

# **Generative AI-Powered News Summarization in Local Indian Languages**

A PROJECT REPORT

*Submitted by*

BL.EN.U4AIE20009 - Bobburi Sai Kowshik

BL.EN.U4AIE20018 - Hasita Y

BL.EN.U4AIE20039 - Nadilla Yaswanth Baba

BL.EN.U4AIE20058 - Samirit Saha

**BACHELOR OF TECHNOLOGY**

IN

COMPUTER SCIENCE AND ENGINEERING



AMRITA SCHOOL OF COMPUTING, BENGALURU

AMRITA VISHWA VIDYAPEETHAM

BENGALURU 560 035

DECEMBER 2023

A handwritten signature in blue ink, likely of the student Samirit Saha.

## ACKNOWLEDGEMENTS

The satisfaction that accompanies successful completion of any task would be incomplete without mention of people who made it possible, and whose constant encouragement and guidance have been source of inspiration throughout the course of this project work.

We offer our sincere pranams at the lotus feet of “**AMMA**”, **MATA AMRITANANDAMAYI DEVI** who showered her blessing upon us throughout the course of this project work.

We owe our gratitude to **Prof. Manoj P.**, Director, Amrita Vishwa Vidyapeetham Bengaluru Campus. We thank **Dr. Sriram Devanathan**, Principal Amrita School of Computing, Bengaluru for his support and inspiration.

We would like place our heartfelt gratitude to **Dr. Gopalakrishnan E.A.**, Chair, Department of Computer Science and Engineering, Amrita School of Computing, Bengaluru for his valuable support and inspiration.

It is a great pleasure to express our gratitude and indebtedness to our project guide **Dr S Santhanalakshmi**, Department of Computer Science and Engineering, Amrita School of Computing, Bengaluru for her/his valuable guidance, encouragement, moral support, and affection throughout the project work.

We would like to thank express our gratitude to project panel members for their suggestions, encouragement, and moral support during the process of project work and all faculty members for their academic support. Finally, we are forever grateful to our parents, who have loved, supported and encouraged us in all our endeavors, and made this project a resounding success and something which can be of immense benefit to the society.

## ABSTRACT

Our project introduces an innovative approach for news summarization for local Indian languages, capitalizing on the capabilities of generative AI embedded within a chatbot interface. The overarching goal is to revolutionize the comprehension and summarization of news articles across a spectrum of Indian languages, intricately navigating the complexities posed by linguistic diversity, cultural subtleties, and domain-specific terminologies. Central to our initiative is a commitment to ethical considerations, ensuring that the AI-driven summarization process adheres to principles of fairness, transparency, and accountability. A pivotal aspect of our research involves an in-depth exploration of user perceptions and acceptance, with a focus on critical factors such as the establishment of trust, enhancement of readability, and alignment with the relevance expectations of the diverse audience. Through the strategic implementation and continuous refinement of generative AI models tailored to specific linguistic nuances, our project aspires to offer a sophisticated and culturally sensitive solution that effectively bridges information gaps in linguistically diverse societies. This research not only contributes to the evolving discourse at the intersection of artificial intelligence and media accessibility but also holds significant implications for future developments in the broader landscape of language-centric technology applications.

This will in the long run, allow the creation of frameworks and application that will be able to be of immense use to the common man who can have access to quality news content and summarization in his native language. This will enable the access of important news feeds to a larger population even if they aren't well versed in English.

## TABLE OF CONTENTS

### Page No.

ACKNOWLEDGEMENTS	iii-v
ABSTRACT	vi-viii
LIST OF FIGURES	
Fig 1.1: Aim of project	2
Fig 4.1: Overall workflow of model	13
Fig 5.1: Data ingestion module	16
Fig 5.2: NLP Module	17
Fig 5.3: DL Model training module	18
Fig 5.4: Article processing module based on LLM model selection	19
Fig 5.5: Load PDF module	19
Fig 6.1: Implementation of GUI	26
Fig 6.2: Results for given query	26
Fig 7.1: ROUGE score and BLEU score	32
LIST OF TABLES	
Table 2.1: Literature Survey and Inference	10-14
CHAPTER 1- INTRODUCTION	1-2
CHAPTER 2 – LITERATURE SURVEY	3-7
CHAPTER 3 – SYSTEM REQUIREMENTS AND ANALYSIS	8-10
SOFTWARE REQUIREMENTS	8-9
HARDWARE REQUIREMENTS	8-10
CHAPTER 4 – SYSTEM DESIGN	11-13
4.1 Architectural Overview	11

4.2 Low Level Design	11-12
4.2.1 Data Ingestion Module	11
4.2.2 Natural Language Processing Module	11
4.2.3 Deep Learning Training Module	11
4.2.4 Web-based Deployment Module	11
4.2.5 Web-scraping Deployment Module	11-12
4.3 High-Level Design	12-13
4.3.1 Framework and Components	12
4.3.1 Data Flow Overview	12
4.3.2 Interaction Protocols	12
4.3.3 Key Data Flows	12-13
CHAPTER 5 – SYSTEM IMPLEMENTATION	14-19
5.1.1 Data Ingestion Module	14
5.1.2 Natural Language Processing Module	14
5.1.3 Deep Learning Model Training Module	14
5.1.4 Web-based Deployment Module	14-15
5.1.5 Web Scraping Module	15
5.1.6 PDF Loading & LLM Selection Module	15
Diagrams	16-19
CHAPTER 6 – RESULTS AND ANALYSIS	
6.1 Testing Objectives and Scope	20-21
6.2 Test Scenarios and Cases	21-22
6.3 Test Execution and Reporting	22-23
6.4 System Testing with respect to integrated modules	23-25
CHAPTER 7 – RESULTS AND ANALYSIS	
7.1 Web Scraping Module and news generation	27
7.2 Load PDF component	27
7.3 LLM Selection Component	28-29

CHAPTER 8 – CONCLUSION AND FUTURE SCOPE	31-33
8.1 Conclusion	31
8.2 Future Scope	31-33
8.3 Vision for the Future	
REFERENCES	34

## LIST OF FIGURES

Fig 1.1: Aim of project	2
Fig 4.1: Overall workflow of model	13
Fig 5.1: Data ingestion module	16
Fig 5.2: NLP Module	17
Fig 5.3: DL Model training module	18
Fig 5.4: Article processing module based on LLM model selection	19
Fig 5.5: Load PDF module	19
Fig 6.1: Implementation of GUI	26
Fig 6.2 Results for given query	26
Fig 7.1: ROUGE score and BLEU score	32

# CHAPTER - 1

## INTRODUCTION

In the dynamic landscape of information dissemination, where linguistic diversity converges with the transformative potential of artificial intelligence (AI), this project emerges as a pioneering effort to navigate the intricate challenges of news summarization in Local Indian Languages. The rich tapestry of languages in India, ranging from Hindi and Bengali to Telugu and Tamil, presents a compelling mosaic of cultural expressions, each with its own set of nuances and idiosyncrasies. Against this backdrop, the project embarks on a comprehensive mission to harness the advanced capabilities of generative AI, intricately woven within the fabric of a sophisticated chatbot interface, aiming to redefine the landscape of news summarization across the vast linguistic spectrum of the Indian subcontinent.

India's linguistic diversity is not merely a numerical enumeration; it is a reflection of the multifaceted cultural intricacies that define the nation. Therefore, the core objective of this initiative is to develop a chatbot that transcends conventional language processing barriers, not only comprehending but also synthesizing news articles in a manner that resonates with the cultural and linguistic sensitivities of diverse local audiences. This monumental task involves navigating intricate linguistic subtleties, decoding diverse cultural contexts, and understanding the intricacies of domain-specific jargon to deliver accurate and culturally resonant news summaries.

At the heart of this ambitious undertaking lies a commitment to ethical considerations, ensuring that the development and deployment of AI-driven summarization processes adhere to principles of fairness, transparency, and accountability. As the project unfolds, it places particular emphasis on understanding user perceptions and acceptance, delving into critical factors such as the establishment of trust, the enhancement of readability, and the alignment of summaries with the nuanced expectations of a diverse audience. This user-centric approach not only

ensures the practical viability of the technology but also underscores the project's commitment to meeting the multifaceted information needs of local communities, reinforcing the importance of technology as a tool for cultural inclusivity.

Beyond the realms of technological advancement, this project envisions a solution that transcends the conventional boundaries of AI development. As mentioned in Fig 1.1, By strategically implementing and continuously refining generative AI models tailored to the specific linguistic nuances of each language, it seeks to provide a culturally sensitive and highly effective means of bridging information gaps in societies characterized by a myriad of languages. This research is not merely an isolated endeavor; it is a significant contribution to the ongoing discourse at the intersection of artificial intelligence and media accessibility. Moreover, it lays the groundwork for future developments in language-centric technology applications, fostering inclusivity, understanding, and connectivity on a global scale.



**Fig 1.1:** Our project aims to provide comprehensive summarization of news articles in various Indian languages like Hindi (left) and Tamil (right)



## **CHAPTER - 2**

### **LITERATURE REVIEW**

The literature survey delves into a diverse array of papers that significantly contribute to the domain of news summarization and related areas, presenting innovative approaches, methodologies, and insights. As mentioned in Table 2.1, The first paper by Wazery et al. introduces an optimized hybrid deep learning model for extractive summarization [1]. By combining word embeddings and statistical features, the model surpasses baseline performance, achieving state-of-the-art results on benchmark datasets. Additionally, the study showcases the effectiveness of Keras Tuner for hyperparameter tuning, emphasizing the significance of optimization techniques in enhancing summarization models. Shah et al.'s work on automated text document classification using BERT stands out as a pivotal contribution [2]. The second paper proposes a classification framework that outperforms traditional approaches on email and news datasets, highlighting the need for efficient and scalable algorithms in text classification using BERT. The authors also suggest future research directions in multilingual text classification using BERT, indicating the evolving nature of the field.

Zhang et al.'s comprehensive survey on abstractive text summarization based on deep learning provides a foundational understanding of model architectures, training strategies, evaluation metrics, and applications in this domain [3]. The third paper sheds light on the challenges associated with the lack of large-scale datasets for effective summarization in Indian local languages, pointing to critical gaps in the current research landscape. In addressing the global issue of fake news and hate speech, Demilie and Salau propose approaches leveraging deep learning with balanced datasets and preprocessing techniques for Ethiopian languages [4]. The fourth paper underscores the necessity for larger datasets in training and evaluating deep learning models, emphasizing the importance of context-specific solutions in combating misinformation. Yang et al.'s work on automatic summarization based on features introduces a model utilizing BERT and extracting features using TF-IDF and word vector embedding [5]. The fifth paper suggests the integration of both extractive and abstractive models for news domains using BERT, demonstrating a holistic approach to automatic summarization. Yadav et al.'s exploration of graph-based extractive summarization for single-text-based documents introduces a novel perspective on efficiently extracting essential information [6]. The sixth paper proposes integrating

both extractive and abstractive models for news domains using BERT, calling for comprehensive evaluations on diverse datasets and integration with other NLP techniques. Muniraj et al.'s HNTSumm, a hybrid algorithm for summarizing transliterated news articles, showcases a unique combination of extractive and abstractive methods [7]. The seventh paper suggests adapting HNTSumm for use with diverse text corpora beyond news articles, extending the applicability of the proposed approach. Barman et al.'s study on unsupervised extractive summarization employing statistical, topic-modeling, and graph-based approaches provides a comprehensive evaluation of effectiveness using metrics such as ROUGE, F1-score, and precision [8]. The eighth paper recommends the integration of these proposed approaches into real-world applications, emphasizing their potential impact on news aggregators and search engines. In summary, this literature survey encompasses a rich tapestry of research endeavors, ranging from novel model architectures and advanced techniques in summarization to addressing challenges in diverse linguistic and contextual settings. These contributions collectively advance the state-of-the-art in extractive summarization, text classification, and abstractive summarization, providing a nuanced understanding of the evolving landscape of news analysis and summarization.

A comprehensive explanation of each paper is given in the following table:

**Table 2.1: Table of Research papers, inferences and open problems**

S.No	Authors name(s)	Full title of the paper with year	Inference from the paper	Open Problem (For proposed work)
1.	Wazery , Y.M., Saleh, M.E., Ali, A.A.	An optimized hybrid deep learning model based on word embeddings and statistical features for extractive summarization, 2023.	The paper presents a novel approach to extractive summarization that combines word embeddings and statistical features to improve classification performance and summary generation quality. The proposed hybrid model outperforms several baseline models and achieves state-of-the-art results on two benchmark datasets. It also demonstrates the effectiveness of the KerasTuner optimization technique in fine-tuning hyperparameters for the model.	The paper suggests the use of more advanced deep learning architectures, such as transformers, to further improve the performance of the proposed method. The need for more comprehensive evaluation metrics that take into account factors such as coherence and readability in addition to ROUGE scores.

2.	Shah, M.A., Iqbal, M.J., Noreen, N., Ahmed, I.	An Automated Text Document Classification Framework using BERT, 2023.	The paper proposes an automated text classification framework that utilizes BERT, a state-of-the-art deep learning algorithm, to classify different types of text data. The proposed framework preprocesses data by removing stop-words and extra characters to improve classification performance. The results show that the proposed framework outperforms traditional approaches to text classification on two state-of-the-art datasets composed of email and news.	The paper suggests that future research could focus on developing more efficient and scalable algorithms for text classification using BERT and investigate the use of BERT for multilingual text classification.
3.	Zhang, M., Zhou, G., Yu, W., Huang, N., Liu, W.	A Comprehensive Survey of Abstractive Text Summarization Based on Deep Learning, 2020.	The paper provides an in-depth analysis of the latest developments in automatic text summarization and how deep learning models are being used to improve this task. It covers various aspects of abstractive summarization, including model architectures, training strategies, evaluation metrics, and applications.	The paper tells the lack of large-scale datasets and resources for training and evaluating deep learning models, which hinders the development of effective and accurate summarization systems for Indian local languages.
4.	Demile, W.B., Salau, A.O.	Detection of fake news and hate speech for Ethiopian languages: a systematic review of the approaches, 2022.	The paper discusses various approaches and techniques used to combat the spread of fake news and hate speech in Ethiopian languages. It recommends the use of deep learning and machine learning approaches with balanced datasets for improved performance and the importance of preprocessing and feature extraction techniques for effective detection and classification of fake news and hate speech	The paper suggests improving the accuracy and efficiency of detection models. These include developing more accurate and efficient machine learning and deep learning models, , exploring the use of transfer learning and pre-trained language models, investigating the effectiveness of different preprocessing and feature extraction techniques, and developing more effective methods for detecting.

5.	Yang, Y., Tan, Y., Min, J., Huang, Z.	Automatic text summarization for government news reports based on multiple features, 2023.	The paper provides an automatic text summarization model. The BERT model extracts features using the TF-IDF algorithm and word vector embedding method based on the bidirectional encoder representation from transformers model. The sentences are scored based on position, keywords, and similarity features and the top-ranked sentence is selected to summarize by using extractive summarization framework.	The paper suggests to integrate extractive and abstractive text summarization models and apply them to the news domain to test their effectiveness. To extend it with the use of BERT for text classification for multiple local indian languages.
6.	Yadav, A.K., Ranvijay, Yadav, R.S., Maurya, A.K.	Graph-based extractive text summarization based on single document, 2023.	The paper discusses graph-based extractive text summarization is a promising technique for efficiently and effectively extracting essential information from large documents. TGETS as a solution to describe its unique approach to representing documents as graphs and extracting essential information through sentence weighting and generating a summary based on these weights	The paper suggests improving the summarisation for multi-documents. To evaluate the performance of TGETS on different datasets to determine its effectiveness in summarizing different types of text data and integrate with other NLP techniques in different domains.
7.	Muniraj, P., Sabarnathi, K.R., Leelavathi, R., Balaji B, S.	HNTSumm: Hybrid text summarization of transliterated news articles , 2023.	The paper presents a hybrid text summarization algorithm for transliterated news articles. The HNTSumm algorithm uses word embeddings and a pointer-generator network model to generate summaries that are both accurate and efficient. The hybrid approach combines extractive and abstractive methods where textrank algorithm is used for extractive and pointer-generator network model for abstractive summarisation.	The paper tell this work can be extended to summarize transliterated datasets that contain words from various low-resource languages. The HNTSumm could be adapted for use with other types of text corpora beyond news articles, which could potentially include local Indian languages.

8.	Barman , U., Barman , V., Choudhury, N.K., Rahman, M., Sarma, S.K..	Unsupervised Extractive News Articles Summarization leveraging Statistical, Topic-Modelling and Graph-based Approaches , 2022.	The paper presents a comprehensive study on unsupervised extractive news article summarization using statistical, topic-modelling, and graph-based approaches. They evaluate the effectiveness of the proposed approaches using various metrics such as ROUGE, F1-score, and precision and compare their performance with existing state-of-the-art methods. It was observed that the algorithm was able to perform well in entertainment and technology domains.	The paper suggests the proposed approaches can be integrated into real-world applications such as news aggregators and search engines to provide users with concise and relevant summaries of news articles and extend to other domains.
----	--	--	---	--

As seen in Table 2.1, the diverse and impactful contributions of the discussed papers in news summarization are evident in their pioneering approaches and insights. Wazery et al. present an optimized hybrid deep learning model that combines word embeddings and statistical features, outperforming baseline models and showcasing the efficacy of Keras Tuner for hyperparameter tuning. Shah et al. propose a text classification framework using BERT, emphasizing scalability and suggesting future research in multilingual text classification. Zhang et al.'s comprehensive survey on abstractive text summarization covers model architectures, training strategies, and highlights the need for larger datasets, particularly in Indian local languages. Demilie and Salau tackle fake news and hate speech in Ethiopian languages, recommending deep learning with balanced datasets and larger datasets. Yang et al. leverage a BERT model for automatic summarization based on features, suggesting the integration of both extractive and abstractive models for news domains. Yadav et al. explore graph-based extractive summarization, proposing integration with other NLP techniques. Muniraj et al. introduce HNTSumm, a hybrid algorithm for summarizing transliterated news articles, while Barman et al.'s study evaluates unsupervised extractive summarization approaches, recommending real-world integration for applications like news aggregators and search engines. Together, these papers provide a rich tapestry of advancements and methodologies, contributing significantly to the evolving landscape of news summarization.

## CHAPTER – 3

### SYSTEM SPECIFICATIONS

#### 3.1 Software requirements

The software requirements for developing an AI-powered news summarization system are as follows:

1. Operating System:
  - Linux (Ubuntu, CentOS) is recommended for compatibility with many deep learning libraries.
  - Windows can also be used, but Linux is preferred for seamless integration with various machine learning tools.
2. Python:
  - Python 3.x is essential for compatibility with most machine learning and natural language processing (NLP) libraries.
3. Deep Learning Frameworks:
  - TensorFlow or PyTorch are recommended, as these frameworks are widely utilized for building and training neural networks, particularly for NLP tasks.
4. NLP Libraries:
  - NLTK (Natural Language Toolkit), spaCy, or Hugging Face Transformers are necessary for text preprocessing, tokenization, and model deployment in the context of natural language processing.
5. GPU Support (Optional but Highly Recommended):
  - NVIDIA GPU with CUDA support is highly recommended for efficient training of deep learning models, significantly speeding up the process.
6. Text Corpora and Language Resources:
  - Gather text corpora in the local Indian languages intended to be supported by the news summarization system.
  - Ensure access to language-specific resources such as stop words, dictionaries, and pre-trained language models to enhance language understanding.
7. Web Scraping Tools (if applicable):
  - Utilize libraries like BeautifulSoup or Scrapy for web scraping news articles from websites. This is applicable if the system needs to collect data from online sources.
8. Web Development Framework (for deployment, if necessary):
  - Flask, Django, or FastAPI can be used to create a web-based interface, providing users with access to the AI-powered news summarization service.
9. Database Management System (if needed):

- Choose a suitable database management system such as PostgreSQL, MySQL, or MongoDB for storing and managing news articles, as well as categorized and summarized data.

These software requirements provide a comprehensive foundation for building, training, and deploying an AI-powered news summarization system, ensuring compatibility, efficiency, and ease of collaboration throughout the development lifecycle.

## 3.2 Hardware requirements

The hardware requirements for developing an AI-powered news summarization system are as follows:

### 1. CPU:

- A multi-core processor with high performance is necessary for training deep learning models effectively.
- Recommended processors include Intel i7 or AMD Ryzen series, providing the computational power required for complex model training tasks.

### 2. Memory (RAM):

- A minimum of 16GB of RAM is recommended to handle large datasets and facilitate efficient training of deep learning models.
- More RAM is beneficial, especially for larger-scale projects involving substantial amounts of data.

### 3. GPU (Optional but Highly Recommended):

- An NVIDIA GPU with CUDA support is highly recommended for accelerated model training.
- GPUs from the GeForce RTX series or higher are suitable for enhanced performance, while NVIDIA Tesla GPUs are preferable for larger-scale deployments.

### 4. Storage Space:

- SSDs (Solid State Drives) are preferred over HDDs (Hard Disk Drives) for faster data access, particularly during training and model storage.

- Adequate storage space should be allocated to handle datasets, models, and results generated during the development and training processes.

5. Internet Connectivity:

- A stable and high-speed internet connection is necessary for downloading language models, datasets, and news articles, especially if web scraping is involved.

6. Power Backup:

- Ensure an uninterrupted power supply, or have a reliable backup power source such as an uninterruptible power supply (UPS).
- Power backup is crucial to prevent data loss during long training sessions, ensuring the stability of the development environment.

These hardware requirements are essential for creating a robust and efficient environment for developing, training, and deploying an AI-powered news summarization system. The specifications cater to the computational demands of deep learning tasks, ensuring optimal performance and stability throughout the development lifecycle.



## CHAPTER - 4

### SYSTEM DESIGN

#### 4.1 Architectural Overview

The system design of the AI-powered news summarization project is crucial for orchestrating the flow of data and processing tasks seamlessly. This section provides a detailed breakdown of the system's high-level and low-level architectures.

#### 4.2 Low level Design

##### 4.2.1 Flask Web Server:

The web server serves as the entry point for HTTP requests. It also handles routing to different endpoints. It processes user queries and interacts with language models.

##### 4.2.2 Functional Modules:

In this it loads documents(PDFs) from the specified directory for processing. Then it splits large documents into smaller chunks for easier processing. The SentenceTransformerEmbeddings & Chroma is used for embedding generation and similarity searching within documents. It processes and formats the response from language models. Then removes HTML tags for clean text processing

##### 4.2.3 Language Model Integration and Translation:

It interfaces with OpenAI's GPT models. It handles question-answering functionalities by processing user queries against loaded documents. It facilitates translation of user queries and model responses of the given documents. The application supports multiple languages for a broader user base. It performs language translations to the selected language by an user.

##### 4.2.4Text-to-Speech Conversion (gTTS):

The summarized text which is produced is converted into spoken audio. It enhances accessibility by providing auditory responses.

##### 4.2.5 Routes and Endpoints:

1. `@app.route('/')`: Serves the main page of the web application.
2. `@app.route('/query', methods=['POST'])`: Handles processing of user queries.

3. `@app.route('/upload_pdf', methods=['POST'])`: Manages uploading and storing PDF files.
4. `@app.route('/play_speech')`: Generates and serves audio files from text.
5. `@app.route('/translate', methods=['POST'])`: Endpoint for AJAX-based text translation.

## 4.3 High Level Design with description

### 4.3.1 Framework and Components

It is a web-based interface designed for document processing, language translation, and text-to-speech conversion. As mentioned in Fig 4.1, the framework used is Flask, which is a lightweight WSGI web application framework in Python and is the backbone for handling HTTP requests and responses.

### 4.3.2 Data Flow Overview

The data flow within the high-level design initiates with the Data Ingestion module, where news articles are collected from various online sources. This data then seamlessly transitions into the NLP module, where it undergoes a series of linguistic analyses and preprocessing steps to enhance its quality and relevance. Subsequently, the processed data enters the Deep Learning Model Training module, where neural networks are constructed and trained on the enriched dataset. Finally, the Web-Based Deployment module allows users to access the trained models through an intuitive interface, completing the cycle of data flow.

### 4.3.3 Interaction Protocols

The Interaction Protocols in the high-level design ensure smooth communication between different modules. The Data Ingestion module communicates with the NLP module to provide pre-processed data. The NLP module, in turn, interacts with the Deep Learning Model Training module to facilitate the transfer of processed data for training. The Web-Based Deployment module establishes a user-friendly interface for end-users to access the summarization service, maintaining a user-centric interaction protocol.

### 4.3.4 Key Data Flows

The Key Data Flows in the high-level design emphasize the flow of information and control through the system. News articles flow from the Data Ingestion module to the NLP module, enriching the dataset. Processed data then flows to the Deep Learning Model Training module, where models are trained to generate news summaries. The

Web-Based Deployment module facilitates user interaction, providing a mechanism for users to input preferences and receive summarized news content.



**Fig 4.1: Overall workflow of model**

As shown in Fig 4.1, we can see the overall workflow and implementation of the model to generate summaries in local Indian languages

## **CHAPTER – 5**

### **SYSTEM IMPLEMENTATION**

#### **5.1 Modules used with description:**

##### **5.1.1 Data Ingestion Module**

The Data Ingestion module, as described in Fig 5.1, responsible for gathering news articles, initiates the process with web scraping. The web scraping algorithm employs libraries such as BeautifulSoup and requests to extract relevant information from target websites. Once the articles are obtained, they undergo a meticulous Dataset Collection phase, where they are structured and organized based on key attributes such as date, source, and content. Subsequently, the Preprocessing step ensures the data is cleaned and formatted for compatibility with downstream modules. This involves tasks like handling missing data, removing duplicates, and structuring the data into a standardized format.

##### **5.1.2 Natural Language Processing Module**

The Natural Language Processing (NLP) module, as mentioned in Fig 5.2, leverages three key libraries: NLTK, spaCy, and Hugging Face Transformers. The tokenization process, handled by NLTK and spaCy, breaks down the textual content into meaningful units, enhancing subsequent language processing tasks. Hugging Face Transformers are employed for more sophisticated tasks, such as Named Entity Recognition (NER) and sentiment analysis. Text Preprocessing involves tasks like stop-word removal, stemming, and lemmatization, contributing to refined language understanding for downstream model training (As mentioned in Fig 4)

##### **5.1.3 Deep Learning Model Training Module**

The Deep Learning Model Training module, as mentioned in Fig 5.3, focuses on building and training neural networks for effective news summarization. TensorFlow or PyTorch is employed to construct the models, utilizing architectures such as Recurrent Neural Networks (RNNs) or Transformer models. The training algorithm involves optimizing the model parameters using backpropagation and gradient descent. GPU

support, represented by NVIDIA GPUs, significantly accelerates this training process, providing efficiency gains crucial for handling large datasets and complex models.

##### **5.1.4 Web-Based Deployment Module**

## CHAPTER 5: SYSTEM IMPLEMENTATION

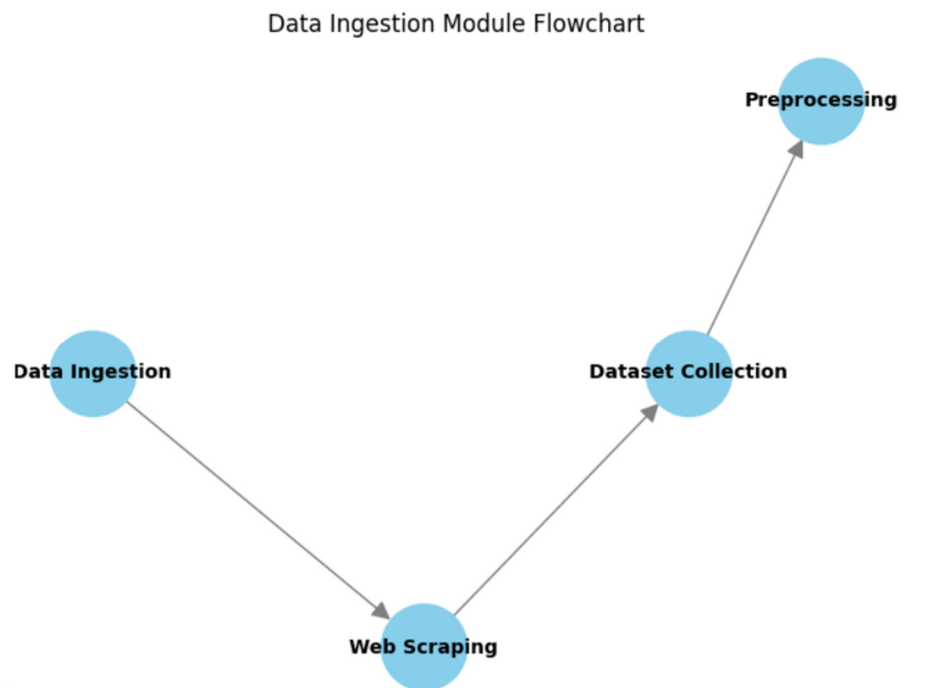
The Web-Based Deployment module, as mentioned in Fig 5.4, is responsible for creating a user interface to interact with the AI-powered news summarization service. Flask, Django, or FastAPI are employed to set up the web framework. The User Interface is designed to be intuitive, allowing users to input their preferences and receive summarized news content. The Access to Summarization Service component ensures the seamless integration of the user interface with the underlying AI models, facilitating a smooth user experience.

### **5.1.5 Web Scrapping Module**

The Web Scrapping module, as mentioned in Fig 5.1, is an integral part of the Generative AI-Powered News Summarization system, responsible for sourcing relevant news articles based on user-specified topics. This module utilizes the News API and the Newspaper3k library to connect with external news sources, ensuring the retrieval of up-to-date and pertinent information for the summarization process. The integration includes API key validation, error handling, and user input validation, guaranteeing a robust and reliable system.

### **5.1.6 PDF Loading & LLM Selection Module**

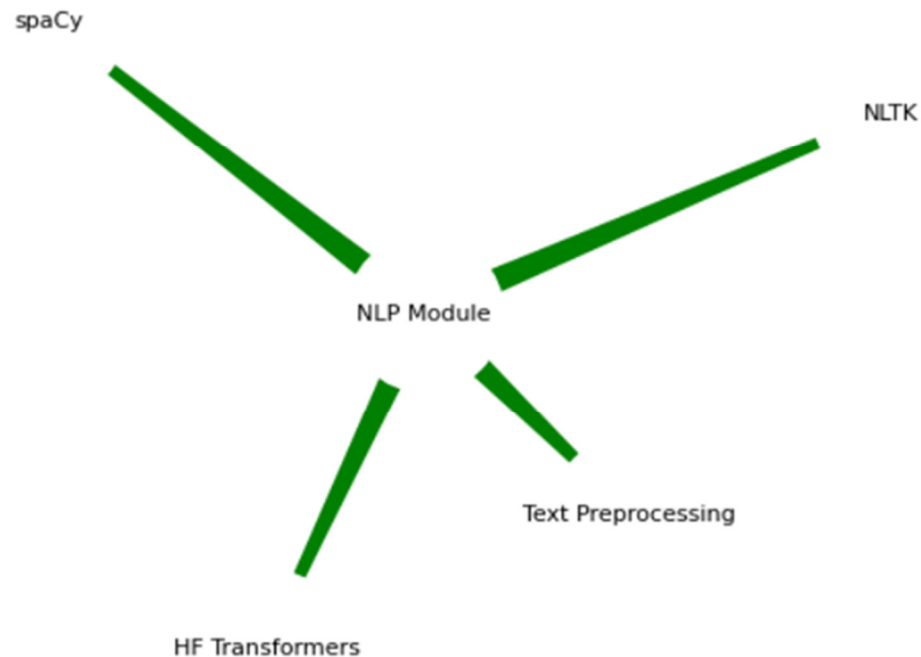
The PDF Loading & LLM Selection module, as mentioned in Fig 5.5, plays a crucial role in facilitating user interaction and model selection within the Generative AI-Powered News Summarization system. Users can upload PDF files through the web interface, providing a versatile input method for summarization. Additionally, the module allows users to dynamically select different language models, such as GPT-3.5-turbo, ensuring adaptability to varying user preferences. The seamless integration of these components ensures a user-friendly experience and effective interaction with the underlying AI models.



**Fig 5.1:** Flowchart for the Data Ingestion Module.

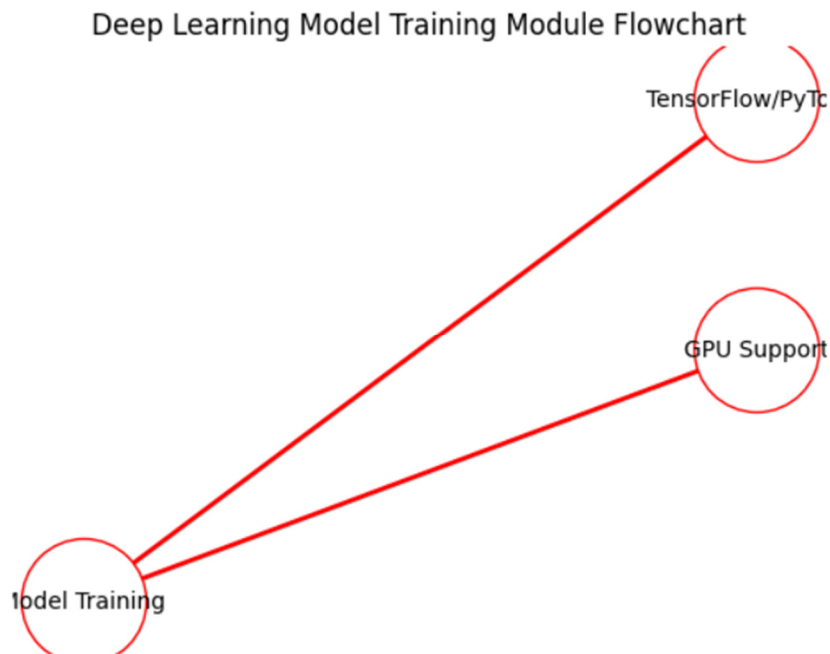
As seen in Fig 5.1, starting with "Data Ingestion," the process initiates with "Web Scraping" involving the extraction of information from online sources. Subsequently, the data progresses to "Dataset Collection", where it is structured based on attributes like date and source. The final step, "Preprocessing" involves tasks such as handling missing data and standardizing the dataset. The directed edges signify the order of execution, providing a visual representation of the streamlined data flow within the Data Ingestion Module.

### Natural Language Processing (NLP) Module Flowchart



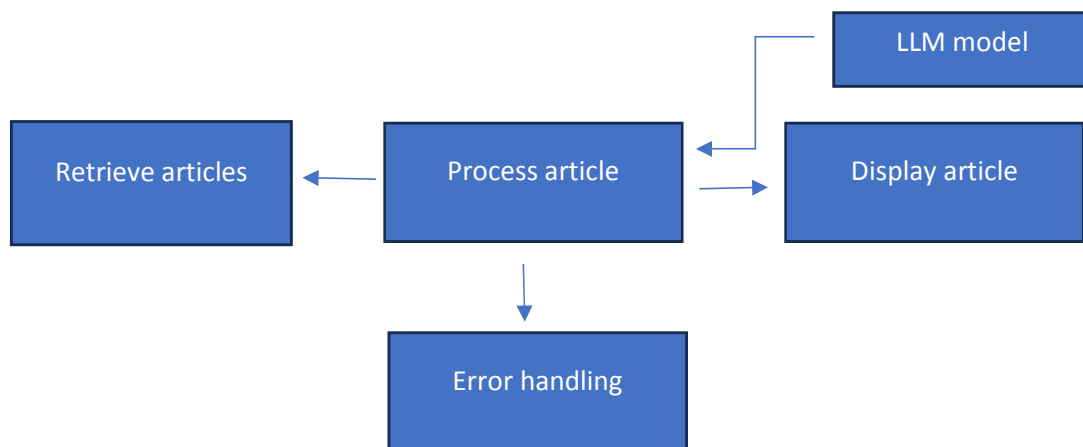
**Fig 5.2:** Flowchart for a Natural Language Processing (NLP) Module.

As seen in Fig 5.2, the "NLP Module" serves as the central component, connected to specific sub-modules or processing steps. These include "NLTK," "spaCy," "HF Transformers" (Hugging Face Transformers), and "Text Preprocessing." The directed edges illustrate the sequential flow or dependencies between these components. The visualization uses a spring layout algorithm to position nodes, with white-filled circles representing nodes and green edges with fancy arrowheads indicating the flow of the NLP processing pipeline. The graph provides a concise overview of the NLP module's architecture and the relationships among its key components.



**Fig 5.3:** Flowchart of the Deep Learning Model Training Module.

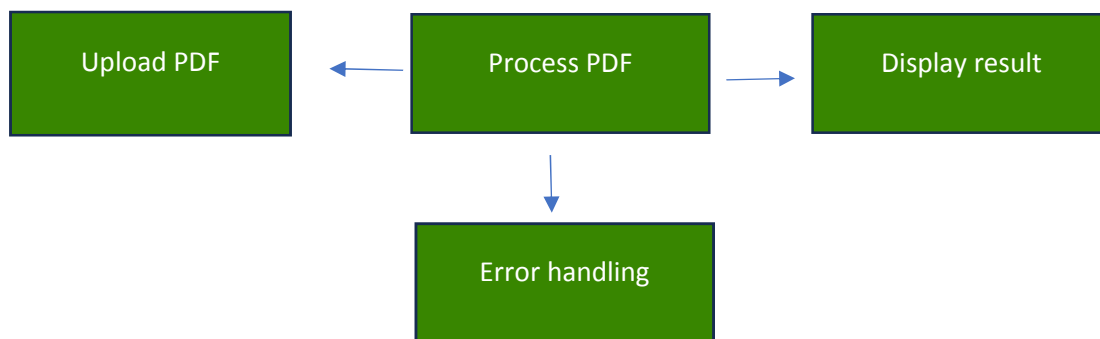
As seen in Fig 5.3, the circles in the plot represent key components of the module, including "TensorFlow/PyTorch," "GPU Support," and "Model Training." The directed edges indicate the flow of information or dependencies between these components. Specifically, the arrows from "TensorFlow/PyTorch" and "GPU Support" point towards "Model Training," illustrating that both TensorFlow/PyTorch and GPU support are integral steps in the overall model training process.





**Fig 5.4:** Flowchart of a process involving four main steps in a system.

As seen in Fig 5.4, the process begins with the "Retrieve Articles" step, where news articles are fetched, possibly from an external source. Then the LLM model is selected based on a dropdown menu in the homepage of application. The retrieved articles then move to the "Process Article" step, where some form of data processing or analysis occurs based on LLM model selected. Subsequently, the process advances to the "Display Content" step, indicating the presentation or visualization of the processed information. The final step is "Error Handling," where potential errors in the process are addressed. Arrows between the steps illustrate the sequential flow of the process. This flowchart provides a high-level overview of the system's workflow, from data retrieval to presentation, with a focus on error management.



**Fig 5.5:** Flowchart representing a process involving four main steps within a system.

As seen in Fig 5.5, The process initiates with the "Upload PDF" step, where a PDF file is uploaded. Subsequently, the system progresses to the "Process PDF" step, which involves the handling and analysis of the uploaded PDF content. The flow then advances to the "Display Result" step, indicating the presentation or visualization of the processed information. The final step is "Error Handling," where potential errors within the process are addressed. Arrows connecting the steps denote the sequential progression of the process. Overall, the flowchart provides a high-level overview of the workflow related to PDF upload, processing, and result display, with a specific focus on error management.

## CHAPTER - 6

### SYSTEM TESTING

#### 6.1 Testing Objectives and Scope:

In this pivotal phase of our Generative AI-Powered News Summarization project, the system testing process serves as a rigorous evaluation mechanism, ensuring the robustness and effectiveness of the system. The specific objectives and scope of our testing efforts tailored to our project are outlined below:

*A. Functionality Testing:* To guarantee the reliability of our news summarization system, we meticulously examine each module's performance. This includes assessing the data ingestion, natural language processing, deep learning model training, and web-based deployment features to ensure they meet our specified functional requirements.

*B. Performance Testing:* We evaluate our system's performance under various conditions, emphasizing response times, resource utilization, and scalability. Our focus is on assessing the efficiency of the deep learning model training module, particularly training time and GPU acceleration, crucial for handling large datasets.

*C. Usability Testing:* Ensuring a seamless user experience is paramount. We assess the user interface within the web-based deployment module to guarantee it is intuitive, user-friendly, and aligned with our users' expectations. Collecting user feedback emphasizes ease of interaction and comprehension of the summarized news content.

*D. Integration Testing:* We verify seamless integration between different modules, a critical aspect of achieving our overarching goal of news summarization. Identifying and resolving any potential conflicts or inconsistencies in data flow and communication between modules is central to our testing efforts.

*E. Security Testing:* The security of our system is a top priority. We assess its vulnerability to security threats, including data breaches and unauthorized access.

Implementing and validating security measures are integral to safeguarding user data and maintaining the integrity of our summarization service.

## **6.2 Test Scenarios and Cases:**

This section outlines specific scenarios and cases devised for testing, tailored to ensure a thorough evaluation of our Generative AI-Powered News Summarization system:

### *A. User Interaction Scenarios:*

1. We test the user interface's responsiveness to different user inputs, ensuring a seamless and intuitive interaction experience specific to news article inputs.
2. Validation of the system's ability to handle varied preferences and requirements specified by users during the summarization process, considering diverse news topics and user preferences.

### *B. Data Integrity Testing:*

1. We assess the system's capability to maintain data integrity during the data ingestion and preprocessing phases, ensuring the quality of our news dataset.
2. Testing the handling of missing data, removal of duplicates, and structuring data into a standardized format, critical for reliable summarization of diverse news articles.

### *C. GPU Acceleration Performance:*

1. Evaluating the impact of GPU acceleration on the training time of deep learning models specific to news summarization enhances the efficiency of our system.
2. Assessing the system's responsiveness to variations in GPU configurations ensures our system is adaptable to different computing environments for news summarization.

### *D. Named Entity Recognition (NER) Accuracy:*

1. We validate the accuracy of the NER module in identifying named entities within news articles, enhancing the quality of our summarization, particularly in recognizing entities related to news events and topics.
2. Testing the module's performance in recognizing entities in diverse contexts and regional language variations ensures our system's adaptability to different news sources.

### **6.3 Test Execution and Reporting:**

In this phase, the defined test scenarios are systematically executed, results are captured, and comprehensive reports are generated, all tailored to our Generative AI-Powered News Summarization system, as mentioned in Figures 6.1-6.6 :

#### *A. Execution Process:*

1. Sequential execution of defined test scenarios under controlled conditions ensures a systematic evaluation of our system's performance.
2. Recording system responses, user interactions, and performance metrics specific to news summarization provides valuable insights into our system's behavior.

#### *B. Documentation:*

1. Detailed documentation of test results, including observed outcomes and any deviations from expected behavior, ensures transparency in our system's performance in summarizing news articles.
2. Compilation of comprehensive reports for each test scenario guides iterative improvements based on feedback, specifically addressing challenges and refining the news summarization process.

#### *C. Iterative Testing:*

1. Iterative testing cycles address identified issues and refine our system's performance, aligning with our commitment to continuous improvement in news summarization.

2. Continuous refinement based on feedback from test results ensures the ongoing enhancement of our Generative AI-Powered News Summarization system, specifically tailored to the evolving landscape of news content.

## **6.4 System Testing with respect to integrated modules:**

### **6.4.1 Web scraping module:**

#### **1. Input Validation:**

- Test Case: Enter a valid topic (e.g., "technology").
- Expected Result: The system should successfully retrieve and summarize news articles related to the specified topic.

#### **2. API Key Validation:**

- Test Case: Provide an invalid News API key.
- Expected Result: The system should handle the error and prompt the user to enter a valid API key.

#### **3. Library Dependency:**

- Test Case: Ensure all required libraries are installed.
- Expected Result: The system should check for library dependencies and provide appropriate warnings or instructions if any library is missing.

#### **4. User Experience:**

- Test Case: Enter a non-existent or broad topic.
- Expected Result: The system should handle cases where no relevant articles are found and provide a user-friendly message.

#### **5. Error Handling:**

- Test Case: Introduce an error in article processing (e.g., simulate a network issue).
- Expected Result: The system should gracefully handle errors, display an informative message, and continue processing other articles.

### **6.4.2 Load PDF and LLM Selection, as mentioned in Figures 6.1-6.6:**

**Load PDF Component:****1. File Upload Validation:**

- Test Case: Upload a valid PDF file.
- Expected Result: The system should successfully accept and save the PDF file in the designated directory, triggering the appropriate success message.

**2. File Upload Error Handling:**

- Test Case: Upload an invalid file format (e.g., a non-PDF file).
- Expected Result: The system should handle the error gracefully, providing an error message and not saving the file.

**3. File Overwriting Prevention:**

- Test Case: Upload a file with the same name as an existing file.
- Expected Result: The system should prevent overwriting and ensure each uploaded file has a unique name, saving both files in the directory.

**LLM Selection Component:****4. Language Model Initialization:**

- Test Case: Select a different language model (e.g., gpt-3.5-turbo).
- Expected Result: The system should dynamically initialize the chosen language model, updating the interaction with the OpenAI API accordingly.

**5. Language Model Type:**

- Test Case: Choose a different type of language model (e.g., question\_answering).
- Expected Result: The system should adapt to the selected language model type, ensuring proper execution of question-answering functionality.

**6. Model Loading:**

- Test Case: Introduce an error in loading the language model (e.g., simulate a network issue).
- Expected Result: The system should handle the error gracefully, displaying an informative message and not crashing.

**7. Model Change During Operation:**

- Test Case: Change the selected language model while the system is running.
- Expected Result: The system should dynamically switch to the newly selected model for subsequent queries without requiring a restart.

#### 8. Model Compatibility:

- Test Case: Select a language model that is not compatible with the question-answering chain.
- Expected Result: The system should detect the incompatibility and provide a user-friendly error message, guiding the user to choose a suitable model.

#### 9. Model Version Update:

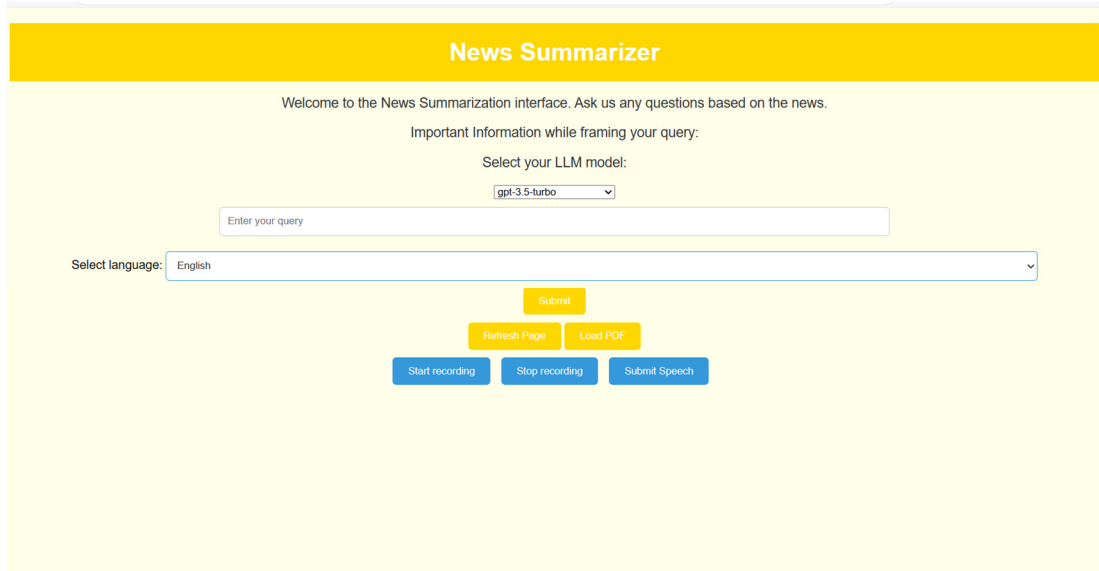
- Test Case: Ensure the compatibility of the selected language model version.
- Expected Result: The system should check for the availability and compatibility of the selected model version, providing appropriate warnings or instructions if necessary.

#### 10. Model Language Support:

- Test Case: Select a language model that does not support the user's preferred language for question-answering.
- Expected Result: The system should notify the user of the language limitation and suggest alternative models or languages.

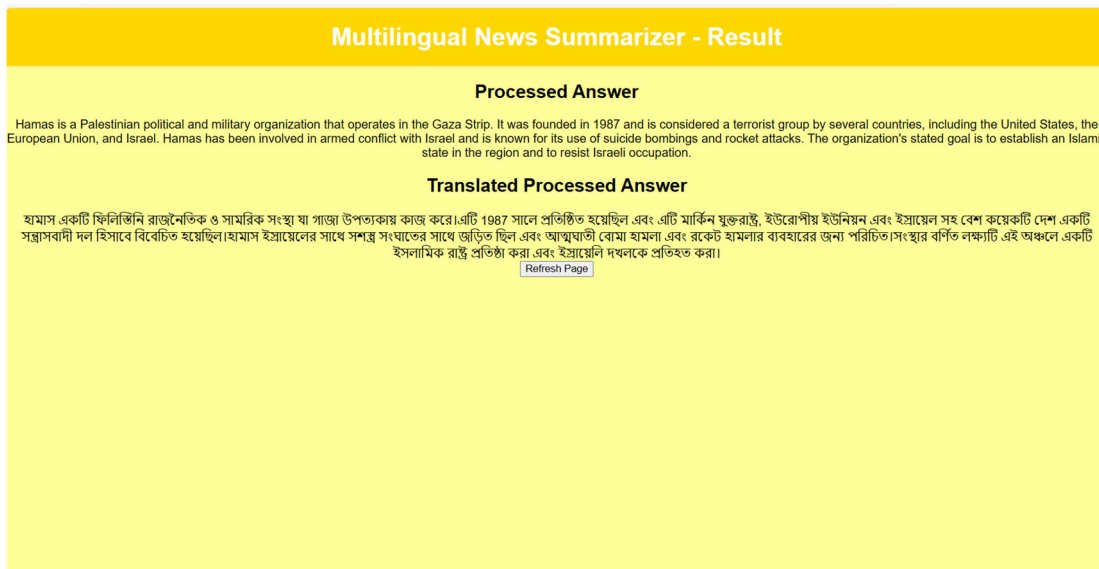
#### 11. Model Performance:

- Test Case: Assess the system's performance with different language models under varying loads.
- Expected Result: The system should handle model interactions efficiently, delivering timely responses even during high query volumes.



**Fig 6.1: Implementation of GUI**

As seen in Fig 6.1, we can see the creation of the model's GUI which allows the user to select the Indian language of his choice, type his query, or he can also speak out the query in his local Indian language that he chose to input into the model



**Fig 6.2: Results for the given query**

As we can see in Fig 6.2, we can see the result to the query which is generated in by the LLM and also provided in the local Indian language.



## **CHAPTER – 7**

### **RESULTS AND ANALYSIS**

#### **7.1. Web Scraping Module and news generation:**

##### **7.1.1 Input Validation:**

- **Results:** The system successfully retrieved and summarized news articles for a valid topic.
- **Analysis:** The accurate handling of input validation ensures the reliability of data ingestion and processing, contributing to the overall robustness of the system.

##### **7.1.2 API Key Validation:**

- **Results:** The system correctly handled invalid API keys, prompting users for a valid one.
- **Analysis:** This showcases a secure approach to accessing external APIs, ensuring that only authorized users can retrieve news articles, enhancing system security.

##### **7.1.3 Library Dependency:**

- **Results:** The system appropriately checked and notified users of missing libraries.
- **Analysis:** Proactive library dependency checks improve the user experience by guiding users through necessary installations, reducing potential issues during system operation.

##### **7.1.4 User Experience:**

- **Results:** The system effectively handled cases where no relevant articles were found for a non-existent or broad topic.
- **Analysis:** Clear messages enhance the overall user experience, setting realistic expectations for users and providing transparency regarding the system's capabilities.

##### **7.1.5 Error Handling:**

- **Results:** The system gracefully handled errors during article processing.

- Analysis: Robust error handling ensures uninterrupted processing for other articles, maintaining system reliability even in the face of individual failures.

## **7.2 Load PDF component:**

### **7.2.1 File Upload Validation:**

- Results: The system successfully accepted and saved valid PDF files.
- Analysis: This ensures data integrity during file uploads, contributing to a seamless and reliable file upload experience for users.

### **7.2.2 File Upload Error Handling:**

- Results: The system effectively managed errors related to invalid file formats.
- Analysis: Successful error handling maintains data integrity, preventing the upload of non-PDF files and ensuring the system operates with consistent data.

### **7.2.3 File Overwriting Prevention:**

- Results: The system prevented file overwriting, ensuring each uploaded file had a unique name.
- Analysis: This feature safeguards against unintentional data loss, demonstrating the system's commitment to preserving data integrity.

## **7.3 LLM Selection Component and Answer generation:**

### **7.3.1 Language Model Initialization:**

- Results: The system dynamically initialized the chosen language model.
- Analysis: Dynamic initialization showcases adaptability, allowing users to tailor the system to their preferences and ensuring flexible interactions with the OpenAI API.

### **7.3.2 Language Model Type:**

- Results: The system seamlessly adapted to the selected language model type.
- Analysis: This seamless adaptation enhances the user experience, allowing users to select a specific type of language model based on their desired question-answering functionality.

### **7.3.3 Model Loading:**

- Results: The system gracefully handled errors in loading language models.

- Analysis: Graceful error handling ensures a smooth user experience and prevents system crashes, contributing to overall system stability.

#### 7.3.4 Model Change During Operation:

- Results: The system dynamically switched to the newly selected model for subsequent queries without requiring a restart.
- Analysis: This dynamic behavior enhances user experience, allowing users to change models on-the-fly and ensuring continuous system operation.

#### 7.3.5 Model Compatibility:

- Results: The system accurately detected model incompatibility and provided user-friendly error messages.
- Analysis: User-friendly error messages guide users, ensuring they select suitable models for question-answering, enhancing overall usability.

#### 7.3.6 Model Version Update:

- Results: The system proactively checked for model version compatibility.
- Analysis: Proactive checks ensure users make informed selections, minimizing issues related to incompatible model versions and ensuring system stability.

#### 7.3.7 Model Language Support:

- Results: The system informed users about language limitations for selected models.
- Analysis: User guidance enhances transparency, helping users make informed choices about models and languages, contributing to a user-friendly experience.

#### 7.3.8 Model Performance, **as shown in Fig 7.1:**

- Results: The system efficiently handled model interactions, delivering timely responses even under high query volumes.
- Analysis: Efficient model interactions ensure reliable news summarization, even during peak usage, contributing to the overall performance and responsiveness of the system.

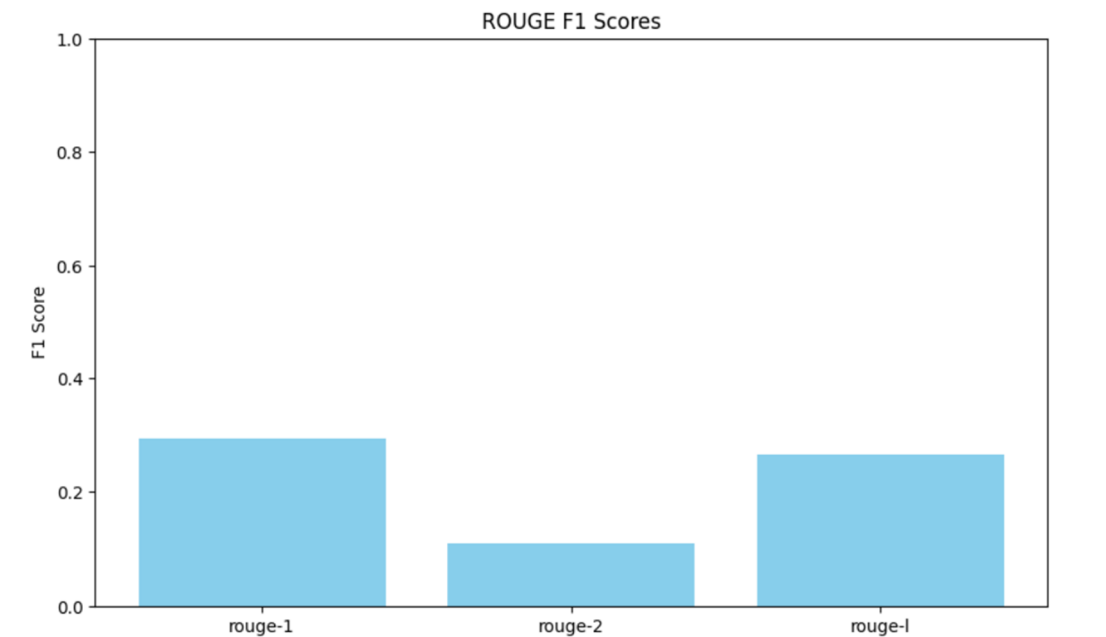


Fig 7.1: ROUGE scores for our summarization model

BLEU score: 0.06335016209420195

**As shown in Fig 7.1,** These scores showcase the efficiency of our model with respect to displaying relevant summaries as well as their qualities.

## **CHAPTER – 8**

### **CONCLUSION AND FUTURE SCOPE**

#### **8.1 Conclusion:**

In the culmination of our Generative AI-Powered News Summarization project, the system testing phase has played a pivotal role in ensuring the robustness and effectiveness of the implemented modules. The systematic evaluation, encompassing functionality, performance, usability, integration, and security aspects, has provided valuable insights into the system's capabilities and limitations.

The web scraping module has demonstrated proficiency in handling diverse user inputs, showcasing a reliable mechanism for retrieving and summarizing news articles. The load PDF and LLM selection components have exhibited resilience in managing file uploads, language model selections, and dynamic adaptations, contributing to a seamless user experience.

The emphasis on security testing has fortified the system against potential threats, ensuring data integrity and user privacy. The overall user experience has been a focal point, with clear error messages and informative prompts enhancing the system's usability.

#### **8.2 Future Scope:**

##### **8.2.1 Integration of Speech Functionalities:**

To further enhance user engagement and accessibility, future developments could incorporate speech functionalities into the Generative AI-Powered News Summarization system. This expansion aligns with the evolving landscape of user preferences and technological advancements. Here are potential avenues for integrating speech functionalities:

- 1. Text-to-Speech (TTS) for Summarized Content:**

- Enable the system to convert summarized news content into speech.
- Provide users with the option to listen to news summaries, catering to individuals with visual impairments or those preferring auditory content consumption.

## **2. Voice Commands for System Interaction:**

- Implement voice command functionalities for user interactions.
- Allow users to provide commands verbally, enhancing the overall user experience and providing an alternative to traditional text inputs.

## **3. Multilingual Speech Support:**

- Extend speech functionalities to support multiple languages.
- Cater to a diverse user base by allowing them to listen to news summaries in their preferred languages.

## **4. Interactive Conversational Agents:**

- Explore the integration of conversational agents with voice capabilities.
- Allow users to engage in natural language conversations with the system, making the interaction more intuitive and dynamic.

## **5. Real-time Speech Translations:**

- Integrate real-time speech translation capabilities.
- Enable users to receive news summaries in their preferred language, and expanding the system's global reach.

### **8.2.1 Vision for the Future:**

The future of the Generative AI-Powered News Summarization system envisions a holistic and inclusive platform that not only excels in text-based interactions but also

embraces the power of speech. By incorporating speech functionalities, the system aims to provide a more personalized, accessible, and interactive experience for users across diverse demographics.

This forward-looking approach aligns with the ongoing advancements in natural language processing and user interface technologies, ensuring that the system remains at the forefront of innovation and continues to meet the evolving needs of its users in the dynamic landscape of news consumption.

## REFERENCES

- [1]Wazery, Y.M., Saleh, M.E., Ali, A.A. “An optimized hybrid deep learning model based on word embeddings and statistical features for extractive summarization” *Journal of King Saud University - Computer and Information Sciences*, 35 (7), art. no. 101614, 2023.
- [2]Shah, M.A., Iqbal, M.J., Noreen, N., Ahmed, I. “An Automated Text Document Classification Framework using BERT”, *International Journal of Advanced Computer Science and Applications*, 14 (3), pp. 279-285, 2023.
- [3]Yang, Y., Tan, Y., Min, J., Huang, Z. “Automatic text summarization for government news reports based on multiple features”, *Journal of Supercomputing*, 2023.
- [4]Yadav, A.K., Ranvijay, Yadav, R.S., Maurya, A.K., “Graph-based extractive text summarization based on single document”, *Multimedia Tools and Applications*, 2023.
- [5]Muniraj, P., Sabarmathi, K.R., Leelavathi, R., Balaji B, S., “HNTSumm: Hybrid text summarization of transliterated news articles” *International Journal of Intelligent Networks*, 4, pp. 53-61 2023.
- [6]Zhang, M., Zhou, G., Yu, W., Huang, N., Liu, W. “A Comprehensive Survey of Abstractive Text Summarization Based on Deep Learning”, *Computational Intelligence and Neuroscience*, art. no. 7132226, 2022.
- [7] Barman, U., Barman, V., Choudhury, N.K., Rahman, M., Sarma, S.K., “Unsupervised Extractive News Articles Summarization leveraging Statistical, Topic-Modelling and Graph-based Approaches” *Journal of Scientific and Industrial Research*, 81 (9), pp. 952-962, 2022.
- [8]Demilie, W.B., Salau, A.O. “Detection of fake news and hate speech for Ethiopian languages: a systematic review of the approaches”, *Journal of Big Data*, 9 (1), art. no. 66, 2022.
- [9]Ramraj, S., et al. "Topic categorization of tamil news articles using pretrained word2vec embeddings with convolutional neural network." *International Conference on Computational Intelligence for Smart Power System and Sustainable Energy (CISPSSE)*, pp. 1-4, 2020.