A Dynamic News Aggregation Platform with Personalization

Sania Khalid
University College of engineering and Bikaner Technical
Rajasthan, India
Khalidsania24@gmail.com

Jai Bhaskar
Assistant ProfessorUniversity College of engineering
andBikaner Technical university
Rajasthan, India
jai bhaskar@cet-gov.ac.in

Abstract: **- This paper presents the development and implementation of a dynamic news aggregation website that provides personalized news, text summarization, multilingual support, and text-to-speech features. The system utilizes the NewsAPI to fetch real-time news, processes the data using JavaScript, and enhances user experience through dark mode, language preferences, and voice-based interactions. The paper discusses the architecture, features, implementation challenges, testing results, and future enhancements for improving user engagement**

Keywords— News Aggregation, Personalization, Text-to-Speech, AI Summarization, Multilingual Support.

Introduction

With the exponential growth of online news sources, users face challenges in accessing relevant and credible information efficiently. Traditional news platforms often lack personalization and accessibility features, leading to information overload. This paper introduces a modern news website that integrates AI-driven summarization, voice-based interactions, and multilingual capabilities to enhance the news-reading experience. The system leverages APIs for real-time news retrieval and applies natural language processing (NLP) for text summarization.

FEATURES

A. Real-time News Updates

The system fetches and displays the latest news articles from various categories, ensuring users stay informed with up-to-date information. The system dynamically updates content to display the most recent headlines across multiple categories, keeping users informed of ongoing events.

B. Multilingual Support

To cater to a diverse audience, the system provides multilingual support, allowing users to choose their preferred language for reading news articles. This feature enhances accessibility and ensures that users from different linguistic backgrounds can engage with the content effectively

C. Text Summarization

To enhance readability and comprehension, the system includes an automatic text summarization feature. This functionality extracts key points from news articles and presents them in a concise bullet-point format. This allows users to quickly grasp the essence of an article without reading lengthy content, improving information retention and efficiency.

D. Text-to-Speech (TTS)

To improve user experience and reduce eye strain, especially in low-light environments, the system incorporates a dark mode option. When activated, the website switches to a darker theme, making it more comfortable for users to read news for extended periods. This feature also helps in conserving battery life on OLED and AMOLED screens.

E. Category-based Navigation

The website organizes news articles into multiple categories such as Technology, Sports, Politics, and more. Users can easily browse news articles based on their interests by selecting a specific category. This structured navigation enhances usability and ensures a seamless browsing experience

F. Search Functionality

A built-in search feature allows users to find news articles by entering relevant keywords. This functionality helps users access specific topics or past articles without manually browsing through multiple categories. The search algorithm ensures relevant and accurate results, improving content discoverability.

II. System Architecture and Methodology

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

A. Overview

The proposed news website is designed to provide users with real-time news updates while incorporating various features such as multilingual support, text summarization, and text-to-speech functionality. The system follows a client-server architecture, where the front end interacts with a back-end server that fetches news articles from an external API. The website is developed using modern web technologies to ensure a seamless user experience.

B. System Architecture

Frontend Interface: The user interface (UI) is developed using HTML, CSS, and JavaScript to ensure responsiveness and accessibility. The UI dynamically updates based on user

preferences, such as selected news categories and language settings.

Backend Server: The server-side logic is responsible for handling user authentication, API requests, and data processing. It acts as an intermediary between the front end and the external news API.

External News API: The system retrieves news articles from the **NewsAPI** service, which provides real-time news articles from various sources.

Database (for authentication): A secure database is integrated to manage user authentication details, including admin and user credentials. This allows personalized experiences based on user preferences.

Text Processing Module: This module processes the news content to extract and summarize key information. It removes redundant data, extracts bullet-point summaries, and ensures readability.

Text-to-Speech (TTS) Module: The TTS module converts textual news content into speech, allowing users to listen to articles instead of reading them.

C. Workflow of the system

a) User request: A user selects a news category or enters a search query.

b) API Call: The backend fetches relevant news articles from the external **NewsAPI** based on the user's selection.

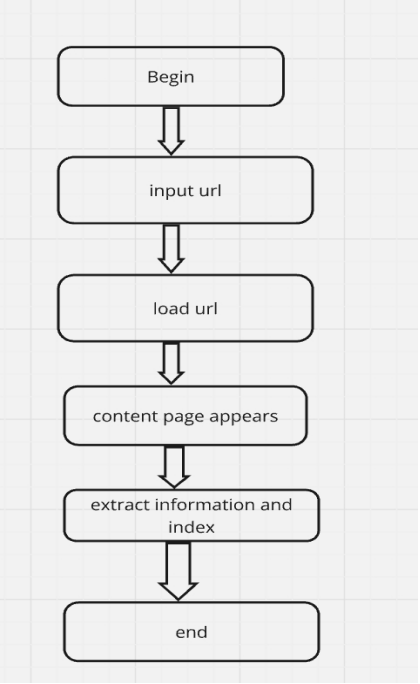
c) Data Processing: The fetched data is processed to remove unnecessary content and summarize key points.

d) Rendering on UI: The front end dynamically updates the news feed with processed articles, displaying headlines, images, sources, and summaries.

e) User Interaction: Users can switch between languages, activate dark mode, listen to articles via TTS, or search for specific news topics.

f) User Authentication: Registered users and admins can log in to access personalized settings or manage news content.

A. The system is designed to retrieve text from both the news index page and the content page of the target URL.
B. It extracts the news index page's URL, title, and publication date.
C. The program formats the extracted text content according to the display format of the page.
D. If there are multiple pages or related news, the system ensures that all relevant news content is extracted to maintain completeness.
E. The extracted news and information can be stored either in a text file or a database to facilitate easy access for user queries.



a.)

Fig. 1. Design flow chart

D. Technologies Used

- ***Frontend***: HTML, CSS, JavaScript
- ***Backend:*** Node.js
- ***API Integration***: NewsAPI for fetching news articles
- ***Text Processing***: JavaScript-based summarization logic
- ***Text-to-Speech:*** Web Speech API for voice output

API Fetching and Summarization

The news application fetches real-time news articles using the **NewsAPI** service. By making asynchronous API requests, the system retrieves articles based on user-selected categories or search queries. The fetched data includes details such as the title, description, source, and image URL, which are dynamically displayed on the webpage.

To enhance readability, the application implements **text summarization** by extracting key sentences from the article content. A local summarization function processes the fetched text, breaking it into meaningful bullet points. This ensures that users can quickly grasp essential information without reading the entire article, improving accessibility and efficiency.

CONCLUSION

The news application fetches real-time news articles using the **NewsAPI** service. By making asynchronous API requests, the system retrieves articles based on user-selected categories or search queries. The fetched data includes details such as the title, description, source, and image URL, which are dynamically displayed on the webpage.

FUTURE SCOPE

Looking ahead, we aim to further enhance the website by incorporating more interactive features such as live updates, video content, and user-generated stories. We plan to introduce personalized news feeds, powered by AI, to deliver content that aligns with each user's interests. Additionally, expanding coverage to include global news, integrating social media features, and exploring multimedia storytelling will allow us to connect with a broader audience. Over time, we envision the website becoming a go-to source for news, offering innovative tools for readers and a platform for diverse voices to be heard.

ACKNOWLEDGMENT

I would like to express my sincere gratitude to all those who have contributed to the success of this research paper. First and foremost, would like to thank my advisor for their continuous support, valuable insights, and constructive feedback throughout the course of my research. Their guidance has been instrumental in shaping the direction of this paper.

REFERENCES

1. HTML: Mozilla Contributors. (2023). *HTML: HyperText Markup Language*. Mozilla Developer Network (MDN). Retrieved from https://developer.mozilla.org/en-US/docs/Web/HTML
2. **CSS:** W3C. (2018). *CSS: Cascading Style Sheets*. World Wide Web Consortium (W3C). Retrieved from https://www.w3.org/Style/CSS/
3. **JavaScript:** Mozilla Contributors. (2023). *JavaScript: A Programming Language for the Web*. Mozilla Developer Network (MDN). Retrieved from https://developer.mozilla.org/en-US/docs/Web/JavaScript
4. **API:** Mozilla Contributors. (2023). *Introduction to APIs*. Mozilla Developer Network (MDN). Retrieved from https://developer.mozilla.org/en-US/docs/Web/API
5. **W3Schools.** (n.d.). *Learn HTML, CSS, JavaScript, and more*. W3Schools. Retrieved from https://www.w3schools.com
6. **MDN Web Docs.** (n.d.). *JavaScript Guide*. Mozilla Developer Network (MDN). Retrieved from https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide