Protocol for WEB API for Members

NCMS (CM Segment)

Version 1.1

The NSE Clearing Limited (National Clearing)
Exchange Plaza, Plot No. C/1, G Block,
Bandra-Kurla Complex, Bandra (E),
Mumbai - 400 051

NSE Clearing Confidential

Notice

Confidential

Revision History

Date	Change Description	Edited By	Version
29-Oct-2025	Initial version		1.0
02-Mar-2026	Second version		1.1

Confidential

Table of Contents

Confidential

Introduction

This document provides information on the Web APIs used for programmatic access to trade management related data between NCL's NCMS Platform and its Members. It details the messaging protocols and structures required to develop this interface.

General Instructions

1. Following headers need to be provided in all API calls made to clearing corporation.
    - **Content-Type**: This header should be provided in all requests with method as "POST". Header value should be "application/json".
    - **User-Agent**: All requests should contain this header. The value of "User-Agent" header can be "/".
    - **Accept-Encoding**: This header is required in all API calls to CC. The value of this header should be blank.
    - **Accept**: This header value should be "application/json"
2. Some of the key specifications related to JSON and standards followed for the API's are as follows
    - JSON is built on 2 structures. Map containing key value pairs and an ordered list of values.
    - A value could be boolean (true / false), number, decimal, String or a structure (List or Object).
    - Object or key value pair structure consists of keys which are strings and values of any of the above types. E.g. {"name":"Amit", "age":25}
    - List contains list of values. E.g. ["Amit", "Ajay", "Vikas"]
    - A Boolean has only 2 values true or false.
    - String values are enclosed in double quotes. e.g. "name", "Amit", "Pending"
    - Numbers and decimals are represented without any thousand - separator character. Decimal indicator is dot (".")
    - Numbers have an optional maximum number of digits. If not specified, then it is defaulted to 18.
    - Decimals have 2 mandatory length parameters. The first length parameter indicates number of digits in the whole part (before decimal place) and the second length parameter indicates number of digits in the decimal part (after decimal place).
3. All URLs for API will be always in lower case.
4. All JSON field names will follow camel-hump style of naming. A field with multiple words would be concatenated without spaces. All characters will be in lower case. First characters of words other than the first word in the field name will be in upper case. For e.g. field for "Order Number" could be represented by field name "orderNumber". Other examples are "firstName", "lastName".
5. In case of JSONs representing an object or a key-value pair, keys with null values could be omitted from the JSON.

HTTP Status Codes

All API's will respond with an HTTP status code. A status code of 200 would indicate successful execution of the API and the response body would be as defined in the API specification.

In case of an error a HTTP status code other than 200 will be returned. The API may or may not return an error response JSON depending upon the type of error encountered. Following are the HTTP status codes that could be returned by the APIs

#	Status Code	Reason	Description
1	200	SUCCESS	Request was handled successfully
2	400	BAD REQUEST	Indicates a validation / business logic error / json parsing errors
3	401	UNAUTHORIZED: Failed to authenticate the request	Indicates that the credentials / access token shared for authentication is invalid or expired.
4	404	NOT FOUND	Incorrect URL or Resource does not exist
5	405	METHOD_NOT_ALLOWED	Unsupported HTTP Method: A request was made for a resource using a request method not supported by that resource (e.g. using GET instead of POST).
6	500	UNKNOWN_ERROR	Internal Server Error. Such errors are to be reported to the support desk.
7	503	SVC_UNAVAILABLE	Service unavailable.

Common Error Response JSON

Field	Type	Mandatory	Description
code	Number	Yes	Http Status Code. See above
messages	List<String>	Yes	One or more error messages

Sample Response

```
{
   "code": 400,
   "messages": [
      "Invalid JSON."
   ]
}
```

Environment Details

Base URL for all CM Segment Trade Management API endpoints mentioned in this document will be as follows:

Testing Environment: https://uat.connect2nsccl.com/CM/TRD/

Live Environment: https://www.connect2nsccl.com/CM/TRD/

API Consumer Registration

To initiate data consumption through the API endpoints, members are required to submit necessary information, including their IP address and registered email address, to NCL. Additionally, members must provide their public key certificates to NCL to enable payload encryption. The public key should be generated using the RSA algorithm and comply with the X.509 standard to ensure compatibility. Once this information is received, the member will be registered for API access and provided with a Consumer Key and Secret.

API Security

OAuth 2.0, an industry-standard authorisation protocol, is employed to facilitate access to API endpoints. Members can generate bearer tokens through the designated API call (refer to details below). The token response payload's data field will be asymmetrically encrypted using the Member's Public Key Certificate with the RSA algorithm. This encrypted payload will be delivered as a Base64-encoded string.

Furthermore, an AES secret key and IV unique to the member will be included within the access token payload and retained by both NCL and the member. This will serve to enable secure encryption and decryption of API payloads.

Confidential

Clearing Corporation APIs

This chapter gives details of the API's exposed by clearing corporation and to be consumed by members.

POST /<version>/request/token

To obtain a token, the member's consumer app must request for the access token using API POST /<version>/request/token endpoint. The access token can be reused to access NCL API data until it expires (after 'n' minutes). During API registration, the member receives a consumer key and secret, which are validated for token authorization. The access token payload also contains aes_secret_key and aes_iv required to decrypt response payloads.

Request

Get Token Request Header Parameters

#	Parameter Name	Data Type	Description	Sample Value
1	Authorization	String	The format should be as follows: Basic <member_credentials> Here, member_credentials refers to a base64-encoded string consisting of the following data: cons_key:cons_secret	Basic MRZmwzCl6.....SGq lCaxH9rAM3hVlMJzFg==
2	nonce	String	A nonce uniquely identifies each server request. It should be a base64-encoded string in the format: ddMMyyyyHHmmssSSS:<6-digit random number>.	MjAwMTIwMTcxNjEyMjE1OTE6
3	grant_type	String	Value MUST be set to "client_credentials".	client_credentials

Sample Request

```
POST /auth/token HTTP/1.1
Host: www.connect2nsccl.com
Content-Type: application/x-www-form-urlencoded
Authorization: Basic MRZmwzCl6.....SGqXlCaxH9rAM3hVlMJzFg==
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
x-www-form-urlencoded
grant_type=client_credentials
```

Response

The response's data field includes the encrypted token response payload as a Base64-encoded string. To access the raw token payload, first decode the Base64 data string, then decrypt the resulting bytes using the Member private key associated with the public certificate provided during the API Consumer Registration process.

Success Response Sample

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
   "data":
"GHzovnrYUaw6X8J9GE1vfLLIgb6b/KIVp6B0uKttHP91FlFNEpEZIMI43eWMcyOUEsvqr5fj4snHA125K8++8U/RtCYC7r3bW+2U/P6J/nG2qNtFGRoM1Koc0KVMcFgNptJC6BK2Bs6Fo44KAOtJ97NBlf9R0/WPxJy3dqi2A6zXo9tqn22JfgaFq/2JWZT0kX1grGkBEJZZImUiA0+ftpV3JfqrnYwZAtCr+cM7nbhab8Mri8cWBeHNG1pAlU/A1jcDvar5/NTdMDSClmkuw7ngXQpnOFX1mP1AlTAYLHOTnuau3KoE653lze2+ruleMuk9ceIEuL+vahYqtZfz7w=="
}
```


Failure Response Sample

```
HTTP/1.1 401 UNAUTHORIZED
Content-Type: application/json

{
  "messages":{"code":"0100401"},
  "status":"error"
}
```

Token Response Raw Parameters

#	Parameter Name	Data Type	Description	Sample Value
1	access_token	String	The access token that is issued by the authorization server.	ee1073de-45d0-4040-b9c2-eddfa80280c0
2	token_type	String	The type of the token issued.	bearer
3	expires_in	int	The lifetime in seconds of the access token. For example, the value "32400" denotes that the access token will expire in nine hour from the time the response was generated.	32400
4	Scope	String	If identical to the scope requested by the client otherwise, REQUIRED.	api_scope
5	key	String	aes_secret_key and aes_iv collectively used to encrypt and decrypt further API request-response	
6	iv	String	aes_secret_key and aes_iv collectively used to encrypt and decrypt further API request-response	

Sample output of the decrypted **raw token payload** in JSON format:

```
HTTP/1.1 200 OK
Content-Type: application/json
{
   "access_token": "ee1073de-45d0-4040-b9c2-eddfa80280c0",
   "token_type": "bearer",
   "expires_in": "3600",
   "scope":"api_scope"
   "key":"aes_secret_key",
   "iv": "aes_iv"
}
```


Confidential

POST /<version>/inquire/trd-act

This API will allow members to inquire for trades & actions using API POST /<version>/inquire/trd-act endpoint. A maximum of 1,00,000 messages can be inquired in a single API call.

Request

Request Header Parameters

#	Parameter Name	Data Type	Description	Sample Value
1	Authorization	String	Bearer token encrypted using the AES received as part of token response. <access_token>	Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
2	nonce	String	A nonce uniquely identifies each server request. It should be a base64-encoded string in the format: ddMMyyyyHHmmssSSS:<6-digit random number>.	MjAwMTIwMTcxNjEyMjE1OTE6
3	consumerKey	String	The Member Consumer Key received as part of API Registration process. <consKey>	consKey

Request Body Payload (JSON)

#	Parameter Name	Data Type	Description	Sample Value
1	Version	String	API version	1.0
2	Data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnn> Member / Custodian Code (Length: 4 or 5) • YYYYMMDD – Date format • nnnnnnn – Running sequence no. starting from one i.e. For first request of the day, it should be (0000001).	XXXXX201310140000001
3	data.dataformat	String	Request data format: Response data format	CSV:CSV
4	data.trdactInquiry	CSV	Data Structure specified below	0,ALLTRDACT,,

Trade Action Inquiry (trdactInquiry) Request Packet Structure (CSV)

Field Name	Description	Data Type	Size (in Bytes)	Sample
seqNo	The trade sequence should reflect the last trade sent by the server. For the first	INT	8	0

Field Name	Description	Data Type	Size (in Bytes)	Sample
	request of the day, set it to 0.			
srchFilter	Search Filter	String	50	• ALLTRDACT – Download all trades and actions • TMTRDACT – Download trades and actions of the Clearing Member as a Trading Member • CPTRDACT – Download only CP trades and related actions • ERRRORACT-Download only all failure actions performed
fill1	Filler	String	10	Leave it blank
fill2	Filler	String	10	Leave it blank

Sample Request

Request Header:

```
POST /api/ncms-cm/trades-inquiry HTTP/1.1
Host: uat.connect2nsccl.com
Authorization: Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
consumerKey: consKey
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
Content-Type: application/json
```

Request Body:

```
{
   "data":
"i3fJhLKZHhGdanX8csAP4sfqaXse/PO2ek84FMMocd8hLMVgHuOQREft6QsruHisVrqTBjDqAL4guyyVLLV3RNr
YRRa3uuhRj+BdJI7UJE……A6dy/yJaem0qa40X+5iUvteGpQ7BIpQ=="
}
```

Sample output of the decrypted **request body payload data** in JSON format:

The access token in the Authorization header, as well as the data parameter in the request body, are required to be AES-encrypted. When making an API call, the Base64-encoded string of these encrypted values must be used. Members should perform encryption using the AES secret key and IV provided at the time of token generation alongside the access token.

```
{
        "version": "1.0",
        "data":
        {
                "msgId": "00001201310140000001",
                "dataFormat": "CSV:CSV",
                "trdactInquiry": "0,TMTRDACT,,"

        }
```

```
}
```

The access token in the authorization header, as well as the data parameter in the request body, are required to be AES-encrypted. When making an API call, the Base64-encoded string of these encrypted values must be transmitted. Members should perform encryption using the AES secret key and IV provided at the time of token generation alongside the access token.

Response

Response Payload Structure (JSON)

#	Parameter Name	Data Type	Description	Sample Value
1	status	String	Response Status	success/error
2	messages.code	String	Refer to section "Message based response code"	01010000
3	data.msgId	String	Unique request number sent back in the response	XXXXX201310140000001
4	data.trdactinquiry	CSV	Data Structure specified below. Structure Separator "^"	Refer Sample Response

Trade Action Inquiry (trdactInquiry) Response Packet Structure (CSV – Structure Separator "^")

Field Name	Description	Data Type	Size (in Bytes)	Sample
sysinfoResData	Control Record Structure	CSV	-	System Info Response Structure given below
trdResData	Detail Record Structure	CSV	-	Field Separator – "," Record Separator – "^" Trades Response Data Structure given below

Control Record Structure (sysinfoResData) (CSV)

Field Name	Description	Data Type	Size (in Bytes)	Sample
mktSts	Market Status For list of Market Status please refer "Reference Codes" section	Short	2	6
currTrdDate	Current Trade Date (YYYYMMDD)	Long	8	20251028
sfill1	Filler	String	10	
sfill1	Filler	String	10	
maxSeqNo	Max sequence number sent in response	long	8	47
noOfRec	Count of trades sent in the response	int	4	2

Detail Record Structure (trdResData) (CSV) (Field Separator – "," | Record Separator – "^")

#	Field Name	Description	Data Type	Size (in Bytes)	Sample
1	seqNo	Unique Sequence Number	Int	8	46
2	trdNo	Trade Number	Double	8	20250429800000001
3	actType	Activity Type. Refer Section "Reference Codes"	Short	2	2
4	trdPr	Trade price in paise	Float	4	12300.0
5	trdTime	Trade Time in jiffy format	Long	8	93824483328000
6	trdTmCd	TM code	String	5	XXXXX
7	trdTmBrnCd	TM Branch code	Short	2	2
8	trdStpType	STP type	String	1	5
9	trdStpNo	STP number	String	7	2025586
10	trdSecToken	Securities Token	String	11	757139
11	trdSecSymbl	Securities Symbol	String	10	ACC
12	trdSecSeries	Securities Series	String	2	EQ
13	ordNo	OrderNumber	Double	8	1400000000000001
14	Quantity	Trade Quantity	Int	4	110000
15	mktType	Market Type	String	1	N
16	cpCd	Cp code	String	12	XXXXX0000001
17	cusCd	Custodian Code	String	5	XXXX
18	trdAcc	Client Code / Account No	String	20	10001
19	ordTime	Order Date Time	Date		10-OCT-25 09:22:33
20	transCd	Transaction Code. Refer Section "Reference Codes"	Short	2	6001
21	booktype	Book Type. Refer Section "Reference Codes"	Int	2	1
22	trdPan	Pan number	String	10	QAZZZZZZZZ
23	exchangeCd	Exchange Code	Short	2	1
24	trdUniqId	Trade Unique Id	String	20	75713920250429800 0000011
25	trdCmCd	CM code	String	14	XXXXX
26	trdOrdUniqCd	Order Unique Code	Double	30	05900100000000000 001811
27	actCliCd	Action Client Code	String	10	A1410353
28	actBy	Action taken by	String	12	XXXXX
29	actDate	Action Date	Date		29-MAR-23
30	actTime	Action Time in jiffy format	Double	8	1372609300
31	actRcvdTime	Action Received Time in jiffy format	Double	8	1372010664
32	actErrCd	Action Response error code. Refer Sub Section "ANNEXURE 1 – Error Codes" in Section "Reference Codes".	Int	2	9
33	bsFlg	Buy/Sell Flag. Refer Section "Reference Codes - Buy Sell Flag"	String	1	B
34	actId	Action Type. Refer Section "Reference Codes"	Short	2	1

#	Field Name	Description	Data Type	Size (in Bytes)	Sample
35	Filler1	Filler	Double	8	
36	Filler2	Filler	Double	8	
37	Filler3	Filler	String	16	
38	Filler4	Filler	String	16	
39	Filler5	Filler	String	16	

Sample Failure Response

Wrong access token or expired access token

```
HTTP/1.1 401 UNAUTHORIZED
Content-Type: application/json
{
  "messages":{"code":"0100401"},
  "status":"error"
}
```

Error in encryption

```
HTTP/1.1 400 BAD_REQUEST
Content-Type: application/json
{
  "messages":{"code":"0100400"},
  "status":"error"
}
```

Sample Success Response

The payload in the response to the API call, will be AES-encrypted string. The Base64-encoded string of this encrypted value will be transmitted. Members should perform decryption using the AES secret key and IV provided at the time of token generation alongside the access token.

Actual Response

```
HTTP/1.1 200 OK
Content-Type: application/json
{
        "status": "success",
        "messages":
        {
                "code": "01010000"
        },
        "data":
        "i1iw0PPNS0DJNSX8bswCpY65aWYSobTpBCR/UZAqacBiO8smQeHRa338+Ro7qGi8VOXSVzPCEP04o
        XWHd/AjOozKTge2vO9WiOJdJY3VJVhdcwZL3Gj4tEbXi4vxft+SfJ9bRxyfh5kMHZXvclnC55mpkHca9fd
        gm8pKmT0SdnQsKMJ11GMUYxKQtDvdzQXyxWI4GY3f"
}
```

Response with Raw Data

```
{
        "status": "success",
        "messages":
        {
                "code": "01010000"
        },
```

```
"data":
{
"msgId": "XXXXX201310140000001",
"trdactInquiry":"6,20251028,,,47,2^46,20250429800000001,2,12300.0,93824483328000,XXXXX,2,
5,2025586,757139,ACC,EQ,1400000000000001,110000,N,XXXXX0000001,XXXX,10001,10-OCT-25
09:22:33,6001,1,QAZZZZZZZZ,1,75713920250429800000011,XXXXX,0590010000000000000001811,
A1410353,XXXXX,29-MAR-23,1372609300,1372010664,9,B,1,,,,,^47,
20250429800000001,2,12300.0,93824483328000,XXXXX,2,5,2025586,757139,ACC,EQ,1400000000
000001,110000,N,XXXXX0000001,XXXX,10001,10-OCT-25
09:22:33,6001,1,QAZZZZZZZZ,1,75713920250429800000011,XXXXX,0590010000000000000001811,
A1410353,XXXXX,29-MAR-23,1372609300,1372010664,9,S,1,,,,,^"
}
}
```


POST /<version>/request/cp-modification

This API will allow members for CP modification using API POST /<version>/ request/cp-modification

Below is the list of modifications which can be carried out through the CP modification API

- CP Trade to CP Trade Modification
- CP Trade to Client Modification
- CP Trade to INST Modification
- Client to CP Trade Modification
- Client to INST Modification
- INST to CP Trade Modification
- INST to Client Modification

Request

Request Header Parameters

#	Parameter Name	Data Type	Description	Sample Value
1	Authorization	String	Bearer token encrypted using the AES received as part of token response. <access_token>	Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
2	nonce	String	A nonce uniquely identifies each server request. It should be a base64-encoded string in the format: ddMMyyyyHHmmssSSS:<6-digit random number>.	MjAwMTIwMTcxNjEyMjE1OTE6
3	consumerKey	String	The Member Consumer Key received as part of API Registration process. <consKey>	consKey

Request Body Payload (JSON)

#	Parameter Name	Data Type	Description	Sample Value
1	Version	String	API version	1.0
2	Data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnn> Member Code (Length: 5) • YYYYMMDD – Date format • nnnnnnn – Running sequence no. starting from one i.e. For first request of the day, it should be (0000001).	XXXXX201310140000001
3	data.stpType	String	Settlement Type	M
4	data.stpNo	String	STP Number	2025115
5	data.cpMod	JSON	Array of order Modification structure	Max 50000 records allowed per messageID. Structure details given below

Request Packet Structure:

#	Field Name	Description	Data Type	Size (in Bytes)	Sample
1	trdSecToken	Securities Token	String	11	757139
2	ordNo	Order No	Long	12	2600000001529680
3	bsFlg	Buy/Sell Flag. Refer Section “Reference Codes - Buy Sell Flag”	Short	2	1
4	newCPCode	Below are the modifications which can be carried out and 1. CP Trade to CP Trade Modification (newCPCode: CP0000000001) 2. CP Trade to Client Modification (newCPCode: Blank) 3. CP Trade to INST Modification (newCPCode: INST)	String	12	CP0000000001

#	Field Name	Description	Data Type	Size (in Bytes)	Sample
		4. Client to CP Trade Modification (newCPCode: CP0000000002) 5. Client to INST Modification (newCPCode: INST) 6. INST to CP Trade Modification (newCPCode: CP0000000002) 7. INST to Client Modification (newCPCode: Blank)			

Sample Request

Request Header:

POST /api/<ncms-cm>/<cp-modification> HTTP/1.1
Host: uat.connect2nsccl.com
Authorization: Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
consumerKey: consKey
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
Content-Type: application/json

Request Body:

{
"data":
"i3fJhLKZHhGdanX8csAP4sfqaXse/PO2ek84FMMocd8hLMVgHuOQREft6QsruHisVrqTBjDqAL4guyyVLLV3RNr
YRRa3uuhRj+BdJI7UJE……A6dy/yJaem0qa40X+5iUvteGpQ7BIpQ=="
}

Sample output of the decrypted **request body payload data** in JSON format:

The access token in the Authorization header, as well as the data parameter in the request body, are required to be AES-encrypted. When making an API call, the Base64-encoded string of these encrypted values must be used. Members should perform encryption using the AES secret key and IV provided at the time of token generation alongside the access token.

➔ Sample request call (cp trade to cp trade modification)

{
"version": "1.0",
"data":
{

```
                "msgId": "00001201310140000001",
                “stpType”: “M”,
                “stpNo”: “2025115”,
                " cpMod ": [
                {
                “trdSecToken”: “757139”,
                " ordNo ": 1100000000000404,
                "bsFlg": 1,
                " newCPCode": " CP0000000002",
                }
    ]

  }

}
```

➔ Sample request call (cp trade to client modification)

```
{
        "version": "1.0",
        "data":
        {
                "msgId": "00001201310140000001",
                “stpType”: “M”,
                “stpNo”: “2025115”,
                " cpMod ": [
                {
                “trdSecToken”: “757139”,
                " ordNo ": 1100000000000404,
                "bsFlg": 1,
                " newCPCode": "",
                }

    ]

  }

}
```

➔ Sample request call (cp trade to INST modification)

```
{
        "version": "1.0",
        "data":
        {
                "msgId": "00001201310140000001",
                “stpType”: “M”,
                “stpNo”: “2025115”,
                " cpMod ": [
                {
                “trdSecToken”: “757139”,
                " ordNo ": 1100000000000404,
                "bsFlg": 1,
                " newCPCode": " INST",
                }
    ]
```

```
   }

}
```

➔ Sample request call (client to cp trade modification)

```
{
        "version": "1.0",
        "data":
        {
                "msgId": "00001201310140000001",
                “stpType”: “M”,
                “stpNo”: “2025115”,
                " cpMod ": [
                {
                “trdSecToken”: “757139”,
                " ordNo ": 1100000000000404,
                "bsFlg": 1,
                " newCPCode": " CP0000000001",
                }
    ]

  }

}
```

➔ Sample request call (client to INST modification)

```
{
        "version": "1.0",
        "data":
        {
                "msgId": "00001201310140000001",
                “stpType”: “M”,
                “stpNo”: “2025115”,
                " cpMod ": [
                {
                “trdSecToken”: “757139”,
                " ordNo ": 1100000000000404,
                "bsFlg": 1,
                " newCPCode": " INST",
                }
    ]
```

```
   }

}
```

➔ Sample request call (INST to cp trade modification)

```
{
        "version": "1.0",
        "data":
        {
                "msgId": "00001201310140000001",
                “stpType”: “M”,
                “stpNo”: “2025115”,
                " cpMod ": [
                {
                “trdSecToken”: “757139”,
                " ordNo ": 1100000000000404,
                "bsFlg": 1,
                " newCPCode": " CP0000000001",
                }
    ]

   }

}
```

➔ Sample request call (INST to client modification)

```
{
        "version": "1.0",
        "data":
        {
                "msgId": "00001201310140000001",
                “stpType”: “M”,
                “stpNo”: “2025115”,
                " cpMod ": [
                {
                “trdSecToken”: “757139”,
                "ordNo ": 1100000000000404,
                "bsFlg": 1,
                "newCPCode": "",
                }
    ]
```

```
  }

}
```

Response

Response Data Payload (JSON)				
Sr. No.	**Parameter Name**	**Data Type**	**Description**	**Sample Value**
1	status	String	Response status	success/error
2	messages.success	String	Message	Request submitted successfully
3	data.code	String	Refer Section “Message based response code”.	01010000

```
{
  "status": "Success",
  "messages": {
  "success": " Request submitted successfully."
},
  "data": {
    "code": "01010000"
  }
}
```

POST /<version>/inquire/otr-inquiry

This API will allow members to inquire the OTR file using API POST /<version>/inquire/otr-inquiry endpoint.

Request

Request Header Parameters

#	Parameter Name	Data Type	Description	Sample Value
1	Authorization	String	Bearer token encrypted using the AES received as part of token response. <access_token>	Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
2	nonce	String	A nonce uniquely identifies each server request. It should be a base64-encoded string in the format: ddMMyyyyHHmmssSSS:<6-digit random number>.	MjAwMTIwMTcxNjEyMjE1OTE6

#	Parameter Name	Data Type	Description	Sample Value
3	consumerKey	String	The Member Consumer Key received as part of API Registration process. <consKey>	consKey

Request Body Payload (JSON)

#	Parameter Name	Data Type	Description	Sample Value
1	Version	String	API version	1.0
2	Data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnn> Member / Custodian Code (Length: 4 or 5) • YYYYMMDD – Date format • nnnnnnn – Running sequence no. starting from one i.e. For first request of the day, it should be (0000001).	XXXXX201310140000001
3	data.dataformat	String	Request data format: Response data format	CSV:CSV
4	data.otrInquiry	CSV	Data Structure specified below	0, OTRINQ,,

OTR Inquiry (otrInquiry) Request Packet Structure (CSV)

Field Name	Description	Data Type	Size (in Bytes)	Sample
otrNo	The otr sequence should reflect the last otr no sent by the server. For the first request of the day, set it to 0.	String	16	0
srchFilter	Search Filter	String	50	• OTRINQ – Download all OTR files
fill1	Filler	String	10	Leave it blank
fill2	Filler	String	10	Leave it blank

Sample Request

Request Header:

```
POST /api/ncms-cm/otr-inquiry HTTP/1.1
Host: uat.connect2nsccl.com
Authorization: Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
consumerKey: consKey
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
Content-Type: application/json
```

Request Body:

```
{
   "data":
"i3fJhLKZHhGdanX8csAP4sfqaXse/PO2ek84FMMocd8hLMVgHuOQREft6QsruHisVrqTBjDqAL4guyyVLLV3RNr
YRRa3uuhRj+BdJI7UJE……A6dy/yJaem0qa40X+5iUvteGpQ7BIpQ=="
}
```


Sample output of the decrypted **request body payload data** in JSON format:

The access token in the Authorization header, as well as the data parameter in the request body, are required to be AES-encrypted. When making an API call, the Base64-encoded string of these encrypted values must be used. Members should perform encryption using the AES secret key and IV provided at the time of token generation alongside the access token.


```
{
        "version": "1.0",
        "data":
        {
                "msgId": "00001201310140000001",
                "dataFormat": "CSV:CSV",
                " otrInquiry ": "0,OTRINQ,,"

        }
}
```


The access token in the authorization header, as well as the data parameter in the request body, are required to be AES-encrypted. When making an API call, the Base64-encoded string of these encrypted values must be transmitted. Members should perform encryption using the AES secret key and IV provided at the time of token generation alongside the access token.

Response

Response Payload Structure (JSON)

#	Parameter Name	Data Type	Description	Sample Value
1	status	String	Response Status	success/error
2	messages.code	String	Refer to section “Message based response code”	01010000
3	data.msgId	String	Unique request number sent back in the response	XXXXX201310140000001
4	data.otrInquiry	CSV	Data Structure specified below. Structure Separator “^”	Refer Sample Response

Trade Action Inquiry (otrInquiry) Response Packet Structure (CSV – Structure Separator “^”)

Field Name	Description	Data Type	Size (in Bytes)	Sample
sysinfoResData	Control Record Structure	CSV	-	System Info Response Structure given below
otrResData	Detail Record Structure	CSV	-	Field Separator – “,” Record Separator – “^”

Field Name	Description	Data Type	Size (in Bytes)	Sample
				OTR Response Data Structure given below

Control Record Structure (sysinfoResData) (CSV)

Field Name	Description	Data Type	Size (in Bytes)	Sample
mktSts	Market Status For list of Market Status please refer "Reference Codes" section	Short	2	6
currTrdDate	Current Trade Date (YYYYMMDD)	Long	8	20251028
sfill1	Filler	String	10	
sfill1	Filler	String	10	
maxSeqNo	Max sequence number sent in response	long	8	47
noOfRec	Count of trades sent in the response	int	4	2

Detail Record Structure (otrResData) (CSV) (Field Separator – "," | Record Separator – "^")

#	Field Name	Description	Data Type	Size (in Bytes)	Sample
1	seqNo	Sequence number of Order	String	16	421523
2	otrNo	OTR Number	INT	4	3
3	cpCode	CP CODE	String	12	CP0000000001
4	trdTmCd	TM code	String	5	11782
5	trdSecToken	Securities Token	String	11	0
6	otrQty	Allocation/Unallocation Quantity	INT	4	100
7	otrPrc	Allocation/Unallocation Price	Double	8	646272000.00
8	trdDt	Trade Date	Date		20251028
9	bsFlag	Buy/Sell Flag. Refer Section "Reference Codes - Buy Sell Flag"	Short	2	1
10	trdSTPNo	Settlement No	String	1	M
11	trdSTPType	Settlement Type	String	7	2026012
12	contractNoteNo	Contract Note	String	20	123456
13	confFlag	Confirmation Flag	String	2	Y
14	otrErrCd	OTR Error Code	String	20	NA
15	trdSecSymbl	Securities Symbol	String	10	TCS
16	trdSecSeries	Securities Series	String	2	EQ
17	uniqId	Trade Unique ID	String	20	1544221684
18	exchangeCd	Exchange Code Refer Section "Reference Codes - Exchange Code"	Short	2	1
19	Filler1	Filler	Double	8	

#	Field Name	Description	Data Type	Size (in Bytes)	Sample
20	Filler2	Filler	Double	8	
21	Filler2	Filler	Double	8	
22	Filler3	Filler	String	16	
23	Filler4	Filler	String	16	
24	Filler5	Filler	String	16	

Sample Failure Response

Wrong access token or expired access token

```
HTTP/1.1 401 UNAUTHORIZED
Content-Type: application/json
{
  "messages":{"code":"0100401"},
  "status":"error"
}
```

Error in encryption

```
HTTP/1.1 400 BAD_REQUEST
Content-Type: application/json
{
  "messages":{"code":"0100400"},
  "status":"error"
}
```

Sample Success Response

The payload in the response to the API call, will be AES-encrypted string. The Base64-encoded string of this encrypted value will be transmitted. Members should perform decryption using the AES secret key and IV provided at the time of token generation alongside the access token.

Actual Response

```
HTTP/1.1 200 OK
Content-Type: application/json
{
        "status": "success",
        "messages":
        {
                "code": "01010000"
        },
        "data":
        "i1iw0PPNS0DJNSX8bswCpY65aWYSobTpBCR/UZAqacBiO8smQeHRa338+Ro7qGi8VOXSVzPCEP04o
        XWHd/AjOozKTge2vO9WiOJdJY3VJVhdcwZL3Gj4tEbXi4vxft+SfJ9bRxyfh5kMHZXvclnC55mpkHca9fd
        gm8pKmT0SdnQsKMJ11GMUYxKQtDvdzQXyxWI4GY3f"
}
```

Response with Raw Data

```
{
        "status": "success",
        "messages":
        {
                "code": "01010000"
        },
```

```
"data":
{
"msgId": "XXXXX201310140000001",
"trdactInquiry":"6,20251028,,,47,2^ 421523, 3, CP0000000002, 11782, 0, 100, 646272000.00, 20251028, 1, M, 2026012, 123456, Y,NA, TCS, EQ, 1544221684,1,,,,,,^ 7,2026018, CP0000000001,321456,5,RELIANCE,EQ,B,2,100,N,,,,,^ 421524, 2, CP0000000003, 11784, 0, 100, 556272000.00,20251028, 1, M, 2026012, 312456, N,NA, TCS, EQ, 1544221684, 1,,,,,,^
}
}
```


POST /<version>/request/otr-allocation

Request

This API will allow members for OTR allocation, reallocation and unallocation using API POST /<version>/ request/ otr-allocation.

Request Header Parameters

#	Parameter Name	Data Type	Description	Sample Value
1	Authorization	String	Bearer token encrypted using the AES received as part of token response. <access_token>	Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
2	nonce	String	A nonce uniquely identifies each server request. It should be a base64-encoded string in the format: ddMMyyyyHHmmssSSS:<6-digit random number>.	MjAwMTIwMTcxNjEyMjE1OTE6
3	consumerKey	String	The Member Consumer Key received as part of API Registration process. <consKey>	consKey

Request Body Payload (JSON)

#	Parameter Name	Data Type	Description	Sample Value
1	Version	String	API version	1.0
2	Data.msgId	Stsring	Unique request number for each request <CODE><YYYYMMDD><nnnnnnn> Member Code (Length: 5) • YYYYMMDD – Date format • nnnnnnn – Running sequence no. starting from one i.e. For first request of the day, it should be (0000001).	XXXXX201310140000001
3	data.stpType	String	Settlement Type	M
4	data.stpNo	String	STP Number	2025115

#	Parameter Name	Data Type	Description	Sample Value
5	data.otrAlloc	JSON	Array of OTR Allocation/Unallocation structure	Max 50000 records allowed per messageID. Structure details given below

Request Packet Structure:

#	Field Name	Description	Data Type	Size (in Bytes)	Sample
1	memCd	TM code	String	5	11782
2	bsFlg	Buy/Sell Flag. Refer Section "Reference Codes - Buy Sell Flag"	Short	2	1
3	secSymbol	Symobl	String	10	TCS
4	secSeries	Series	String	2	EQ
5	trdSecToken	Security token	String	11	0
6	oldcupCd	Old CP CODE	String	12	INST
7	newcupCd	New CP Code	String	12	CP0000000001
8	qty	Allocation/Unallocation Quantity	INT	4	100
9	val	Allocation/Unallocation value	Double	8	646272000.00
10	contractNo	Contract No.	String	20	3487
11	otrNo	OTR Number	INT	4	3
12	uniqId	Trade Unique ID	String	20	1544221684
13	actType	allocaiton or unallocation Refer Section "Reference Codes – Allocation/Unallocation Type"	Short	2	1

Sample Request

Request Header:

POST /api/<ncms-cm>/< otr-allocation > HTTP/1.1
Host: uat.connect2nsccl.com
Authorization: Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
consumerKey: consKey

```
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
Content-Type: application/json

Request Body:

{
   "data":
"i3fJhLKZHhGdanX8csAP4sfqaXse/PO2ek84FMMocd8hLMVgHuOQREft6QsruHisVrqTBjDqAL4guyyVLLV3RNr
YRRa3uuhRj+BdJI7UJE……A6dy/yJaem0qa40X+5iUvteGpQ7BIpQ=="
}
```

Sample output of the decrypted **request body payload data** in JSON format:

The access token in the Authorization header, as well as the data parameter in the request body, are required to be AES-encrypted. When making an API call, the Base64-encoded string of these encrypted values must be used. Members should perform encryption using the AES secret key and IV provided at the time of token generation alongside the access token.

```
{
"version": "1.0",
"data":
{
"msgId": "00001201310140000001",
“stpType”: “M”,
“stpNo”: “2025115”,
"otrAlloc": [
        {
          “memCd”: “11782",
           "bsFlg": 1,
          “secSymbol”: “TCS”
          “secSeries”: “EQ”
          “trdSecToken”: “0”
          “oldcupCd”: “INST”
          “newcupCd”: “CP0000000001”
          "qty": “100”,
          “val”: “646272000.00”
          “contractNo”: “3487”
           " otrNo": "3",
           " uniqId ": "1544221684",
           " actType ": "1",

        }
     ]
   }
}
```

Response

Confidential

Response Data Payload (JSON)				
Sr. No.	**Parameter Name**	**Data Type**	**Description**	**Sample Value**
1	status	String	Response status	success/error
2	messages.success	String	Message	Request submitted successfully
3	data.code	String	Refer Section “Message based response code”.	01010000


```
{
   "status": "Success",
   "messages": {
   "success": " Request submitted successfully."
},
   "data": {
      "code": "01010000"
   }
}
```


POST /<version>/request/generate-iso

This API will allow custodian for generating the iso as well as csv file.

Request

Request Header Parameters

#	Parameter Name	Data Type	Description	Sample Value
1	Authorization	String	Bearer token encrypted using the AES received as part of token response. <access_token>	Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
2	Nonce	String	A nonce uniquely identifies each server request. It should be a base64-encoded string in the format: ddMMyyyyHHmmssSSS:<6-digit random number>.	MjAwMTIwMTcxNjEyMjE1OTE6
3	consumerKey	String	The Member Consumer Key received as part of API Registration process. <consKey>	consKey

Request Body Payload (JSON)

#	Parameter Name	Data Type	Description	Sample Value
1	Version	String	API version	1.0
2	Data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnn> Member Code (Length: 5) • YYYYMMDD – Date format • nnnnnnn – Running sequence no. starting from one i.e. For first request of the day, it should be (0000001).	XXXXX201310140000001
3	Data.fileType	String	File Type can be either CSV or ISO	CSV or ISO
4	Data.batchNo	Number	Batch No is not Mandatory, The Batch No will provide the requested batch file	01

Response

Multipart File

Complete data stream of the file contents.

Sample Failure Response

Wrong access token or expired access token

```
HTTP/1.1 401 UNAUTHORIZED
Content-Type: application/json
{
  "messages":{"code":"0100401"},
  "status":"error"
}
```

Error in encryption

```
HTTP/1.1 400 BAD_REQUEST
Content-Type: application/json
{
  "messages":{"code":"0100400"},
  "status":"error"
}
```

Sample Success File Response

Response with ISO Data

The received file structure would be in format <XXXX><YYYYMMDD>.Cnn and sample file is attached below.
XXXX – Custodian Code,
YYYYMMDD – Date Format,
nn – Batch No

The example provided is the ISO file

CUST_20250421.C02.txt

Response with CSV Data

The received file structure would be in format <XXXX>_CP<YYYYMMDD>.csv and <XXXX>_OT<YYYYMMDD>.csv.

XXXX – Custodian Code,
YYYYMMDD – Date Format,

CUST_CP20241104.csv

CUST_OT20241104.csv

POST /<version>/list-inquire/iso

This API will allow custodian for inquiring the list of files uploaded and return file generated.

Request

Request Header Parameters

#	Parameter Name	Data Type	Description	Sample Value
1	Authorization	String	Bearer token encrypted using the AES received as part of token response.	Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==

#	Parameter Name	Data Type	Description	Sample Value
			<access_token>	
2	nonce	String	A nonce uniquely identifies each server request. It should be a base64-encoded string in the format: ddMMyyyyHHmmssSSS:<6-digit random number>.	MjAwMTIwMTcxNjEyMjE1OTE6
3	consumerKey	String	The Member Consumer Key received as part of API Registration process. <consKey>	consKey

Request Body Payload (JSON)

#	Parameter Name	Data Type	Description	Sample Value
1	Version	String	API version	1.0
2	Data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnn> Member Code (Length: 5) • YYYYMMDD – Date format • nnnnnnn – Running sequence no. starting from one i.e. For first request of the day, it should be (0000001).	XXXXX201310140000001
3	Data.currTrdDate	Long	Current Trade Date (YYYYMMDD)	20251028

Response

Response Data Payload (JSON)				
Sr. No.	**Parameter Name**	**Data Type**	**Description**	**Sample Value**
1	status	String	Response status	success/error
2	timestamp	Timestamp	Timestamp of response	
3	data	Object	Contains categorized file list	
4	uploadedFiles	Array	List of uploaded files	
5	returnFiles	Array	List of return files	

```
{
```

```
"status": "Success",
“timestamp”: “”,
“data”: {
            uploadedFiles [
              20250421.I01,
              20250421.I02,
            ],
            returnFiles [
              CUST_20250421.C02,
              CUST_CP20241104.csv,
              CUST_OT20241104.csv,
            ],
          }

}
```

POST /<version>/inquire/return-iso

This API will allow custodian for getting the return file.

Request

Request Header Parameters

#	Parameter Name	Data Type	Description	Sample Value
1	Authorization	String	Bearer token encrypted using the AES received as part of token response. <access_token>	Basic MRZmwzdkje382jdw8ue93j dCaxH9rAM3hVlMJzFg==
2	nonce	String	A nonce uniquely identifies each server request. It should be a base64-encoded string in the format: ddMMyyyyHHmmssSSS:<6-digit random number>.	MjAwMTIwMTcxNjEyMjE1OTE6
3	consumerKey	String	The Member Consumer Key received as part of API Registration process. <consKey>	consKey

Request Body Payload (JSON)

#	Parameter Name	Data Type	Description	Sample Value
1	Version	String	API version	1.0
2	Data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnn> Member Code (Length: 5) • YYYYMMDD – Date format	XXXXX201310140000001

#	Parameter Name	Data Type	Description	Sample Value
			• nnnnnnn – Running sequence no. starting from one i.e. For first request of the day, it should be (0000001).	
3	Data.fileType	String	File Type can be either CSV or ISO	CSV or ISO
4	Data.fileName	String	Enter the name received from list-inquiry-iso api. For e.g <XXXX>_<YYYYMMDD>.C<nn> or <XXXX>_<XX><YYYYMMDD>.csv • XXXX – Custodian Code • YYYYMMDD – Date format • nn – Batch No • XX – CP file or OTR file (CO or OT)	CUST_20250421.C02 or CUST_CP20241104.csv or CUST_OT20241104.csv

Response

Multipart File

Complete data stream of the file contents.

Sample Failure Responses

Wrong access token or expired access token

```
HTTP/1.1 401 UNAUTHORIZED
Content-Type: application/json
{
  "messages":{"code":"0100401"},
  "status":"error"
}
```

Error in encryption

```
HTTP/1.1 400 BAD_REQUEST
Content-Type: application/json
{
  "messages":{"code":"0100400"},
  "status":"error"
}
```

POST /<version>/request/upload-iso

This API can be used to upload the iso file by Custodian for approval and rejection of trade and OTR confirmation.

Request

Multipart File

Complete data stream of the file contents

Request Header Parameters

#	Parameter Name	Data Type	Description	Sample Value
1	Authorization	String	Bearer token encrypted using the AES received as part of token response. <access_token>	Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
2	nonce	String	A nonce uniquely identifies each server request. It should be a base64-encoded string in the format: ddMMyyyyHHmmssSSS:<6-digit random number>.	MjAwMTIwMTcxNjEyMjE1OTE6
3	consumerKey	String	The Member Consumer Key received as part of API Registration process. <consKey>	consKey

Request Body Payload (form-data)

#	Parameter Name	Data Type	Description	Sample Value
1	Version	String	API version	1.0
2	Data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnn> Member Code (Length: 5) • YYYYMMDD – Date format • nnnnnnn – Running sequence no. starting from one i.e. For first request of the day, it should be (0000001).	XXXXX201310140000001
3	Data.fileType	String	File Type will be ISO	ISO
4	Data.fileName	String	The format of the file should be as below <YYYYMMDD>.I<nn> • YYYYMMDD – Date format • nn – Batch No starting from one . i.e For first request of the day, it should be (01).	20250421.I01

Request file ISO Data

Sample file format which needs to be uploaded should

20250421.I02.txt

Response

Multipart File

Complete data stream of the file contents.

Sample Failure Response

Wrong access token or expired access token

```
HTTP/1.1 401 UNAUTHORIZED
Content-Type: application/json
{
  "messages":{"code":"0100401"},
  "status":"error"
}
```

Error in encryption

```
HTTP/1.1 400 BAD_REQUEST
Content-Type: application/json
{
  "messages":{"code":"0100400"},
  "status":"error"
}
```

Sample Success File Response

Response with Log file

The received file structure would be in format <XXXX>_<XXXX>_STP_CNF_ < ddMMHHmmss >.log and sample file are attached below.

XXXX – Custodian Code

ddMMHHmmss - Date and Time


CUST_CUST_STP_CN F_2405175248.LOG


CUST_CUST_STP_CN F_2405182902.LOG

POST /<version>/list-inquire/upload-iso

This API will allow Custodian for inquiring the list of files uploaded and return file generated.

Request

Request Header Parameters

#	Parameter Name	Data Type	Description	Sample Value
1	Authorization	String	Bearer token encrypted using the AES received as part of token response. <access_token>	Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
2	nonce	String	A nonce uniquely identifies each server request. It should be a base64-encoded string in the format: ddMMyyyyHHmmssSSS:<6-digit random number>.	MjAwMTIwMTcxNjEyMjE1OTE6
3	consumerKey	String	The Member Consumer Key received as part of API Registration process. <consKey>	consKey

Request Body Payload (JSON)

#	Parameter Name	Data Type	Description	Sample Value
1	Version	String	API version	1.0
2	Data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnn> Member Code (Length: 5) • YYYYMMDD – Date format • nnnnnnn – Running sequence no. starting from one i.e. For first request of the day, it should be (0000001).	XXXXX201310140000001
3	Data.currTrdDate	Long	Current Trade Date (YYYYMMDD)	20251028

Response

Response Data Payload (JSON)				
Sr. No.	**Parameter Name**	**Data Type**	**Description**	**Sample Value**
1	status	String	Response status	success/error
2	timestamp	Timestamp	Timestamp of response	
3	data	Object	Contains categorized file list	
4	uploadedFiles	Array	List of uploaded files	
5	returnFiles	Array	List of return files	

```
{
   "status": "Success",
   “timestamp”: “”,
   “data”: {
                uploadedFiles [
                  20250421.I01,
                  20250421.I02,
                ],
                returnFiles [
                  CUST_CUST_STP_CNF_2405175248,
                  CUST_CUST_STP_CNF_2405182902,
                ],
             }

}
```

POST /<version>/inquire-upload/return-file

This API will allow custodian for getting the return file.

Request

Request Header Parameters

#	Parameter Name	Data Type	Description	Sample Value
1	Authorization	String	Bearer token encrypted using the AES received as part of token response. <access_token>	Basic MRZmwzdkje382jdw8ue93j dCaxH9rAM3hVlMJzFg==
2	nonce	String	A nonce uniquely identifies each server request. It should be a base64-encoded string in the format: ddMMyyyyHHmmssSSS:<6-digit random number>.	MjAwMTIwMTcxNjEyMjE1OTE6
3	consumerKey	String	The Member Consumer Key received as part of API Registration process. <consKey>	consKey

Request Body Payload (JSON)

#	Parameter Name	Data Type	Description	Sample Value
1	Version	String	API version	1.0
2	Data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnn> Member Code (Length: 5) • YYYYMMDD – Date format	XXXXX201310140000001

#	Parameter Name	Data Type	Description	Sample Value
			• nnnnnnn – Running sequence no. starting from one i.e. For first request of the day, it should be (0000001).	
3	Data.fileName	String	Enter the name field received from inquiry-upload-iso api. For e.g <XXXX>_<XXXX>_STP_CNF_<ddMMHHmmss >.log XXXX – Custodian Code ddMMHHmmss - Date and Time	CUST_CUST_STP_CNF_2405175248

Response

Multipart File

Complete data stream of the file contents.

Appendix A - Reference Codes

Market Type

Code	Description
1	Normal
2	Odd Lot
3	Spot
4	Auction
5	Call Auction 1
6	Call Auction 2

Market Status

Code	Description
1	Preopen shutdown
2	Normal Market Preopen ended
3	Open Msg
4	Close Msg
5	Closing Start
6	Closing End

Transaction Code

Code	Description
6001	Original Trade
5445	Trade Modification (Client Modification)
9001	All Actions (CP modification)

Activity Type

Code	Description
1	Trade Modification (Client Modification)
2	Original Trade
5	CP Modifications

Book Type

Code	Description
1	Regular Lot
2	Special Terms
3	Stop Loss / MIT
4	Negotiated Trade
5	Odd Lot
6	Spot
7	Auction
11	Call Auction 1
12	Call Auction 2

Client Type

Code	Description
1	Cli
2	Pro

Buy Sell Flag

Code	Description
1	BUY
2	SELL

Allocation/Unallocation Type

Code	Description
1	Allocation
2	Unallocation

Trade Status

Code	Description
P	Pending

Exchange Code

Code	Description
1	NSE
2	BSE
3	MSE

Action Type

Code	Description
1	Original Trade
6	Buy Side Pre Image (Old CP)
7	Sell Side Pre Image (Old CP)
8	Buy Side Post Image (New CP)
9	Sell Side Post Image (New CP)

ANNEXURE 1 – Error Codes

Async response code shall be populated in the field "actErrCd" of the message

Error	Error Code
MODIFY ORDER DOES NOT EXISTS IN TRADES	1
ORDER MODIFICATION – MEMBER DISABLED	3
ORDER MODIFICATION – MEMBER ID MISSMATCH	4
MODIFY ORDER – INVALID NEW CUP CODE	7
MODIFY ORDER – INVALID OLD CUP CODE	8
MODIFY CP – CLIENT CODE NOT NULL	11
MODIFY CP – NOT LATEST IMAGE	12
MODIFY CP – CUP CODE NOT ACTIVE	14
MODIFY ORDER – BUY SELL FLG MISMATCH	15
MODIFY CP – OLD NEW CP CODE SAME	16
MODIFY CP – INCORRECT LENGTH	17
MODIFY CP – MARGIN INDICATOR	18
MODIFY OTR – NOT EXIST IN TABLE	51
MODIFY OTR – TM CD MISMATCH	52
MODIFY OTR – INVALID CONTRACT NO	54
MODIFY OTR – BUY SELL FLAG MISMATCH	55
MODIFY OTR – CUP CD MISMATCH	56
MODIFY OTR – QTY EXCEEDS TOTAL QTY	57
MODIFY OTR – VALUE EXCEEEDS TOTAL VAL	58
MODIFY OTR – TRADE DATE MISMATCH	59
MODIFY OTR – MODIFATION TIMING DISABLED	60
MODIFY OTR – ALREADY CONFIRMED	61
MODIFY OTR – MAX VAL QTY MISMATCH	62
MODIFY OTR – CUPCD NOT ACTIVE	63
MODIFY OTR – INVALID QTY VAL ACTYTYPE	64

MODIFY OTR – ALLOCTION TO INST NOT VALID	65
MODIFY OTR – SEC SYMBOL SERIES MISMATCH	66
MODIFY OTR – SEC TOKEN MISMATCH	67
MODIFY OTR – INVALID ACTYTYPE	68
MODIFY OTR –ALREADY UNALLOCATED	69
MODIFY OTR – DB UPDATE EXCEPTION	70
INVALID CP MOD ACTTYPE NORMAL	91
INVALID PARTIALCLI MOD ACTTYPE NORMAL	92
INVALID PARTIAL CP MOD ACTTYPE NORMAL	93
INVALID PARTIAL CLICP MOD ACTTYPE NORMAL	94
MODIFY OTR - QTY NOT IN MULTIPLES OF LOT SIZE	98
INVALID CC CODE	110
INVALID OTR	111
OTR REMOVAL REQUEST ALREADY SENT TO OTHER CC	112
OTR REMOVAL REQUEST REJECTED BY OTHER CC	113
OTR ASSIGNMENT NOT ESISTS FOR OTR REMOVAL REQUEST FROM OTHER CC	114
CHECKSUM ALREADY SENT TO OTHER CC	115

Appendix B - Response Codes

There can be two types of response codes

- HTTP response codes
- Message based response codes

HTTP response code

- HTTP responses shall be generated during login with success or failure status
- HTTP response shall also be generated in case of any authentication/input validation failure of the message.

HTTP response codes are as follows:

HTTP Response Codes			
Sr. No.	**Reason**	**Meaning**	**HTTP Response Code**
1	SUCCESS	Request was handled successfully	200
2	UNKNOWN_ERROR	Internal Server Error: Internal server error has occurred in our platform	500
3	SVC_UNAVAILABLE	The server is currently unable to handle the request due to a temporary overloading or maintenance of the server	503

HTTP Response Codes			
Sr. No.	**Reason**	**Meaning**	**HTTP Response Code**
4	METHOD_NOT_ALLOWED	Unsupported HTTP method: A request was made for a resource using a request method not supported by that resource (e.g. using POST instead of GET)	405
5	BAD REQUEST	PARAMETER_ABSENT – There's a required parameter which is not present in the request	400
6	BAD REQUEST	DATA_INVALID – The data is not in correct format and not recognized by our system	400
7	BAD REQUEST	DATA_FORMAT_REJECTED – Unsupported Data format parameter value	400
8	UNAUTHORIZED: Failed to authenticate the request	CONSUMER_KEY_UNKNOWN – The provided Consumer Key (API key) is not registered in our system OR service is not registered	401
9	UNAUTHORIZED: Failed to authenticate the request	TOKEN_INVALID – The provided token is not registered in our system	401
10	UNAUTHORIZED: Failed to authenticate the request	UNAUTHORIZED: • Unauthorized requestor IP address • API access disabled	401
11	PERMISSION_DENIED	Subscriber has temporarily disallowed access to his private data	403
12	The requested URL was not found	The requested URL was not found	404
13	REQUEST_NOT_FOUND	Registered request not found	570

Message based response code

- Message based response code shall be populated in the field "code" of the JSON response message
- It shall be of the format below
  - First four characters (Field Identifier): refers to specific field or the entire message
  - Next characters (Validation code): refer to specific validation failure or success. Success code shall be populated only on successful acceptance of the message.

Field Identifier is as follows:

Sr. No.	Module	Field Name	Field Identifier
1	Entire Message	NA	0101
2	Input Data Parameter	msgId	0102
3	Input Data Parameter	seqNo	0107
4	Input Data Parameter	srchFilter	0108
5	Input Data Parameter	noOfRec	0110
6	Input Data Parameter	otrNo	0112

Sr. No.	Module	Field Name	Field Identifier
7	Input Data Parameter	fileType	0132
8	Input Data Parameter	stpType	0121
9	Input Data Parameter	stpNo	0122
10	Input Data Parameter	bsFlag	0113
11	Input Data Parameter	currTrdDate	0133
12	Input Data Parameter	fileName	0134
13	Input Data Parameter	batchNo	0135
14	Input Data Parameter	uniqId	0119
15	Input Data Parameter	trdSecToken	0125
16	Input Data Parameter	ordNo	0115
17	Input Data Parameter	newCPCode	0111
18	Input Data Parameter	memCd	0114
19	Input Data Parameter	secSymbol	0123
20	Input Data Parameter	secSeries	0124
21	Input Data Parameter	oldcupCd	0136
22	Input Data Parameter	qty	0126
23	Input Data Parameter	val	0127
24	Input Data Parameter	contractNo	0128
25	Input Data Parameter	actType	0130

Validation codes are as follows:

Sr. No.	Validation	Validation Type	Validation Code	Validation performed on Field
1	Submitted to server successfully	Message Level	0000	Entire Message
2	Duplicate request received	Message Level	0001	Entire Message
3	All HTTP status codes	HTTP error codes	HTTP Response codes. Refer section “HTTP Response Code”.	Entire Message
4	Mismatch in control and data record	Message Level	0200	Entire Message
5	Minimum Required Length	Generic	0201	msgId, memCd
6	Maximum Required Length	Generic	0202	msgId, memCd
7	Mandatory field	Generic	0204	msgId, memCode, noOfRec, seqNo, srchFilter, trdNo, stpType, uniqId, bsFlag, dataFormat, filetype, file, stpNo, otrNo
8	Data Format like Message Id	Generic	0206	msgId

Sr. No.	Validation	Validation Type	Validation Code	Validation performed on Field
9	Minimum allowed value	Generic	0207	seqNo, noOfRec, trdno, bsFlag, stpType, stpNo, otrNo, newCPCode, oldcupCd
10	Maximum allowed value	Generic	0208	noOfRec, bsFlag,
11	Invalid Value	Generic	0209	seqNo, srchFilter, trdno, stpType, stpNo, bsFlag, otrNo, batchNo, filename, uniqId, trdSecToken, ordNo, secSymbol, secSeries, secToken, qty, val, contractNo, actType
12	System Error	Generic	0241	NA
13	Service Unavailable	Generic	0242	NA
14	Request Parsing Error: Invalid Request Structure	Generic	0243	NA
15	Invalid Member Code	Generic	0114	Msgid, memCd
16	Invalid Date	Generic	0246	Msgid, currTrdDate
17	Window Closed	Generic	0122	NA

***** End of Document *****