AI NEWS Web Application

IF4: Roubin Jeanne, STRIEBIG Maximilien, THOMAS Nicolas, THOMASSIN Pablo

April 6, 2024

1 Introduction

This report provides an in-depth analysis of a Flask web application designed to interact with a chatbot, extract articles from Google News, and provide article summarization functionalities. The application is built using Python with libraries such as Flask, Selenium, BeautifulSoup, and requests.

2 Functionality Overview

The Flask application consists of several routes, each serving a specific purpose:

2.1 Scraping Google News

The application starts by scraping Google News for articles related to artificial intelligence. We chose Google News as a source because it provides a wide range of up-to-date articles on various topics. Scraping is done using the Requests library, which is simple and efficient for making HTTP requests. BeautifulSoup is used for parsing the HTML content of the Google News page, allowing us to extract relevant article links.

2.2 Storing Links in JSON

Once the links are extracted, we store them in a JSON file for future use. This approach ensures that we don't need to scrape Google News every time the application is run, reducing unnecessary network requests and improving performance. Storing data in JSON format is convenient and straightforward, allowing easy retrieval and manipulation of link data within the application.

2.3 Search Functionality with difflib

We implemented a search functionality using 'difflib' to enable users to find routes easily through a search bar. This allows users to input a query, and the application finds the closest match among predefined routes using difflib's function. Even with typos, users can still find the correct route, enhancing user experience and usability.

2.4 Sub-routes for Links

The application provides sub-routes for accessing different functionalities related to links:

- **/links**: This route displays all the links found about artificial intelligence. Users can browse through these links to access relevant articles.
- **/links/favourite**: This route displays the main article on the Google News page, which is considered as the favorite. It provides quick access to the most prominent article.
- **/links/favourite/text**: This route shows the content of the main article in text format. Users can read the content directly within the application.

2.5 Chat Functionality

The chat functionality allows users to interact with a chatbot implemented using Selenium. Selenium is a powerful tool for automating web browsers, and in this context, it is used to simulate user interactions with the Perplexity AI Labs website's chat interface. When a user sends a message through the chat interface on the Flask application, Selenium launches a headless browser in the background, navigates to the Perplexity AI Labs website, fills in the message, and retrieves the chatbot's response. This response is then displayed to the user within the Flask application.

2.6 Article Summarization

Users can summarize articles by providing a link to the article. The application extracts the article content using BeautifulSoup, which is well-suited for parsing HTML and extracting specific elements from web pages. Once the content is extracted, it is sent to the chatbot for summarization. We chose this approach because it leverages the capabilities of the chatbot to provide concise summaries, saving users time and effort in reading lengthy articles.

2.7 Enhanced Appearance with HTML and CSS

To improve the appearance and user experience of the Flask web application, we implemented small HTML and CSS enhancements. These enhancements include styling for better layout, font selection for readability, and color schemes for aesthetics. By enhancing the appearance of the application, we aim to create a more engaging and visually appealing user interface.

2.8 Error Handling

The application handles errors gracefully to provide a smooth user experience. For example, if a requested article is not found or inaccessible, appropriate error

messages are displayed to the user. Error handling is crucial for ensuring that users are informed of any issues and can take appropriate actions.

2.9 Python Package Requirements

The Flask web application has the following Python package requirements:

- Flask
- BeautifulSoup (pip install beautifulsoup4)
- requests
- Selenium (pip install selenium)
- difflib

3 Conclusion

The Flask web application effectively integrates chatbot interaction, article extraction, and summarization functionalities. It provides a user-friendly interface for users to interact with the chatbot and access summarized article content. The choice of technologies enables seamless automation and web scraping, contributing to the application's robustness and efficiency.