# CARD PAYMENT API SPECIFICATION

# Table of Contents

**Compliance required**

Using direct card charge involves handling some very sensitive customer data, so a PCI DSS compliance certificate is required. When you've got one, reach out to compliance@lipad.io for verification first and enabling of this feature.

# 1. Introduction

The Card Payment API enables merchants to process Credit and Debit card payments securely and efficiently. This document provides and overview of the API, including its endpoints, request-response formats, authentication and data encryption methods.

# 2. API Overview

## 2.1 Purpose

The Card Payment API enables merchants to process Credit and Debit card payments securely and efficiently. This document provides and overview of the API, including its endpoints, request-response formats, authentication and data encryption methods.

## 2.2 Scope

The API allows merchants to perform the following key operations:

- Process card payments for various currencies.
- Refund payments for eligible transactions.
- View transaction details and status.

## 2.3 Functionality

The Card Payment API provides endpoints to initiate payment processing and handle payment-related tasks. It supports standard HTTP methods (POST, GET, PUT) for communication with the API server.

## 2.4 Integration Flow

### 2.4.1 Authentication

Merchants must obtain an access token to authenticate with the API.

### 2.4.2 Process Payment

Merchants can initiate payment processing by sending card and order details. The API returns the status and transaction details.

### 2.4.3 Refund Payment

Merchants can request a refund for a specific transaction, and the API processes the refund accordingly.

### 2.4.4 View Transaction

Merchants can retrieve transaction details, such as transaction ID, status, and payment details.

## 3. Versioning

| Version | Active | Notes |
|---|---|---|
| 1 | YES | |
| | | |

## 4. Changelog

| Version | Date | Author | Notes |
|---|---|---|---|
| 1.0 | 18/07/2023 | Fred Mainga | Document Creation |
| | | | |

## 5. Authentication and Authorization

Requests to the API will require token authentication. Here's a walkthrough on how to generate the access token:

Authentication Endpoints:

**Production**: **https://api.lipad.io/v1/auth**
**Testing: https://dev.lipad.io/v1/auth**

Method: **POST**

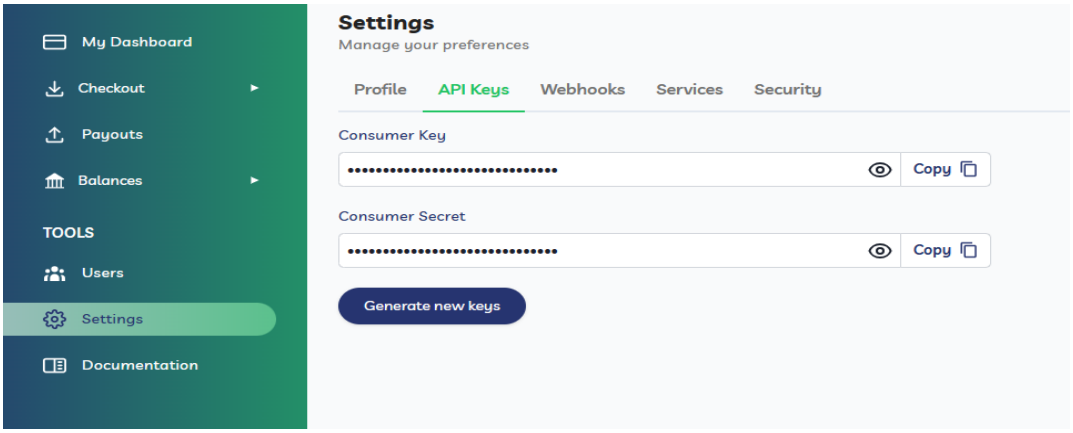
```
Request Body: {
   "consumer_key": "consumer_key",
   "consumer_secret": "consumer_secret"
}
```

```
Response:
{
  "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJjbGllbnRfaWQiOiIxNiIsImRlc2lnbmF0aW9uIjoiQXBpIFVzZXIiLCJhdXRob3JpemVkIjp0cnVlLCJpYXQiOjE2ODk4NDQ0NzEsImV4cCI6MTY4OTg1MDQ3MX0.6bUwqdvfXZf-4LdwoT1rB2bYHgnTcuPfUHHTPy1c6Tk",
  "expiresIn": 6000
}
```

The **access_token** from the response will be passed as **x-access-token** header.

```
Error response: {
  "statusCode": 401,
  "message": "Incorrect consumerKey and consumerSecret combination",
  "error": "Unauthorized"
}
```

The **Consumer Key** and **Secret** will be obtained from https://dashboard.lipad.io Settings API Keys tab. See screenshot for reference.




### 6. Methods and Endpoints

Test Base URL: **https://dev.charge.lipad.io**

Live Base URL: **https://api.charge.lipad.io**

#### 6.1.1 Charge Request

Charge Endpoint: **/v1/cards/charge**

Method: **POST**

### 6.1.1.1 Request Body


```
{
   "external_reference": "stat_test",
   "origin_channel_code": "API",
   "client_code": "DEMO",
   "originator_msisdn": "254*****89",
   "payer_ msisdn ": "254*****89",
   "service_code": "DEMCHE1",
   "account_number": "254*****89",
   "invoice_number": "gdhss-shgsj-sh",
   "currency_code": "KES",
   "country_code": "KEN",
   "amount": 5,
   "add_transaction_charge": false,
   "transaction_charge": 0,
   "payment_method_code": "CARD",
   "extra_data": {
      "store_number": 20,
      "location": "Nairobi"
   },
   "payer_name": "John Doe",
   "payer_email": "jdoe@email.com",
   "description": "Flight booking payment",
   "notify_client": 1,
   "notify_originator": 1,
   "device": {
      "browser": "MOZILLA",
      "ipAddress": “11.80.112.207",
      "browserDetails": {
         "3DSecureChallengeWindowSize": "FULL_SCREEN",
         "acceptHeaders": "application/json",
         "colorDepth": 24,
         "javaEnabled": true,
         "language": "en-US",
         "screenHeight": 400,
         "screenWidth": 250,
         "timeZone": 273
      }
   },
   "card":
"Txm6JgN/3jRfUVieK+w5rtFaumVuhQY1lEogTpPUMbFiQlh6DunJGHqh/7BCrdoKot2kmGXlt1TkBdcV5npHLdwQEEg+Rg
bbIYkHDzagyOxtMWl8e3WFgy3OfblOEwX+5n1/y17IpUkIsn1NblaPf/IzKCqfeWXKicRZd08g40AUbhWDUVhjlHOiiRGlCw
j2XfCozRIHd2QQDTF4vsntGw2L4nta/LXQKswHs9ndHTGngnY3yZfJc5aQTe0v/AlKHfAFBCT+8l+ijXLFogonMHbPH35At
pnRK472QUnsKYO54bf2GAsDtBOGN4VoYh/u3RbTVtMDLbXqiiNuOxuFcw=="
}
```

### 6.1.1.2 Request Parameters definition

| Parameter | DataType | Required | Description |
|---|---|---|---|
| external_reference | string | YES | This is the unique reference generated by the external/internal party requesting charge. It will be used to send a callback once charge is successful. |
| origin_channel_code | string | YES | This is an optional parameter indicating where the request to charge originated from. It has to be a valid channel code (API, WEB, USSD, APP) |
| client_code | string | YES | Client Identifier |
| originator_msisdn | string | YES | This is the mobile number of the individual initiating the request for charge. |
| payer_msisdn | string | YES | This is the mobile number that will making the payment. Can be same as originatorMSISDN |
| service_code | string | YES | This is the service against which charge is being made. Once successful, a payment will be logged against this service |
| account_number | string | YES | This is the account number against which the payment is being made. |
| invoice_number | string | YES | This is an internal invoice number to be used by the originating system. |
| currency_code | string | YES | This is the currency to be used to charge the client. |
| country_code | string | YES | This is the country from where the request is originated |
| amount | number | YES | This is the service amount to be charged from the client. |
| add_transaction_charge | Boolean | NO | Indicates whether the customer should be charged extra fees. |
| transaction_charge | number | NO | The transaction charge amount to be applied if the addTransactionCharge parameter is set to true. |
| payment_method_code | string | YES | The code to be used to identify the payment method. For cards it's **CARD** |
| extra_data | json | YES | JSON array of any additional information to be captured during charge. The same information will be relayed back when sending payment callbacks. |
| payer_name | string | YES | Full name of the payer |
| payer_email | string | YES | Email address of the payer |
| description | string | YES | The charge request narration |
| notify_client | number | YES | Flag to identify is a payment result callback should be sent to the Client 1 = True, 0= False |
| notify_originator | number | YES | |
| device | json | YES | Information about the device making the request |
| card | string | YES | Base64 encoded ciphertext of card information. See encryption section for guidance |

### 6.1.1.3 Response


```
{
   "status_code": 176,
   "status_description": "Request to charge customer was successfully placed.",
   "total_amount": 5,
   "service_amount": 5,
   "transaction_charge": 0,
   "charge_request_id": "1686",
   "external_reference": "stat_test",
   "auth_available": true,
   "html": "<div id=\"threedsChallengeRedirect\" xmlns=\"http://www.w3.org/1999/html\" style=\" height: 100vh\"> <form id
=\"threedsChallengeRedirectForm\" method=\"POST\" action=\"https://ap.gateway.mastercard.com/acs/mastercard/v2/prompt\"
target=\"challengeFrame\"> <input type=\"hidden\" name=\"creq\"
value=\"eyJ0aHJlZURTU2VydmVyVHJhbnNJRCI6ImIwNzdkMWU1LWUyNTctNDYzNC1iZjRiLTcxN2EyNWY2YjY2MSJ
9\" /> </form> <iframe id=\"challengeFrame\" name=\"challengeFrame\" width=\"100%\" height=\"100%\" ></iframe> <script
id=\"authenticate-payer-script\"> var e=document.getElementById(\"threedsChallengeRedirectForm\"); if (e) { e.submit(); if
(e.parentNode !== null) { e.parentNode.removeChild(e); } } </script> </div>"
}
```


### 6.1.1.4 Response Parameters definition

| Parameter | DataType | Provided | Description |
|---|---|---|---|
| status_code | number | YES | Describes the result of the charge request. See statusCodes section for complete description |
| status_description | number | YES | statusCode description message |
| total_amount | number | YES | Total amount to be charged from the customer. |
| service_amount | number | YES | Service amount from the total amount paid. |
| transaction_charge | number | YES | Transaction charge amount from the total amount paid. |
| charge_request_id | string | YES | Internal Charge request ID. |

| | | | |
|---|---|---|---|
| external_reference | string | YES | This is the unique reference generated by the external/internal party that requested the charge. |
| auth_available | Boolean | YES | Indicates if the card is 3DS enrolled. Close payment if no authentication is available |
| html | HTML string | YES | HTML Snippet to inject to your page and allow the payer to interact with their bank for authorization. |

## 13. Security Considerations

### 13.1 Encryption

The API employs **Partial Encryption** where only the card data is encrypted.

The following card data payload should be encrypted using the Public Key provided and the Base64 encoded ciphertext passed as the card parameter.

```
```
{
    "expiry": {
      "month": "01",
      "year": "39"
    },
    "name_on_card": "Test Card",
    "number": "5123450000000008",
    "security_code": "100"
  }
```
```

## 13.1.2 Encryption Examples

### 13.1.2.1 JavaScript

```
const publicKey = Buffer.from(
  await fs.readFile("keypath", { encoding: 'utf-8' }),
);

const encryptedData = crypto.publicEncrypt(
  {
    key: publicKey,
    padding: crypto.constants.RSA_PKCS1_OAEP_PADDING,
  },
  Buffer.from(JSON.stringify(dataToEncrypt.card)));
const encryptedPayload = encryptedData.toString('base64');
return encryptedPayload;
```

### 13.1.2.2 PHP

```

```

```
<?php

$publicKeyRsa = file_get_contents('lipadPublic.pem');
$data = "test";

openssl_public_encrypt($data, $encryptedData, $publicKeyRsa, OPENSSL_PKCS1_OAEP_PADDING);

$encodedData = base64_encode($encryptedData);
return $encodedData;
```

## 14. Request Response codes

| Status Code | Status Message | Comment |
|---|---|---|
| **176** | CHARGE POSTED SUCCESSFULLY | This is the success code for posting a charge request. |
| **173** | GENERIC SUCCESS | GENERIC SUCCESS |
| **174** | GENERIC FAILURE | GENERIC FAILURE |
| **172** | GENERIC VALIDATION FAILURE | This is the generic code to be returned for generic validations. |
| **132** | Invalid Credentials. Authentication failed | Invalid Credentials. Authentication failed |
| **131** | Authentication was a success | Authentication was a success |
| **120** | INVOICE ACCOUNT NUMBER NOT SPECIFIED | This is the error code for when the invoice account number is not specified. |
| **110** | INVALID CUSTOMER MSISDN | This is the error code for a failed msisdn |
| **109** | CUSTOMER MSISDN MISSING | Missing Payer MSISDN |
| **115** | INVALID CURRENCY CODE | Currency error |

## 15. Callbacks

Once payment is made against a charge request by the customer, a callback shall be sent back to the client of the service or originator of the payment request. This payment notification is sent back to the client depending on the notify_client or notify_originator parameters that were sent during charge initiation.

**NOTE:**

1. All callbacks from LIPAD employ Basic authentication with the username and password registered alongside the callback url. See Basic Authentication rfc for guidance.
2. Do not rely entirely on webhooks for Charge request status. Use the LIPAD Charge Status API to verify a Payout Status
3. Acknowledge webhook receipt with the following response

**Payment result acknowledgement Response Body:**

```
{
"charge_request_id": "123",
"acknowledgement_code": "900",
"acknowledgement_description": "payment accepted",
"payment_id": "12345"
}
```

Payment result acknowledgement Response Body Parameters description

| Parameter | Description |
|---|---|
| charge_request_id | The incoming charge request ID |
| acknowledgement_code | A status code that indicates whether a request was accepted or rejected.<br>• 900 - Accepted<br>• 901 - Rejected |
| acknowledgement_description | Callback receipt acknowledgement status description |
| payment_id | The incoming IPN payment ID |

**NOTE:** Unacknowledged callbacks will be retried 3 times over the next 48 hours after which no further webhooks will be sent.

### Payment Result Body


```
{
    "event": "successful_payment",
    "amount": "2.00",
    "extra_data": "extraData",
    "client_code": "ATL-6815O1F",
    "country_code": "KEN",
    "payer_msisdn": "2547XXXXXXXX",
    "payment_date": "2023-02-21T10:15:47.862Z",
    "service_code": "EATZ_CHECKOUT",
    "currency_code": "KES",
    "account_number": "2547XXXXXXXX",
    "payment_status": 700,
    "transaction_id": "2019",
    "payer_narration": "2547XXXXXXXX",
    "charge_request_id": "873",
    "external_reference": "1431",
    "receiver_narration": "The service request is processed successfully.",
    "payment_method_code": "CARD",
    "payer_transaction_id": "EATX6373HGED"
}
```


### Result Parameters description

| Name | Type | Description |
|---|---|---|
| event | String | Indicates payment event e.g failed_payment or successful_payment |
| amount | String | Amount paid by customer. |
| extra_data | String | Extra data passed during redirect |
| client_code | String | Merchant client code |
| country_code | String | The 3 digit ISO of the country which the payment was made from. |
| payer_msisdn | String | This is the mobile number that will be debited. It will receive the USSD or STK push. |
| payment_date | String | Date which payment was made. |
| service_code | String | The service for which the payment request was made. |

| Name | Type | Description |
|---|---|---|
| currency_code | String | This is the currency to be used to charge the client. |
| account_number | String | This is the account number against which the payment is being made. |
| payment_status | String | Indicates payment status i.e<br>• 700 - Successful payment received<br>• 701 - Failed transaction |
| transaction_id | String | Unique transaction id generated by Lipad. |
| payer_narration | String | Request description initally sent by Merchant. |
| charge_request_id | String | Unique charge id generated by Lipad. |
| external_reference | String | Unique charge id generated by Checkout. |
| receiver_narration | String | Narration generated by payment gateway. |
| payment_method_code | String | The payment method to be used to charge customer. |
| payer_transaction_id | String | Id generated by payment gateway. |

## Charge Request Status

For status verification or in the event when result callbacks fail, the Charge Request Status API can be employed. Requests to check payout status are sent to:

**GET** <https://dev.charge.lipad.io/v1/transaction/{charge_request_id}/status

**NOTE:** For charge status Response Body and Parameters description, see Callbacks section.

### Charge Request-Result Payment Status Codes

This flag indicates the individual payment status i.e successful or failed.

| Status | Description |
|---|---|
| 700 | Successful payment made. |

| Status | Description |
|---|---|
| 701 | Failed payment made |
| 702 | Payment was Reversed |
| 703 | Payment is Pending Processing |
| 704 | Payment is Jammed. Requires manual intervention to continue processing |