**Explanation of the our Approach**

The script is designed to automate the analysis of textual data extracted from web pages. Here’s how I approached the solution:

**1. Understanding the Requirements**

* The objective was to process a list of URLs, extract their textual content, and compute various text metrics for analysis.
* The results had to be structured in an Excel file for easy interpretation.

**2. Designing the Solution**

* **Input and Output**: Use Excel files for structured input (Input.xlsx) and output (Output\_Data\_Structure.xlsx).
* **Text Processing**: Implemented preprocessing and analytical functions to compute text statistics.
* **Modularity**: Each functionality (text extraction, preprocessing, metric calculation) is encapsulated in separate functions for reusability and clarity.

**3. Key Steps in the Script**

1. **Dependencies and Data Loading**:
   * Loaded positive and negative word lists for sentiment analysis.
   * Loaded stop word lists to clean text.
2. **Text Extraction**:
   * Extracted HTML content using requests and parsed it with BeautifulSoup to retrieve title and paragraph texts.
3. **Text Preprocessing**:
   * Removed stop words using a custom function.
   * Converted the text to lowercase and tokenized it into words.
4. **Text Metrics**:
   * Calculated metrics like polarity, subjectivity, average word length, and Fog Index using built-in libraries (textstat) and custom logic.
5. **Result Compilation**:
   * Appended results for each URL into a list and saved them to an Excel file using pandas.
6. **Error Handling**:
   * Used try-except blocks to gracefully handle exceptions during URL processing or file access.

**How to Run the Script**

1. **Install Required Dependencies** Ensure the following Python libraries are installed:

pip install pandas openpyxl requests beautifulsoup4 textstat

1. **Set Up the File Structure**
   * Create the following directories in the script's working directory:
     + MasterDictionary: Place positive-words.txt and negative-words.txt here.
     + StopWords: Include files containing stop words (e.g., .txt files).
2. **Prepare Input File**
   * The input Excel file (Input.xlsx) should have the following columns:
     + URL\_ID: A unique identifier for each URL.
     + URL: The URL to analyze.
3. **Run the Script**
   * Save the script as main.py.
   * Execute the script:

python main.py

1. **View the Results**
   * The extracted text files will be saved in an articles directory.
   * The analysis results will be saved in Output\_Data\_Structure.xlsx.

**Dependencies Required**

Here’s the complete list of Python packages and their purpose:

* **pandas**: For reading and writing Excel files and data manipulation.
* **openpyxl**: Backend for handling Excel files with pandas.
* **requests**: For fetching HTML content from URLs.
* **beautifulsoup4**: For parsing HTML and extracting text.
* **textstat**: For calculating readability and text complexity metrics.
* **re**: For regular expressions in text processing.
* **os**: For file and directory management.