

Protocol for WEB API for Members

Margin Derivatives (FO/CD/CO)

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

© Copyright NSE Clearing Ltd (NCL). All rights reserved. Unpublished rights reserved under applicable copyright and trades secret laws.

The contents, ideas and concepts presented herein are proprietary and confidential.
Duplication and disclosure to others in whole, or in part is prohibited

Revision History

Date	Change Description	Edited By	Version
20-Jan-2026	Initial version		1.0
05-Jun-2026	New endpoints added		1.1

Table of Contents

Revision History	2
Introduction	4
General Instructions.....	4
HTTP Status Codes	4
Common Error Response JSON	5
Segment Environment Details	5
FO Segment.....	5
CD Segment.....	5
CO Segment.....	6
API Consumer Registration	6
API Security	6
Clearing Corporation APIs	7
POST /<version>/request/token	7
POST /<version>/request/cm-margins	9
POST /<version>/request/tm-margins.....	12
POST /<version>/request/cli-margins.....	16
POST /<version>/request/positions.....	20
POST /<version>/request/moi	23
POST /<version>/request/mwpl	26
POST /<version>/request/cli-margins/generate-all.....	29
POST /<version>/request/cli-margins/inquiry.....	31
POST /<version>/request/positions/generate-all	33
POST /<version>/request/positions/inquiry.....	35
Appendix A - Response Codes.....	37
HTTP response code.....	37
Message based response code	38
Sample example for success or failure code.....	39

Introduction

This document provides information on the Web APIs used for programmatic access margin and positions related data between NCL's MARGINS 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 APIs 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."
  ]
}
```

Segment Environment Details

FO Segment

Base URL of RISK and Margin Management API endpoints mentioned in this document will be as follows:

Testing Environment: <https://uat.connect2nsccl.com/Margins-FO-API/>

Live Environment: <https://www.connect2nsccl.com/ Margins-FO-API />

CD Segment

Base URL of RISK and Margin Management API endpoints mentioned in this document will be as follows:

Testing Environment: <https://uat.connect2nsccl.com/Margins-CD-API/>

Live Environment: <https://www.connect2nsccl.com/ Margins-CD-API />

CO Segment

Base URL of RISK and Margin Management API endpoints mentioned in this document will be as follows:

Testing Environment: <https://uat.connect2nsccl.com/Margins-CO-API/>

Live Environment: <https://www.connect2nsccl.com/ Margins-CO-API />

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.

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/margin-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 ICaxH9rAM3hVIMJzFg==
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>.	MjAwMTIwMTcxNjEyMjE0TE6
3	grant_type	String	Value MUST be set to "client_credentials".	client_credentials

Sample Request

UAT URL sample:

<https://uat.connect2nsccl.com/Margins-FO-API/v2/request/token>

<https://uat.connect2nsccl.com/Margins-CD-API/v2/request/token>

<https://uat.connect2nsccl.com/Margins-CO-API/v2/request/token>

```
POST /v2/request/token HTTP/1.1
Host: uat.connect2nsccl.com
Content-Type: application/x-www-form-urlencoded
Authorization: Basic MRZmwzCl6.....SGqXlCaxH9rAM3hVlMJzFg==
nonce: MjAwMTIwMTcxNjEyMjE0TE6ODk0MjY3
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/KIVp6B0uKttHP91FlFNEpEZIMI43eWMcyOUEsvqr5fj
4snHA125K8++8U/RtCYC7r3bW+2U/P6J/nG2qNtFGRoM1Koc0KVMcFgNptJC6BK2Bs6Fo44KA
OtJ97NB1f9R0/WPxJy3dqi2A6zXo9tqn22JfgaFq/2JWZT0kX1grGkBEJZZImUiA0+ftpV3Jf
qrnYwZAtCr+cM7nbhab8Mri8cWBeHNG1pAlU/AljcDvar5/NTdMDSClmkuw7ngXQpnOFX1mPl
AlTAYLHOTnuau3KoE653lze2+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 "3600" denotes that the access token will expire in one hour from the time the response was generated.	3600
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"
}
```

POST /<version>/request/cm-margins

This API will allow members to inquire for margins using API POST /<version>/request/ cm-margins

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 dCaxH9rAM3hVIMJzFg==
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>.	MjAwMTIwMTcxNjEyMjE0 TE6
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	2.0
2	data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnnn> Member Code (Length: 4 or 5) • YYYYMMDD – Date format • nnnnnnnn – Running sequence no. starting from one i.e. For first request of the day, it should be (0000001).	XXXXX201310140000001
3	data.memCode	String	Member code	XXXXX

Sample Request

Request Header:

```
POST /v2/request/cm-margins HTTP/1.1
Host: uat.connect2nsccl.com
Authorization: Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
consumerKey: consKey
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
Content-Type: application/json
```

Request Body:

```
{
  "data":
  "i3fJhLKZHhGdanX8csAP4sfqaxse/PO2ek84FMMocd8hLMVgHuOQREft6QsruHisVrqTBjDq
AL4guyyVLLV3RnrYRRa3uuhRj+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": "2.0",
  "data": {
    "msgId": "XXXXXX202509030000001",
    "memCode": "XXXXXX"
  }
}
```

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	String	Refer to section “Message based response code”	
3	Timestamp	String	Date time stamp	17 Nov 2025 14:11:58

Detail Record Structure (CSV) (Separator – “,”)

#	Field Name	Description	Data Type	Size(In Byte)	Sample
1	cmCode	CM Code	String	6	XXXXXX
2	cashCollateral	Cash Collateral(A)	Double	8	6350002144.25
3	nonCashCollateral	Non-Cash Collateral(B)	Double	8	50136240.00
4	effeNonCash	Effective Non-Cash(C)	Double	8	50136240.00
5	effeCollateral	Effective Collateral(D=A+C)	Double	8	6400138384.25
6	propMargin	Prop Margin(E=J+K+L+M)	Double	8	3334843727.87
7	tmExcessMargin	TM Margin>90%(F)	Double	8	181848144.38
8	totalMargin	Total margin(G=E+F)	Double	8	3516691872.25
9	MLnw	MLNW(H)	Double	8	5000000.00
10	Utilization	Utilization%(I=(G*100)/(D-H))	Double	8	54.99
11	iniMarginProp	Initial Margin Prop(J)	Double	8	404419232.63
12	cobgProp	FO - COBG Prop (K) CD - COBG Prop (K) CO - ICMTM Prop Value (K)	Double	8	0.00
13	deliveryMarginProp/additionalMargin	FO - Delivery Margin Prop(L) CD – Not applicable CO – Additional Margin (L)	Double	8	12732175.15
14	extremeLossMargin	Extreme Loss Margin(M)	Double	8	2917692320.09
15	mtmP/Lprop	MTM Profit/Loss Prop(N)	Double	8	-38638348.10
16	Filler1	Filler	Double	8	
17	Filler2	Filler	Double	8	
18	Filler3	Filler	Double	8	
19	Filler4	Filler	String	16	
20	Filler5	Filler	String	16	
21	Filler6	Filler	String	16	

Sample Failure Response

Wrong access token or expired access token

```
HTTP/1.1 401 UNAUTHORIZED
Content-Type: application/json
{
  "messages":{"code":"0101401"}
  "status":"error"
}
```

Error in encryption

```
HTTP/1.1 400 BAD_REQUEST
Content-Type: application/json
{
  "messages":{"code":"0101400"}
  "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"
  "message": ""
  "timeStamp": "17 Nov 2025 14:11:58"
  "data":
  "iliw0PPNS0DJNSX8bswCpY65aWYSobTpBCR/UZAqacBiO8smQeHRa338+Ro7qGi8VOXSVzPC
EP04oXWHd/AjOozKTge2vO9WiOJdJY3VJVhdcwZL3Gj4tEbXi4vxft+SfJ9bRxyfh5kMHZXvc
lnC55mpkHca9fdgm8pKmT0SdnQsKMJl1GMUYxKQtDvdzQXyxWI4GY3f"
}
```

Response with Raw Data

```
{
  "status": "SUCCESS"
  "message": ""
  "timeStamp": "17 Nov 2025 14:11:58"
  "data": [
    {
      XXXXXX, 6350002144.25, 50136240.00, 50136240.00, 6400138384.25, 3334843727.87,
      181848144.38, 3516691872.25, 5000000.00, 54.99, 404419232.63, 0.00, 12732175.15
      , 2917692320.09, -38638348.10, , , , , , }
    ]
}
```

POST /<version>/request/tm-margins

This API will allow members to inquire for margin using API POST /<version>/request/tm-margins

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 dCaxH9rAM3hVIMJzFg==
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>.	MjAwMTIwMTcxNjEyMjE0 TE6
3	consumerKey	String	The Member Consumer Key received as part of API Registration process.	consKey

#	Parameter Name	Data Type	Description	Sample Value
			<consKey>	

Request Body Payload (JSON)

#	Parameter Name	Data Type	Description	Sample Value
1	Version	String	API version	2.0
2	data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnn> Member 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).	XXXXXX201310140000001
3	data.memCode	String	Member code	XXXXX

Sample Request

Request Header:

```
POST /v2/request/tm-margins HTTP/1.1
Host: uat.connect2nsccl.com
Authorization: Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
consumerKey: consKey
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
Content-Type: application/json
```

Request Body:

```
{
  "data":
  "i3fJhLKZHhGdanX8csAP4sfqaXse/PO2ek84FMMod8hLMVgHuOQREft6QsruHisVrqTBjDq
AL4guyyVLLV3RnrYRRa3uuhRj+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": "2.0",
  "data": {
    "msgId": "XXXXXX202509030000001",
    "memCode": "XXXXXX"
  }
}
```

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	String	Refer to section “Message based response code”	
3	timestamp	String	Date time stamp	17 Nov 2025 14:11:58

Detail Record Structure (CSV) (Separator – “,”)

#	Field Name	Description	Data Type	Size(In Byte)	Sample
1	tm/CpCode	TM /CP Code	String	12	XXXXXX
2	tm/CpName	TM /CP Name	String	200	Mr.ABC Ltd
3	cashCollateral	Cash Collateral(A)	Double	8	36200000.00
4	nonCashCollateral	Non-Cash Collateral(B)	Double	8	19160897.19
5	effectiveDeposit	Effective Deposit (C=A+MIN (A, B))	Double	8	55360897.19
6	propMargin	Prop Margin(D=L+M+N+O)	Double	8	57713443.38
7	cliMargin	CLI Margin>90%(E)	Double	8	408974.68
8	nonCashBenTM	Non-cash Benefit TM(F)	Double	8	4410000.00
9	nonCashBenCM	Non-cash Benefit CM(G)	Double	8	3080236.90
10	totalMargin	Total margin(H=D+E-F-G)	Double	8	50632181.16
11	utilization	FO/CD - Utilization%(I=H/C) CO - Utilization%(I=D+E/C)	Double	8	91.45
12	tmMargin	FO/CD - TM Margin>90% (J=MAX(H-(C*0.9),0) CO -TM/CP Margin>90% (J=MAX(D+E)-(C*0.9),0)	Double	8	807373.68
13	iumtm	FO - Intraday Uncrystallized MTM Loss(K) CD - NA CO - NA	Double	8	10755.00
14	iniMargin	Initial Margin Prop(L)	Double	8	5052272.00
15	cobg	FO - COBG Prop (M) CD - COBG Prop (M) CO - ICMTM Prop Value (M)	Double	8	22620.00

16	deliveryMargin/additionalMargin	FO - Delivery Margin Prop(N), CD – Not applicable CO – Additional Margin (N)	Double	8	1308243.28
17	extremeLossMargin	Extreme Loss Margin Prop(O)	Double	8	51330308.10
18	Filler1	Filler	Double	8	
19	Filler2	Filler	Double	8	
20	Filler3	Filler	Double	8	
21	Filler4	Filler	String	16	
22	Filler5	Filler	String	16	
23	Filler6	Filler	String	16	

Sample Failure Response

Wrong access token or expired access token

```
HTTP/1.1 401 UNAUTHORIZED
Content-Type: application/json
{
  "messages": {"code": "0101401"},
  "status": "error"
}
```

Error in encryption

```
HTTP/1.1 400 BAD_REQUEST
Content-Type: application/json
{
  "messages": {"code": "0101400"},
  "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",
  "message": "",
  "timeStamp": "17 Nov 2025 14:11:58",
  "data":
  "i1iw0PPNS0DJNSX8bswCpY65aWYSobTpBCR/UZAqacBiO8smQeHRa338+Ro7qGi8VOXSVzPC
  EP04oXWHd/AjOozKTge2vO9WiOJdJY3VJVhdcwZL3Gj4tEbXi4vxft+SfJ9bRxyfh5kMHZXvc
  lnC55mpkHca9fdgm8pKmT0SdnQsKMJ11GMUYxKQtDvdzQXyxWI4GY3f"
}
```

Response with Raw Data

```
{
  "status": "SUCCESS",
  "message": "",
```

```

"timestamp": "17 Nov 2025 14:11:58",
"data": [
  {
    XXXXX,Mr.ABCLtd,36200000.00,19160897.9,55360897.19,57713443.38,408974.68,
    4410000.00,3080236.90,50632181.16,91.45,807373.68,10755.00,5052272.00,226
    20.00,1308243.28,51330308.10,,,,, }
  ]
}

```

POST /<version>/request/cli-margins

This API will allow members to inquire for margin using API POST /<version>/request/client-margins

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 dCaxH9rAM3hVIMJzFg==
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>.	MjAwMTIwMTcxNjEyMjE0 TE6
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	2.0
2	data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnnn> Member Code (Length: 4 or 5) • YYYYMMDD – Date format • nnnnnnnn – Running sequence no. starting from one i.e. For first request of the day it should be (0000001).	XXXXX201310140000001
3	data.memCode	String	Member code	XXXXX
4	data.cliList	JSON	Array of client code. Max 50000 records allowed per messageID.	[“CLI0000000”,“CLI1000”]

#	Parameter Name	Data Type	Description	Sample Value

Sample Request

Request Header:

```
POST /v2/request/cli-margins HTTP/1.1
Host: uat.connect2nsccl.com
Authorization: Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
consumerKey: consKey
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
Content-Type: application/json
```

Request Body:

```
{
  "data":
  "i3fJhLKZHhGdanX8csAP4sfqaXse/PO2ek84FMMocd8hLMVgHuOQREft6QsruHisVrqTBjDq
AL4guyyVLLV3RNRyRRa3uuhRj+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": "2.0"
  "data": {
    "msgId": "XXXXXX2025090300000001"
    "tmCode": "XXXXXX",
    "cliList": ["CLI0000000", "CLI1000"]
  }
}
```

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	String	Refer to section “Message based response code”	
3	timeStamp	String	Date time stamp	17 Nov 2025 14:11:58

Detail Record Structure (CSV) (Separator – “,”)

#	Field Name	Description	Data Type	Size(In Byte)	Sample
1	cmCode	CM Code	String	6	XXXXXX
2	cmName	CM Name	String	200	Mr.XYZ Ltd
3	tmCode	TM Code	String	5	XXXXX
4	tmName	TM Name	String	200	Mr.ABC Ltd
5	cliCode	Client Code	String	10	CLI0000000
6	cashCollateral	Cash Collateral	Double	8	19881.09
7	nonCashCollateral	Non-cash Collateral	Double	8	567037.70
8	iniMargins	Initial margins	Double	8	131242.00
9	cobg	FO - COBG CD - COBG CO - ICMTM value	Double	8	21072.50
10	delMargin/additionalMargin	FO - Delivery margin CD – Not applicable CO - Additional Margin	Double	8	0.00
11	extremeLossMargin	Extreme loss margin	Double	8	49624.72
12	totalMargins	Total margins	Double	8	180866.72
13	mtmP/L	MTM Profit/Loss	Double	8	0.00
14	cliExcessMargins	Client margins>90%	Double	8	145080.76
15	cliExcessNoncash	90% of clients excess noncash	Double	8	492440.95
16	eligibleNoncash	Eligible noncash (subject to the extent of TM excess cash/CM excess cash)	Double	8	145080.76
17	Filler1	Filler	Double	8	
18	Filler2	Filler	Double	8	
19	Filler3	Filler	Double	8	
20	Filler4	Filler	String	16	
21	Filler5	Filler	String	16	
22	Filler6	Filler	String	16	

Sample Failure Response

Wrong access token or expired access token

```
HTTP/1.1 401 UNAUTHORIZED
Content-Type: application/json
{
  "messages":{"code":"0101401"}
  "status":"error"
}
```

Error in encryption

```
HTTP/1.1 400 BAD_REQUEST
Content-Type: application/json
{
  "messages":{"code":"0101400"}
  "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"
  "message": ""
  "timeStamp": "17 Nov 2025 14:11:58"
  "data":
  "iliw0PPNS0DJNSX8bswCpY65aWYSobTpBCR/UZAqacBiO8smQeHRa338+Ro7qGi8VOXSVzPC
EP04oXWHd/AjOozKTge2vO9WiOJdJY3VJVhdcwZL3Gj4tEbXi4vxft+SfJ9bRxyfh5kMHZXvc
lnC55mpkHca9fdgm8pKmT0SdnQsKMJl1GMUYxKQtDvdzQXyxWI4GY3f"
}
```

Response with Raw Data

```
{
  "status": "SUCCESS"
  "message": ""
  "timeStamp": "17 Nov 2025 14:11:58"
  "data": [
    {
      XXXXXX,Mr.XYZLtd,XXXXX,Mr.ABCLtd,CLIO000000,19881.09,567037.70,131242.00,
      21072.50,0.00,49624.72,180866.72,0.00,145080.76,492440.95,145080.76^CLI10
      00,19881.09,567037.70,131242.00,21072.50,0.00,49624.72,180866.72,0.00,145
      080.76,492440.95,145080.76,,,,, }
    ]
  }
}
```

POST /<version>/request/positions

This API will allow members to inquire client position using API POST /<version>/request/positions.

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 dCaxH9rAM3hVIMJzFg==
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>.	MjAwMTIwMTcxNjEyMjE0TE6
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	2.0
2	data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnnn> Member Code (Length: 4 or 5) • YYYYMMDD – Date format • nnnnnnnn – Running sequence no. starting from one i.e. For first request of the day it should be (0000001).	XXXXX201310140000001
3	data.memCode	String	Member code	XXXXX
4	data.cliList	JSON	Array of client code. N no of records allowed per messageID. 'N' is scalable.	["CLI0000000"]

Sample Request

Request Header:

```
POST /v2/request/positions HTTP/1.1
Host: uat.connect2nsccl.com
Authorization: Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
consumerKey: consKey
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
Content-Type: application/json
```

Request Body:

```
{
  "data":
  "i3fJhLKZHhGdanX8csAP4sfqaXse/PO2ek84FMMocd8hLMVgHuOQREft6QsruHisVrqTBjDq
AL4guyyVLLV3RnrYRRa3uuhRj+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": "2.0"
  "data": {
    "msgId": "XXXXXX2025090300000001"
    "memCode": "XXXXXX"
    "cliList": ["CLI0000000"]
  }
}
```

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	String	Refer to section "Message based response code"	
3	timeStamp	String	Date time stamp	17 Nov 2025 14:11:58

Detail Record Structure (CSV) (Separator – “”)

#	Field Name	Description	Data Type	Size(In Byte)	Sample
1	tmCode	TM Code	String	5	XXXXX
2	cliCode	Client Code	String	10	CLI0000000
3	insType	Ins Type	String	6	FUTSTK
4	symbol	Symbol	String	12	UNITDSPR
5	expDate	Expiry Date	Date		20251125
6	strikePrice	Strike Price	Double	8	0
7	optType	Option Type	String	2	FF
8	Filler1	Filler	Double	8	
9	Filler2	Filler	Double	8	
10	Filler3	Filler	Double	8	
11	Filler4	Filler	Double	8	
12	Filler5	Filler	Double	8	
13	Filler6	Filler	Double	8	
14	openPosQty	Open Position Qty	Double	8	4
15	Filler7	Filler	Double	8	
16	Filler8	Filler	Double	8	
17	Filler9	Filler	Double	8	
18	Filler10	Filler	String	16	
19	Filler11	Filler	String	16	
20	Filler12	Filler	String	16	

Sample Failure Response

Wrong access token or expired access token

```
HTTP/1.1 401 UNAUTHORIZED
Content-Type: application/json
{
  "messages":{"code":"0101401"}
  "status":"error"
}
```

Error in encryption

```
HTTP/1.1 400 BAD_REQUEST
Content-Type: application/json
{
  "messages":{"code":"0101400"}
  "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"
  "message": ""
  "timeStamp": "17 Nov 2025 14:11:58"
  "data":
  "i1iw0PPNS0DJNSX8bswCpY65aWYSobTpBCR/UZAqacBiO8smQeHRa338+Ro7qGi8VOXSVzPC
EP04oXWHd/AjOozKTge2vO9WiOJdJY3VJVhdcwZL3Gj4tEbXi4vxft+SfJ9bRxyfh5kMHZXvc
lnC55mpkHca9fdgm8pKmT0SdnQsKMJ11GMUYxKQtDvdzQXyxWI4GY3f"
}
```

Response with Raw Data

```
{
  "status": "SUCCESS"
  "message": ""
  "timeStamp": "17 Nov 2025 14:11:58"
  "data": [
    {
      XXXXX,CLI0000000,FUTSTK,UNITDSPR,20251125,0,FF,,,,,,,,4,,,,,,,, }
    ]
}
```

POST /<version>/request/moi

This API will allow members to inquire for member open interest using API POST
/<version>/request/moi.

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 dCaxH9rAM3hVIMJzFg==
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>.	MjAwMTlwMTcxNjEyMjE0 TE6
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	2.0

#	Parameter Name	Data Type	Description	Sample Value
2	data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnnn> Member Code (Length: 4 or 5) • YYYYMMDD – Date format • nnnnnnnn – Running sequence no. starting from one i.e. For first request of the day it should be (0000001).	XXXXX201310140000001
3	data.memCode	String	Member code	XXXXX

Sample Request

Request Header:

```
POST /v2/request/moi HTTP/1.1
Host: uat.connect2nsccl.com
Authorization: Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
consumerKey: consKey
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
Content-Type: application/json
```

Request Body:

```
{
  "data":
  "i3fJhLKZHhGdanX8csAP4sfqaXse/PO2ek84FMMocd8hLMVgHuOQREft6QsruHisVrqTBjDq
AL4guyyVLLV3RnrYRRa3uuhRj+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": "2.0"
  "data": {
    "msgId": "XXXXX202509030000001"
    "memCode": "XXXXX"
  }
}
```

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	String	Refer to section “Message based response code”	
3	timeStamp	String	Date time stamp	17 Nov 2025 14:11:58

Detail Record Structure (CSV) (Separator – “,”)

#	Field Name	Description	Data Type	Size(In Byte)	Sample
1	tm/CPcode	TM / CP Code	String	12	XXXXX
2	symbol	Symbol	String	12	PPP
3	instType	Instrument Type	String	6	Stock Position
4	nseclrOpenInterest	NSE Clearing Open Interest	Double	8	1750
5	limit	Limit	Double	8	36989400.00
6	Percentage	Percentage	Double	8	0
7	Filler1	Filler	Double	8	
8	Filler2	Filler	Double	8	
9	Filler3	Filler	String	16	
10	Filler4	Filler	String	16	

Sample Failure Response

Wrong access token or expired access token

```
HTTP/1.1 401 UNAUTHORIZED
Content-Type: application/json
{
  "messages":{"code":"0101401"}
  "status":"error"
}
```

Error in encryption

```
HTTP/1.1 400 BAD_REQUEST
Content-Type: application/json
{
  "messages":{"code":"0101400"}
  "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"
  "message": ""
  "timeStamp": "17 Nov 2025 14:11:58"
  "data":
  "iliw0PPNS0DJNSX8bswCpY65aWYSobTpBCR/UZAqacBiO8smQeHRa338+Ro7qGi8VOXSVzPC
EP04oXWHd/AjOozKTge2vO9WiOJdJY3VJVhdcwZL3Gj4tEbXi4vxft+SfJ9bRxyfh5kMHZXvc
lnC55mpkHca9fdgm8pKmT0SdnQsKMJ11GMUYxKQtDvdzQXyxWI4GY3f"
}
```

Response with Raw Data

```
{
  "status": "SUCCESS"
  "message": ""
  "timeStamp": "17 Nov 2025 14:11:58"
  "data": [
    {
      XXXXX,PPP,Stock Position,1750,36989400.00,0,,, }
    ]
}
```

POST /<version>/request/mwpl

This API will allow members to inquire for market wide position limit using API POST /<version>/request/mwpl. This API will be applicable only for FO segment.

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 dCaxH9rAM3hVIMJzFg==
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>.	MjAwMTIwMTcxNjEyMjE1O TE6
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	2.0
2	data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnn> Member 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.memCode	String	Member code	XXXXX

Sample Request

Request Header:

```
POST /v2/request/mwpl HTTP/1.1
Host: uat.connect2nsccl.com
Authorization: Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
consumerKey: consKey
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
Content-Type: application/json
```

Request Body:

```
{
  "data":
  "i3fJhLKZHhGdanX8csAP4sfqaXse/PO2ek84FMMocd8hLMVgHuOQREft6QsruHisVrqTBjDq
AL4guyyVLLV3RnrYRRa3uuhRj+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": "2.0"
  "data": {
    "msgId": "XXXXXX202509030000001"
    "memCode": "XXXXXX"
  }
}
```

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	String	Refer to section “Message based response code”	
3	timeStamp	String	Date time stamp	17 Nov 2025 14:11:58

Detail Record Structure (CSV) (Separator – “,”)

#	Field Name	Description	Data Type	Size(In Byte)	Sample
1	symbol	Symbol	String	12	011NSETEST
2	ncIFUTEQOI	NSE CLEARING FUTEQ OI	Double	8	10350.00
3	limit	LIMIT	Double	8	126327288.00
4	ncIFUTEQOI %	NSE CLEARING FUTEQ OI%	Double	8	0.01
5	Filler1	Filler	Double	8	8
6	Filler2	Filler	Double	8	8
7	Filler3	Filler	String	16	16
8	Filler4	Filler	String	16	16

Sample Failure Response

Wrong access token or expired access token

```
HTTP/1.1 401 UNAUTHORIZED
Content-Type: application/json
{
  "messages":{"code":"0101401"}
  "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"
    "message": ""
    "timeStamp": "17 Nov 2025 14:11:58"
  "data":
    "i1iw0PPNS0DJNSX8bswCpY65aWYSobTpBCR/UZAqacBiO8smQeHRa338+Ro7qGi8VOXSVzPC
    EP04oXWHd/AjOozKTge2vO9WiOJdJY3VJVhdcwZL3Gj4tEbXi4vxft+SfJ9bRxyfh5kMHZXvc
    lnC55mpkHca9fdgm8pKmT0SdnQsKMJ11GMUYxKQtDvdzQXyxWI4GY3f"
  }

```

Response with Raw Data

```

{
  "status": "SUCCESS"
  "message": ""
  "timeStamp": "17 Nov 2025 14:11:58"
  "data": [
    {
      011NSETEST,10350.00,126327288.00,0.01,,,, }
    ]
  }

```

POST /<version>/request/cli-margins/generate-all

This API will allow members to generate all the client margin for the respective member.

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 dCaxH9rAM3hVIMJzFg==
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>.	MjAwMTIwMTcxNjEyMjE0 TE6
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	2.0
2	data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnnn>	XXXXX201310140000001

#	Parameter Name	Data Type	Description	Sample Value
			Member Code (Length: 4 or 5) - YYYYMMDD – Date format - nnnnnnnn – Running sequence no. starting from one i.e. For first request of the day, it should be (0000001). Note: The 'msgId' provided will be used as a response during identifying the record in Inquiry API. POST/<version>/request/cli-margins/inquiry	

Sample Request

Request Header:

```
POST /v2/request/cli-margins/generate-all HTTP/1.1
Host: uat.connect2nsccl.com
Authorization: Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
consumerKey: consKey
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
Content-Type: application/json
```

Request Body:

```
{
  "data":
  "i3fJhLKZHhGdanX8csAP4sfqaXse/PO2ek84FMmocd8hLMVgHuOQREft6QsruHisVrqTBjDq
AL4guyyVLLV3RNRyRRa3uuhRj+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": "2.0"
  "data": {
    "msgId": "XXXXXX202509030000001"
  }
}
```

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 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/cli-margins/inquiry

This API will allow members to download the generated file of all the client margin for the respective member.

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 dCaxH9rAM3hVIMJzFg==
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>.	MjAwMTIwMTcxNjEyMjE0TE6
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	2.0

#	Parameter Name	Data Type	Description	Sample Value
2	data.msgId	String	This message id should be match with message id provided in POST/<version>/request/cli-margins/generate-all	XXXXX201310140000001

Sample Request

Request Header:

```
POST /v2/request/cli-margins/inquiry HTTP/1.1
Host: uat.connect2nsccl.com
Authorization: Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
consumerKey: consKey
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
Content-Type: application/json
```

Request Body:

```
{
  "data":
  "i3fJhLKZHhGdanX8csAP4sfqaXse/PO2ek84FMMocd8hLMVgHuOQREft6QsruHisVrqTBjDq
AL4guyyVLLV3RnrYRRa3uuhRj+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": "2.0"
  "data": {
    "msgId": "XXXXX202509030000001"
  }
}
```

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

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"
}
```

POST /<version>/request/positions/generate-all

This API will allow members to generate all the positions for the respective member.

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 dCaxH9rAM3hVIMJzFg==
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>.	MjAwMTlwMTcxNjEyMjE0 TE6
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	2.0
2	data.msgId	String	Unique request number for each request <CODE><YYYYMMDD><nnnnnnn> Member 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

#	Parameter Name	Data Type	Description	Sample Value
			Note: The 'msgId' provided will be used as a response during identifying the record in Inquiry API. POST/<version>/request/positions /inquiry	

Sample Request

Request Header:

```
POST /v2/request/positions/generate-all HTTP/1.1
Host: uat.connect2nsccl.com
Authorization: Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
consumerKey: consKey
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
Content-Type: application/json
```

Request Body:

```
{
  "data":
  "i3fJhLKZHhGdanX8csAP4sfqaXse/PO2ek84FMMocd8hLMVgHuOQREft6QsruHisVrqTBjDq
AL4guyyVLLV3RNRrYRRa3uuhRj+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": "2.0"
  "data": {
    "msgId": "XXXXX202509030000001"
  }
}
```

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 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

Response Data Payload (JSON)				
Sr. No.	Parameter Name	Data Type	Description	Sample Value
3	data.code	String	Refer Section “Message based response code”.	01010000

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

POST /<version>/request/positions/inquiry

This API will allow members to download the generated file of all the positions for the respective member.

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 dCaxH9rAM3hVIMJzFg==
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>.	MjAwMTIwMTcxNjEyMjE0TE6
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	2.0
2	data.msgId	String	This message id should be match with message id provided in POST/<version>/request/positions/generate-all	XXXXX201310140000001

#	Parameter Name	Data Type	Description	Sample Value

Sample Request

Request Header:

```
POST /v2/request/positions/inquiry HTTP/1.1
Host: uat.connect2nsccl.com
Authorization: Basic MRZmwzdkje382jdw8ue93jdCaxH9rAM3hVlMJzFg==
consumerKey: consKey
nonce: MjAwMTIwMTcxNjEyMjE1OTE6ODk0MjY3
Content-Type: application/json
```

Request Body:

```
{
  "data":
  "i3fJhLKZHhGdanX8csAP4sfqaXse/PO2ek84FMMocd8hLMVgHuOQREft6QsruHisVrqTBjDq
AL4guyyVL3RNRrYRRa3uuhRj+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": "2.0"
  "data": {
    "msgId": "XXXXX202509030000001"
  }
}
```

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**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"
}

```

Appendix A - 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. N o.	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
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	TOKEN_INVALID – The provided token is not registered in our system	401

HTTP Response Codes			
Sr. No.	Reason	Meaning	HTTP Response Code
	authenticate the request		
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	memCode	0114
4	Input Data Parameter	cliList	0115

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

Sr. No.	Validation	Validation Type	Validation Code	Validation performed on Field
5	Minimum Required Length	Generic	0201	msgId
6	Maximum Required Length	Generic	0202	msgId
7	Mandatory field	Generic	0204	msgId, memCode, cliList
8	Data Format like Message Id	Generic	0206	msgId
12	System Error	Generic	0241	NA
13	Service Unavailable	Generic	0242	NA
14	Request Parsing Error: Invalid Request Structure	Generic	0243	NA
15	No Data found	Generic	0244	msgId
16	File Generation In Progress	Generic	0245	NA

Sample example for success or failure code

Example for Generic Error Code

Let's assume that msgId field holds value ABCD201340402132165, which turns out to be an error "Invalid Data Format". Error Code that will be generated is as shown below:

Field Identifier: 0102

Validation Code: 0206

code = combination of "Field Identifier" and "Validation Code" = 01020206

Example for Success code (Submitted to server successfully)

Let's assume that message for approval/rejection is successful, success code that will be generated is as shown below:

Field Identifier: 0101 (which is the identifier of the entire message)

Validation Code: 0000

code = combination of "Field Identifier" and "Validation Code" = 01010000

Example for HTTP error code

Let's assume that the invalid request scenario due to UNAUTHORIZED Access Request, error code that will be generated is as shown below:

Field Identifier: 0101 (which is the identifier of the entire message)

Validation Code: 401

code = combination of "Field Identifier" and "Validation Code" = 0101401

Note:

For HTTP error code the above code will not be valid for every response. It will be valid only if there is an error in request header other than this it will populate HTTP code only.

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