RagSeek: A RAG-Based Multimodal Assistant for Dynamic News Retrieval and Question Answering

Atharva Surendra Chaudhari
D.Y. Patil College of Engineering, Akurdi
Savitribai Phule Pune University
India

Aniket Somnath Lokhande
D.Y. Patil College of Engineering, Akurdi
Savitribai Phule Pune University
India

Prof. Rasika Vishal Wattamwar
D.Y. Patil College of Engineering, Akurdi
Savitribai Phule Pune University
India

***Abstract*— The rapid evolution of news data presents significant challenges for AI-driven question-answering systems. Traditional Large Language Models (LLMs) struggle with context length limitations, making it impractical to process large volumes of daily news updates. Moreover, fine-tuning LLMs frequently to incorporate the latest news is computationally expensive. Additionally, news content is multimodal, consisting of both textual articles and visual elements (images, infographics, etc.). News stories often report on visually driven events such as natural disasters, protests, sporting events, or political campaigns. Visual data like photos and charts serve as verifiable evidence, making news content more believable. This requires an AI system capable of handling diverse data types effectively. To address these challenges, we propose RagSeek, a multimodal Retrieval-Augmented Generation (RAG) system tailored for news domain question-answering. Instead of relying solely on static knowledge stored in LLM parameters, our approach separates knowledge retrieval from reasoning by leveraging external, dynamically updated knowledge sources. This allows the model to fetch the latest news data in real time while maintaining accurate and contextually relevant responses. Furthermore, our system integrates multimodal capabilities, enabling users to query using text, images, or a combination of both, and receive responses in the appropriate format. We evaluate RagSeek across diverse news datasets, demonstrating its effectiveness in real-time knowledge retrieval, multimodal understanding, and factual accuracy. Experimental results highlight the potential of RAG-powered architectures in transforming AI-driven journalism, news summarization, and real-time information synthesis.**

Keywords*— *Large Language Models, AI, Machine Learning, Retrieval Augmented Generation (RAG), Multimodal AI, News Question Answering

I. INTRODUCTION

In recent years, the landscape of Natural Language Processing (NLP) has been radically transformed by the advent of transformer-based models, beginning with the landmark paper “Attention is All You Need” by Vaswani et al. in 2017. This architecture introduced the self-attention mechanism that eliminated the need for recurrence and convolution in deep learning for language, unlocking unprecedented scalability and performance. Building on this foundation, a new class of Large Language Models (LLMs) emerged, including OpenAI’s GPT series, Meta’s LLaMA, Anthropic’s Claude, Mistral, and others. These models demonstrate remarkable fluency, contextual understanding, and generation abilities, positioning them as state-of-the-art tools for a wide range of applications—from code generation and dialogue agents to content creation and question answering.

Despite their successes, LLMs exhibit a critical limitation: they are inherently static in nature. Once trained, their knowledge is frozen, meaning they are incapable of adapting to newly emerging information, such as breaking news events. This is especially problematic in dynamic domains like journalism, where real-time information access and reasoning are essential. Attempts to overcome this limitation via fine-tuning pose several drawbacks: the process is expensive, lacks transparency, and becomes obsolete quickly as news updates in real time. Furthermore, fine-tuned models operate as "black boxes," offering no visibility into how or why a response was generated—an unacceptable risk in domains that demand factual correctness and accountability.

The application of LLMs in journalism is already gaining traction, with systems like **BloombergGPT** being fine-tuned specifically for financial reporting, and tools like **GENIE** designed to assist editors with content generation. These models can perform summarization, fact-checking, and even generate headlines. However, their reliance on historical data and closed training corpora means they often hallucinate facts or misrepresent context when confronted with novel events. Other tools such as ChatGPT offer browsing capabilities, enabling limited access to real-time content via search, but even these are restricted to text-based responses and lack structured integration with external knowledge systems. Notably, they cannot process or return **visual data**, such as graphs, satellite imagery, or photojournalism, which are integral to understanding many real-world events.

This gap has led researchers to explore **Retrieval-Augmented Generation (RAG)** as a more practical solution. In a RAG system, the language model retrieves relevant documents or content from external, up-to-date sources and conditions its output on that context—allowing it to dynamically incorporate new information without retraining. RAG frameworks can be further enhanced by integrating external APIs, real-time RSS feeds, or curated news datasets. Unlike fine-tuning, RAG enables **query-specific personalization**, filtering content based on topic, region, time, or language, thus improving both relevance and user experience. Moreover, it adds an important layer of explainability, as retrieved documents can be shown to the user alongside the generated response.

However, a major shortcoming persists across nearly all existing RAG systems in the news domain: they operate exclusively on **textual content**. This is problematic because

modern news is inherently **multimodal**. Visual information such as images, infographics, heat maps, and statistical charts is often essential to understanding complex or rapidly evolving events. For instance, satellite imagery is crucial for interpreting natural disasters, graphs are essential for understanding election results, and stock charts convey financial shifts at a glance. Systems that fail to include visual media present an incomplete and potentially misleading view of the news. To address these gaps, we propose a novel system that combines **multimodal retrieval capabilities** with a RAG-enabled LLM architecture tailored for real-time news-based question answering. Our approach allows users to submit queries in **text, image, or hybrid form**, and receive a rich, grounded response that includes **relevant articles, images, and references**.

When a user enters a **text query**, it provides a direct answer along with relevant text snippets from recent news and matching images that visually support the topic. If a user uploads an **image**, the system identifies visually related news events and returns associated articles and similar images. For **combined text and image input**, the system retrieves and presents both relevant textual content and visuals that align with the user's intent. This flexible input-output behavior ensures that users receive rich, informative, and context-aware responses, regardless of the input type.

II. Literature Review

The integration of Large Language Models (LLMs) into news production and dissemination has significantly transformed digital journalism, enabling more personalized, responsive, and scalable content delivery. One of the most prominent domain-specific applications is BloombergGPT, a 50-billion-parameter model fine-tuned on a vast corpus of financial data. This system excels at tasks such as sentiment analysis, question answering, and summarizing finance-related content, owing to its specialized training. While it offers high precision within the financial news sector, its domain specificity presents a limitation when applied to broader news topics.

Interactive systems like the Time AI Chatbot, introduced alongside the 2024 *Person of the Year* coverage by *Time* magazine, showcase the use of LLMs to improve user engagement. This chatbot enables users to interact with articles through natural language queries, obtain summaries, and translate content into multiple languages. Such functionalities enhance the accessibility of news and illustrate the growing use of conversational AI in journalism.

In addition to these targeted systems, fully automated, real-time news generation platforms like Realtime.org represent a shift towards scalable, data-driven journalism. By combining LLMs with public data repositories, Realtime.org automatically produces summaries, visualizations, and full-length articles. This architecture supports the generation of current and context-aware news content without manual intervention, highlighting the potential of AI for autonomous newsroom functions.

Another promising direction involves the integration of Retrieval-Augmented Generation (RAG) to improve the contextual relevance and factual accuracy of LLM responses. A notable proof-of-concept from RedBuffer demonstrated how a fine-tuned DistilBERT model, combined with a RAG mechanism, could effectively analyze sentiment and answer questions based on BBC News data. Similarly, the Generative AI Newsroom project explored the deployment of LLMs for building a live news platform, emphasizing retrieval from structured public datasets and the generation of summaries aligned with journalistic standards. These examples illustrate the ongoing innovation in hybrid architectures that blend generative and retrieval-based components to meet the factual demands of news production.

Despite these advancements, significant challenges remain unaddressed. Domain specificity continues to hinder the adaptability of certain models, while the lack of support for multimodal content restricts expressiveness in modern news formats that increasingly rely on images, videos, and infographics. Moreover, the rapid pace of news requires continuous fine-tuning to keep models aligned with emerging information, resulting in substantial computational and operational overhead.

To address these limitations, our research introduces a Multimodal Retrieval-Augmented Generation (RAG) framework designed for general news applications. Unlike domain-constrained systems, our model integrates visual and textual modalities to provide more comprehensive and engaging content. Furthermore, by leveraging dynamic retrieval capabilities, it reduces the reliance on frequent fine-tuning, ensuring that the system remains current with minimal intervention. This approach not only enhances the flexibility and factuality of AI-generated news but also makes it suitable for a wide spectrum of topics and media formats, aligning with the evolving demands of digital journalism.

III. Proposed Multimodal RAG Assistant-RagSeek

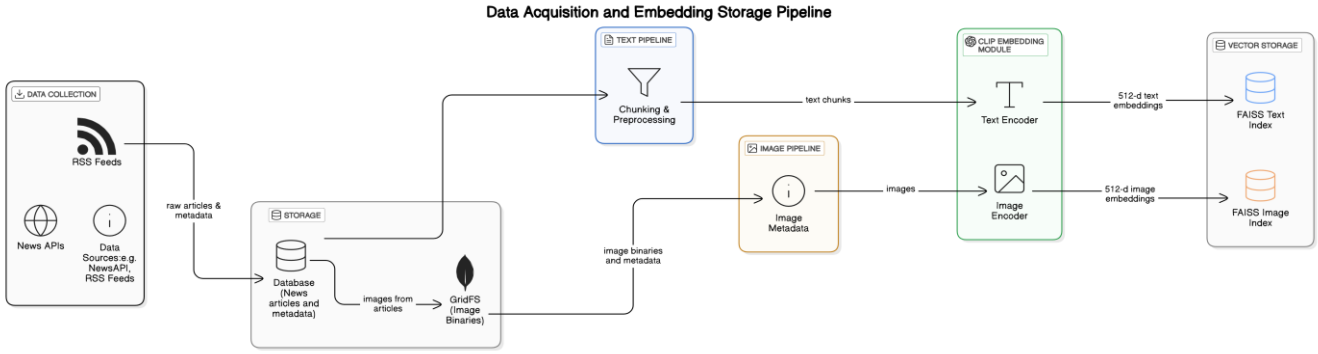
This section elaborates the detail methodology of our proposed approach. The proposed architecture consists of six core components operating in a pipeline configuration: (1) Data Acquisition and Management, (2) Data Processing and Representation, (3) Embedding Generation and Storage, (4) Retrieval Engine and Ranking, (5) Query Processing and Answer Generation, and (6) User Interface and Visualization. Each module is responsible for a distinct function but designed to interface seamlessly with the rest of the pipeline to support scalability and modular development.

A. Data Acquisition and Management

The Data Acquisition component is responsible for the systematic collection, processing, and storage of both textual and visual content from a broad spectrum of news sources. The system is designed to support scalability and content diversity, maintaining a curated registry of trustworthy news providers spanning domains such as politics, technology, business, and entertainment, as well as covering multiple geographic regions. Refer fig 3.1

To ensure comprehensive data ingestion, the system integrates multiple collection strategies, with RSS feed parsing serving as one of the primary examples. A scheduled polling mechanism is employed to periodically query configured sources based on their update frequency. High-velocity sources may be polled every few minutes, whereas low-frequency outlets are accessed on an hourly or daily basis. This adaptive scheduling ensures the timely inclusion of breaking news while managing resource efficiency.

For each collected news item, the system performs deep content extraction by resolving article URLs and retrieving full-length content beyond the metadata typically available in



feed structures. This raw content is then enriched with

Fig3.1

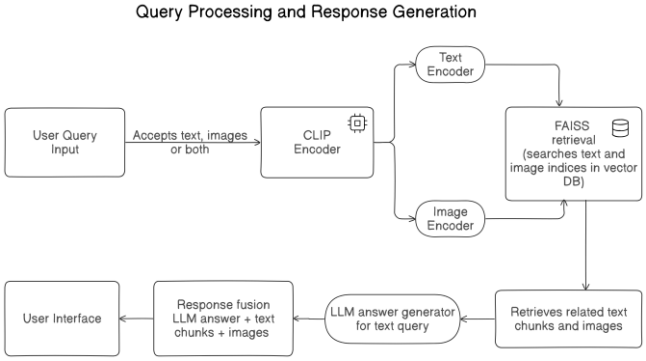


Fig 3.2

structured metadata including publication timestamps, source identifiers, content categories, and regional tags. In parallel, the Image Acquisition Subsystem identifies and processes visual elements associated with the retrieved news articles. Upon discovery, images are subjected to a series of validation checks to ensure quality, relevance, and format compatibility. Validated images are then stored in a scalable binary storage system such as MongoDB GridFS, with comprehensive metadata associations maintained to preserve the link between visual content and its source article. This metadata includes contextual cues, source URLs, resolution details, and semantic tags extracted during processing. This approach enables the system to handle large volumes of image data while maintaining efficient retrieval capabilities.

B. Data Processing and Representation

Prior to embedding generation, both textual and visual modalities undergo dedicated preprocessing pipelines to ensure compatibility, uniformity, and semantic richness for downstream embedding and retrieval operations. This component transforms raw multimodal content into structured, normalized representations.

Textual data processing : The text processing pipeline is designed to convert raw news content into a clean, normalized format optimized for language modeling and embedding generation. Initial processing involves HTML structure cleaning to strip away markup artifacts while preserving meaningful paragraph and section hierarchies. Subsequently, text normalization procedures are applied to remove extraneous characters, control whitespace, standardize punctuation, and ensure UTF-8 encoding compliance. This step also eliminates boilerplate text, advertisements, and navigation elements commonly embedded within web-based news articles.

Image data processing : For the visual modality, the system implements a standardized image preprocessing workflow tailored for compatibility with vision-language models such as CLIP. Each image is then resized to the CLIP model's expected input dimensions of 224×224 pixels.

C. Embedding Generation and Vector Storage

The embedding generation module is a central component of the RAGSeek architecture, enabling unified representation of both textual and visual modalities within a shared semantic space. This is achieved using CLIP (Contrastive Language–Image Pre-training), a vision-language model developed by OpenAI that jointly learns representations of images and their corresponding natural language descriptions. CLIP enables the encoding of both modalities into a common vector space, supporting cross-modal retrieval, semantic similarity matching, and multimodal reasoning. The system employs the pre-trained openai/clip-vit-base-patch32 model, which generates 512-dimensional embeddings for both text and image inputs. These embeddings facilitate direct comparison between text and visual content through dot product or cosine similarity metrics, enabling cross-modal alignment and retrieval. Refer fig 3.1

1)Text embedding Pipeline : Textual data, preprocessed as described in the prior section, is passed through CLIP's text encoder. This encoder utilizes a Transformer-based architecture, where the input text is tokenized and normalized using CLIP's dedicated processor. The output is a dense vector embedding residing in the joint multimodal embedding space. These vectors capture the semantic essence of the text in a way that aligns with the visual representation space, allowing for effective multimodal similarity computations.

2)Image Embedding Pipeline : Simultaneously, images undergo a parallel embedding process via CLIP's image encoder. This encoder is based on a modified Vision Transformer (ViT), which transforms normalized and resized images into the same 512-dimensional vector space. The

Identify applicable funding agency here. If none, delete this text box.

shared representation allows direct semantic matching between an image and related textual descriptions, reinforcing the model's ability to support multimodal retrieval tasks.

3)FAISS-Based Vector Storage and Retrieval : To enable efficient similarity search and retrieval, the system utilizes FAISS (Facebook AI Similarity Search), a high-performance library designed for large-scale vector indexing and search.

a) Text embedding storage : Text embeddings are stored in a dedicated FAISS index using the IndexFlatL2 configuration, which provides exact nearest neighbor search based on L2 distance. Each embedding corresponds to a semantically meaningful chunk of text, and is stored alongside rich metadata such as source ID, timestamp, and contextual tags, ensuring traceability and contextual relevance during retrieval.

b) Image Embedding Storage : Similarly, image embeddings are stored in a parallel FAISS index using the same `IndexFlatL2` scheme to maintain consistency across modalities. Each image embedding corresponds to a preprocessed visual input and is linked to associated metadata including article ID, image source, and original resolution. While embeddings are used for retrieval, the actual binary image data is stored in a separate scalable storage system, such as MongoDB GridFS, with persistent references maintained for efficient lookup.

c) Serialization and Persistence : Both FAISS indices (text and image) are periodically serialized and saved to disk for reusability and recovery. During serialization, the vector indices, their associated metadata, and original content references are bundled using Python's pickle protocol. This process ensures data integrity and minimizes reprocessing time during system restarts or updates.

D. Retrieval Engine and Ranking

The retrieval engine serves as the semantic core of the RAGSeek architecture, enabling efficient, accurate access to relevant textual and visual information based on user queries. The system is designed to handle multimodal search operations by leveraging a shared embedding space powered by CLIP and optimized retrieval through FAISS. Refer fig 3.2

1) Supported Search Modalities

The retrieval framework supports four primary query-response modalities, enabling comprehensive and flexible information access:

Bullet 1 : Text-to-Text (text2text): Retrieves semantically relevant textual content in response to a textual query. This mode is optimized for natural language question answering and contextual text retrieval.

Bullet 2 : Text-to-Image (text2image): Locates images that align with a user-provided text query. This facilitates visual discovery using natural language descriptions.

Bullet 3 : Image-to-Text (image2text): Returns descriptive or contextually relevant text segments in response to a visual query. This supports content identification and visual explanation tasks.

Bullet 4 : Image-to-Image (image2image): Retrieves visually similar images based on a given image input, supporting visual similarity search and content clustering.

These multimodal retrieval paths are enabled by the shared CLIP embedding space and optimized using FAISS vector search techniques.

2) Text Query Search Workflow

When a user submits a textual query, the system processes and executes retrieval as follows:

a) Query Processing

The text query is normalized and encoded using CLIP's text encoder, yielding a 512-dimensional embedding within the shared multimodal space. This vector is cast into float32 format for FAISS compatibility.

b) FAISS Search Execution

The appropriate FAISS index—either for text (text2text) or image embeddings (text2image)—is dynamically loaded. A k-nearest neighbor (k-NN) search is executed via index.search(query_embedding, top_k), returning the indices and distance scores of the top matching entries. Lower L2 distances correspond to higher semantic similarity.

c) Result Processing

Based on the query mode:

In text2text, the retrieved text chunks are returned along with their metadata (e.g., source URL, timestamp, content tags).

In text2image, semantically related images are returned with corresponding metadata.

Results are sorted by ascending L2 distance to prioritize the most semantically relevant entries.

3) Image Query Search Workflow

For image-based queries, the pipeline mirrors the text query process with the following adjustments:

a) Query Processing

The uploaded image is preprocessed (resizing, normalization) and passed through CLIP's image encoder to generate the corresponding 512-dimensional embedding, subsequently cast to float32.

b) FAISS Search Execution

The system loads the relevant FAISS index—image embeddings (image2image) or text embeddings (image2text)—and performs a k-NN search as before.

c) Result Processing

In image2image, visually similar images are retrieved and returned with metadata.

In image2text, the system surfaces text chunks semantically related to the content of the image.

Again, results are ordered based on proximity in the embedding space, using L2 distance as the similarity metric.

4) Multimodal Query Search Workflow

For queries that include both text and image inputs, the system performs a dual retrieval process:

a) Dual Encoding

The text and image inputs are separately encoded using their respective CLIP encoders. The resulting embeddings co-exist in the same joint semantic space.

b) Combined Search Strategy

The system concurrently executes k-NN searches across both embedding indices:

The text query is used to search both text and image indices.

The image query is similarly used to retrieve both visual and textual content.

c) Result Fusion

Search results from all paths are aggregated and ranked, with metadata and confidence scores preserved. Fusion algorithms may consider normalized distances and modality-specific weights.

d) Comprehensive Response Generation

The final response includes a curated selection of the most relevant text chunks and images, structured to offer a unified view of cross-modal content associated with the user's query.

E. Response Synthesis and Presentation

Following retrieval of relevant content, the RAGSeek system enters its final phase—synthesizing a coherent, multimodal response and presenting it through the user interface. This stage is responsible for contextualizing the retrieved information, generating natural language answers where applicable, and integrating text and visual results in an accessible, user-friendly format.

1) Text-Based Queries: For textual queries, the system first retrieves the most relevant text chunks and associated images using the mechanisms outlined in the retrieval engine. These chunks are then passed to a Large Language Model (LLM), which synthesizes a direct answer to the user's question. This is accomplished using a prompt-completion paradigm, where the retrieved context is embedded within the prompt template to guide the LLM in generating a precise, context-aware response. Output includes:

*a) A synthesized **answer to the query** generated by the LLM.*

*b) A **ranked list of supporting text chunks**, each presented with metadata such as title, source, and publication date.*

*c) A **collection of semantically related images** to enhance the user's understanding and engagement.*

2) Image-Based Queries : When the user submits an image, the system processes it through the CLIP encoder and performs image-to-text and image-to-image retrieval. In this case, the system does not generate a new answer via the LLM, as no explicit textual question has been provided. Instead, the interface presents:

*a) **Relevant text chunks** that describe or relate to the visual content of the query.*

*b) **Visually similar images** retrieved from the indexed dataset. This approach supports use cases such as content exploration, reverse image lookup, and contextual discovery.*

3) Multimodal Queries : In scenarios where both text and image are submitted, the system conducts dual retrieval and fuses results across all modalities. Retrieved text chunks serve as grounding content for the LLM, which is prompted with both the textual query and relevant contextual passages. The model then generates a synthesized response that reflects both the user's question and the contextual cues extracted from the image-based search. Multimodal response includes :

*a) A comprehensive **answer generated by the LLM**, integrating signals from both the textual and visual domains.*

b) A set of relevant text segments semantically aligned with the query and/or image.

*c) A **collection of matching images**, visually linked to both the query and retrieved textual context.*

4) User interface Integration : All synthesized responses and retrieved assets are delivered through a structured user interface. The interface includes dedicated sections for :

*a) Displaying the **LLM-generated answer** (if applicable).*

*b) Presenting **retrieved textual and visual elements** in scrollable, filterable panes.*

*c) Offering **source attribution** and **confidence scores** for transparency and trustworthiness.*

IV. Experimental Results and Evaluation

Due to the novelty of the proposed system—combining multimodal Retrieval-Augmented Generation (RAG) with real-time news data—there are no existing baseline systems that support the same feature set. General-purpose LLMs such as GPT-4, Gemini, and Claude provide strong generative capabilities and limited web search functionality, but none support integrated multimodal retrieval across both input and output channels. Specifically, no existing system accepts combined text and image queries and returns both textual and visual content in the response.

A. Screenshots showing different query modes

Figure	Content
4.1	Text only query - Retrieved text chunks + metadata - Retrieved relevant images - LLM generated answer
4.2	Image only query - Retrieved text chunks - Retrieved similar images - Metadata shown alongside each
4.3	Text + Image (Multimodal Query) - Retrieved text chunks - Retrieved relevant images

	- LLM generated answer - All elements with metadata (source attribution)

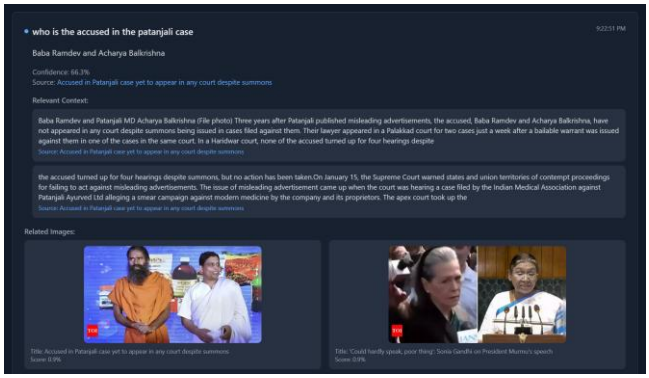


Fig 4.1

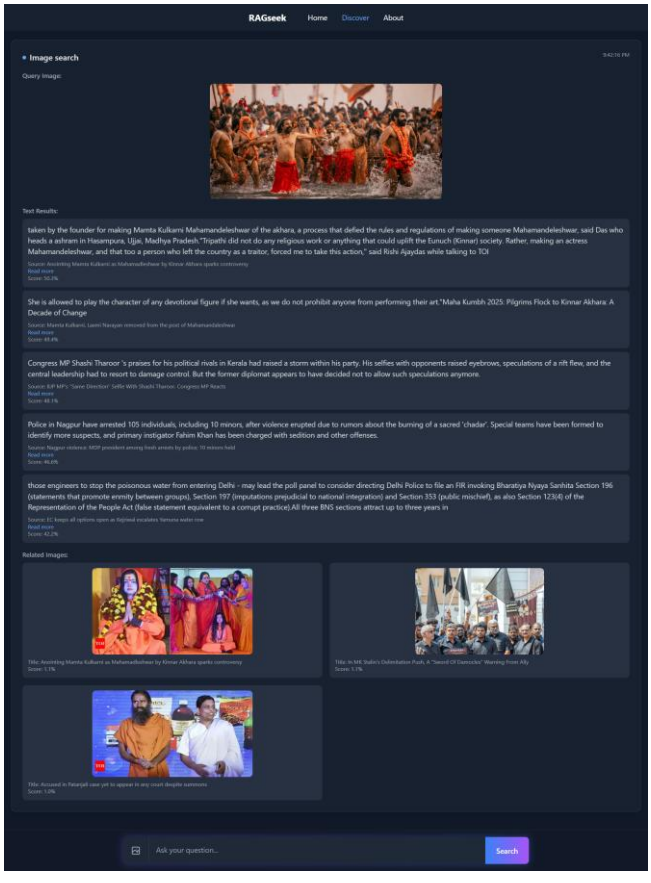
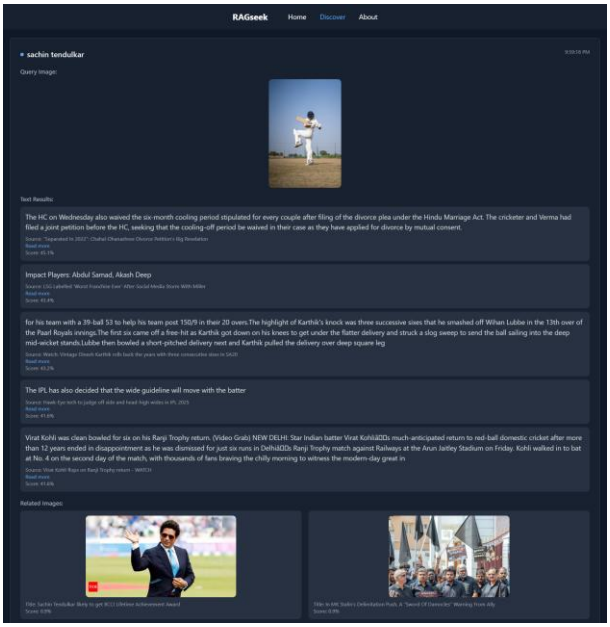


Fig 4.2



REFERENCES

[1] J. Johnson *et al.*, "The Faiss Library," *arXiv preprint arXiv:2401.08281*, 2025.

[2] Z. Jing, Y. Su, Y. Han, B. Yuan, H. Xu, C. Liu, K. Chen, and M. Zhang, "When Large Language Models Meet Vector Databases: A Survey," *CoRR*, vol. abs/2402.01763, 2024.

[3] E. K. Elena, "Why You Should Be Careful Using FAISS," *Medium*, Jul. 20, 2022.

[4] L. Caspari, K. G. Dastidar, S. Zerhoudi, J. Mitrovic, and M. Granitzer, "Beyond Benchmarks: Evaluating Embedding Model Similarity for Retrieval Augmented Generation Systems," *arXiv preprint arXiv:2407.08275*, Jul. 2024.

[5] S. S. R. Kumar, R. S. M. R. Kumar, and K. R. S. Kumar, "Performance Evaluation of Vector Embeddings with Retrieval-Augmented Generation," in *Proc. 2024 IEEE International Conference on Artificial Intelligence and Data Science (ICAIDS)*, 2024.

[6] B. Zhang, Z. Liu, C. Cherry, and O. Firat, "When scaling meets llm finetuning: The effect of data, model and finetuning method," arXiv preprint arXiv:2402.17193, 2024.

[7] A. Tayal and A. Tyagi, "Dynamic contexts for generating suggestion questions in rag based conversational systems," in Companion Proceed ings of the ACM on Web Conference 2024, pp. 1338–1341, 2024.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need.(nips), 2017," arXiv preprint arXiv:1706.03762, vol. 10, p. S0140525X16001837, 2017.

[9] K. Ethayarajh, "How contextual are contextualized word representa tions? comparing the geometry of bert, elmo, and gpt-2 embeddings," arXiv preprint arXiv:1909.00512, 2019.