```
[ZERODHA][MARKET_PRICE] Failed {
  symbol: 'RELIANCE',
  error: {
    status: 'error',
    message: 'Insufficient permission for that call.',
    data: null,
    error_type: 'PermissionException'
  }
}
AxiosError: Request failed with status code 403
    at settle
(C:\Users\ADMIN\OneDrive\Documents\Files\VisionX\BullTrek\node_modules\axios\dist\nod
e\axios.cjs:2106:12)
    at Unzip.handleStreamEnd
(C:\Users\ADMIN\OneDrive\Documents\Files\VisionX\BullTrek\node_modules\axios\dist\nod
e\axios.cjs:3491:11)
    at Unzip.emit (node:events:519:28)
    at Unzip.emit (node:domain:489:12)
    at endReadableNT (node:internal/streams/readable:1698:12)
    at process.processTicksAndRejections (node:internal/process/task_queues:90:21)
    at Axios.request
(C:\Users\ADMIN\OneDrive\Documents\Files\VisionX\BullTrek\node_modules\axios\dist\nod
e\axios.cjs:4731:41)
    at process.processTicksAndRejections (node:internal/process/task_queues:105:5)
    at async fetchZerodhaMarketPrice
(C:\Users\ADMIN\OneDrive\Documents\Files\VisionX\BullTrek\dist\services\stocks\exchang
e\zerodhaService.js:283:26)
    at async Object.fetchMarketPrice
(C:\Users\ADMIN\OneDrive\Documents\Files\VisionX\BullTrek\dist\sockets\stocks\marketDa
ta\marketDataManager.js:258:24)
    at async
C:\Users\ADMIN\OneDrive\Documents\Files\VisionX\BullTrek\dist\utils\scheduler\scheduler.j
s:32:33 {
  code: 'ERR_BAD_REQUEST',
  config: {
    transitional: {
      silentJSONParsing: true,
      forcedJSONParsing: true,
      clarifyTimeoutError: false
    },
    adapter: [ 'xhr', 'http', 'fetch' ],
    transformRequest: [ [Function: transformRequest] ],
    transformResponse: [ [Function: transformResponse] ],
    timeout: 0,
    xsrfCookieName: 'XSRF-TOKEN',
    xsrfHeaderName: 'X-XSRF-TOKEN',
    maxContentLength: -1,
    maxBodyLength: -1,
```

```
  env: { FormData: [Function], Blob: [class Blob] },
  validateStatus: [Function: validateStatus],
  headers: Object [AxiosHeaders] {
    Accept: 'application/json, text/plain, */*',
    'Content-Type': undefined,
    'X-Kite-Version': '3',
    Authorization: 'token ...:...',
    'User-Agent': 'axios/1.13.2',
    'Accept-Encoding': 'gzip, compress, deflate, br'
  },
  params: { i: 'NSE:RELIANCE' },
  method: 'get',
  url: 'https://api.kite.trade/quote/ltp',
  allowAbsoluteUrls: true,
  data: undefined
},
request: <ref *1> ClientRequest {
  _events: [Object: null prototype] {
    abort: [Function (anonymous)],
    aborted: [Function (anonymous)],
    connect: [Function (anonymous)],
    error: [Function (anonymous)],
    socket: [Function (anonymous)],
    timeout: [Function (anonymous)],
    finish: [Function: requestOnFinish]
  },
  _eventsCount: 7,
  _maxListeners: undefined,
  outputData: [],
  outputSize: 0,
  writable: true,
  destroyed: true,
  _last: true,
  chunkedEncoding: false,
  shouldKeepAlive: true,
  maxRequestsOnConnectionReached: false,
  _defaultKeepAlive: true,
  useChunkedEncodingByDefault: false,
  sendDate: false,
  _removedConnection: false,
  _removedContLen: false,
  _removedTE: false,
  strictContentLength: false,
  _contentLength: 0,
  _hasBody: true,
  _trailer: '',
  finished: true,
  _headerSent: true,
```

```
_closed: true,
_header: 'GET /quote/ltp?i=NSE:RELIANCE HTTP/1.1\r\n' +
  'Accept: application/json, text/plain, */*\r\n' +
  'X-Kite-Version: 3\r\n' +
  'Authorization: token ...:...\r\n' +
  'User-Agent: axios/1.13.2\r\n' +
  'Accept-Encoding: gzip, compress, deflate, br\r\n' +
  'Host: api.kite.trade\r\n' +
  'Connection: keep-alive\r\n' +
  '\r\n',
_keepAliveTimeout: 0,
_onPendingData: [Function: nop],
agent: Agent {
  _events: [Object: null prototype],
  _eventsCount: 2,
  _maxListeners: undefined,
  defaultPort: 443,
  protocol: 'https:',
  options: [Object: null prototype],
  requests: [Object: null prototype] {},
  sockets: [Object: null prototype] {},
  freeSockets: [Object: null prototype],
  keepAliveMsecs: 1000,
  keepAlive: true,
  maxSockets: Infinity,
  maxFreeSockets: 256,
  scheduling: 'lifo',
  maxTotalSockets: Infinity,
  totalSocketCount: 2,
  maxCachedSessions: 100,
  _sessionCache: [Object],
  [Symbol(shapeMode)]: false,
  [Symbol(kCapture)]: false
},
socketPath: undefined,
method: 'GET',
maxHeaderSize: undefined,
insecureHTTPParser: undefined,
joinDuplicateHeaders: undefined,
path: '/quote/ltp?i=NSE:RELIANCE',
_ended: true,
res: IncomingMessage {
  _events: [Object],
  _readableState: [ReadableState],
  _maxListeners: undefined,
  socket: null,
  httpVersionMajor: 1,
  httpVersionMinor: 1,
```

```
  httpVersion: '1.1',
  complete: true,
  rawHeaders: [Array],
  rawTrailers: [],
  joinDuplicateHeaders: undefined,
  aborted: false,
  upgrade: false,
  url: '',
  method: null,
  statusCode: 403,
  statusMessage: 'Forbidden',
  client: [TLSSocket],
  _consuming: true,
  _dumped: false,
  req: [Circular *1],
  _eventsCount: 4,
  responseUrl: 'https://api.kite.trade/quote/ltp?i=NSE:RELIANCE',
  redirects: [],
  [Symbol(shapeMode)]: true,
  [Symbol(kCapture)]: false,
  [Symbol(kHeaders)]: [Object],
  [Symbol(kHeadersCount)]: 22,
  [Symbol(kTrailers)]: null,
  [Symbol(kTrailersCount)]: 0
},
aborted: false,
timeoutCb: null,
upgradeOrConnect: false,
parser: null,
maxHeadersCount: null,
reusedSocket: false,
host: 'api.kite.trade',
protocol: 'https:',
_redirectable: Writable {
  _events: [Object],
  _writableState: [WritableState],
  _maxListeners: undefined,
  _options: [Object],
  _ended: true,
  _ending: true,
  _redirectCount: 0,
  _redirects: [],
  _requestBodyLength: 0,
  _requestBodyBuffers: [],
  _eventsCount: 3,
  _onNativeResponse: [Function (anonymous)],
  _currentRequest: [Circular *1],
  _currentUrl: 'https://api.kite.trade/quote/ltp?i=NSE:RELIANCE',
```

```
  _timeout: null,
  [Symbol(shapeMode)]: true,
  [Symbol(kCapture)]: false
},
[Symbol(shapeMode)]: false,
[Symbol(kCapture)]: false,
[Symbol(kBytesWritten)]: 0,
[Symbol(kNeedDrain)]: false,
[Symbol(corked)]: 0,
[Symbol(kChunkedBuffer)]: [],
[Symbol(kChunkedLength)]: 0,
[Symbol(kSocket)]: TLSSocket {
  _tlsOptions: [Object],
  _secureEstablished: true,
  _securePending: false,
  _newSessionPending: false,
  _controlReleased: true,
  secureConnecting: false,
  _SNICallback: null,
  servername: 'api.kite.trade',
  alpnProtocol: false,
  authorized: true,
  authorizationError: null,
  encrypted: true,
  _events: [Object: null prototype],
  _eventsCount: 9,
  connecting: false,
  _hadError: false,
  _parent: null,
  _host: 'api.kite.trade',
  _closeAfterHandlingError: false,
  _readableState: [ReadableState],
  _writableState: [WritableState],
  allowHalfOpen: false,
  _maxListeners: undefined,
  _sockname: null,
  _pendingData: null,
  _pendingEncoding: '',
  server: undefined,
  _server: null,
  ssl: [TLSWrap],
  _requestCert: true,
  _rejectUnauthorized: true,
  timeout: 5000,
  parser: null,
  _httpMessage: null,
  autoSelectFamilyAttemptedAddresses: [Array],
  [Symbol(alpncallback)]: null,
```

```
    [Symbol(res)]: [TLSWrap],
    [Symbol(verified)]: true,
    [Symbol(pendingSession)]: null,
    [Symbol(async_id_symbol)]: -1,
    [Symbol(kHandle)]: [TLSWrap],
    [Symbol(lastWriteQueueSize)]: 0,
    [Symbol(timeout)]: Timeout {
      _idleTimeout: 5000,
      _idlePrev: [TimersList],
      _idleNext: [Timeout],
      _idleStart: 40850,
      _onTimeout: [Function: bound ],
      _timerArgs: undefined,
      _repeat: null,
      _destroyed: false,
      [Symbol(refed)]: false,
      [Symbol(kHasPrimitive)]: false,
      [Symbol(asyncId)]: 1617,
      [Symbol(triggerId)]: 1615,
      [Symbol(kAsyncContextFrame)]: undefined
    },
    [Symbol(kBuffer)]: null,
    [Symbol(kBufferCb)]: null,
    [Symbol(kBufferGen)]: null,
    [Symbol(shapeMode)]: true,
    [Symbol(kCapture)]: false,
    [Symbol(kSetNoDelay)]: false,
    [Symbol(kSetKeepAlive)]: true,
    [Symbol(kSetKeepAliveInitialDelay)]: 1,
    [Symbol(kBytesRead)]: 0,
    [Symbol(kBytesWritten)]: 0,
    [Symbol(connect-options)]: [Object]
  },
  [Symbol(kOutHeaders)]: [Object: null prototype] {
    accept: [Array],
    'x-kite-version': [Array],
    authorization: [Array],
    'user-agent': [Array],
    'accept-encoding': [Array],
    host: [Array]
  },
  [Symbol(errored)]: null,
  [Symbol(kHighWaterMark)]: 16384,
  [Symbol(kRejectNonStandardBodyWrites)]: false,
  [Symbol(kUniqueHeaders)]: null
},
response: {
  status: 403,
```

```
  statusText: 'Forbidden',
  headers: Object [AxiosHeaders] {
    date: 'Fri, 02 Jan 2026 07:58:49 GMT',
    'content-type': 'application/json',
    'transfer-encoding': 'chunked',
    connection: 'keep-alive',
    'cf-cache-status': 'DYNAMIC',
    'strict-transport-security': 'max-age=15552000; includeSubDomains',
    'set-cookie': [Array],
    server: 'cloudflare',
    'cf-ray': '9b78b6a9ffd63aef-BOM',
    'alt-svc': 'h3=":443"; ma=86400'
  },
  config: {
    transitional: [Object],
    adapter: [Array],
    transformRequest: [Array],
    transformResponse: [Array],
    timeout: 0,
    xsrfCookieName: 'XSRF-TOKEN',
    xsrfHeaderName: 'X-XSRF-TOKEN',
    maxContentLength: -1,
    maxBodyLength: -1,
    env: [Object],
    validateStatus: [Function: validateStatus],
    headers: [Object [AxiosHeaders]],
    params: [Object],
    method: 'get',
    url: 'https://api.kite.trade/quote/ltp',
    allowAbsoluteUrls: true,
    data: undefined
  },
  request: <ref *1> ClientRequest {
    _events: [Object: null prototype],
    _eventsCount: 7,
    _maxListeners: undefined,
    outputData: [],
    outputSize: 0,
    writable: true,
    destroyed: true,
    _last: true,
    chunkedEncoding: false,
    shouldKeepAlive: true,
    maxRequestsOnConnectionReached: false,
    _defaultKeepAlive: true,
    useChunkedEncodingByDefault: false,
    sendDate: false,
    _removedConnection: false,
```

_removedContLen: false,
_removedTE: false,
strictContentLength: false,
_contentLength: 0,
_hasBody: true,
_trailer: '',
finished: true,
_headerSent: true,
_closed: true,
_header: 'GET /quote/ltp?i=NSE:RELIANCE HTTP/1.1\r\n' +
  'Accept: application/json, text/plain, */*\r\n' +
  'X-Kite-Version: 3\r\n' +
  'Authorization: token ...:...\r\n' +
  'User-Agent: axios/1.13.2\r\n' +
  'Accept-Encoding: gzip, compress, deflate, br\r\n' +
  'Host: api.kite.trade\r\n' +
  'Connection: keep-alive\r\n' +
  '\r\n',
_keepAliveTimeout: 0,
_onPendingData: [Function: nop],
agent: [Agent],
socketPath: undefined,
method: 'GET',
maxHeaderSize: undefined,
insecureHTTPParser: undefined,
joinDuplicateHeaders: undefined,
path: '/quote/ltp?i=NSE:RELIANCE',
_ended: true,
res: [IncomingMessage],
aborted: false,
timeoutCb: null,
upgradeOrConnect: false,
parser: null,
maxHeadersCount: null,
reusedSocket: false,
host: 'api.kite.trade',
protocol: 'https:',
_redirectable: [Writable],
[Symbol(shapeMode)]: false,
[Symbol(kCapture)]: false,
[Symbol(kBytesWritten)]: 0,
[Symbol(kNeedDrain)]: false,
[Symbol(corked)]: 0,
[Symbol(kChunkedBuffer)]: [],
[Symbol(kChunkedLength)]: 0,
[Symbol(kSocket)]: [TLSSocket],
[Symbol(kOutHeaders)]: [Object: null prototype],
[Symbol(errored)]: null,

```
      [Symbol(kHighWaterMark)]: 16384,
      [Symbol(kRejectNonStandardBodyWrites)]: false,
      [Symbol(kUniqueHeaders)]: null
    },
    data: {
      status: 'error',
      message: 'Insufficient permission for that call.',
      data: null,
      error_type: 'PermissionException'
    }
  },
  status: 403
}
[STOCK_FETCH_PRICE_ERROR] {
  exchange: 'ZERODHA',
  userId: 'f295974c-14b6-4fe0-8799-56f769110c6d',
  symbol: 'RELIANCE',
  err: {
    code: 'AUTH_INVALID',
    message: 'Insufficient permission for that call.'
  }
}
```

Service Function:

```
export async function fetchZerodhaMarketPrice(params: {
  tradingSymbol: string; // e.g. INFY
  exchange?: "NSE" | "BSE";
  userId: string;
}) {
  try {
    const { tradingSymbol, userId } = params;
    const rawCredentials = await getStocksCredentials(
      userId,
      StocksExchange.ZERODHA
    );
    const credentials = Array.isArray(rawCredentials)
      ? rawCredentials[0]
      : rawCredentials;

    if (!credentials) {
      throw new Error("Credentiala not found");
    }


    const exchange = params.exchange ?? "NSE";

    if (!tradingSymbol) {
      throw new Error("Trading symbol must be provided");
```

```
  }

  const instrumentKey = ${exchange}:${tradingSymbol.toUpperCase()};

  const response = await axios.get(${ZERODHA_BASE_URL}/quote/ltp, {
    params: {
      i: instrumentKey,
    },
    headers: {
      "X-Kite-Version": "3",
      Authorization: token ${credentials.apiKey}:${credentials.accessToken},
    },
  });

  const data = response.data?.data?.[instrumentKey];

  if (!data || typeof data.last_price !== "number") {
    throw new Error(LTP not available for ${instrumentKey});
  }

  return data.last_price;
} catch (error: any) {
  console.error("[ZERODHA][MARKET_PRICE] Failed", {
    symbol: params.tradingSymbol,
    error: error?.response?.data || error.message,
  });
  handleZerodhaError(error);
}
```