Project Overview: Efficient and Robust Web Scraping Script

Objective: Develop a highly efficient and powerful web scraping script utilizing Scrapy, incorporating asynchrony, proxy management, logging, and retry logic. The script will process approximately 3,900 eBay links provided in an input.json file, extract the HTML body of each link, and save it in a specified format. Detailed logging of the process is also required.

Functionality Requirements

A) Making Requests:

1. Asynchronous Requests:
    - Request a chunk of links asynchronously.
    - Implement sleep intervals between chunks to avoid blocking.

2. Retry Logic:
    - Retry on failure with the following status codes: [500, 503, 504, 400, 429, 408].
    - Implement exponential backoff for retry attempts, starting with a base sleep time and increasing with each retry.

3. Proxy Management:
    - Initialization:
        - Load a list of proxies from a proxy.txt file.
    - Proxy Assignment:
        - Distribute proxies evenly across all requests in a chunk.
        - Assign a proxy to each request dynamically from the list of available proxies.
    - Proxy Monitoring and Exclusion:
        - Implement a monitoring system to track the status of each proxy.
        - Exclude a proxy from the pool if it fails multiple requests due to blocking or other errors.
        - Maintain a log of excluded proxies and the reasons for their exclusion.
    - Proxy Rotation:
        - Rotate proxies to avoid overloading a single proxy.
        - Use a random selection method for proxy assignment to ensure fair usage.
    - Retry with New Proxy:
        - On request failure due to proxy issues, retry the request with a different proxy.

- Implement logic to ensure the same proxy is not reused for a failed request.

B) Saving Files:

1. File Formats:
    - Save output in one of the following formats: json, jsonlines, or parquet.
    - The format will be specified in the configuration file.

2. Asynchronous Saving:
    - Save data after processing each chunk to avoid large, time-consuming writes at the end.
    - Perform file saving asynchronously to avoid blocking the main scraping process.

3. Data Structure:
    - Follow the structure provided in the output.json file for consistency.

C) Logging:

1. Comprehensive Logging:
    - Log both to a file and print to the console during execution.
    - Include the following details in each log entry: [Timestamp, Input URL, Proxy Used, Returned Status Code, Comment].
    - Log a message when a proxy is excluded.
    - Comment: "Proxy [proxy] excluded due to failure"

2. Log Comments:
    - On success (status 200), log "success".
    - On failure, log "retrying_num" for each retry attempt.
    - If all retries fail, log "failed after all retrying".

D) Configuration File:

- Control various script settings and variables through a configuration file:
    - Chunk Size (e.g., 100)
    - Sleep time between chunks (e.g., 2 seconds)
    - Retry times (e.g., 3)
    - Initial sleep for each retry
    - Proxy (true or false)
    - Output file type (json|jsonlines|parquet)