Photon-SOL Exchange API Research

Overview

Photon-SOL (https://photon-sol.tinyastro.io/) is a high-speed trading platform focused on token discovery and rapid transaction execution on the Solana blockchain. The platform emphasizes speed advantages, real-time monitoring, and token filtering, catering to traders seeking quick and efficient crypto transactions.

This research document outlines the key aspects of the Photon-SOL API for integration into our existing Telegram crypto trading bot.

1. Authentication Methods and Security Requirements

Based on the available documentation, Photon-SOL's API authentication follows these patterns:

- **API Keys**: While not explicitly detailed in public documentation, typical REST API practices suggest the use of API keys or tokens for secure access.
- **Basic Authentication**: For certain endpoints, especially those related to key management, Basic Authentication with base64-encoded credentials is used.
- **Wallet Integration**: The platform requires users to connect a Solana-compatible wallet (primarily Phantom) to initiate trading. Users generate a Photon-specific wallet linked to their Solana wallet.
- **Security Measures**:
  - Rate limiting to prevent brute-force attacks
  - Encrypted key storage and management
  - Verification steps for user and PIN resets
  - The platform only discloses the private key once during wallet creation

2. Available Endpoints for Trading Functionality

The Photon-SOL API provides several endpoints for trading operations:

Core Trading Endpoints

- **Wallet Connection and Management**:
  - Endpoints for connecting and managing Solana-compatible wallets
  - Deposit and withdrawal management
- **Token Discovery and Filtering**:
  - Real-time token discovery with advanced filtering options
  - Filters based on liquidity, market cap, transaction volume, social presence, and audit status

- **Buy and Sell Operations**:
  - Quick Buy: Endpoints to select a token, specify amount, set slippage and priority fee
  - Quick Sell: Similar to buying, with options to sell a specified percentage or all holdings
- **Channel Management**:
  - `PUT /api/1/channels`: Opens new channels with specified partner, token, and balance
  - `GET /api/1/channels`: Lists all unsettled channels
  - `GET /api/1/channels/{channel_identifier}`: Gets detailed info on specific channels
  - `PUT /api/1/withdraw/{channel_identifier}`: Facilitates cooperative withdrawal
- **Token Operations**:
  - `GET /api/1/tokens`: Lists registered tokens
  - `PUT /api/1/tokens/{token_address}`: Registers new tokens
  - `GET /api/1/tokens/{token_address}/partners`: Retrieves token partners
- **Transfer Operations**:
  - `POST /api/1/transfers/{token_address}/{target_address}`: Initiates token transfers
  - `POST /api/1/registersecret`: Registers secrets for secure transfers
  - `GET /api/1/transferstatus/{token_address}/{locksecrethash}`: Checks transfer status

Additional Endpoints

- **Node Information**:
  - `GET /api/1/address`: Returns the node's address
- **Fee Management**:
  - `GET /api/1/fee_policy`: Retrieves fee policy information
  - `POST /api/1/fee_policy`: Updates fee policy settings

3. Rate Limits and Other Constraints

While specific rate limit information is not explicitly documented, the following constraints should be considered:

- **Transaction Speed**: The platform emphasizes sub-second transaction execution, leveraging Solana's high throughput.
- **API Call Limitations**: Standard REST API practices suggest there may be rate limits on API calls to prevent abuse.
- **Error Handling**: The API uses error codes to indicate success or specific issues:
  - `0`: Success

- `-1`: Unknown error
- `1002`: Insufficient balance
- `1007`: Invalid state
- `1008`: Transfer when channel is closed
- `1011`: Invalid nonce
- `1020`: Transaction timeout
- `2000`: Insufficient gas balance
- `2001`: Error during channel closure

4. Methods for Implementing Required Features

Price Alerts

To implement price alerts, we can: - Use the token discovery endpoints to monitor token prices in real-time - Set up a polling mechanism to check prices at regular intervals - Trigger alerts when price conditions are met

Automated Trading

For automated trading functionality: 1. Connect to user's wallet via the wallet integration endpoints 2. Use token discovery endpoints to identify trading opportunities 3. Execute trades using the buy/sell endpoints with preset parameters 4. Monitor transactions via the transfer status endpoints

Wallet Integration

Wallet integration can be implemented by: - Using the wallet connection endpoints to link users' Solana wallets - Managing deposits and withdrawals through the platform's interface - Storing wallet addresses securely in our bot's database

Limit Orders

For limit orders: - Monitor token prices continuously - Execute buy/sell operations when price conditions are met - Use the transfer operations endpoints to complete transactions

Meme Coin Sniping

Meme coin sniping can be implemented by: - Using token discovery with filters for new tokens - Setting up automated buying based on predefined criteria - Implementing quick sell functionality for profit-taking

5. Documentation and SDK Availability

Based on the research, the following resources are available:

- **API Documentation**: The primary API documentation is available at PhotonNetwork's documentation site

- **GitHub Repositories**:
  - helius-labs/photon: Maintained by Helius Labs, focuses on indexing and data management
  - fccview/photon-token-analisys: Provides tools for analyzing trading patterns
- **SDK Support**: While there's no explicit mention of a dedicated SDK for Photon-SOL, the API is RESTful and can be integrated using standard HTTP clients in various programming languages.

6. Specific Considerations for Solana-based Trading

When integrating Photon-SOL for Solana-based trading, consider the following:

- **Transaction Speed**: Solana's high throughput enables near-instant transaction execution, which is crucial for time-sensitive operations like meme coin sniping.
- **Gas Fees**: Solana's low transaction fees make it ideal for high-frequency trading, but our bot should still monitor and manage gas costs.
- **Wallet Compatibility**: Ensure compatibility with Solana wallets, particularly Phantom, which is the primary wallet supported by Photon-SOL.
- **Solana RPC Nodes**: Consider using optimized RPC nodes to reduce latency, as Photon-SOL emphasizes speed advantages with reports of achieving response times faster than standard Solana RPC calls.
- **Security Measures**: Implement additional security for Solana-specific vulnerabilities, particularly around wallet private keys and transaction signing.
- **Token Standards**: Ensure compatibility with Solana's SPL token standard for all token operations.

Conclusion

Photon-SOL offers a comprehensive API for high-speed trading on the Solana blockchain, with features that align well with our Telegram bot's requirements. The platform's emphasis on speed, security, and token filtering makes it a suitable addition to our existing exchange integrations.

For implementation, we should focus on: 1. Establishing secure authentication and wallet integration 2. Leveraging the token discovery and filtering capabilities 3. Implementing the buy/sell endpoints for automated trading 4. Setting up monitoring for price alerts and limit orders 5. Optimizing for Solana's high-speed, low-fee environment

Additional testing will be required to determine exact rate limits and performance characteristics, but the available documentation provides a solid foundation for

integration.

References

1. Photon's API Documentation
2. Photon - Your Trusted Platform for Token Discovery & Trading
3. GitHub - helius-labs/photon
4. GitHub - fccview/photon-token-analisys
5. Photon SOL Trading Platform Guide