# **Main Architecture and System Flow of HilalBot**

| Figure 01: HilalBot System Architecture and Flow Diagram |
| --- |
|  |

A workflow and system architecture for HilalBot, and additional considerations:

**1. User Input:**

* User inputs protocol/token name or ticker via a user interface (UI).
* UI validates input for format and potential errors.

**2. URL Retrieval:**

* System calls a search function to locate URLs for:
  + Official website
  + Official docs (Gitbook, whitepaper, etc.)
  + CoinMarketCap
  + CoinGecko
  + Other relevant data sources
* It verifies URLs for validity and accessibility.
* It stores retrieved URLs in a structured format (e.g., list or dictionary).

**3. Web Scraping with BeautifulSoup:**

* System iterates through each retrieved URL.
* For each URL:
  + Downloads HTML content using a library like Requests.
  + Passes HTML content to BeautifulSoup for parsing.
  + Extracts relevant text data based on defined criteria.
  + Stores extracted text in a temporary format (e.g., list or JSON).

**4. Data Pre-Processing:**

* System consolidates extracted text from multiple sources.
* It cleans and normalizes text:
  + Removes irrelevant elements (HTML tags, scripts, formatting).
  + Applies text normalization techniques (lowercase, tokenization, lemmatization/stemming).
  + Filters out unwanted content (ads, navigation elements).

**5. Feature Extraction with HilalBot:**

* System feeds pre-processed text to HilalBot.
* HilalBot leverages LangChain and agent to:
  + Analyze text for key features relevant to Shariah compliance.
  + Extract information about:
    - Token/protocol purpose and utility
    - Economic model and incentives
    - Governance structure
    - Use of interest-bearing mechanisms
    - Involvement in prohibited activities
    - Other relevant Shariah criteria

**6. Report Generation:**

* HilalBot combines extracted features with knowledge from its training data.
* It generates a comprehensive report containing:
  + Summary of key features
  + Shariah compliance assessment (compliant, non-compliant, uncertain)
  + Detailed explanations for compliance rulings
  + Recommendations or insights based on analysis
* Report is presented to the user in a clear and informative format (User Interface).

**7. Shariah Scholar Review:**

* The generated report is first sent to a Shariah Scholar.
* The Scholar reviews the report for accuracy, correctness, and compliance with Shariah principles.
* If approved, the report is then released to the user.
* If inaccuracies or errors are found, the Scholar makes necessary corrections.

**8. Report Presentation and Learning Feedback Loop:**

* The finalized report, either directly approved or corrected by the Shariah Scholar, is presented to the user.
* The same report (especially any corrected version) is stored in the database of reviewed token reports.
* HilalBot accesses this database to learn from these reports, enhancing its future analyses and reports.

**Additional Considerations:**

* Error Handling: Implement robust error handling mechanisms for URL retrieval, web scraping, and data processing.
* Data Storage: Consider storing extracted data and reports for future reference or analysis.
* Training Data: Ensure HilalBot's training data aligns with Islamic financial principles and is regularly updated.
* Feedback and Improvement: Incorporate user feedback and expert reviews to refine HilalBot's accuracy and Shariah compliance assessment.

# **Retrieving Accurate URLs**

Here's a system flow and process sequence for retrieving accurate URLs for cryptos using a combination of API integration, search engine APIs, refined web searches, community-sourced databases, and user feedback:

**1. User Input:**

* User enters protocol/token name or ticker via the User Interface (UI).
* UI performs initial validation for format correctness and potential input errors.

**2. Initial API Query:**

* System queries established cryptocurrency data platforms like CoinMarketCap and CoinGecko using the provided token name or ticker.
* System checks for URL data (official website, whitepaper, etc.) in the API response.

**3. API Response Analysis:**

* If URLs are found in API responses, proceed to URL Validation.
* If URLs are not found, proceed to Search Engine API Query.

**4. Search Engine API Query with Custom Logic:**

* System uses search engine APIs (like Google Custom Search) with specific query terms (e.g., "Ethereum official website" or "Bitcoin whitepaper").
* Custom logic is applied to prioritize official websites and filter out irrelevant or sponsored results.

**5. Search Engine API Response Analysis:**

* Analyze search engine API results for relevancy.
* If relevant URLs are found, proceed to URL Validation.
* If relevant URLs are not found, proceed to Refined Web Search.

**6. Refined Web Search:**

* System conducts a manual web search using refined query parameters and advanced search operators to find the correct URLs.
* Leverage community-sourced databases or blogs (e.g., Medium articles) for additional information.

**7. URL Validation:**

* Cross-verify URLs from different sources (APIs, search engine API, web search) for authenticity.
* Implement a validation step to confirm the accuracy and relevancy of URLs.

**8. User Feedback Integration:**

* Allow users to flag incorrect URLs and suggest correct ones.
* Implement a review mechanism for user-submitted URLs.

**9. Final URL Output:**

* Present the validated URLs to the user or proceed with further processing (like web scraping for HilalBot analysis).

**10. Continuous Improvement:**

* Continuously refine search queries, update API integrations, and incorporate user feedback for system improvement.