
        "transaction_type":"card_commercial_card_check",
        "card_information":{
            "card_number":"4484070000000000"
        }
    }
```

Response:

```
    {
        "message":"",
        "details":{
            "card_country_of_origin":"826",
```



```
        "card_product":"VB",
        "card_type":"VISA",
        "transaction_id":"LTT2-256904849"
    }
}
```

Commercial Card Check - Consumer Card

In this example, we process a commercial card check example on a consumer card.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_commercial_card_check",
    "card_information":{
        "card_number":"4111111111111111"
    }
}
```

Response:

```
{
    "message":"NON-COMMERCIAL CARD",
    "details":{
        "card_last_four_digits":"1111",
        "card_type":"VISA",
        "reason_code":"201080",
        "transaction_id":"LTT2-256904866"
    }
}
```

Level 2/Level 3 Transaction

In this set of examples, we will demonstrate a transaction with Level 2 and Level 3 commercial card data.

Sale With Level 2 Data

In this example, we will perform a Sale transaction and include Level 2 data in the L2 field.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"card_sale",
    "reference":"L2L3-SALE-EXAMPLE",
    "payment":{
        "amount":10235
    },
    "token":{
        "token":"EXAMPLETOKEN2"
    },
    "invoice":{
```



```
            "version":"PC3.02",
            "invoice_reference":"INV1234",
            "customer_name":"ABC COMPANY",
            "invoice_total":10235,
            "pst_rate":0,
            "gst_rate":1300,
            "pst_amount":0,
            "gst_amount":1177,
            "customer_po_number":"PO123569",
            "order_date":20120914,
            "line_item_count":1,
            "requester_buyer_name":"REQBUY JSMITH",
            "customer_reference":"REF1234",
            "invoice_number":"INV1234",
            "ship_to_prov_state":"ON",
            "ship_to_postal_code":"L9K9K9"
        }
    }
```

Response:

```
    {
        "message":"",
        "details":{
            "authorization_code":"T04843",
            "card_country_of_origin":"826",
            "card_product":"VB",
            "card_type":"VISA",
            "unique_id":"19000MY",
            "expiry_date":"0419",
            "transaction_id":"LTT2-256904843"
        }
    }
```

Level 3 Detail Submission Transaction (Line Item)

Using the unique identifier from the previous example's Sale transaction, we will attach an invoice line item to the transaction using a Level 3 Detail Submission transaction. Since this is the first line item, we will send a Level 3 Sequence Number of 1.

Request:

```
    {
        "terminal_id":"EXAMPLE1",
        "transaction_type":"card_invoice_line_item",
        "reference":"L2L3-L3-EXAMPLE",
        "unique_id":"19000MY",
        "invoice_line_item":{
            "version":"PC3.02",
            "sequence_number" :1,
            "customer_po":"CUSTPO12",
            "customer_sku":"CSKU786756",
            "division_number":"DIV8765",
            "extended_item_amount":10235,
            "gl_account_number":"GL8765",
```



```
            "gst_amount":1178,
            "gst_rate":1300,
            "item_commodity_code":"65985",
            "item_description":"PENCILS",
            "manufacturer_sku":"MSKU",
            "po_line_number":5,
            "product_code":"123569",
            "quantity":100,
            "supplementary_data":"SUPPDATA456",
            "merchant_reference":"INV1234",
            "unit_cost":9058,
            "unit_of_measure":"EA"
        }
    }
```

Response:

```
    {
        "message":"",
        "details":{
            "transaction_id":"LTT2-256904844"
        }
    }
```

Delivery of the line item detail was successful.

Fee Calculation

Approved Fee Calculation Transaction

In this example we process a fee calculate transaction using card information with a payment amount of $500.00, which is successful.

Request:

```
    {
        "terminal_id":"EXAMPLE1",
        "transaction_type":"fee_calculate",
        "fee_reference":"USERPAYFEE",
        "card_information":{
            "card_number":4111111111111111,
            "expiry_month":6,
            "expiry_year":2019
        },
        "payment":{
            "amount":50000
        }
    }
```

Response:

```
    {
```



```
    "message":"",
    "details":{
        "fee_amount":"975",
        "fee_reference":"USERPAYFEE",
        "amount":"50000",
        "total_amount":"50975",
        "terminal_id":"EXAMPLE1",
    }
}
```

Rejected Fee Calculation Transaction - Invalid Payment Amount

In this example we process a fee calculate transaction using a token with an invalid payment amount, which is rejected.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"fee_calculate",
    "fee_reference":"USERPAYFEE2",
    "token":{
        "token":"EXAMPLETOKEN1"
    },
    "payment":{
        "amount":"INVALID"
    }
}
```

Response:

```
{
    "message":"invalid amount",
    "details":{
        "reason_code":"300005"
    }
}
```

Fee Payment

Approved Fee Payment Transaction

In this example we process a fee payment transaction, which is authorized. This payment transaction references the approved fee calculate transaction from the example above.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type":"fee_payment",
    "fee_reference":"USERPAYFEE",
    "reference":"USERPAYPAYMENT",
    "card_information":{
```



```
        "card_number":4111111111111111,
        "expiry_month":6,
        "expiry_year":2019
    },
    "payment":{
        "amount":50000,
        "fee_amount":975,
        "total_amount":50975
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "terminal_id":"EXAMPLE1",
        "card_last_four_digits":"1111",
        "expiry_date":"0619",
        "amount":"50000",
        "fee_amount":"975",
        "total_amount":"50975",
        "reference":"USERPAYPAYMENT",
        "fee_reference":"USERPAYFEE",
        "fee_response":"T18327",
        "payment_response":"T18328"
    }
}
```

Approved Fee Payment Transaction with Fee Override

In this example we process a fee payment transaction with a fee override, which is authorized. This payment transaction does not reference a fee calculate transaction.

Request:

```
{
    "terminal_id": "EXAMPLE1",
    "transaction_type": "fee_payment",
    "fee_calc_override": "Y",
    "fee_reference": "USERPAYFEEOVERRIDE",
    "reference": "USERPAYPAYMENTOVERRIDE",
    "card_information": {
        "card_number": 4111111111111111,
        "expiry_month": 6,
        "expiry_year": 2019
    },
    "payment": {
        "amount": 10000,
        "fee_amount": 100,
        "total_amount": 10100
    }
}
```

Response:


```
{
    "message":"",
    "details":{
        "terminal_id":"EXAMPLE1",
        "card_last_four_digits":"1111",
        "expiry_date":"0619",
        "amount":"10000",
        "fee_amount":"100",
        "total_amount":"10100",
        "fee_reference":"USERPAYFEEOVERRIDE",
        "reference":"USERPAYPAYMENTOVERRIDE",
        "fee_response":"T18329",
        "payment_response":"T18330"
    }
}
```

Fee Void

Approved Fee Void Transaction

In this example we process a fee payment void transaction, which is approved. This void transaction references the approved payment transaction from the example above.

Request:

```
{
    "terminal_id":"EXAMPLE1",
    "transaction_type": "fee_void",
    "fee_reference": "USERPAYFEE",
    "reference": "USERPAYPAYMENT",
    "payment": {
        "amount":50000,
        "fee_amount":975,
        "total_amount":50975
    }
}
```

Response:

```
{
    "message":"",
    "details":{
        "terminal_id":"EXAMPLE1",
        "fee_reference":"USERPAYFEE",
        "reference":"USERPAYPAYMENT",
        "amount":"50000",
        "fee_amount":"975",
        "total_amount":"50975",
        "fee_response":"VOID+OK",
        "payment_response":"VOID+OK"
    }
}
```

Rejected Fee Void Transaction

In this example we process a fee payment void transaction that doesn't have a matching fee reference number, which is rejected.

Request:

```
{
    "terminal_id": "EXAMPLE1",
    "transaction_type": "fee_void",
    "fee_reference": "USERPAYFEE-NOMATCH",
    "reference": "USERPAYPAYMENT",
    "payment": {
        "amount":50000,
        "fee_amount":975,
        "total_amount":50975
    }
}
```

Response:

```
{
    "message":"invalid reference number",
    "details":{
        "reason_code":"300103",
        "fee_response":"",
        "payment_response":""
    }
}
```

Testing Transactions

If a test terminal ID is being used then varying the amount in the transaction will cause various responses and behaviours. The table below details each test amount and its corresponding non-approved response. In order to generate an approval response then an amount should be used that is not listed in the table.

Notes:

1. The amount for the transaction must have no currency signs or decimal.
2. Amounts 909 and 1010 can be used to generate an HTTP 503, Service Unavailable, condition.

Amount	Reason Code	Response Text
2101	201285	CARD OK
2102	201201	CALL
2103	201202	CALL
2104	201228	NO REPLY
2105	201291	NO REPLY
2106	201204	PICK UP CARD
2107	201207	HOLD-CALL
2108	201241	PICK UP CARD
2109	201243	HOLD-CALL

... continued on next page

Amount	Reason Code	Response Text
2110	201299	ACCT LENGTH ERR
2111	201213	AMOUNT INVLD
2112	201214	CARD NO. INVLD
2201	201299	CHECK DIGIT ERR
2202	201299	CID FORMAT INVLD
2203	201280	DATE INVLD
2204	201205	DECLINE
2205	201251	DECLINE
2206	201299	DECLINE
2207	201261	DECLINE
2208	201262	DECLINE
2209	201265	DECLINE
2210	201293	DECLINE
2211	201254	EXPIRED CARD
2212	201292	INVALID ROUTING
2301	201212	INVALID TRANS
2302	201278	NO ACCOUNT
2303	201221	NO ACTION TAKEN
2304	201215	NO SUCH ISSUER
2305	201219	RE ENTER
2306	201263	SEC VIOLATION
2307	201257	SERV NOT ALLOWED
2308	201299	CVV2 MISMATCH
2310	200050	DECLINE
2311	200051	PHONE HELP 901
2312	200052	TEL ### ####
2401	200053	PHONE HELP 200
2402	200054	PHONE HELP 097
2403	200055	PHONE HELP 150
2404	200056	PICK UP CARD
2405	200057	HOLD CARD CALL
2406	200058	ERROR ACCOUNT
2407	200059	EXPIRY ERROR
2408	200060	PHONE HELP 055
2409	200061	PHONE HELP 801
2411	200201	TRANSMIT ERROR
2412	200202	INVALID MERCH ID
2501	200204	CALL 05
2502	200205	INV EXPIRY DATE
2503	200206	CARD EXPIRED
2504	200207	INVALID MERCH ID
2505	200208	INVALID AUTH CODE
2506	200210	LOST/STOLEN
2507	200216	CALL AUTH CENTRE
2508	200218	CALL AUTH CENTRE

... continued on next page

Amount	Reason Code	Response Text
2509	200219	CALL AUTH CENTRE
2510	200220	INV MSG TYPE
2511	200221	CALL AUTH CENTRE
2512	200222	CALL AUTH CENTRE
2601	200224	CALL AUTH CENTRE
2602	200225	CALL AUTH CENTRE
2603	200226	CALL CENTRE ####
2604	200227	CALL AMEX
2605	200228	CALL VOICE CENTRE
2606	200229	INV CARD TYPE
2607	200230	CALL AUTH CENTRE
2608	200231	TELEPHONE
2609	200232	CALL AUTH CENTRE
2610	200233	CALL 03
2611	200234	CALL 04
2701	200290	ERROR UNKNOWN
2702	200291	ERROR 10: FORMAT
2703	200292	ERROR 11: MAPP
2704	200293	ERROR 13: ACCT #
2705	200294	ERROR 15: MERCH
2706	200295	PLEASE RETRY
2708	200351	INVALID TERM ID
2709	200352	INVALID MERCH ID
2710	200354	INVALID OPERATOR
2711	200356	INVALID TRAN NUM
2801	200360	CALL AUTH CENTRE
2802	200362	INVALID TRAN CODE
2803	200363	INVALID ACCOUNT
2804	200364	INVALID TRANS
2805	200365	HOLD/CALL CENTRE
2806	200366	DECLINE
2807	200370	INSTIT NOT FOUND
2808	200373	ERROR - NO MATCH
2809	200374	DECL - HOLD CARD
2810	200375	HOLD/CALL CENTRE
2811	200376	HOLD/CALL CENTRE
2812	200377	HOLD/CALL CENTRE
3601	200378	CARD EXPIRED
3602	200379	DECLINED EXPIRED
3603	200410	INVALID LICENCE
3604	200420	ACCOUNT FROZEN
3605	200440	LIMIT EXCEEDED
3606	200443	DECLINE/CALL CS
3607	200451	EXCESS ACTIVITY
3608	200452	DO NOT ACCEPT

... continued on next page

Amount	Reason Code	Response Text
3610	200605	INVALID AMOUNT
3612	208001	EXPIRY INVALID
3710	201210	PARTIAL APPROVAL