SIRIUS SOFTWARE

Back-end Challenge

News Web Crawler

V1.0

LAUNCHING YOU INTO THE FUTURE

www.sirius.com.ar
Parque Empresarial Austral (Av. Sgto. Cayetano Beliera 3025, Pilar). Edificio M3.
administracion@sirius.com.ar

Pilar, 21 de Octubre de 2024

News Web Crawler

Task Overview

This task involves developing a web crawler that generates concise summaries of news articles that users provide. The system will receive requests with news article URLs through a REST endpoint that you need to develop. It will fetch these pages, extract the main content of the articles, and generate summaries. You can use any method or tool you prefer to perform the summarization.

The initial source will only be articles from the BBC (https://www.bbc.com/news)

Technical Details

- **Scalability**: You can assume that the load of the requests is about 1/sec, but make sure that your design can support a load several orders of magnitude larger than that.
- **Other providers:** The system should be able to escalate to other news providers easily (Clarin, La Nacion, etc). If the User inputs a URL that is not from a supported news provider, the system should respond with an error message that the provider has not yet been implemented.
- **URL Handling**: URLs might repeat themselves, and as with any good engineering system, we would like to avoid fetching and processing the same page over and over again.
- **Content Extraction**: News articles come from various websites with different HTML structures. Your system should reliably extract the main content of the article, excluding navigation menus, ads, comments, and other non-essential elements.
- **Text Processing**:
  - **Summarization**: Implement an efficient way to generate a concise summary of the article's content. You can use any method or tool you prefer to perform the summarization.
- **Error Handling**:
  - If the site cannot access the news article because it requires an account or log in to view it, your service should return a custom error indicating that the article cannot be accessed.

- **Performance Optimization**: Assume the corpus can be very large, so naive methods may not be quick enough to render results promptly. You will need to address this through your choice of algorithms, data structures, or technologies.
- **Output Format**: The summaries should be returned in a JSON format. This is not a front-end development assignment, so you don't need to focus on making the output pretty.
- **Caching and Storage**: Implement caching to store processed articles to avoid redundant fetching and processing, thereby improving performance

The list of URLs will be provided through a set of CURL calls. Please use simulateRequests.sh script to simulate those requests while you develop.
For example, you can run it as follows:

```
./simulateRequests.sh localhost 8080 productUrl 1
```