Authentication

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)
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>" \ -H "Content-Type: application/json" \ -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
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; 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
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