Python Development Internship: BreakoutAI

This assessment involves working with options trading data in the context of Indian financial markets. The task will require creating two functions in Python, integrating with an options chain API, and calculating specific values related to options trading. Below are detailed instructions, definitions, examples, and encouragement to leverage AI tools for research and code generation.

Internship Details:

Stipend: €200-400 per month, based on experience and performance
Duration: 3 months
Location: Remote

Additional Note for Applicants

Prior knowledge of stock markets is not required for this assessment. The task is intended to be approached from a purely data-centric perspective, focusing on understanding data structures, API integration, and performing calculations on structured data.

Your objective is to develop functions that handle and process data as defined in the instructions. Financial terms and concepts have been simplified, and any deeper knowledge needed can be acquired through research or AI assistance. Please leverage AI tools to explore any unfamiliar terms or concepts, as understanding this context will enhance the accuracy of your solution.

This approach allows you to complete the task by focusing on data processing skills without the need to be an expert in finance or options trading.

Note on API Selection:

Over 10 brokers in India offer free API access for market data, options chains, and margin calculations. You are encouraged to explore and choose any API that best suits your workflow, such as Upstox, Fyers, Zerodha, or others. Each API may have slight variations in response format or authentication process, but all provide the essential data needed for this task.

Feel free to experiment with different APIs, and if needed, use AI tools to compare documentation and quickly understand the specifics of each broker's API. Select the one that aligns with your familiarity, access ease, or preferred data structure.

Prerequisites and Background Information

1. **Options Trading**:
    - An option is a financial contract giving the buyer the right, but not the obligation, to buy or sell an asset at a specified strike price before a specified date.
    - **Option Types**:
        - **CE (Call Option)**: Gives the buyer the right to buy the underlying asset.

- **PE (Put Option)**: Gives the buyer the right to sell the underlying asset.

2. **Strike Price**:
   - This is the price at which an option can be exercised. For example, if you buy a call option with a strike price of 19500, you have the right to buy at 19500 even if the market price is higher.
3. **Expiry Date**:
   - Options have a specific date when they expire. This is the last day the option can be exercised.
4. **Bid and Ask Prices**:
   - **Bid Price**: The highest price a buyer is willing to pay.
   - **Ask Price**: The lowest price a seller is willing to accept.
   - **Highest Bid/Ask Price**: The goal of this assessment is to retrieve either the highest bid price for put options (PE) or the highest ask price for call options (CE).
5. **Option Selling**:
   - Selling options means that you take on the obligation to buy (for a put) or sell (for a call) at the strike price if the buyer exercises the option.
   - **Margin Requirements**: Brokers require a margin when selling options, as it involves higher risk. APIs like Upstox or Fyers can calculate these margins.
6. **Premium Earned**:
   - When you sell an option, you earn a premium. This premium is calculated by multiplying the bid or ask price by the lot size (the number of units in one contract).

Step-by-Step Assessment

Part 1: Retrieve Option Chain Data

Objective: Create a function that fetches the option chain data for a given instrument, such as NIFTY or BANKNIFTY, on a specified expiry date. The function should return the highest bid price for put options (PE) or the highest ask price for call options (CE) for each strike price.

Function:
`def get_option_chain_data(instrument_name: str, expiry_date: str, side: str) -> pd.DataFrame:`

1. **Inputs**:
   - `instrument_name`: Name of the instrument (e.g., NIFTY or BANKNIFTY).
   - `expiry_date`: The expiration date of the options, in `YYYY-MM-DD` format.
   - `side`: Type of option to retrieve. Use "PE" for Put and "CE" for Call.

2. **Function Logic**:
    - Retrieve the option chain data from the broker's API.
    - For each strike price, check the side variable:
        - If side == "PE", select the highest bid_price.
        - If side == "CE", select the highest ask_price.
    - Organize this data into a DataFrame with columns:
        - instrument_name, strike_price, side, and bid/ask.

Sample Output:

instrument_name | strike_price | side | bid/ask

NIFTY | 19500 | PE | 0.65

NIFTY | 19500 | CE | 2302.25

Part 2: Calculate Margin and Premium Earned

Objective: Create a second function that takes the DataFrame from Part 1 and calculates two additional columns: margin_required and premium_earned.

Function:
def calculate_margin_and_premium(data: pd.DataFrame) -> pd.DataFrame:

1. **Inputs**:
    - data: The DataFrame returned by get_option_chain_data.
2. **Function Logic**:
    - **Margin Calculation**:
        - For each row (representing an option contract), request the margin requirement from the API.
        - The API will calculate the margin based on the transaction type "Sell".
    - **Premium Calculation**:
        - Multiply the bid/ask price by the lot size for each option to get the premium earned.
3. **Output**:
    - Return the modified DataFrame with new columns:
        - margin_required and premium_earned.

Sample Output:

instrument_name | strike_price | side | bid/ask | margin_required | premium_earned

---

NIFTY | 19500 | PE | 0.65 | 1950 | 780

Instructions and Guidance

1. **Choosing and Using an API**:
    - Options chain and margin calculation data are available through APIs like Upstox or Fyers. Upstox's documentation for retrieving option chain data and margin calculation API endpoints is linked here.
    - Follow the API documentation to obtain an API key and understand the request and response formats.
2. **AI Assistance**:
    - **Encouragement to Use AI**: We encourage using AI tools like ChatGPT or Copilot to generate code snippets, clarify stock market concepts, and debug any issues. Include notes in your final documentation about how AI was used in each part of the task, such as generating sample code, researching API usage, or understanding financial terminology.
    - **Documentation**: Note each AI-assisted step, such as code generation, syntax help, or error debugging. For example:
        - "Used ChatGPT to generate the function skeleton for API data retrieval."
        - "Asked AI to explain options margin requirements and adapted it for this script."
3. **Testing and Validation**:
    - **Data Validation**: Test your functions with sample data or sandbox environments provided by the API.
    - **Error Handling**: Implement error handling for API responses, missing data, or incorrect inputs.
4. **Documentation**:
    - Include a brief but comprehensive explanation of how each function operates, assumptions made, and any specific API information.
    - Provide examples of API responses and show how the function processes this data to achieve the desired output.
5. **Submission**:
    - Submit your Python code, along with a README file or documentation explaining the structure and logic of your functions.

- Include comments explaining how AI tools were used throughout your code and any other resources consulted.

Examples and Additional Tips

- **Example API Response**: Refer to the Upstox API sample provided in the instructions.
- **Data Formatting**: Ensure all output DataFrames are cleanly formatted and columns are correctly labeled for easy readability.

Submission Guidelines

Please follow these guidelines to ensure a smooth and complete submission process:

1. **Deadline**:
   - All submissions are due by **5th November 2024**. Late submissions may not be considered.
2. **Submission Components**:
   - **Python Files**: Include all Python scripts used to complete the task.
   - **Documentation**: A short README or documentation file explaining your code structure, logic, and any specific details about your approach.
     - Be sure to mention any AI tools used, including how they assisted you in the project.
   - **Loom Video**: Record a short video (3-5 minutes) using Loom. This should include:
     - A brief introduction about yourself.
     - A walkthrough of your project, explaining the functions, logic, and any challenges you encountered.
     - Any insights or unique approaches you applied.
3. **Submission Process**:
   - **Email**: Send all files and a link to your Loom video to **kapil@breakoutinvesting.in**.
   - Ensure all attached files and links are accessible and correctly named.
4. **Stay Connected**:
   - Follow **Kapil Mittal** on LinkedIn for updates and opportunities: LinkedIn Profile.
   - Join the Telegram channel for updates in trading and investment: Telegram @breakoutinvesting.
   - Check out other trading insights and strategies on TradingView: TradingView Profile.
   - Visit the Breakout AI website for more about what we do: breakoutai.tech.