**MightyPay API Interfacing Guide**

**Version 0.1**

**All rights reserved. This document is proprietary and confidential. This document may contain trade secrets and information proprietary to MVI Technologies (S) Pte Ltd. No part of this document may be disclosed in any manner to a third party without the prior written consent of MVI Technologies (S) Pte Ltd. This document is protected under copyright laws as unpublished work of MVI Technologies (S) Pte Ltd.**

**Version History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Date** | **Description** | **Version** |
| Vijayakumar | 05-Dec-2019 | Initial version | 0.1 |

Contents

[1. About Mightypay System 4](#_Toc26546273)

[2. Topup 4](#_Toc26546274)

[3. Onboard User Entity 5](#_Toc26546275)

[4. Request/Response Samples 7](#_Toc26546276)

[4.1 Topup 7](#_Toc26546277)

[4.2 Onboard 8](#_Toc26546278)

[5. Security Implementation 8](#_Toc26546279)

[5.1 Acquiring App ID & App secret 8](#_Toc26546280)

[5.2 Securing Messages using JOSE (JWT, JWE, JWS). 9](#_Toc26546281)

## About Mightypay System

In this system there are Players, Sponsors, Game owners, etc., First Phase will have 2 APIs (Topup and Onboard) only for Players.

## Topup

**Resource URL:**

https://<baseURL>/1.0/fe-api-gw/topup

|  |  |
| --- | --- |
| **Resource** | **Description** |
| POST  fe-api-gw/topup | Topup digital asset into an ewallet account. The quantity will be credited into the ewallet immediately upon successful transaction. |

**Request Private Claim Fields:**

| **Parameter** | **Data Type** | **M/O** | **Size** | **Value** |
| --- | --- | --- | --- | --- |
| platformCode | Alphanumeric | M | 5 | This indicates the platform code assigned to the requestor which is also the source of this transaction |
| userEntityId | Numeric | M | 40 | User Entity ID |
| walletId | Alphanumeric | M | 20 | eWallet ID |
| digitalAssetTypeCode | Alphanumeric | M | 5 | Digital Asset Type Code |
| quantity | Numeric as string | M | 18 | Quantity |
| terminalId | Alphanumeric | O | 50 | Terminal ID at the terminal where transaction is done |
| platformRef | Alphanumeric | M | 50 | This is for platform to cross reference with their transaction |
| timestamp | Alphanumeric | M | 14 | This is the time stamp of the transaction at the source in the format “yyyyMMddHHmmss” |
| remarks | Alphanumeric | O | 100 | Transaction description |

**Response Private Claim Fields:**

| **Parameter** | **Data Type** | **M/O** | **Size** | **Value** |
| --- | --- | --- | --- | --- |
| responseCode | Alphanumeric | M | 3 | 000 – success, any other response code is considered as an error. Refer to message field for detail. |
| message | Alphanumeric | O | 200 | Additional information on the status |
| rrn | Alphanumeric | M | 20 | Retrieval Reference Number |
| authidresp | Alphanumeric | O | 10 | Authorization ID Response, available only for successful transactions |
| txnLogUid | Alphanumeric | M | 20 | Transaction Log Unique ID / Transaction reference |
| digitalAssetTypeCode | Alphanumeric | M | 5 | Digital Asset Type Code |
| platformRef | Alphanumeric | M | 50 | This is for platform to cross reference with their transaction |
| quantity | Numeric as string | O | 18 | Quantity (available only if responseCode=000) |
| trxnDate | Alphanumeric | M | 17 | This is the time stamp of the transaction in the format “yyyyMMddHHmmssS” |

## Onboard User Entity

**Resource URL:**

https://<baseURL>/1.0/ fe-api-gw/onboard

|  |  |
| --- | --- |
| **Resource** | **Description** |
| POST  fe-api-gw/onboard | Onboard a new user entity (Player, sponsors etc.,) |

**Request Private Claim Fields:**

| **Parameter** | **Data Type** | **M/O/C** | **Size** | **Value** |
| --- | --- | --- | --- | --- |
| platformCode | Alphanumeric | M | 5 | This indicates the platform code assigned to the requestor which is also the source of this transaction |
| terminalId | Alphanumeric | M | 50 | Terminal ID at the terminal where transaction is done |
| name | Alphanumeric | M | 100 | User Entity Name (Player/Sponsor Name) |
| mobileNo | Alphanumeric | M | 20 | Mobile number (numbers only) |
| userEntityIdentityType | Alphanumeric | O | 20 | Identify Type  (  PASSPORT,  TAX\_ID,  NATIONAL\_ID  ) |
| userEntityIdentity | Alphanumeric | C | 20 | Identify Number  This field is mandatory if the “userEntityIdentityType” field has value. |
| userEntityType | Alpha | M | 20 | User Entity Type.  (PLAYER,  SPONSOR,  STREAMER,  INFLUENCER,  GAME\_OWNER) |
| address1 | Alphanumeric | O | 80 | Address Line 1 |
| address2 | Alphanumeric | O | 80 | Address Line 2 |
| address3 | Alphanumeric | O | 80 | Address Line 3 |
| postalCode | Alphanumeric | 0 | 20 | Postal code |
| city | Alphanumeric | O | 40 | City |
| countryCode | Alphanumeric | O | 3 | Country code |
| timestamp | Alphanumeric | M | 14 | This is the time stamp of the transaction at the source in the format yyyyMMddHHmmss” |
| state | Alphanumeric | O | 40 | State |
| emailId | Alphanumeric | M | 75 | Email ID |
| kycFlag | Alpha | O | 1 | KYC Flag (‘Y’ or ’N’)  Y – KYC completed N – KYC pending |

**Response Private Claim Fields:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Parameter** | **Data Type** | **M/O** | **Size** | **Value** |
| responseCode | Alphanumeric | M | 3 | 000 – success, any other response code is considered as error. Refer to message field for detail. |
| message | Alphanumeric | O | 200 | Additional information on the status |
| rrn | Alphanumeric | M | 20 | Retrieval Reference Number |
| authidresp | Alphanumeric | O | 10 | Authorization ID Response, available only for successful transactions |
| txnLogUid | Alphanumeric | M | 20 | Transaction Log Unique ID / Transaction reference |
| userEntityId | Alphanumeric | M | 40 | User Entity Account ID |
| walletId | Alphanumeric | M | 20 | eWallet ID |
| trxnDate | Alphanumeric | M | 17 | This is the time stamp of the transaction in the format “yyyyMMddHHmmssS” |

## Request/Response Samples

### 4.1 Topup

**Request**:

{

"platformCode": "00001",

"userEntityId": "CUST35",

"walletId": "MA10001",

"digitalAssetTypeCode": "DGC02",

"quantity": "10",

"terminalId": "T0000004",

"platformRef": "20190412111910",

"timestamp": "20190412111910",

"remarks": "Topup"

}

**Response**:

{

"responseCode": "000",

"message": "Success",

"rrn": "191206094551",

"authidresp": "3633928205",

"txnLogUid": "19120600109451",

"trxnDate": "20191206105324735",

"digitalAssetTypeCode": "DGC02",

"availableQuantity": "52.000",

"platformRef": "20190412111910"

}

### 4.2 Onboard

**Request:**

{

"platformCode": "00001",

"terminalId": "T0000004",

"name": "Player 1",

"mobileNo": "8765987",

"userEntityIdentityType": " PASSPORT ",

"userEntityIdentity": "G19992244X",

"userEntityType": "PLAYER",

"address1": "address1",

"address2": "address2",

"address3": "address3",

"postalCode": "345990",

"city": "SG",

"countryCode": "SG",

"timestamp": "20190412111910",

"state": "SG",

"emailId": "abc@abc.com",

"kycFlag": "N"

}

**Response:**

{

"responseCode": "000",

"message": "Success",

"rrn": "191206094553",

"authidresp": "2223862797",

"userEntityId": "CUST00111501PLAYER",

"txnLogUid": "19120600109453",

"trxnDate": "20191206110817119",

"walletId": "MPW00110904"

}

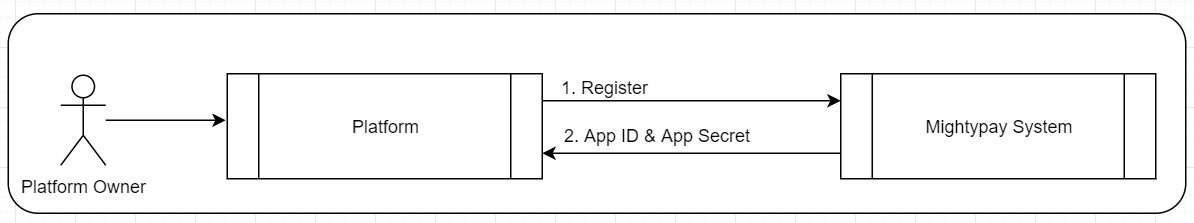
## Security Implementation

In phase 2, the Mightypay system will use JOSE to send/receive the message (payload) securely from the platform owner.

### 5.1 Acquiring App ID & App secret

This is a manual step (for now).

Each platform owner who needs to call the Mightypay system will have to be registered and be provided with an application identifier and an application secret. Application identifier and application secret are used in the API calls for the system to be able to identify and authenticate that the transactions come from a valid source.



### 5.2 Securing Messages using JOSE (JWT, JWE, JWS).

The secure messaging between the platform owner system and Mightypay system is designed based on JavaScript Object Signing and Encryption (JOSE), thereby ensuring confidentiality, integrity and authentication during communication between the two parties.

EX.

Messages will be sent in the form of **JWS** to the Mightypay system.

**JWS**:

Format:

JWS = Header.Payload.Signature

* **Header** = BASE64URL(UTF8(**JWS Protected Header**))

**JWS Protected Header** is in the form of a JSON message containing the following fields: “alg” and “typ”

Example = {“typ”: “JWT”, “alg”: “HS256”}, indicates that encoded object is a JSON Web Token (JWT) and the payload is signed with HMAC SHA-256

* **Payload**= BASE64URL(UTF8(**JSON message**))

JSON message

|  |  |
| --- | --- |
| Claims (Fields) | Description |
| iss | indicates the Application ID |
| iat | issue at time, indicates timestamp of the current request |
| sub | type of request (ex: Load Wallet/Onboard User) |
| jwe | **Actual JSON message** **payload** in encrypted form |
| appId | indicates the Application ID |
| authToken | session ID after successful authentication.  This will be used for all the request except login. |

* **Signature** = BASE64URL(HMAC SHA256**appSecret**(**data**))

**data** = ASCII(BASE64URL(UTF8(JWS Protected **Header**)) || '.' || BASE64URL(JWS **Payload**))

**JWE:**

**Actual JSON Message Payload in encrypted form.**

To encrypt the **CEK** (Content Encryption Key) **Auth token & Auth Secret** (This will be obtained upon successful login using the valid platform owner credentials. Details related to login API (which will be provided in phase 2) needs to use for all the request except login.

For login, App ID & App Secret will be used for CEK encryption.

Format:

JWE = Header.encryptedCEK**.**IV**.**Ciphertext**.**AuthenticationTag

* **Header** = BASE64URL(UTF8(**JWS Protected Header**))

JWS Protected Header is in the form of a JSON message containing the following fields: “enc” and “alg”

|  |  |
| --- | --- |
| Claims (Fields) | Description |
| enc | content encryption algorithm to produce the ciphertext and the Authentication Tag |
| alg | identifies the cryptographic algorithm used to encrypt or determine the value of the CEK |

* **encryptedCEK** = BASE64URL(Encrypted CEK)

CEK – Content encryption key. This is used to encrypt the actual payload.

Encrypted CEK - Encrypt the CEK with shared secret (**Auth token & Auth Secret** or **App ID & App Secret**) using A256KW algorithm

* **IV**

JWE Initialization Vector

* **Ciphertext** = BASE64URL(encrypted payload)

Additional Authenticated Data = ASCII(BASE64URL(UTF8(JWE Protected Header)))

Encrypted payload – Encrypt payload with AES GCM using CEK with IV and additional authenticated data.

Authentication Tag – Obtain Authentication Tag using the encrypted payload.

* **AuthenticationTag** = BASE64URL(Authentication Tag)

Complete JWE Representation

Assemble the final representation: The Compact Serialization of this result is the string BASE64URL(UTF8(JWE Protected Header)) || '.' || BASE64URL(JWE Encrypted Key) || '.' || BASE64URL(JWE Initialization Vector) || '.' || BASE64URL(JWE Ciphertext) || '.' || BASE64URL(JWE Authentication Tag).

Refer this [link](https://tools.ietf.org/html/rfc7516) for more information related to the JWE.