Kite Connect, Uplink and Smart API: A Comprehensive Study of Python API Libraries

Somnath Hase[1] and Vikas T. Humbe[2]
[1]*Department of Computer Science, Smt. S. K. Gandhi Arts, Amolak Science and P. H. Gandhi Commerce College, Kada 414202, Maharashtra, India*
[2]*School of Technology, SRTM University, Sub Center Latur, Maharashtra, India*

Keywords: WebSocket, SmartAPI, Uplink, Kite Connect, API.

Abstract: The ability of machines to efficiently execute complicated and high-frequency trading strategies has made algorithmic trading, or "algo trading," an essential part of the financial markets. With a focus on three well-known platforms like SmartAPI, Uplink, and Kite Connect this research paper offers an in-depth study of Python APIs in the context of the Indian financial markets. The basics of algorithm trading are covered in the first section of the study, along with the value of Python as a programming language for creating algorithmic techniques. The selection of Kite Connect, Uplink, and SmartAPI was driven by their notable positions in the Indian financial scene, each providing traders and developers with special features and functionalities. Factors including order execution speed, accuracy of market data, and the variety of supported financial instruments are considered in this study. Case studies and real-world examples show how each API is used in algorithmic trading scenarios. The study additionally looks at each API's WebSocket streaming capabilities, which are essential for real-time data updates in the market.

1 INTRODUCTION

The financial system has shifted its paradigm in recent years due to the use of technology into trading activities. Algorithmic trading, or "algo trading," has become an effective tool that is changing the way the financial market function. This study explores the complex world of algorithmic trading, with a particular emphasis on its application utilizing Python API inside the framework. Algo trading is the process of automatically executing high-frequency trades using mathematical models and pre-established methods. Its ability to quickly assess market conditions, identify trading opportunities, and carry out orders at speeds faster than humans makes it attractive. As the Indian share market keeps developing and embracing new technology, algo trading strategies especially those that use Python APIs are becoming more and more popular. After approving the Direct-Market-Access (DMA) technology, the Securities Exchange Board of India (SEBI) approved Algo Trading in 2008 (S. Acharya and Dr. A. Ps 2022).

Artificial intelligence is a technology that can think and act for itself. As such, it is ideal for complicated trading applications where efficiency and speed are critical. Its use can alter trading in a variety of ways (Vignesh CK 2020), as is already clear. Many factors are responsible for the daily changes in the market, which makes it challenging for businesses and stockbrokers to choose where to trade (Bali 2021). Python's versatility, user-friendliness, and availability of libraries and frameworks make it a popular choice for algo trading. For traders and engineers looking to build advanced algorithms in the dynamic and complex environment of the Indian stock market, Python is a great option due to its readability and strong community involvement. With a focus on Python API integration complexities, this research study attempts to offer a thorough grasp of API libraries. It looks at the main benefits and characteristics of using Python for algorithmic trading, as well as the difficulties encountered and how they affect in trading procedures Algorithmic trading is a method of order execution. Using automated, pre-modified trading rules that represent variables like volume, cost, and time (M. Mathur et al., 2021). When compared to human brokers, this type of trading aims to take advantage of the speed and computational power of PCs.

Hase, S. and Humbe, V. T.
Kite Connect, Uplink and Smart API: A Comprehensive Study of Python API Libraries.
DOI: 10.5220/0013944200004919

In *Proceedings of the 1st International Conference on Research and Development in Information, Communication, and Computing Technologies (ICRDICCT'25 2025) - Volume 5*, pages 814-819
ISBN: 978-989-758-777-1

This study explores the complex world of Python API libraries, concentrating on the Python APIs offered by three major platforms: Kite Connect, Uplink, and SmartAPI. These APIs, which are provided by top Indian financial institutions, are essential in enabling algorithmic trading methods because they give developers the resources and connection they need to communicate programmatically with financial markets.

2 PYTHON API

2.1 Kite Connect from Zerodha

One of the top stockbrokers in India, Zerodha, offers Kite Connect, a well-liked trading API. Using Python and other computer languages, it enables developers to include stock trading capabilities into their own apps. An interface for easy interaction with the Kite Connects API is provided by the Kite Connect Python library.

2.1.1 Installation

You can install the Kite Connect Python library using pip:

```bash
pip install kiteconnect
```

2.1.2 Authentication

Your Zerodha API credentials are required in order to utilize the Kite Connect API. An access token can be generated to authenticate your API queries once you have the API key and secret. Methods for managing authentication are provided by the library.

```python
from kiteconnect import KiteConnect

kite = KiteConnect(api_key='your_api_key')
print(kite.login_url())

# After obtaining the request token, you can generate the access token
data = kite.generate_session('your_request_token', api_secret='your_ap
kite.set_access_token(data['access_token'])
```

2.1.3 Place Orders

You can place a variety of orders with the Kite Connect Python library, such as market, limit, and stop-loss orders.

```python
# Place a market order to buy 10 shares of RELIANCE
order_id = kite.place_order(tradingsymbol='RELIANCE',
                            exchange='NSE',
                            transaction_type='BUY',
                            quantity=10,
                            order_type='MARKET',
                            product='CNC')
```

2.1.4 Fetch Market Data

You can retrieve market data, including live market quotes, historical data, and more, using the Kite Connect API.

```python
# Fetch live market data for RELIANCE
quote = kite.ltp('NSE:RELIANCE')
print(quote)
```

2.1.5 Historical Data

Access historical market data for a certain financial instrument.

```python
# Fetch historical data for RELIANCE
historical_data = kite.historical_data('NSE:RELIANCE',
                                       from_date='2022-01-01',
                                       to_date='2022-01-10',
                                       interval='day')
print(historical_data)
```

2.1.6 WebSocket Streaming

WebSocket streaming is supported by Kite Connect to provide real-time data updates.

```python
from kiteconnect import WebSocket

kws = WebSocket(api_key='your_api_key', access_token='your_access_toke

def on_ticks(ws, ticks):
    print("Ticks: {}".format(ticks))

kws.on_ticks = on_ticks
kws.subscribe(['NSE:RELIANCE'])
kws.connect(threaded=True)
```

2.1.7 Account Information

Retrieve details on the user's trading account.

```python
# Fetch user's account information
user_profile = kite.profile()
print(user_profile)
```

2.2 Uplink from Upstox

Leading Indian stock brokerage Upstox has an official API called Uplink. With the help of the Uplink API, developers can incorporate Upstox trading features into their apps, allowing users place orders, get market data, and carry out a number of other programmatic tasks. The basics of the Upstox Uplink API is shown below:

2.2.1 Installation

Upstox uplink API can be installed by using following command.

```bash
pip install Uplink
```

2.2.2 Authentication

To use the Uplink API, developers need to obtain API credentials (API key and secret) from Upstox. These credentials are used to authenticate and authorize API requests. Once authenticated, developers can access various endpoints to interact with the Upstox platform.

```python
import uplink

class AuthHandler(uplink.auth.AuthHandler):

    def apply(self, request):
        # Add your authentication logic here
        request.headers['Authorization'] = 'Bearer YOUR_ACCESS_TOKEN'
        return request

@uplink.headers({"Content-Type": "application/json"})
class MyApi(uplink.Consumer):

    @uplink.get("/endpoint")
    @uplink.returns.json()
    def get_data(self):
        """Retrieve data from the API."""
```

2.2.3 Order Placement

The Uplink API allows developers to make a variety of orders, such as limit orders, stop-loss orders, market orders, and more. Users may change attributes including instrument, quantity, order type, and validity details of order.

```python
# Example: Place a market order to buy 10 shares of RELIANCE
order_details = {
    'symbol': 'RELIANCE',
    'quantity': 10,
    'transaction_type': 'BUY',
    'order_type': 'MARKET',
    'product': 'CNC'
}
response = upstox.place_order(order_details)
```

2.2.4 Market Data

Developers can access historical data, live quotes, and market depth in real-time via the Uplink API. Making customized charts, examining patterns, and deciding on trades with knowledge can all benefit from this.

```python
# Example: Fetch real-time market data for RELIANCE
market_data = upstox.get_live_data('NSE:RELIANCE')
```

2.2.5 Historical Data

Developers can use the API to fetch historical market data for backtesting and analysis purposes. Historical data can be retrieved in different time intervals, such as daily, hourly, or minute-wise.

```python
# Example: Fetch historical data for RELIANCE
historical_data = upstox.get_historical_data('NSE:RELIANCE', '2022-01-
```

2.2.6 WebSocket Streaming

WebSocket streaming is enabled by Uplink API so that real-time updates can be received. Through the WebSocket connection, developers can subscribe to specific tools and get real-time market data, order updates, and more.

```python
# Example: WebSocket streaming for real-time market data
upstox.subscribe(['NSE:RELIANCE'])
```

2.2.7 Account Information

Developers may obtain account-related data via the API, such as positions, margin details, and user profile details.

```python
# Example: Fetch user's account information
account_info = upstox.get_profile()
```

2.3 SmartAPI

One of the well-known stockbrokers in India, Angel One (previously known as Angel Broking), offers an official API (Application Programming Interface) called SmartAPI. With the help of SmartAPI, developers can integrate stock trading features into their apps, allowing users place orders, get market data, and carry out a number of other programmed activities. The basics of Angel One's SmartAPI is shown below.

2.3.1 Installation

The following command can be used to install AngelOne's SmartAPI.

```bash
pip install smartapi
```

2.3.2 Authentication

Angel One provides developers with API credentials (API key and secret) in order for them to use the SmartAPI. These login credentials are required for API request authorization and authentication. Once developers have registered for API access, they can generate API keys via the Angel One developer site.

```python
import requests

# Replace 'YOUR_API_KEY' with your actual API key or token
api_key = 'YOUR_API_KEY'
url = 'https://api.example.com/endpoint'

# Include the API key in the headers
headers = {'Authorization': f'Bearer {api_key}'}

# Make a sample GET request
response = requests.get(url, headers=headers)
```

2.3.3 Order Placement

Developers can place various orders using SmartAPI, such as limit orders, stop-loss orders, market orders, and more. Order attributes including symbol, amount, order type, validity, and product type are configurable by developers.

```python
# Example: Place a market order to buy 10 shares of RELIANCE
order_details = {
    'variety': 'NORMAL',
    'tradingsymbol': 'RELIANCE',
    'transaction_type': 'BUY',
    'quantity': 10,
    'exchange': 'NSE',
    'order_type': 'MARKET',
    'product': 'MIS'
}
response = smartapi.place_order(order_details)
```

2.3.4 Market Data

With SmartAPI, developers may obtain up-to-date market data. This covers historical data, market depth, and real-time quotes. For the purpose of building personalized charts, identifying trends, and making wise trading decisions, access to market data is essential.

```python
# Example: Fetch real-time market data for RELIANCE
market_data = smartapi.get_ltp('NSE', 'RELIANCE')
```

2.3.5 Historical Data

Developers can retrieve historical market data for analysis and backtesting using SmartAPI. One can obtain historical data at several time periods, including hourly, minute, and daily.

```python
# Example: Fetch historical data for RELIANCE
historical_data = smartapi.get_historical_data('NSE', 'RELIANCE', '202
```

2.3.6 WebSocket Streaming

For real-time updates, WebSocket streaming is supported by SmartAPI. Through the WebSocket connection, developers can subscribe to particular instruments and get real-time market data, order updates, and more.

```python
# Example: WebSocket streaming for real-time market data
smartapi.subscribe(['NSE:RELIANCE'])
```

2.3.7 Account Information

Developers can get details about accounts, such as positions, margins, and user profiles, through the API.

```python
# Example: Fetch user's account information
account_info = smartapi.get_profile()
```

3 CONCLUSIONS

The analysis presented in this research has shed light on the distinctive features and functionalities offered by each API, enabling a nuanced understanding of their strengths and limitations.

SmartAPI, Uplink, and Kite Connect represent key players in shaping the future of algorithmic trading in the Indian financial markets. Their distinctive attributes cater to a spectrum of trading needs, providing traders and developers with the tools necessary to navigate the complexities of algorithmic strategies. As we stand at the intersection of technology and finance, these APIs serve as catalysts for innovation, empowering market participants to unlock new possibilities and chart the course for a future where algorithmic trading seamlessly integrates with the heartbeat of Indian financial markets.

REFERENCES

"Kite Connect 3 / API documentation." https://kite.trade/docs/connect/v3/
"SmartAPI." https://smartapi.angelbroking.com/docs

"Trader API - Free Stock Market Trading API with Documentation @Upstox," Upstox - Online Stock and Share Trading, Jul. 25, 2023. https://upstox.com/uplink/trader-api/

A. Bali, "Development of Trading Bot for Stock Prediction Using Evolution Strategy," Sep. 30, 2021. https://easychair.org/publications/preprint/Xrlc

E. P. Chan, Quantitative Trading. 2009. [Online]. Available: http://books.google.ie/books?id=4kJ8tAEACAAJ&dq=Quantitative+Trading&hl=&cd=1&source=gbs_api

M. Mathur, S. Mhadalekar, S. Mhatre, and V. Mane, "Algorithmic Trading Bot," ITM Web of Conferences, vol. 40, p. 03041, 2021, doi: 10.1051/itmconf/20214003041.

S. Acharya and Dr. A. Ps, "Algorithmic Trading-Changing The Paradigm of Stock Trading in The Indian Capital Market," ResearchGate, Nov. 26, 2022.

S. Hase and V. Humbe, "Python-Powered ETF Trading: Unleashing Algorithmic Trading Strategies," in 2024 3rd Edition of IEEE Delhi Section Flagship Conference (DELCON), 2024, pp. 1–4. doi: 10.1109/DELCON64804.2024.10866662.

Vignesh CK, "APPLYING MACHINE LEARNING MODELS IN STOCK MARKET PREDICTION," EPRA International Journal of Research & Development (IJRD), pp. 395–398, Apr. 2020, doi: 10.36713/epra4361.

Y. Hilpisch, Python for Algorithmic Trading. O'Reilly Media, 2020. [Online]. Available: http://books.google.ie/books?id=g5IIEAAAQBAJ&printsec=frontcover&dq=trading+using+python&hl=&cd=1&source=gbs_api MichaelL. Halls_Moore, Successful Algorithmic Trading.