Documentation

Quick Navigation

Getting Started (15 minutes) - Start here if you're new
Authentication Reference - Detailed authentication info
Market Data - Quotes and instrument master
Trading Operations - Place, modify, cancel orders
Portfolio & Funds - Holdings, positions, limits, margin
Reports - Order History, orderbook, tradebook
Troubleshooting - Check here when stuck, common error codes
Complete API Reference - Downloadable reference and curl examples

---

Login with TOTP

Overview

The **Login with TOTP** authentication for Kotak Securities Trade API allows secure and automated user authentication by leveraging Time-based One-Time Passwords (TOTP). This is a three-step process:

1. **Step 1**: Register for TOTP via NEO (one time process)
2. **Step 2:** Validate the user's credentials and TOTP to receive a session token and view token.

3. **Step 3:** Use the session information and MPIN to complete the login and receive a trade token.

Below is comprehensive, user-friendly documentation for both steps.

Step 1: Register for TOTP

TOTP stands for *Time-based One-Time Password*. Unlike SMS OTP, which is sent to your phone, a TOTP is generated every **30 seconds** in an authenticator app (e.g., Google Authenticator, Microsoft Authenticator).

1. On **API Dashboard**, click **TOTP Registration**.
2. Verify with your **mobile number, OTP, and client code**.
3. **Scan QR code** with Google/Microsoft Authenticator (application can be downloaded from playstore/appstore).
4. Enter the generated **TOTP**.
5. Confirm **"TOTP successfully registered"**

Step 2: Login with TOTP

API Access Token is issued from the NEO App. Go to **Invest → Trade API**, create an app under **Your Applications**, and copy the token shown. This token is your **access token**, and must be passed in the `Authorization` header of the Login APIs.

1. Introduction

Authenticate your account using mobile number, UCC, and TOTP. On success, you receive a view token (`token`), along with session identifiers to be used in the next step.

2. API Endpoint

```
POST https://mis.kotaksecurities.com/login/1.0/tradeApiLogin
```

3. Headers

Name	Type	Description
Authorization	string	Token provided in your NEO API dashboard - use plain token
neo-fin-key	string	static value: neotradeapi
Content-Type	string	Always `application/json`

4. Request Body

Example curl Request

```
curl -X POST "https://mis.kotaksecurities.com/login/1.0/tradeApiLogin" \
  -H "Authorization: <access_token>" \
  -H "neo-fin-key: neotradeapi" \
  -H "Content-Type: application/json" \
  -d '{
        "mobileNumber": "<+91XXXXXXXXXX>",
        "ucc": "<client_code>",
        "totp": "<6_digit_totp>"
      }'
```

Name	Type	Description
mobileNumber	string	User's registered mobile number (with ISD)
ucc	string	Unique Client Code (Client ID)
totp	string	TOTP generated using authenticator app

5. Response

Success Example:

```
{
    "data": {
        "token": "eyJhbGciOiJ.....",
        "sid": "*******-****-****-****-**********",
        "rid": "*******-****-****-****-**********",
        "kType": "View",
        "status": "success",
        "greetingName": "*****",
        // other fields as described in common response table
    }
}
```

Error Example:

```
{
    "status": "error",
    "message": "Invalid credentials or TOTP.",
    "errorCode": "401"
}
```

Step 3: Validate MPIN (Trading Token Generation)

1. Introduction

Complete authentication by providing your 6-digit MPIN. You'll receive a trading token and session data required for authorized trading actions.

2. API Endpoint

```
POST https://mis.kotaksecurities.com/login/1.0/tradeApiValidate
```

3. Headers

Name	Type	Description
Authorization	string	Token provided in your NEO API dashboard - use plain token
neo-fin-key	string	static value: neotradeapi
Content-Type	string	Always `application/json`
sid	string	view sid received from Step 1 (`sid` in response)

Name	Type	Description
Auth	string	View token received from Step 1 (`token` in response)

4. Request Body

Example Request:

```
curl -X POST "https://mis.kotaksecurities.com/login/1.0/tradeApiValidate" \
  -H "Authorization: <access_token>" \
  -H "neo-fin-key: neotradeapi" \
  -H "sid: <viewSid_from_previous_step>" \
  -H "Auth: <viewToken_from_previous_step>" \
  -d '{
        "mpin": "<mpin>"
      }'
```

Name	Type	Description
mpin	string	User's 6-digit MPIN

5. Response

📌 **Response** gives you:

- `baseUrl` (use it for **all** post-login APIs)
- token = **session token → used in headers as "Auth" for all successive APIs**
- `Sid` = **session sid**

Success Example:

```
{
    "data": {
        "token": "eyJhbGciOiJ.....",
```

```
        "sid": "******-****-****-****-**********",
        "rid": "******-****-****-****-**********",
        "baseUrl": "https://cis.kotaksecurities.com",
        "kType": "Trade",
        "status": "success",
        "greetingName": "*****",
        ...
    }
}
```

Error Example:

```
{
    "status": "error",
    "message": "Invalid MPIN.",
    "errorCode": "401"
}
```

Common Response Fields

Both Step 1 and Step 2 return a `data` object with many similarities.

Name	Type	Description & Possible Values
token	string	JWT token. "View" token for Step 1 (`kType="View"`), "Trade" token for Step 2 (`kType="Trade"`).
sid	string	Session ID
rid	string	Request ID for tracking

Name	Type	Description & Possible Values
baseUrl	**string**	**returns base url which is to be used post login**
hsServerId	string	Server ID. Usually empty.
isUserPwdExpired	boolean	Indicates if user's password has expired.
ucc	string	Unique Client Code (masked).
greetingName	string	Greeting name for user (masked)
isTrialAccount	boolean	Indicates if the account is a trial type
dataCenter	string	Data center code e.g., E22
searchAPIKey	string	Search API Key. Usually empty.
derivativesRiskDisclosure	string	SEBI Derivatives Risk Disclosure;

Name	Type	Description & Possible Values
		usually lengthy disclaimer text
mfAccess	integer	Mutual Fund access: `1` means active
dataCenterMap	object	Mapping data for centers (can be null)
dormancyStatus	string	Account dormancy status, e.g., `A`
asbaStatus	string	ASBA status (usually empty)
clientType	string	Client type, e.g., `RI`
isNRI	boolean	If client is NRI (`true` or `false`)
kId	string	PAN or similar identification (masked)
kType	string	"View" for Step 1, "Trade" for Step 2

Name	Type	Description & Possible Values
status	string	"success" for successful response, else error
incRange	integer	Income range (numeric, optional)
incUpdFlag	string	Income Update Flag (optional, usually blank)
clientGroup	string	Client Group (optional, usually blank)
kraStatus	string	KRA verification status (optional, blank)
rcFlag	integer	Internal flag (numeric, optional)

Error Codes

Code	Description
401	Unauthorized / Invalid credentials / TOTP / MPIN
422	Invalid request parameters
500	Internal Server Error
403	Forbidden

Notes:

- All tokens and IDs in sample responses are masked for illustration.
- Use your actual values from APIs in production.
- Handle all sensitive information (like tokens and MPINs) securely.
- The Step 2 response `token` (where `kType` is `Trade`) is to be used as the authorization credential for trading APIs.

If you need more endpoints documented or wish for tailored code samples, just let me know!

Instruments

Scrip Master API

1. Introduction

The **Scrip Master API** provides direct download links to the latest scrip master (instrument master) CSV files for all supported exchange segments. These files

include instrument tokens, symbols, and other key data required for trading activities and symbol lookups.

2. API Endpoint

`GET <Base URL>`/script-details/1.0/masterscrip/file-paths
Replace `<Base URL>` with the relevant Kotak environment base URL provided in response from /tradeApiValidate api.

3. Headers

Name	Type	Description
Authorization	string	Token provided in your NEO API dashboard - use plain token

4. Request

Example Request

```
curl --location '<Base URL>/script-details/1.0/masterscrip/file-paths' \
--header 'Authorization: xxxxx-your-neo-token-xxxx'
```

No request parameters or request body required.

5. Response

Example Success Response


```
{
    "data": {
        "filesPaths": [
            "https://lapi.kotaksecurities.com/wso2-scripmaster/v1/prod/yyyy-mm-
dd/transformed/cde_fo.csv",
            "https://lapi.kotaksecurities.com/wso2-scripmaster/v1/prod/yyyy-mm-
dd/transformed/mcx_fo.csv",
            "https://lapi.kotaksecurities.com/wso2-scripmaster/v1/prod/yyyy-mm-
dd/transformed/nse_fo.csv",
            "https://lapi.kotaksecurities.com/wso2-scripmaster/v1/prod/yyyy-mm-
dd/transformed/bse_fo.csv",
            "https://lapi.kotaksecurities.com/wso2-scripmaster/v1/prod/yyyy-mm-
dd/transformed/nse_com.csv",
            "https://lapi.kotaksecurities.com/wso2-scripmaster/v1/prod/yyyy-mm-
dd/transformed-v1/bse_cm-v1.csv",
            "https://lapi.kotaksecurities.com/wso2-scripmaster/v1/prod/yyyy-mm-
dd/transformed-v1/nse_cm-v1.csv"
        ],
        "baseFolder": "https://lapi.kotaksecurities.com/wso2-scripmaster/v1/prod"
    }
}
```


200 Response Fields

Name	Type	Description
filesPaths	array	Array of URLs – each is a download link to a CSV for one segment.
baseFolder	string	The root URL for retrieving CSV files (read-only).

Error Codes

Code	Description
401	Unauthorized: Invalid or missing API token
403	Forbidden: Not enough privileges
429	Too many requests: API rate limited
500	Server error: Unexpected system error

6. Response CSV Files column mapping to Orders & websocket APIs

Column name	mapping	description
pSymbol	Webscocket and Quotes API	this is passed along with pExchSeg with a separator in quotes and websocket APIs. example: nse_cm

Column name	mapping	description
pExchSeg	Orders API as **‘es’ field;**	
Webscocket and Quotes API as part of query url	`Expected values are nse_cm, bse_cm, nse_fo, bse_fo, cde_fo`	
passed as a string in case of orders API		
pTrdSymbol	Orders API as **‘ts’ field**	This refers to trading instrument in a way orders API interpret; passed as a string
lLotSize	Orders API as **‘qt’ field**	Quantity sent in place order should be in multiple of the lot size

Column name	mapping	description
lExpiryDate	-	In case of contracts like in F&O, this represents the expiry date.

7. Example Workflow

1. Make a **GET** request with your Authorization token to the endpoint above.
2. On success, parse `filesPaths` to get the CSV URLs for downloading scrip/instrument master files for each segment (e.g., NSE F&O, BSE CM).
3. Download and use the CSV files to map instrument tokens to trading symbols and other details in your trading application.

Notes

- Always use the latest download links; files update frequently, usually daily.
- Each file relates to a specific exchange segment (e.g., `nse_fo`, `bse_cm`).
- Download and cache these CSVs as needed for fast local symbol lookups.
- Always secure your API token and never share confidential links or files publicly.

Order

This section covers all order-related APIs for trading operations.

Available Order Operations:

- Place Order - Create new orders
- Modify Order - Update existing orders
- Cancel Order - Cancel pending orders

Place Order

1. Introduction

The **Place Order API** allows you to place buy or sell orders across all supported exchange segments and order types. It supports product types like NRML, CNC, MIS, CO, and BO, and includes specialized fields for Bracket Orders (BO) and Cover Orders (CO).

2. API Endpoint

`POST <Base URL>/quick/order/rule/ms/place`

Replace `<Base URL>` with the relevant Kotak environment base URL provided in response from /tradeApiValidate api.

3. Headers

Name	Type	Description
accept	string	Should always be `application/json`
Sid	string	session sid generated on login
Auth	string	session token generated on login
neo-fin-key	string	static value: neotradeapi
Content-Type	string	Always `application/x-www-form-urlencoded`

4. Request Body

The request body is sent as a single field named `jData`, which is a stringified JSON object and must be URL-encoded.

Example Request

```
curl -X POST "<baseUrl>/quick/order/rule/ms/place" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  --data-urlencode 'jData={
```

```
    "am": "NO",
    "dq": "0",
    "es": "nse_cm",
    "mp": "0",
    "pc": "CNC",
    "pf": "N",
    "pr": "0",
    "pt": "MKT",
    "qt": "1",
    "rt": "DAY",
    "tp": "0",
    "ts": "ITBEES-EQ",
    "tt": "B"
  }'
```

Example Request Body (`jData`)

```
{
  "am": "NO",
  "dq": "0",
  "es": "nse_cm",
  "mp": "0",
  "pc": "MIS",
  "pf": "N",
  "pr": "0",
  "pt": "L",
  "qt": "1",
  "rt": "DAY",
  "tp": "0",
  "ts": "******-**",
  "tt": "B",
  "sot": "",    *// Only for BO*
  "slt": "",    *// Only for BO*
  "slv": "",    *// Only for BO*
  "sov": "",    *// Only for BO*
  "tlt": "",    *// Only for BO*
  "tsv": ""     *// Only for BO*
}
```

Request Body Fields

Name	Type	Description	Allowed / Example Values
am	string	After Market Order flag.	"NO" (normal),

Name	Type	Description	Allowed / Example Values
			"YES" (AMO)
dq	string	Disclosed quantity.	"0" or a partial quantity
es	string	Exchange segment code.	"nse_cm", "bse_cm", "nse_fo", "bse_fo", "cde_fo", “mcx_fo”
mp	string	Market protection value (used in some market orders).	"0" or numerical value
pc	string	Product code.	"NRML", "CNC", "MIS", "CO", "BO", “MTF”
pf	string	Portfolio flag.	"N"

Name	Type	Description	Allowed / Example Values
pr	string	Price for limit order, "0" for market order.	e.g. "0", "450.5"
pt	string	Order type.	"L" (Limit), "MKT" (Market), "SL" (Stoploss), "SL-M" (SL-Market)
qt	string	Order quantity.	e.g. "1", "100", etc.
rt	string	Validity or order duration.	"DAY", "IOC"
tp	string	Trigger price **(used for SL/SL-M/CO).**	"0" or actual trigger price.
ts	string	Trading symbol (from scrip master file).	e.g., "ITBEES-EQ"

Name	Type	Description	Allowed / Example Values
tt	string	Transaction type.	"B" (Buy), "S" (Sell)
sot	string	Square off type. Only for BO.	"Absolute", "Ticks"
slt	string	Stop loss type. Only for BO.	"Absolute", "Ticks"
slv	string	Stop loss value. Only for BO.	numeric (e.g. "5.00")
sov	string	Square off value. Only for BO.	numeric (e.g. "10.00")
tlt	string	Trailing stop loss indicator. Only for BO.	"Y" (yes), "N" (no)
tsv	string	Trailing stop loss value. Only for BO and if tlt is Y.	numeric (e.g. "2.00"), else "0"

5. Response

Example Success Response


```
{
  "nOrdNo": "250720000007242",
  "stat": "Ok",
  "stCode": 200
}
```


Success (200) Response Fields

Name	Type	Description
nOrdNo	string	Unique order number assigned to your request
stat	string	Status message, "Ok" if successful
stCode	int	HTTP status code, 200 for success

Example Error Response


```
{
  "stat": "Not_Ok",
  "emsg": "Insufficient balance.",
  "stCode": 1004
}
```

Error Response Fields

Name	Type	Description
stat	string	Status message, "Not_Ok" for errors
emsg	string	Error message in plain English
stCode	int	Error code (see below)

Tips & Notes

- Make sure all header tokens and session information are obtained using prior authentication flows.
- Use the latest Scrip Master file to get correct trading symbols and instrument details.
- Use appropriate BO/CO fields **only** when placing Bracket or Cover orders.
- Handle all non-200 status codes in your integration for robust error management.

Modify order

1. Introduction

The **Modify Order API** allows you to modify an already placed order's parameters—such as quantity, price, validity, product type, and more—across supported segments and order types before it is executed or fully filled.

2. API Endpoint

`POST <Base URL>/quick/order/vr/modify`

Replace `<Base URL>` with the relevant Kotak environment base URL provided in response from /tradeApiValidate api.

3. Headers

Name	Type	Description
accept	string	Should always be `application/json`
Sid	string	session sid generated on login
Auth	string	session token generated on login
neo-fin-key	string	static value: neotradeapi
Content-Type	string	Always `application/x-www-form-urlencoded`

4. Request Body

The request body uses a single field named `jData`, which is a URL-encoded JSON object.

Example Request

```
curl -X POST "<baseUrl>/quick/order/vr/modify" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  --data-urlencode 'jData={
    "am": "NO",
    "dq": "0",
    "es": "nse_cm",
    "mp": "0",
    "pc": "NRML",
    "pf": "N",
    "pr": "0",
    "pt": "MKT",
    "qt": "1",
    "rt": "DAY",
    "tp": "0",
    "ts": "TATAPOWER-EQ",
    "tt": "B",
    "no": "<orderNo>"
  }'
```

Example Request Body (`jData`)

```
{
  "tk": "*****",
  "mp": "0",
  "pc": "NRML",
  "dd": "NA",
  "dq": "0",
  "vd": "DAY",
  "ts": "******-**",
  "tt": "B",
  "pr": "3001",
  "tp": "0",
  "qt": "10",
  "no": "***************",
  "es": "nse_cm",
  "pt": "L"
}
```

Request Body Fields

Name	Type	Description	Allowed / Example Values
tk	string	Token (Instrument token from scrip master, as **pSymbol** column)	"11536", or as from the scrip master pSymbol column
fq	string	Filled Quantity (optional)	"10", "0"
mp	string	Market protection value	"0"
pc	string	Product code	"NRML", "CNC", "MIS", "CO", "BO"
dd	string	Date/Days (trailing validity, if applicable)	"NA" or as required
dq	string	Disclosed quantity	"0" or a partial quantity
vd	string	Validity (order duration)	"DAY", "IOC"
ts	string	Trading Symbol (from scrip master)	"TCS-EQ", etc.

Name	Type	Description	Allowed / Example Values
tt	string	Transaction type	"B" (Buy), "S" (Sell)
pr	string	Price	e.g., "3001"
tp	string	Trigger price (for SL, SL-M)	"0" or actual trigger price
qt	string	Quantity	e.g., "10"
no	string	Nest Order Number (system order id for the original order)	e.g., "220106000000185"
es	string	Exchange Segment	"nse_cm", "bse_cm", "nse_fo", "bse_fo", "cde_fo"
pt	string	Order Type	"L" (Limit), "MKT" (Market), "SL" (Stoploss), "SL-M" (SL-Market)

5. Response

Example Success Response

```
{
```

```
  "nOrdNo": "250720000007242",
  "stat": "Ok",
  "stCode": 200
}
```


200 Response Fields

Name	Type	Description
nOrdNo	string	New Order Number created or modified
stat	string	"Ok" if modification successful
stCode	int	HTTP status code, 200 for success

Example Error Response


```
{
  "stat": "Not_Ok",
  "emsg": "Order cannot be modified as it is already executed.",
  "stCode": 1006
}
```

Error Response Fields

Name	Type	Description
stat	string	"Not_Ok" for errors
emsg	string	Error message in English
stCode	int	Error code (see below)

Notes

- Only orders that are **not** yet executed or completed can be modified.
- Always use valid instrument tokens, symbols, and original order numbers.
- Headers and authorization must be handled securely as in the Place Order API.
- Use the latest scrip master data for token and symbol lookups.
- Use appropriate error handling for non-200 and failure responses.

Cancel Order

Cancel Order APIs (Regular, Bracket Order, Cover Order)

1. Introduction

Kotak Securities provides APIs for cancelling three types of open orders:

- **Regular Orders** (Equity, F&O, etc.)

- **Bracket Orders (BO)**
- **Cover Orders (CO)**

Only the endpoint differs for BO/CO cancellations. All headers, request format, and responses remain consistent.

Use these APIs to cancel pending orders before execution.

2. API Endpoints

Order Type	Endpoint (after)
Regular Order	`/quick/order/cancel`
Bracket Order (BO)	`/quick/order/bo/exit`
Cover Order (CO)	`/quick/order/co/exit`

Replace `<Base URL>` with the relevant Kotak environment base URL provided in response from /tradeApiValidate api.

3. Headers

Name	Type	Description
accept	string	Should always be `application/json`
Sid	string	session sid generated on login

Name	Type	Description
Auth	string	session token generated on login
neo-fin-key	string	static value: neotradeapi
Content-Type	string	Always `application/x-www-form-urlencoded`

4. Request Body

The request body is a single URL-encoded field named `jData`, containing a JSON object.

Example Request

```
# Cancel
curl -X POST "<baseUrl>/quick/order/cancel" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  --data-urlencode 'jData={"on":"<orderNo>","am":"NO"}'

# Exit Cover
curl -X POST "<baseUrl>/quick/order/co/exit" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  --data-urlencode 'jData={"on":"<orderNo>","am":"NO"}'

# Exit Bracket
curl -X POST "<baseUrl>/quick/order/bo/exit" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi" \
```

```
  -H "Content-Type: application/x-www-form-urlencoded" \
  --data-urlencode 'jData={"on":"<orderNo>","am":"NO"}'
```

Example Request Body (`jData`)

```
{
  "am": "NO",
  "on": "********************",
  "ts": "*******-**"
}
```

Request Body Fields

Name	Type	Description	Required	Example
am	string	AMO flag ("YES" for AMO orders; omit/"NO" for others)	Optional	"YES", "NO"
on	string	Nest order number (unique order id)	Required	"2105199703091997"
ts	string	Trading symbol (mandatory for AMO orders)	Optional	"TCS-EQ"

5. Response

Example Success Response

```
{
  "nOrdNo": "2105199703091997",
  "stat": "Ok",
  "stCode": 200
}
```

Success (200) Response Fields

Name	Type	Description
nOrdNo	string	Nest order number of the cancelled order
stat	string	Status, "Ok" if cancellation successful
stCode	int	HTTP status code, 200 for success

Example Error Response

```
{
  "stat": "Not_Ok",
  "emsg": "Order already cancelled or not found.",
  "stCode": 1006
}
```

Error Response Fields

Name	Type	Description
stat	string	"Not_Ok" for errors
emsg	string	Error message in English
stCode	int	Error code (see below)

6. Usage Notes

- **To cancel a regular, bracket, or cover order**, simply use the relevant endpoint with the same request fields and headers.
- For **AMO cancellation**, `"am": "YES"` and `"ts"` (trading symbol) are mandatory.
- Orders already fully executed or cancelled cannot be cancelled again.
- Use the exact `on` (Nest order number) as returned in the order placement or status queries.
- Always check for `"stat":"Ok"` in the response for a successful cancellation.

Order Report APIs

Covers: Order Book, Order History, Trade Book APIs

1. Introduction

Kotak Securities offers APIs to fetch your:

- **Order Book** (all open, completed, and rejected orders)
- **Order History** (all order status changes/updates for a particular order)
- **Trade Book** (details of completed trades for the trading day)

These APIs allow you to fetch and monitor all relevant trading activity programmatically.

2. API Endpoints

API	Endpoint (after)	Method
Order Book	`/quick/user/orders`	GET
Order History	`/quick/order/history`	POST
Trade Book	`/quick/user/trades`	GET

Replace `<Base URL>` with the relevant Kotak environment base URL provided in response from /tradeApiValidate api.

3. Headers

Applicable to all three APIs:

Name	Type	Description
accept	string	Should always be `application/json`

Name	Type	Description
Sid	string	session sid generated on login
Auth	string	session token generated on login
neo-fin-key	string	static value: neotradeapi
Content-Type	string	Always `application/x-www-form-urlencoded`

4. Request

4.1. GET Order Book

Example Request:

```
curl -X GET "<baseUrl>/quick/user/orders" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi"
```

No request body or parameters required.

4.2. POST Order History

Request Body:
Send `jData` as url-encoded JSON.
Example:

```
curl -X GET "<baseUrl>/quick/user/order/history" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi"
--data-urlencode 'jData={"nOrdNo":"250720000007242"}'
```

Field	Type	Description	Required
nOrdNo	string	Nest order number for which history is required	Yes

4.3. GET Trade Book

Example Request:

```
# Trade Book
curl -X GET "<baseUrl>/quick/user/trades" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi"No request body or parameters required.
```

5. Response

5.1. GET Order Book Response

Success Example (truncated):

```
{
  "stat": "Ok",
  "data": [
    {
      "nOrdNo": "250720000007242",
      "ordSt": "rejected",
      "trdSym": "ITBEES-EQ",
      "qty": 1,
      "prc": "0.00",
      "avgPrc": "0.00",
```

```
      "trnsTp": "B",
      "prcTp": "L",
      "vldt": "DAY",
      ...
    },
    {
      "nOrdNo": "250720000007588",
      "ordSt": "after market order req received",
      "trdSym": "ITBEES-EQ",
      "qty": 1,
      "prc": "0.00",
      "avgPrc": "0.00",
      ...
    }
  ],
  "stCode": 200
}
```


Important Fields (Truncated and Simplified)

Field	Type	Description
nOrdNo	string	Nest order number
ordSt / stat	string	Order status ("open", "rejected", etc.)
trdSym	string	Trading symbol e.g., "ITBEES-EQ"
qty	int	Quantity
prc	string	Placed price
avgPrc	string	Average traded price
trnsTp	string	Transaction type - "B"=Buy, "S"=Sell

Field	Type	Description
prcTp	string	Order type: "L"=Limit, "MKT"=Market, etc.
vldt	string	Validity (DAY/IOC)
rejRsn	string	Rejection reason if any
exSeg	string	Exchange segment e.g., "nse_cm"
ordGenTp	string	"AMO" for after-market orders, else blank
ordDtTm	string	Order date/time
stat	string	Overall status at top level: "Ok" for success

The order book contains all available fields per order as per API glossary.

5.2. POST Order History Response

Success Example (truncated):


```
{
  "stat": "Ok",
  "stCode": 200,
  "data": [
    {
      "nOrdNo": "250720000007242",
      "flDtTm": "20-Jul-2025 20:21:42",
      "ordSt": "rejected",
      "rejRsn": "ADAPTER is down",
      "qty": 1,
      "prc": "0.00",
```

```
      "avgPrc": "0.00",
      "prod": "MIS",
      "trnsTp": "B",
      "prcTp": "L",
      ...
    },
    {
      "nOrdNo": "250720000007242",
      "flDtTm": "20-Jul-2025 20:21:42",
      "ordSt": "open pending",
      ...
    }
  ]
}
```

Field	Type	Description
nOrdNo	string	Nest order number
ordSt	string	Status at this stage
flDtTm	string	Date/time for the update
qty	int	Order quantity at this status
prc	string	Order price at this status
avgPrc	string	Average price at this stage
prod	string	Product type ("MIS", "CNC")
trnsTp	string	Transaction type ("B"=Buy, "S"=Sell)
prcTp	string	Order type ("L", "MKT", etc.)

Field	Type	Description
rejRsn	string	Rejection reason if applicable

Full list of all response fields is available in Order Book API Glossary, as most fields are common.

5.3. GET Trade Book Response

Success Example (truncated):


```
{
  "stat": "Ok",
  "stCode": 200,
  "data": [
    {
      "nOrdNo": "221007000000354",
      "trdSym": "TCS-EQ",
      "qty": 11,
      "avgPrc": "3194.00",
      "fldQty": 11,
      "flDt": "07-Oct-2022",
      "exOrdId": "1100000000047870",
      "exTm": "07-Oct-2022 13:04:14",
      "prcTp": "L",
      "prod": "CNC",
      "ordDur": "DAY",
      "trnsTp": "B",
      "usrId": "PRABHAT",
      ...
    }
  ]
}
```


Field	Type	Description
nOrdNo	string	Nest order number
trdSym	string	Trading symbol

Field	Type	Description
qty	int	Trade quantity
avgPrc	string	Average execution price
fldQty	int	Filled quantity
flDt	string	Trade date
exOrdId	string	Exchange order ID
exTm	string	Trade execution datetime
prcTp	string	Order type (L/MKT/SL/etc.)
prod	string	Product code ("CNC","MIS", etc.)
ordDur	string	Order validity (DAY/IOC)
trnsTp	string	Transaction type (B/S)
usrId	string	User/client ID

Additional fields are available and can be referenced from the API glossary.

Common Response Fields

Field	Type	Description
stat	string	"Ok" for success, "Not_Ok" for errors
stCode	int	HTTP status code (200 = success, else error)
data	array	List of order/trade details objects
emsg	string	Present only for errors: error message

6. Usage Notes

- Use correct headers with valid session and auth tokens.
- For **Order Book** and **Trade Book**: Access all your recent orders/trades for the day.
- For **Order History**: Always provide the correct `nOrdNo` (order number) to fetch its full lifecycle/status changes.
- Handle `"stat": "Not_Ok"` and use `"emsg"` for debugging API issues.
- Field meanings and data formats remain consistent across all order-related APIs.
- Reference scrip master for symbol/segment mapping as needed.

Positions

1. Introduction

The **Positions API** provides a consolidated view of your open and closed positions for the current trading day across all supported segments. This allows you to track real-time exposures, quantities bought/sold, and settlement data.

2. API Endpoint

`GET <Base URL>/quick/user/positions`

Replace `<Base URL>` with the relevant Kotak environment base URL provided in response from /tradeApiValidate api.

3. Headers

Name	Type	Description
accept	string	Should always be `application/json`
Sid	string	session sid generated on login
Auth	string	session token generated on login

Name	Type	Description
neo-fin-key	string	static value: neotradeapi
Content-Type	string	Always `application/x-www-form-urlencoded`

4. Request

Example Request:

```
# Position Book
curl -X GET "<baseUrl>/quick/user/positions" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi"
```

No request body or query params required.

5. Response

Example Success Response (truncated, user-friendly)


```
{
  "stat": "Ok",
  "stCode": 200,
  "data": [
    {
      "actId": "******",
      "prod": "CNC",
      "exSeg": "nse_cm",
      "trdSym": "AXISBANK-EQ",
      "sym": "AXISBANK",
      "qty": "9",
      "buyAmt": "5862.90",
      "sellAmt": "0.00",
      "flBuyQty": "9",
      "flSellQty": "0",
```

```
      "brdLtQty": 1,
      "posFlg": "true",
      "sqrFlg": "Y",
      "lotSz": "1",
      "stkPrc": "0.00",
      "hsUpTm": "2022/06/21 15:11:02"
    },
    {
      "actId": "******",
      "prod": "CNC",
      "exSeg": "nse_cm",
      "trdSym": "ITC-EQ",
      "sym": "ITC",
      "qty": "15",
      "buyAmt": "4021.50",
      "sellAmt": "0.00",
      "flBuyQty": "15",
      "flSellQty": "0",
      "brdLtQty": 1,
      "posFlg": "true",
      "sqrFlg": "Y",
      "lotSz": "1",
      "stkPrc": "0.00",
      "hsUpTm": "2022/06/21 15:11:02"
    }
  ]
}
```


200 Response Fields

Field	Type	Description
stat	string	Overall status ("Ok" for success)
stCode	int	Status code (200 = success)
data	array	Array of position objects (see below)

Position Object (Key Fields):

Field	Type	Description
actId	string	Account ID
prod	string	Product code (e.g., CNC, MIS, NRML)
exSeg	string	Exchange segment (e.g., nse_cm, bse_cm)
trdSym	string	Trading symbol (e.g., AXISBANK-EQ)
sym	string	Symbol name (e.g., AXISBANK)
qty	string	Net position quantity
buyAmt	string	Total buy amount
sellAmt	string	Total sell amount
flBuyQty	string	Filled buy quantity
flSellQty	string	Filled sell quantity
brdLtQty	int	Board lot quantity
posFlg	string	Position flag ("true" if open position)

Field	Type	Description
sqrFlg	string	Square-off flag ("Y" = allowed)
lotSz	string	Lot size
stkPrc	string	Strike price (for derivatives)
hsUpTm	string	Last updated time

Other available fields:
cfBuyAmt, cfSellAmt, cfBuyQty, cfSellQty, multiplier, precision, expDt, genNum, genDen, prcNum, prcDen, optTp, type

Note: cf as a prefix refers for carry forward

Example Error Response


```
{
  "stat": "Not_Ok",
  "emsg": "Invalid session",
  "stCode": 1003
}
```


Field	Type	Description
stat	string	"Not_Ok" for errors
emsg	string	Error message in English

Field	Type	Description
stCode	int	Error code (see below)

6. Notes

- Only positions with actual trades for the day will be listed.
- Use the latest session and auth tokens.
- Quantities and amounts are strings for precision; convert as needed.
- Refer to the scrip master for segment, instrument, and symbol details.

Holdings

1. Introduction

The **Holdings API** provides a detailed summary of the securities (stocks, ETFs, etc.) held in your account, including current market value, average price, quantity, and sellable quantity. Use this API to display or manage the current equity portfolio.

2. API Endpoint

`GET <Base URL>/portfolio/v1/holdings`

Replace `<Base URL>` with the relevant Kotak environment base URL provided in response from /tradeApiValidate api.

3. Headers

Name	Type	Description
accept	string	Should always be `application/json`
Sid	string	session sid generated on login
Auth	string	session token generated on login
neo-fin-key	string	static value: neotradeapi
Content-Type	string	Always `application/x-www-form-urlencoded`

4. Request

Example Request:

```
# Portfolio Holdings (new response shape)
curl -X GET "<baseUrl>/portfolio/v1/holdings" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi"
```

No request body or parameters required.

5. Response

Example Success Response


```
{
    "data": [
        {
            "instrumentType": "Equity",
            "sector": "Others",
            "instrumentToken": 18974,
            "commonScripCode": "NIPPON_ETF_NIFTYIT",
            "instrumentName": "Nippon India ETF Nifty IT",
            "quantity": 45,
            "averagePrice": 38.7162,
            "holdingCost": 1742.2293,
            "closingPrice": 39.36,
            "mktValue": 1771.2,
            "scripId": "",
            "isAlternateScrip": "",
            "unrealisedGainLoss": 28.9707,
            "sqGainLoss": -5.6,
            "delGainLoss": -7.074,
            "subTotal": 0,
            "prevDayLtp": 0,
            "subType": "ETF",
            "instrumentStatus": "ACTV",
            "marketLot": 1,
            "expiryDate": "",
            "optType": "",
            "strikePrice": "0.00",
            "symbol": "ITBEES",
            "displaySymbol": "ITBEES",
            "exchangeSegment": "nse_cm",
            "series": "EQ",
            "exchangeIdentifier": "19084",
            "sellableQuantity": 45,
            "securityType": "ETF",
            "securitySubType": "ETF",
            "logoUrl": "https://www.kotaksecurities.com/stockit/stock_logos/74813.png",
            "cmotCode": "74813"
        },
    ]
}
```

200 Response Fields

Field	Type	Description
instrumentType	string	The type of instrument (e.g., Equity, Debt, ETF, Derivative).
sector	string	Industry/sector classification of the company/security.
instrumentToken	integer	Unique token ID for the instrument (used in trading/order APIs).
commonScripCode	string	Common identifier for the instrument across systems.
instrumentName	string	Full name of the instrument/security.
quantity	integer	Total number of units/shares held.
averagePrice	float	Average acquisition price per share/unit.

Field	Type	Description
holdingCost	float	Total cost of acquisition for this holding.
closingPrice	float	Previous closing price from the exchange.
mktValue	float	Current market value of the holding (closingPrice × quantity).
scripId	string	Internal scrip identifier (may be blank for some securities).
isAlternateScrip	string	Flag indicating if this is an alternate scrip (blank if not applicable).
unrealisedGainLoss	float	Unrealized profit/loss on the holding based on current market value.
sqGainLoss	float	Square-off gain/loss (intraday MTM component, if applicable).

Field	Type	Description
delGainLoss	float	Delivery gain/loss (applicable for deliveries).
subTotal	float	Additional subtotal field (value = 0 if unused).
prevDayLtp	float	Previous day's Last Traded Price (LTP).
subType	string	Subtype of instrument (e.g., ETF, EQUITY).
instrumentStatus	string	Status of the instrument (e.g., ACTV = Active).
marketLot	integer	Market lot size (minimum tradable unit).
expiryDate	string	Expiry date (applicable for derivatives, else blank).
optType	string	Option type (CE/PE for derivatives, else blank).

Field	Type	Description
strikePrice	string	Option strike price (string format, "0.00" for non-derivative instruments).
symbol	string	Symbol code of the instrument (e.g., IDEA).
displaySymbol	string	Symbol displayed in UI (same as symbol in most cases, e.g., ITBEES).
exchangeSegment	string	Exchange segment (e.g., nse_cm, bse_cm).
series	string	Series of the instrument (e.g., EQ).
exchangeIdentifier	string	Exchange-provided identifier for the scrip.
sellableQuantity	integer	Quantity available for selling at the moment.
securityType	string	Type of security (e.g., EQUITY STOCK, ETF).

Field	Type	Description
securitySubType	string	Subtype of security (e.g., EQUITY STOCK, ETF).
logoUrl	string	URL for the company/logo image.
cmotCode	string	Internal Kotak code mapped for margin/holdings tracking.

Example Error Response


```
{
  "stat": "Not_Ok",
  "emsg": "Invalid session",
  "stCode": 1003
}
```


Field	Type	Description
stat	string	"Not_Ok" for errors
emsg	string	Error message in English
stCode	int	Error code (see explanations below)

6. Notes

- Only securities currently held in the portfolio are reported.
- `sellableQuantity` reflects what can be sold on the current trading day.
- Fields like `instrumentToken` and `exchangeSegment` are useful for placing new orders.
- Use valid session and auth tokens to avoid authentication errors.

7. Response change from v1 API.

This API along with its endpoint has also gone response field changes, if you were the user of older api, kindly map the new key names as per the need.

Field - OLD API	Example Value	Field - NEW API	Example Value
displaySymbol	ITBEES	instrumentType	Equity
averagePrice	38.7162	sector	Others
quantity	45	instrumentToken	18974
exchangeSegment	nse_cm	commonScripCode	NIPPON_ETF_NIFTYIT
exchangeIdentifier	19084	instrumentName	Nippon India ETF Nifty IT
holdingCost	1742.2293	quantity	45
mktValue	1771.2	averagePrice	38.7162

Field - OLD API	Example Value	Field - NEW API	Example Value
scripId	...	holdingCost	1742.2293
instrumentToken	18974	closingPrice	39.36
instrumentType	Equity	mktValue	1771.2
isAlternateScrip	false	scripId	
closingPrice	39.36	isAlternateScrip	
		unrealisedGainLoss	28.9707
		sqGainLoss	-5.6
		delGainLoss	-7.074
		subTotal	0

Limits

1. Introduction

The **Limits API** allows you to query real-time available limits, margins, collateral, exposure, and cash balances for your trading account, filtered by segment, exchange, and product type.

2. API Endpoint

`POST <Base URL>/quick/user/limits`

Replace `<Base URL>` with the relevant Kotak environment base URL provided in response from /tradeApiValidate api.

3. Headers

Name	Type	Description
accept	string	`application/json`
Sid	string	session sid generated on login
Auth	string	session token generated on login
neo-fin-key	string	static value: neotradeapi
Content-Type	string	`application/x-www-form-urlencoded`

4. Request

Example Request:

```
curl --location '<Base URL>/quick/user/limits' \
 -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi"
```

```
--data-urlencode 'jData={"seg":"ALL","exch":"ALL","prod":"ALL"}'
```

Request Body (`jData`)

Name	Type	Description	Allowed Values	Default
seg	string	Segment to fetch limits for	ALL, CASH, CUR, FO	ALL
exch	string	Exchange to fetch limits for	ALL, NSE, BSE	ALL
prod	string	Product to fetch limits for	ALL, NRML, CNC, MIS	ALL

5. Response

Example Success Response (truncated)


```
{
    "Category": "CLIENT_MTF",
    "EntityId": "account-******",
    "BoardLotLimit": "5000",
    "CollateralValue": "10197.48",
    "Net": "10157.08",
    "MarginUsed": "40.4",
    "AdhocMargin": "0",
    "SpanMarginPrsnt": "0",
    "ExposureMarginPrsnt": "0",
    "NotionalCash": "0",
    "UnrealizedMtomPrsnt": "0",
    "RealizedMtomPrsnt": "0",
    "SpecialMarginPrsnt": "0",
    "PremiumPrsnt": "0",
    "MarginVarPrsnt": "0",
    "stCode": 200,
```

```
    "stat": "Ok"
}
```

200 Response Fields (most relevant)

Field	Type	Description
Category	string	Category
EntityId	string	Account ID
BoardLotLimit	string	Board lot limit
CollateralValue	string	Value of pledged securities/collateral
Net	string	Net available margin/cash
MarginUsed	string	Margin already used
AdhocMargin	string	Extra margin added
SpanMarginPrsnt	string	SPAN margin requirement
ExposureMarginPrsnt	string	Exposure margin requirement
NotionalCash	string	Notional (total) cash

Field	Type	Description
UnrealizedMtomPrsnt	string	Unrealized Mark-to-Market (PnL)
RealizedMtomPrsnt	string	Realized Mark-to-Market (PnL)
SpecialMarginPrsnt	string	Special margin imposed
PremiumPrsnt	string	Premium margin present
MarginVarPrsnt	string	VAR margin present
stCode	int	Status code (200 = success)
stat	string	"Ok" for success
...	...	Additional technical/segment breakdown fields

Example Error Response


```
{
  "stat": "Not_Ok",
  "emsg": "Invalid session",
  "stCode": 1003
```

```
}
```

Field	Type	Description
stat	string	"Not_Ok" for errors
emsg	string	Error message in English
stCode	int	Error code as below

6. Notes

- All numerical limits are provided as strings for precision.
- The response includes total cash, margin, and product/segment-wise breakdowns.
- Supply `"ALL"` for any parameter to fetch a consolidated report.
- Use the response fields such as `CollateralValue`, `Net`, `MarginUsed` to display funding or risk information on your interface.
- Use valid session and authentication tokens to avoid errors.

Margins

1. Introduction

The **Margin API** allows you to check the required margin and available balance for a new order, including advanced order types such as Bracket Orders (BO) and Cover Orders (CO). This API helps you validate whether you have enough margin before placing an order.

2. API Endpoint

`POST <Base URL>/quick/user/check-margin`

Replace `<Base URL>` with the relevant Kotak environment base URL provided in response from /tradeApiValidate api.

3. Headers

Name	Type	Description
accept	string	`application/json`
Sid	string	session sid generated on login
Auth	string	session token generated on login
neo-fin-key	string	static value: neotradeapi
Content-Type	string	`application/x-www-form-urlencoded`

4. Request

Example Request

```
curl --location '<Base URL>/quick/user/check-margin' \
```

```
  -H "Auth: <session_token>" \
   -H "Sid: <session_sid>" \
   -H "neo-fin-key: neotradeapi"
--data-urlencode
'jData={"brkName":"KOTAK","brnchId":"ONLINE","exSeg":"nse_cm","prc":"12500","prcTp":"L","prod
":"CNC","qty":"1","tok":"11536","trnsTp":"B"}'
```

Request Body (`jData`)

Field	Type	Description	Required	Example
brkName	string	Broker name (always send as `KOTAK`)	Yes	KOTAK
brnchId	string	Branch id of user (always send as `ONLINE`)	Yes	ONLINE
exSeg	string	Exchange segment, e.g., `nse_cm`	Yes	nse_cm
prc	string	Order price in Rupees (`0` for market orders)	Yes	12500
prcTp	string	Order type (`L`, `MKT`, etc.)	Yes	L
prod	string	Product code (`CNC`, `NRML`, `MIS` etc.)	Yes	CNC
qty	string	Order quantity	Yes	1

Field	Type	Description	Required	Example
tok	string	Token number (from scrip master)	Yes	11536
trnsTp	string	Transaction type (`B` for Buy, `S` for Sell)	Yes	B
slAbsOrTks	string	Stop loss type (`Absolute` or `Ticks`, BO only)	No	
slVal	string	Stop loss value (BO only)	No	
sqrOffAbsOrTks	string	Square off type (`Absolute` or `Ticks`, BO only)	No	
sqrOffVal	string	Square off value (BO only)	No	
trailSL	string	Trailing stop loss (`Y` or `N`, BO only)	No	
trgPrc	string	Trigger price (for CO only, in Rupees)	No	

Field	Type	Description	Required	Example
tSLTks	string	Trailing SL value (for BO only, "Y" or "N")	No	

For Bracket and Cover Orders, include the additional optional fields as needed.

5. Response

Example Success Response


```
{
    "avlCash": "10197.480000",
    "totMrgnUsd": "12540.400000",
    "mrgnUsd": "40.400000",
    "ordMrgn": "12500.000000",
    "rmsVldtd": "NOT_OK",
    "reqdMrgn": "12540.400000",
    "avlMrgn": "10197.480000",
    "insufFund": "2342.920000",
    "stat": "Ok",
    "stCode": 200
}
```


200 Response Fields

Field	Type	Description
avlCash	string	Total cash available
avlMrgn	string	Available margin after order is placed

Field	Type	Description
insufFund	string	Insufficient funds required to place order
mrgnUsd	string	Margin already used
ordMrgn	string	Margin required for the order
reqdMrgn	string	Net required margin for the order
rmsVldtd	string	Value is typically `OK` or `NOT_OK`
stat	string	API status (`Ok` for success)
totMrgnUsd	string	Total margin used
stCode	int	HTTP status code (200 = success)

Example Error Response

```
{
  "stat": "Not_Ok",
  "emsg": "Invalid session",
  "stCode": 1003
}
```

Field	Type	Description
stat	string	"Not_Ok" for errors
emsg	string	Error message in English
stCode	int	Error code

6. Notes

- Use this API **before** placing any order to check if you have enough funds/margin.
- For market orders, send `prc` as `"0"`.
- For BO/CO, include the SL, trailing stop loss, and square off fields as applicable.
- Use valid session and auth tokens to avoid authentication errors.

Quotes

1. Introduction

The **Quotes API** retrieves live and last traded market data for one or more instruments (including stocks, ETFs, and indices) from supported exchanges. It allows the use of advanced filters to fetch specific, detailed market values including depth, OHLC, circuit limits, and more.

2. API Endpoint

```
GET <Base URL>/script-details/1.0/quotes/neosymbol/<query>[,<query>][/<filter_name>]
```

Replace `<Base URL>` with the relevant Kotak environment base URL provided in response from the `/tradeApiValidate` API.

Key points about endpoint structure:

- `<query>` is formatted as `<exchange_segment>|<instrument>`.
- Multiple queries are separated by commas, e.g. `nse_cm|Nifty 50,bse_cm|SENSEX`.
- Instrument (except indices): Use `pSymbol` from the instrument/scrip master file.
- Indices: Use the **exact case-sensitive name** (e.g. `Nifty 50`, `BANKEX`).
- **Expected values for `exchange_segment` are (string):**
  - `nse_cm` (NSE Cash)
  - `bse_cm` (BSE Cash)
  - `nse_fo` (NSE F&O)
  - `bse_fo` (BSE F&O)
  - `cde_fo` (CDS F&O)

3. Headers

Name	Type	Description
Authorization	string	Token provided in your NEO API dashboard - use plain token
Content-Type	string	`application/json`

4. Request

Example Request:

```
curl --location --request GET '<Base URL>/script-details/1.0/quotes/neosymbol/nse_cm|Nifty 50,nse_cm|Nifty Bank/all' \
--header 'Content-Type: application/json' \
--header 'Authorization: xxxxx-your-api-token-xxxx'
```

Filter Options

After all queries, you may append a filter with `/filter_name`.
Allowed filter values (total 8 including default 'all'):

- `all` (default, returns all fields)
- `52W` (52-week high/low)
- `scrip_details` (scrip basics)
- `circuit_limits` (circuit limits)
- `ohlc` (Open, High, Low, Close)
- `oi` (open interest, if applicable)
- `depth` (order book, top 5 each side)
- `ltp` (last traded price)

5. Response

Example Success Response


```
[
    {
        "exchange_token": "SENSEX",
        "display_symbol": "SENSEX-IN",
        "exchange": "bse_cm",
        "lstup_time": "1757915078",
        "ltp": "81809.3400",
        "last_traded_quantity": "0",
        "total_buy": "0",
        "total_sell": "0",
        "last_volume": "0",
        "change": "-95.3600",
        "per_change": "-0.1200",
        "year_high": "0",
        "year_low": "0",
        "ohlc": {
            "open": "81925.5100",
```

```
            "high": "81998.5100",
            "low": "81779.8200",
            "close": "81904.7000"
        },
        "depth": {
            "buy": [
                {"price": "0", "quantity": "0", "orders": "0"},
                ...
            ],
            "sell": [
                {"price": "0", "quantity": "0", "orders": "0"},
                ...
            ]
        }
    }
]
```

Response Field Mapping

Field	Type	Description
exchange_token	string	Instrument token or index name
display_symbol	string	UI display symbol
exchange	string	Exchange segment (e.g. nse_cm, bse_cm, ...)
lstup_time	string	Last update time (Unix timestamp)

Field	Type	Description
ltp	string	Last traded price
last_traded_quantity	string	Last traded quantity
total_buy	string	Top bid quantity
total_sell	string	Top offer quantity
last_volume	string	Most recent trade volume
change	string	Net price change from previous close
per_change	string	Percent price change
year_high	string	52-week high
year_low	string	52-week low

Field	Type	Description
ohlc	object	Object: open, high, low, close prices
depth	object	Top 5 bid/ask levels (arrays ‘buy’ & ‘sell’)

OHLC Object

Field	Type	Description
open	string	Day’s open price
high	string	Day’s high price
low	string	Day’s low price
close	string	Previous close price

Depth Object

Field	Type	Description
price	string	Price level
quantity	string	Quantity at level
orders	string	Order count at level

Example Error Response

```
{
  "stat": "Not_Ok",
  "emsg": "Invalid instrument/code",
  "stCode": 1009
}
```

Field	Type	Description
stat	string	"Not_Ok" for errors
emsg	string	Error message
stCode	int	Error code

6. Notes

- All fields are returned as strings.
- When using indices, pass the correct case-sensitive index name.
- For stocks/ETFs, use `pSymbol` from instrument/scrip master.
- Multi-instrument queries are comma-separated.
- Valid exchange segments are: **nse_cm, bse_cm, nse_fo, bse_fo, cde_fo** (must be passed as string).
- By default (`/all` or blank) returns all quote data; filters allow more targeted queries.

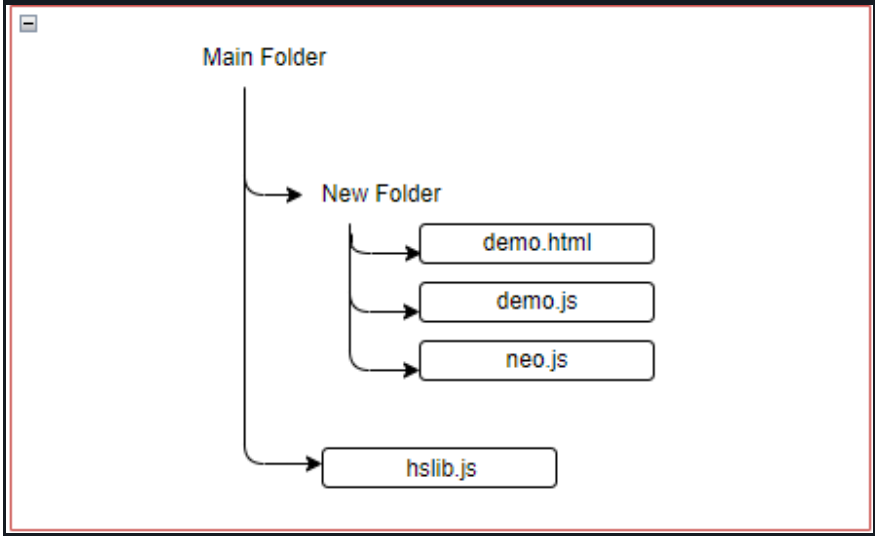
NEO Websocket

Websocket (2).zip
Kotak Neo Websocket zip contains 4 major files :

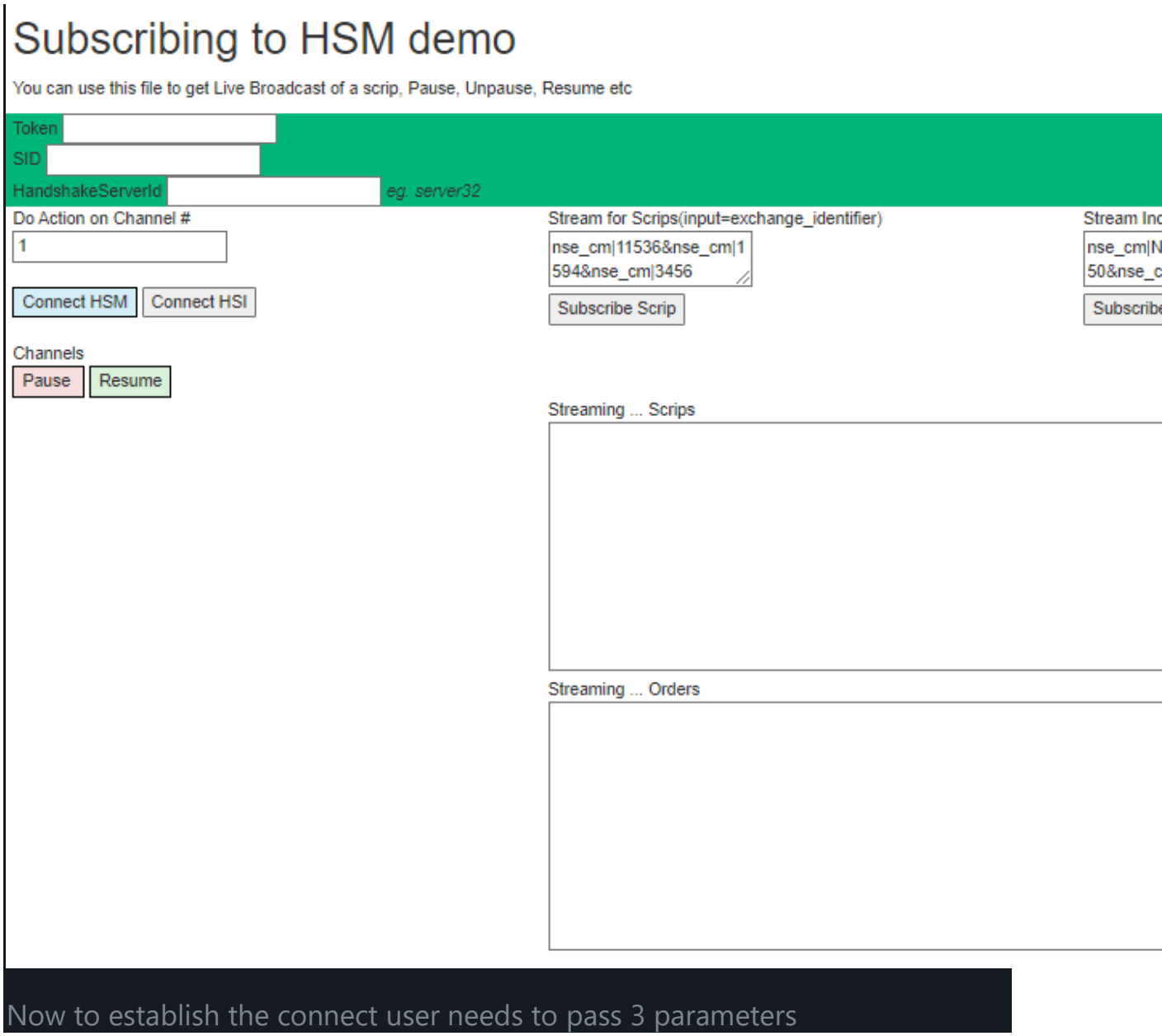
1. HSLib

2. Demo.html
3. Demo.js
4. Neo.js

To Start the websocket make sure the position of these files are aligned as shown in below example :



To Start Websocket open demo.html file, this file will open a web page in your browser which will like this :

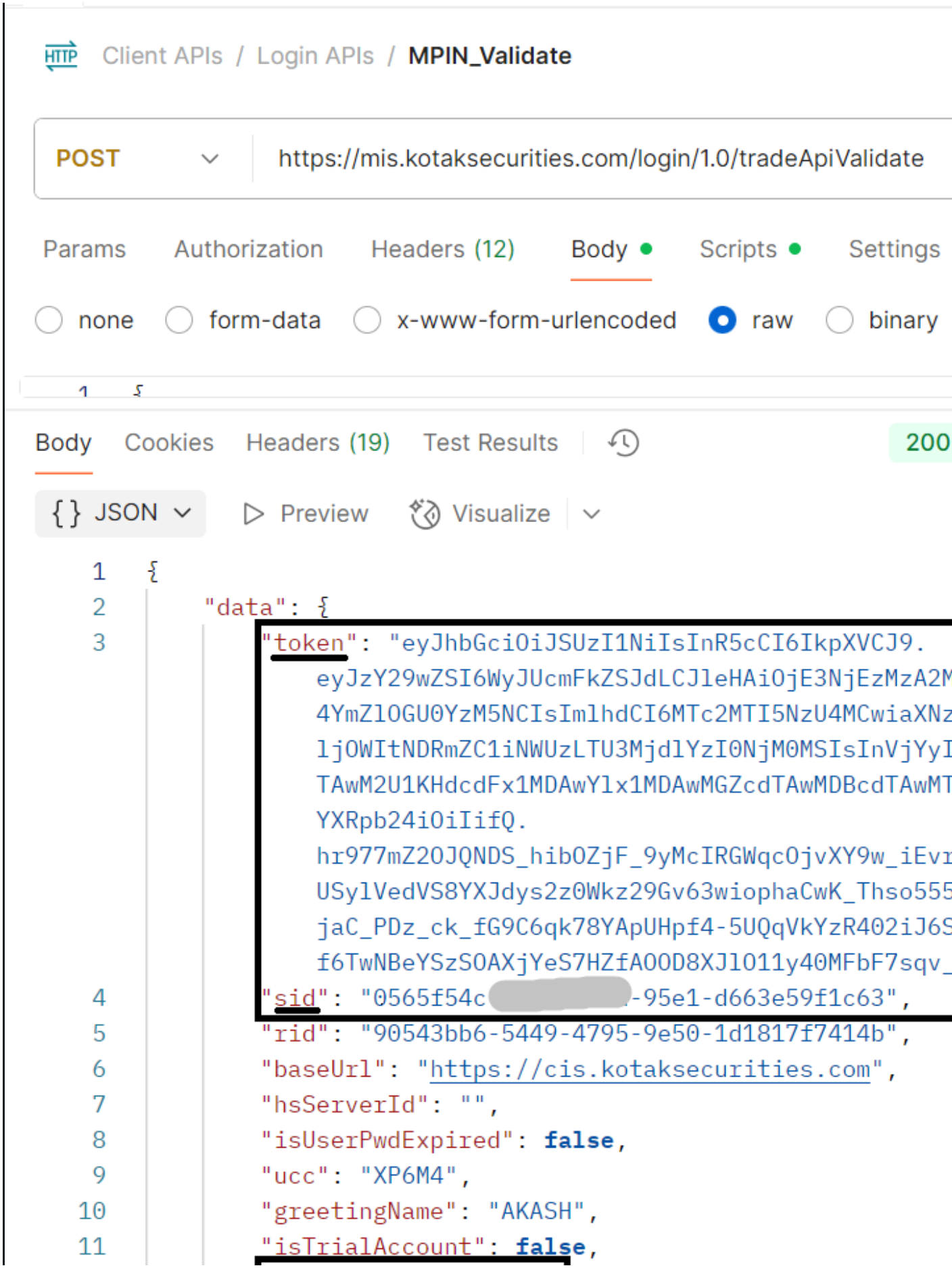


Now to establish the connect user needs to pass 3 parameters

- **Token** : Final token(session token) which user gets after running login api
- **sid** : received along with session sid after running login api
- **Data Center**: In response to login API, you get data center. like E43, E41, etc

```
'dataCenter': '123'
```

Note: If you're using postman for testing https://mis.kotaksecurities.com/login/1.0/tradeApiValidate would return all the above 3 values.
Example:

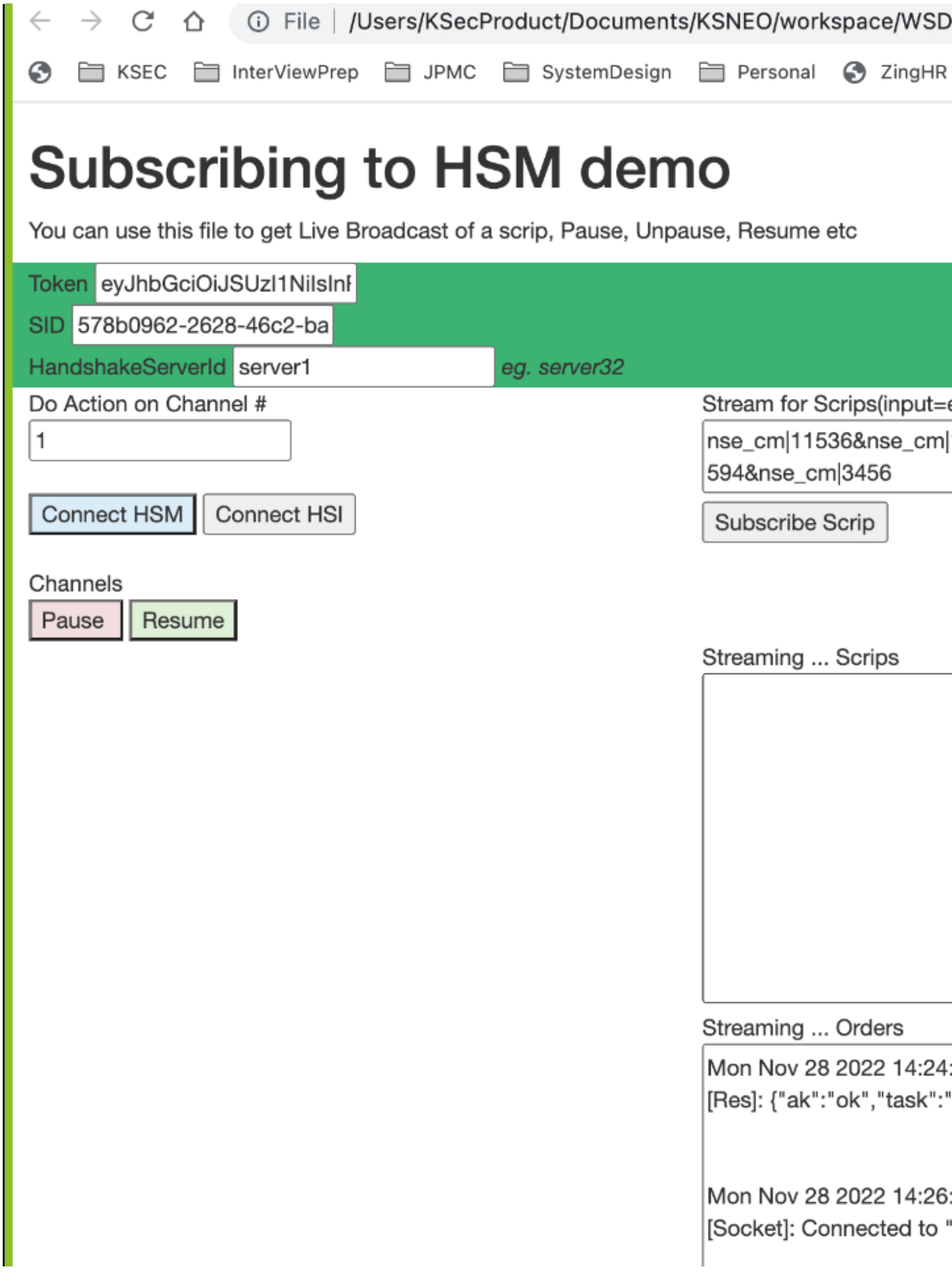

Client APIs / Login APIs / MPIN_Validate
POST
https://mis.kotaksecurities.com/login/1.0/tradeApiValidate
Params
Authorization
Headers (12)
Body
Scripts
Settings
none
form-data
x-www-form-urlencoded
raw
binary
Body
Cookies
Headers (19)
Test Results
200
JSON
Preview
Visualize
{
"data": {
"token": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.
eyJzY29wZSI6WyJUcmFkZSJdLCJleHAiOjE3NjEzMzA2M
4YmZlOGU0YzM5NCIsImlhdCI6MTc2MTI5NzU4MCwiaXNz
ljOWItNDRmZC1iNWUzLTU3MjdlYzI0NjM0MSIsInVjYyI
TAwM2U1KHdcdFx1MDAwY1x1MDAwMGZcdTAwMDBcdTAwMT
YXRpb24iOiIifQ.
hr977mZ2OJQNDS_hibOZjF_9yMcIRGWqcOjvXY9w_iEvr
USylVedVS8YXJdys2z0Wkz29Gv63wiophaCwK_Thso555
jaC_PDz_ck_fG9C6qk78YApUHpf4-5UQqVkYzR402iJ6S
f6TwNBeYSzSOAXjYeS7HZfAOOD8XJlO11y40MFbF7sqv_
"sid": "0565f54c[redacted]-95e1-d663e59f1c63",
"rid": "90543bb6-5449-4795-9e50-1d1817f7414b",
"baseUrl": "https://cis.kotaksecurities.com",
"hsServerId": "",
"isUserPwdExpired": false,
"ucc": "XP6M4",
"greetingName": "AKASH",
"isTrialAccount": false,

Connect to Market feed / websocket :

Lets understand how to connect to HSM

HSM is nothing but that which streams market data.

When we pass Session Token, SID and data center we can connect with HSM.
Click on "Connect HSM"


File | /Users/KSecProduct/Documents/KSNEO/workspace/WSD
KSEC
InterViewPrep
JPMC
SystemDesign
Personal
ZingHR
Subscribing to HSM demo
You can use this file to get Live Broadcast of a scrip, Pause, Unpause, Resume etc
Token eyJhbGciOiJSUzI1NiIsInI
SID 578b0962-2628-46c2-ba
HandshakeServerId server1 eg. server32
Do Action on Channel #
1
Connect HSM
Connect HSI
Channels
Pause
Resume
Stream for Scrips(input=e
nse_cm|11536&nse_cm|
594&nse_cm|3456
Subscribe Scrip
Streaming ... Scrips
Streaming ... Orders
Mon Nov 28 2022 14:24:
[Res]: {"ak":"ok","task":"
Mon Nov 28 2022 14:26:
[Socket]: Connected to "

Subscribe Scrip :

Here user have to provide the exchange identifier to start receiving the Feeds.

How to add scrip :

Ex : nse_cm|11536&

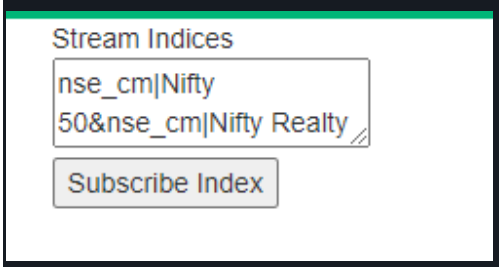
1 input of scrip consist of exchange name then a pipe separation (|) followed by scrip identifier.

To add another scrip simultaneously you can add in the same input by putting &(ampersand)seperation.

Subscribe Index :

Here user have to provide the exchange identifier of indices to start receiving the Feeds.

How to add index:

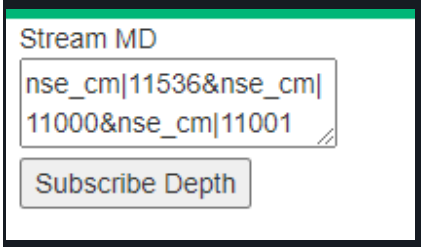


Ex : nse_cm|Nifty 50&

1 input of index consist of exchange name then a pipe separation (|) followed by index identifier.

To add another index simultaneously you can add in the same input by putting &(ampersand)seperation.

Subscribe Depth:



Ex : nse_cm|11536&

1 input of market depth consist of exchange name then a pipe separation (|) followed by scrip identifier.

To add another Market Depth Scrip simultaneously you can add in the same input by putting &(ampersand)seperation.

Connect to Order feed:

HSI is noting but stream that delivers order updates. You can connect with HSI to view feeds of Orders you have placed. The feeds will be reflected in Streaming Orders Column

Integrating market feed or order feed in the code?

By running the inspect to the demo.html file you can get the websocket string.

Total No of Channels user's can use at a time : 16

Total No of Scrips user's can subscribe at a time : 200

For Indepth detail of Websocket Functions: Refer this

📌 Troubleshooting - FAQs/Error codes

🔑 Tokens & Authentication

Q1. What is an access token, and what's new about it?

The **access token** now comes from the **Neo app/web → Invest → TradeAPI → API Dashboard** (not via `/oauth2/token`). Send it as a **plain string** (no `Bearer`). Resetting it **immediately invalidates** all sessions.

Q2. What happens if I reset the token?

All active sessions break **instantly**. Re-login (TOTP → MPIN validate) to obtain **new session token (`Auth`) and session sid (`sid`)**.

Q3. Demystify tokens: access token, view token, session token (trade token), view sid, session sid, neo-fin-key

- **Access Token** – From Neo dashboard. Used for **login APIs** and **Quotes/Scripmaster**.
- **View Token + View SID** – Returned by `/login/1.0/tradeApiLogin` (TOTP step).
- **Session Token (aka Trade Token) + Session SID** – Returned by `/login/1.0/tradeApiValidate` (MPIN step). Use these as `Auth` and `sid` headers for all **post-login APIs**.
- **neo-fin-key** – **Always send** `neotradeapi` (static) **except** in **Quotes/Scripmaster**, where it is **not required**.

⏱ Login & baseUrl

Q4. What are the login endpoints (fixed)?

- **TOTP Login**: `https://mis.kotaksecurities.com/login/1.0/tradeApiLogin`
- **MPIN Validate**: `https://mis.kotaksecurities.com/login/1.0/tradeApiValidate` → returns `baseUrl`, **session token** (header `Auth`) and **session sid** (header `sid`).

Q5. Is `baseUrl` static or dynamic?

It's **stable for the day** and even after that rarely changes. Always capture it after MPIN validate and use it for that session.

Q6. Which APIs need `baseUrl`?

All **post-login** APIs: **Orders**, **Reports**, **Portfolio**, **Limits**, **Margins**, **Quotes**, **Scripmaster**. (Only login endpoints are fixed.)

Q7. Show me how to replace `{{baseUrl}}` with a real example.

- Suppose `baseUrl` returned is: `https://neo-gw.kotaksecurities.com/xyz`
- Specified endpoint: `{{baseUrl}}/quick/order/cancel`
- **Final URL to call**: `https://neo-gw.kotaksecurities.com/xyz/quick/order/cancel`

(Just replace `{{baseUrl}}` with the full string you received—no braces remain.)

📋 Headers & Endpoint Usage

Q8. Which headers do I pass for each category?

API Category	Required Headers
Login (TOTP + MPIN)	`Authorization: <access token>` + `neo-fin-key: neotradeapi`
Orders / Reports / Portfolio / Limits / Margins	`Auth: <session token>` + `sid: <session sid>` + `neo-fin-key: neotradeapi`
Quotes / Scripmaster	`Authorization: <access token>` (no `neo-fin-key`, no `Auth/sid`)

Q9. Do I use `Bearer` with Authorization?

No. Always pass the token **without** `Bearer`.

🔐 TOTP Registration & Troubleshooting

Q10. How do I register for TOTP?

Dashboard → **TOTP Registration in menu** → verify **mobile OTP + client code** → **scan QR** (Google/Microsoft Authenticator) → **enter TOTP** → success toast.

Q11. I reinstalled my authenticator. What now?

Deregister via the same route, then **register** again with a new QR.

Q12. I see "Invalid TOTP" / "Service error"

- **Invalid TOTP**: sync device time (auto time), use the latest code.
- **Service error**: there's a **5-minute cooldown** if you reattempt too quickly—wait and then re-scan QR.

🌐 Static IP & Family Mapping

Q13. Where do I find the network IP ?

For windows system, go to command prompt from the start menu.
type `ipconfig` and press enter. You will get IPv4 address that is to be whitelisted on the NEO API dashboard.
For mac system, open Terminal and type `ipconfig getifaddr en0` for Wi-Fi or `ipconfig getifaddr en1` for Ethernet to see your local IP

Q14. Where can I get a static IP?

You can request one from your **ISP** (Airtel, Jio, ACT, etc.)—this usually needs **Aadhaar/KYC**. Another option is an **IP-over-VPN service** that provides a fixed address.
Kotak Securities is **not associated with any provider**; please do your own due diligence before choosing one.

Q15. Why static IP?

For **SEBI compliance** and security. Ensures only trusted infra can call your APIs. **Optional now**, but will become mandatory once the circular is effective.

Q16. How many IPs can I set and how often can I change?

You can set **one primary & one secondary** (backup). **Each can be changed once per week**.

Q17. Can I reuse the IP for family accounts?

Yes—**self-serve UI** lets you link up to **10 family members** to the same whitelisted IP.

Q18. I don't have a static IP.

Consider using a **registered third-party platform** (e.g., **smallcase**) that manages whitelisting with Kotak.

📊 API Changes

Q19. Portfolio Holdings response—what changed?
The **new response** adds richer fields such as `commonScripCode`, `logoUrl`, `cmotCode`, `unrealisedGainLoss`, `sqGainLoss`, `delGainLoss`, `marketLot`, `securitySubType`, etc.
(We'll show a before/after table in the visuals.)

Q20. What are the new endpoints for Quotes & Scripmaster?

- **Quotes** → `{{baseUrl}}/script-details/1.0/quotes/` *(headers:* `Authorization: <access token>` *only)*
- **Scripmaster files** → `{{baseUrl}}/script-details/1.0/masterscrip/file-paths` *(headers:* `Authorization: <access token>` *only)*

🛠️ Third-Party & SDKs

Q21. I'm on a third-party platform. What should I do?
Check with your provider if they've **migrated to v2**. If they have, **no action** is required on your end.

Q22. I use the Python SDK. Any changes?
No changes needed currently; old endpoints remain valid. For direct REST calls, follow this migration guide.

Q23. Do you provide SDKs in other languages?
For now, **use cURL + Postman codegen** to generate language stubs.

📈 Rate Limits & Cutover

Q24. What are the rate limits?
10 requests/second across APIs. Exceeding this returns rate limit errors.

Q25. What happens after v1 retires?
Calls to old endpoints will **fail**.

✅ Error FAQs

Login API (common)

- **Invalid TOTP** → The code is wrong/expired. Sync device time and retry with a fresh code.
- **Invalid MPIN** → MPIN incorrect. Verify or reset MPIN, then retry.
- **Session expired (after token reset)** → Resetting your access token immediately ends existing sessions. Re-login (TOTP → MPIN).
- **Dependency error (424)** → Temporary backend issue. Retry after a few seconds.

Orders API (common)

- **1006: Invalid Exchange** → Wrong exchange segment. Use correct NSE/BSE segment.
- **1007: Invalid Symbol** → Scrip not found/unsupported. Validate with the scrip master.
- **1009: Invalid Quantity** → Below min or wrong lot step. Match the instrument's lot size/rules.
- **1005: Internal Error** → Transient issue. Retry; if persistent, raise a support ticket.

Reports API (common)

- **400: Request format error** → Check payload schema, types, and required fields.
- **424: Dependency failure** → Upstream dependency issue. Retry later.
- **401/Expired session** → Session token/sid expired or missing. Re-login and pass `Auth` + `sid`.

---

cURL examples

🔑 Login Flow (fixed endpoints)

1) TOTP Login → returns viewToken + viewSid

```
curl -X POST "https://mis.kotaksecurities.com/login/1.0/tradeApiLogin" \
  -H "Authorization: <access_token>" \
  -H "neo-fin-key: neotradeapi" \
  -H "Content-Type: application/json" \
  -d '{
        "mobileNumber": "<+91XXXXXXXXXX>",
        "ucc": "<client_code>",
        "totp": "<6_digit_totp>"
      }'
```

2) MPIN Validate → returns session token (Auth) + session sid (Sid) + baseUrl

In your collection, viewSid and viewToken are sent as headers to this call.

```
curl -X POST "https://mis.kotaksecurities.com/login/1.0/tradeApiValidate" \
  -H "Authorization: <access_token>" \
  -H "neo-fin-key: neotradeapi" \
  -H "sid: <viewSid_from_previous_step>" \
  -H "Auth: <viewToken_from_previous_step>" \
  -H "Content-Type: application/json" \
  -d '{
        "mpin": "<mpin>"
      }'
```

📌 **Response** gives you:

- `baseUrl` (use it for **all** post-login APIs)
- `Auth` = **session token**
- `Sid` = **session sid**

🔁 Using `baseUrl` (important)

If MPIN validate returned:

```
"baseUrl": "https://neo-gw.kotaksecurities.com/xyz"
```

and the spec shows:

```
{{baseUrl}}/quick/order/cancel
```

then your **final URL** is:

```
https://neo-gw.kotaksecurities.com/xyz/quick/order/cancel
```

👉 Just **replace** `{{baseUrl}}` with the returned string. No braces in the final URL.

📜 Orders (Postman: urlencoded, `jData` body)

Place Order

```
curl -X POST "<baseUrl>/quick/order/rule/ms/place" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  --data-urlencode 'jData={
    "am": "NO",
    "dq": "0",
    "es": "nse_cm",
    "mp": "0",
    "pc": "CNC",
    "pf": "N",
    "pr": "0",
    "pt": "MKT",
    "qt": "1",
    "rt": "DAY",
    "tp": "0",
    "ts": "ITBEES-EQ",
    "tt": "B"
  }'
```

Modify Order

```
curl -X POST "<baseUrl>/quick/order/vr/modify" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  --data-urlencode 'jData={
    "am": "NO",
    "dq": "0",
    "es": "nse_cm",
```

```
    "mp": "0",
    "pc": "NRML",
    "pf": "N",
    "pr": "0",
    "pt": "MKT",
    "qt": "1",
    "rt": "DAY",
    "tp": "0",
    "ts": "TATAPOWER-EQ",
    "tt": "B",
    "no": "<orderNo>"
  }'
```

Cancel / Exit (Cover / Bracket)

```
# Cancel
curl -X POST "<baseUrl>/quick/order/cancel" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  --data-urlencode 'jData={"on":"<orderNo>","am":"NO"}'

# Exit Cover
curl -X POST "<baseUrl>/quick/order/co/exit" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  --data-urlencode 'jData={"on":"<orderNo>","am":"NO"}'

# Exit Bracket
curl -X POST "<baseUrl>/quick/order/bo/exit" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  --data-urlencode 'jData={"on":"<orderNo>","am":"NO"}'
```

Reports

Order Book (GET)

```
curl -X GET "<baseUrl>/quick/user/orders" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi"
```

Order History (POST, urlencoded)

```
curl -X POST "<baseUrl>/quick/order/history" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  --data-urlencode 'jData={"nOrdNo":"250720000007242"}'
  }'
```

Trade Book / Positions / Holdings (GET)

```
# Trade Book
curl -X GET "<baseUrl>/quick/user/trades" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi"

# Position Book
curl -X GET "<baseUrl>/quick/user/positions" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi"

# Portfolio Holdings (new response shape)
curl -X GET "<baseUrl>/portfolio/v1/holdings" \
  -H "Auth: <session_token>" \
  -H "Sid: <session_sid>" \
  -H "neo-fin-key: neotradeapi"
```

📈 Quotes (GET; no `neo-fin-key`, no `Auth/Sid`)

Two patterns appear in your collection; here's the **neosymbol** route:

```
curl -X GET "<baseUrl>/script-details/1.0/quotes/neosymbol/nse_cm|26000/all" \
  -H "Authorization: <access_token>"
```

Only Authorization header is required here (plain access token).

Do not send `neo-fin-key`, `Auth`, or `Sid`.

🗁 Scripmaster Files (GET; no `neo-fin-key`, no `Auth/Sid`)

```
curl -X GET "<baseUrl>/script-details/1.0/masterscrip/file-paths" \
  -H "Authorization: <access_token>"
```

Notes :

- **Headers**:
  - Post-login APIs (Orders/Reports/Portfolio/etc.) use `Auth` (session token) + `Sid` (session sid).
  - `neo-fin-key: neotradeapi` is required **except** for **Quotes** and **Scripmaster** (per your instruction).
- **Bodies** for Orders and some Reports are `application/x-www-form-urlencoded` with a `jData` parameter containing JSON (exactly as in your collection).