# Project Proposal

for

# Mirror News Summarizer

**Student Name:** Zijun Li

**Student ID:** 2272583

**Supervisor Name:** Jizheng Wan

**Project Category/Topic:** AI

# 1 Project Aim:

- **Objective**: To construct a "Mirror News Summarizer" system that enables users to swiftly access summarized content from targeted news websites. By mirroring the interface of the original news site and presenting concise summaries, the tool aims to improve the user experience by providing quick insights and reducing the time taken to consume full-length articles.
- **Significance**: With the advent of the digital age and the vast array of information available online, it's becoming increasingly challenging for users to sift through extensive content efficiently. Especially in the realm of news where staying updated is crucial, navigating through lengthy articles can be time-consuming. The "Mirror News Summarizer" strives to bridge this gap by allowing users to view compressed news articles that retain the core essence of the original. This not only ensures the efficient consumption of news but also reduces the cognitive load on the user.
- **Relevance to AI**: This system harnesses the capabilities of advanced Natural Language Processing algorithms and summarization APIs. Leveraging such cutting-edge technology, the project aims to transform the way users engage with news content online.

# 2 Literature Review:

-

# 3 Project Objectives/Deliverables

1. Web Scraping and Content Retrieval
    - Develop a web scraping mechanism to automatically fetch content from targeted news websites based on specific criteria or keywords using Beautifulsoap library in Python for each supported websites.
    - Ensure a robust and efficient system to handle varying webpage structures and unexpected changes in site designs.
2. Content Filtering and Cleanup
    - Implement algorithms to meticulously extract relevant sections of the fetched content, focusing on primary news elements.

- Design the system to identify and remove unwanted sections such as advertisements, sidebars, and non-essential footers to improve content clarity.

3. Summarization and Content Condensation
    - Utilize a state-of-the-art summary-generation API to condense the filtered content into bite-sized, easily digestible summaries while retaining key information
    - Ensure that the summarization process maintains the integrity and accuracy of the original content, providing users with a comprehensive understanding in fewer words.
4. Storage and Quick Retrieval System
    - Design a caching mechanism or database specifically tailored for storing summarized content.
    - Ensure quick retrieval capabilities, allowing users to access stored summaries instantaneously, improving the user experience and reducing wait times.
5. User Interface and Content Presentation
    - Develop a mock version interface that closely resembles the original news website's look and feel.
    - Ensure a seamless integration of the summarized content within this interface, enabling users to navigate and read news articles effortlessly, further enhancing user engagement and overall experience.

These objectives comprehensively address the primary goal of this project: to provide users with an efficient way to access and consume news articles from their favorite websites, without the distractions of unnecessary content. The initial objective guarantees that the system is equipped to fetch content autonomously, enabling a continuous flow of updated news. The following objectives focus on processing and optimizing this content, from extraction and filtering to summarization and storage. The final objective ensures that the users receive their content in a familiar and user-friendly interface, promoting engagement and satisfaction. Overall, this approach ensures users can quickly access the essence of news articles in a format that's convenient and recognizable.

# 4 methodologies

1. **Website Request from User**: Initiate by prompting the user to input or select the target news website or specific news URL they want summarized.

2. **Web Content Retrieval**: Use web scraping techniques to fetch the entire HTML content of the provided news website or URL.
3. **Content Filtering and Cleaning**: Extract the main content sections, filtering out non-essential components like advertisements, sidebars, and footers.
4. **Summarization Process**: Pass the cleaned content to a summarization API.
5. **Storage and Caching Mechanism**:
    - Check a caching database to see if the provided URL's content has already been summarized recently.
    - If it exists, fetch the summarized content from cache. Otherwise, store the new summarized content for future quick access.
6. **User Interface Presentation**: Render the summarized content on a mock version of the original news website's interface, providing a familiar user experience.
7. **Optional Content Personalization**: If desired, introduce features where users can adjust the length or style of summaries based on their preferences.
8. **Performance and Load Management**:
    - Regularly monitor website performance, especially during high traffic times.
    - Implement strategies for load balancing and optimizing web scraping speed.
9. **Security and Ethical Considerations**:
    - Ensure the web scraping process adheres to the terms of service of target news websites.
    - Implement security measures to prevent potential misuse or over-fetching of content from source websites.
10. **Evaluation and User Testing**:
    - Conduct user testing sessions to gauge the effectiveness, accuracy, and user satisfaction with the mirrored news summarizer.
    - Collect insights and iterate on the platform to address any identified concerns or areas for improvement.

# 5 Project Plan

- Week 1-2: Research and select the suitable tools for each component. Start with the implementation of the web crawler.

- Week 3-4: Implement content processing and summary generation functionalities.
- Week 5-6: Develop the caching mechanism or database setup.
- Week 7-8: Build the user interface resembling the target news website.
- Week 9-10: Perform testing, gather feedback, and make necessary refinements.

# 6 Risks and contingency plan

- Risk: Targeted websites implementing anti-crawling measures.

  Contingency: Implement rotating user-agents and IPs, or consider other data acquisition methods.

- Risk: Over-reliance on third-party summary-generation API.

  Contingency: Have backup APIs or explore open-source summary-generation algorithms.

- Risk: Database overload due to frequent read-writes.

  Contingency: Implement an optimal database management strategy, consider using database caching solutions like Redis.

# 7 Hardware/Software Resource

- Hardware: Standard development PC or laptop.
- Software:

  Development Environment: VS Code, PyCharm, or any suitable IDE.

  Web Crawling: Scrapy, BeautifulSoup.

  Backend: Flask, Django, or Node.js.

  Frontend: React, Vue.js, or Angular.

  Database: MySQL, MongoDB, or any other relevant DBMS.

  Others: Git for version control, Postman for API testing.