**MEDIA MONITORING AND ENHANCEMENT HUB**

**A PROJECT REPORT**

***Submitted by***

**HARIHARAN S [211421104088]**

**AJAY S [211421104013]**

**ARJUN V RA [211421104025]**

***in partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

**IN**



**COMPUTER SCIENCE AND ENGINEERING**

**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

APRIL 2025

**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

**BONAFIDE CERTIFICATE**

Certified that this project report **“MEDIA MONITORING AND ENHANCEMENT HUB”** is the bonafide work of **“HARIHARAN S [211421104088], AJAY S [211421104013]** and **ARJUN V RA [211421104025]”**who carried out the project work under my supervision.

|  |  |
| --- | --- |
| **Dr. L. JABASHEELA, M.E., Ph.D.,**  **PROFESSOR AND HEAD,** | **Dr. L. JABASHEELA, M.E., Ph.D.,**  **SUPERVISOR,** |
| Department of Computer Science and Engineering,  Panimalar Engineering College,  Chennai – 123. | Department of Computer Science and Engineering,  Panimalar Engineering College,  Chennai – 123. |

Certified that the above candidate(s) was examined in the End Semester Project Viva - Voce Examination held on ..............................

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**DECLARATION BY THE STUDENT**

We **HARIHARAN S [211421104088], AJAY S [211421104013] and ARJUN V RA [211421104096]** hereby declare that this project report titled **“MEDIA MONITORING AND ENHANCEMENT HUB”** , under the guidance of **Dr. JABASHEELA** is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

HARIHARAN S [211421104088]

AJAY S [211421104013]

ARJUN V RA [211421104025]

**ACKNOWLEDGEMENT**

Our profound gratitude is directed towards our esteemed Secretary and Correspondent, **Dr. P. CHINNADURAI, M.A., Ph.D.,** for his fervent encouragement. His inspirational support proved instrumental in galvanizing our efforts, ultimately contributing significantly to the successful completion of this project.

We want to express our deep gratitude to our Directors, **Tmt. C. VIJAYARAJESWARI, Dr. C. SAKTHI KUMAR, M.E., Ph.D.,** and **Dr. SARANYASREE SAKTHI KUMAR, B.E., M.B.A., Ph.D.,** for graciously affording us the essential resources and facilities for undertaking of this project.

Our gratitude is also extended to our Principal, **Dr. K. MANI, M.E., Ph.D.,** whose facilitation proved pivotal in the successful completion of this project.

We express our heartfelt thanks to **Dr. L. JABASHEELA, M.E., Ph.D.,** Head of the Department of Computer Science and Engineering, for granting the necessary facilities that contributed to the timely and successful completion of project.

We would like to express our sincere thanks to Project Coordinator **Dr. L. JABASHEELA, M.E., Ph.D.,** and Project Guide **Dr. G. SENTHIL KUMAR, M.C.A., M.Phil., M.B.A., M.E., Ph.D.,** and all the faculty members of the Department of CSE for their unwavering support for the successful completion of the project.

**HARIHARAN S [211421104088]**

**AJAY S [211421104013]**

**ARJUN V RA [211421104025]**

**ABSTRACT**

The "Media Monitoring and Feedback System for the Government of India" is a sophisticated technological solution designed to bridge the gap between government authorities and the vast landscape of print, electronic, and digital media. It employs cutting-edge technologies like web scraping, natural language processing, sentiment analysis, OCR, and audio-to-text conversion to track and analyse media coverage across various languages and sources, providing real-time insights to government officials. key functionalities include automated data collection from regional news websites and YouTube channels, data processing for translation, categorization, and sentiment analysis, centralized data storage with an intuitive dashboard, a real-time notification system for negative news content, video analysis, stringent data security measures, scalability, quality assurance, training programs, and continuous support.

Overall, this system empowers the Government of India with a powerful tool for monitoring media coverage, gaining actionable insights, and facilitating timely decision-making, embodying transparency, responsiveness, and a proactive approach to managing public perception and media interactions.

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE NO.** | **TABLE NAME** | **PAGE NO.** |
|  |  |  |
| 3.1 | Content Contibutor Table | 18 |
| 3.2 | Media Content Table | 19 |
| 3.3 | User Table | 19 |
| 3.4 | Search & Feedback Table | 20 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **FIGURE NAME** | **PAGE NO.** |
| 3.1 | Architecture Diagram | 15 |
| 3.2 | Search UI | 21 |
| 4.1 | Sequence Diagram | 23 |
| 4.2 | Activity Diagram | 24 |
| 4.3 | Collaboration Diagram | 25 |
| A.3.1 | Landing Page | 59 |
| A.3.2 | User Login | 59 |
| A.3.3 | Dashboard UI | 60 |
| A.3.4 | Youtube Sentiment Analysis UI | 60 |
| A.3.5 | Youtube Sentiment Analysis Result UI | 61 |
| A.3.6 | PDF Sentiment Analysis UI | 61 |
| A.3.7 | PDF Sentiment Analysis Result UI | 62 |
| A.4.8 | Plagiariam Report | 62 |
| A.4.9 | Paper Publication | 63 |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **TITLE** | **PAGE NO** |
|  | **ABSTRACT** | **v** |
|  | **LIST OF TABLES** | **vi** |
|  | **LIST OF FIGURES** | **vii** |
|  | **LIST OF ABBREVATIONS** | **viii** |
| **1.** | **INTRODUCTION** | **1** |
|  | 1.1. Problem Definition | 1 |
| **2.** | **LITERATURE SURVEY** | **4** |
| **3.** | **THEORETICAL BACKGROUND** | **14** |
|  | 1. Implementation Environment | 14 |
|  | 1. System Architecture | 15 |
|  | 1. Proposed Methodology | 18 |
| **4.** | **SYSTEM IMPLEMENTATION** | **23** |
|  | 1. System Design | 23 |
|  | 1. Module Descriptions | 28 |
| **5.** | **RESULTS & DISCUSSION** | **34** |
|  | 1. Testing | 34 |
| **6.** | **CONCLUSION** | **44** |
|  | Future Enhancement | 45 |
|  | **APPENDICES** | **47** |
|  | 1. SDG Goals | 47 |
|  | 1. Source Code | 48 |
|  | 1. Screen Shots | 59 |
|  | 1. Plagiarism Report | 66 |
|  | 1. Paper Publication | 67 |

**CHAPTER 1**

**INTRODUCTION**

**CHAPTER 1**

**INTRODUCTION**

The Media Monitoring and Feedback System for the Government of India is a sophisticated technological solution aimed at enhancing the government's ability to track, analyze, and respond to media coverage across diverse platforms, including print, electronic, and digital media. In an era where information spreads rapidly, it is crucial for government authorities to stay informed about public sentiment, news trends, and media narratives. This system leverages advanced technologies such as web scraping, natural language processing (NLP), sentiment analysis, optical character recognition (OCR), and audio-to-text conversion to provide real-time insights into media coverage in multiple languages. The system automates data collection, translation, categorization, and sentiment analysis, offering government officials an intuitive dashboard for centralized data storage and analysis. Additionally, it ensures real-time notifications for critical news updates, particularly negative or controversial content that requires immediate attention. To maintain data security and scalability, the system incorporates robust encryption methods and authentication mechanisms, ensuring the confidentiality of sensitive government data. By implementing this system, the Government of India can proactively monitor public perception, identify emerging issues, and refine its communication strategies. This leads to improved governance, timely decision-making, and stronger public engagement, reinforcing the principles of transparency and responsiveness in government operations.

* 1. **Problem Definition**

The primary challenge addressed by this project is the efficient monitoring and analysis of vast amounts of media data in real time. The Government of India requires a comprehensive and automated system capable of tracking news trends, detecting misinformation, and assessing public sentiment across multiple sources and languages. Traditional manual media monitoring approaches are inefficient, time-consuming, and prone to biases, making it difficult for government agencies to respond swiftly to critical issues. With an overwhelming volume of news articles, social media posts, and video content being published daily, manual tracking becomes impractical. Additionally, news is reported in multiple regional languages, necessitating automated translation and analysis for complete and accurate coverage. Understanding the tone and intent behind media coverage requires advanced NLP techniques to differentiate between positive, neutral, and negative sentiments, while the rise of misinformation and disinformation in digital media demands robust fact-checking mechanisms to verify content credibility. Ensuring the confidentiality, integrity, and authenticity of media data while complying with government regulations is another critical challenge. Furthermore, the government requires instant notifications about sensitive or controversial media content to facilitate timely interventions. To address these challenges, this project proposes a centralized, AI-driven media monitoring system with real-time data analysis, multilingual support, sentiment assessment, and enhanced security measures. The integration of web scraping, OCR, speech-to-text conversion, and blockchain verification ensures reliable and actionable insights for government decision-makers. By implementing this solution, the Government of India can enhance its media intelligence capabilities, make data-driven policy decisions, and adopt a proactive approach in addressing public concerns and shaping its communication strategies effectively.

**CHAPTER 2**

**LITERATURE REVIEW**

**CHAPTER 2**

**LITERATURE REVIEW**

**1.** S. S. Korti and S. G. Kanakaraddi, "Depression Detection from Twitter Posts Using NLP and Machine Learning Techniques," 2022.

The paper focuses on identifying signs of depression in Twitter posts by employing Natural Language Processing (NLP) and machine learning techniques. The study demonstrates how social media text data can serve as a valuable resource for early detection of mental health conditions. It utilizes sentiment analysis, lexical feature extraction, and supervised learning models to classify depressive and non-depressive posts. Experimental results show high accuracy in detecting depression indicators, making the approach useful for real-world mental health monitoring. Compared to lexicon-based sentiment analysis, which relies on predefined word lists, this method leverages machine learning to adapt to context variations and improves classification accuracy. While lexicon-based approaches offer interpretability, the proposed model achieves better performance in detecting nuanced depressive expressions.

**2.** A. Dixit, S. Sharma, P. D. Rao, V. Reddy, M. Janaki, R. Thirumalaivasan, and M. M. Subashini, "Audio to Indian and American Sign Language Converter Using Machine Translation and NLP Technique," 2022.

This work presents a system that converts audio input into sign language representation for both Indian and American Sign Languages. It integrates automatic speech recognition (ASR), Natural Language Processing (NLP), and machine translation to facilitate communication for the hearing and speech impaired. The model maps speech inputs to linguistic representations before rendering them as animated sign gestures. Experimental results validate the system’s efficiency in translating spoken language into sign language with high accuracy. Compared to gesture recognition-based systems, which interpret hand movements directly from videos, this approach enables a broader range of linguistic expressions and is better suited for real-time applications. While gesture-based models are effective for capturing visual sign language expressions, the proposed system provides a more scalable and automated translation framework.

**3.** K. Kavita and H. Singh, "Utilizing Mixture Methods for Classifier in NLP: An Essential Consideration," 2023.

The study explores the use of hybrid or mixture methods for classifiers in NLP tasks, highlighting their effectiveness in achieving greater accuracy and robustness in linguistic processing applications. It examines ensemble learning techniques such as stacking, bagging, and boosting to combine multiple classifiers and enhance predictive performance. The proposed approach demonstrates improved accuracy in text classification tasks compared to traditional single-model classifiers. Compared to deep learning-based NLP models, which require extensive computational resources and large datasets, mixture models provide a more interpretable and computationally efficient solution. While deep learning techniques excel in feature learning, the hybrid approach balances performance and explainability, making it suitable for real-world NLP applications.

**4.** V. M. Reddy, T. Vaishnavi, and K. P. Kumar, "Speech-to-Text and Text-to-Speech Recognition Using Deep Learning," 2023.

This paper discusses the development of a bidirectional system capable of converting speech to text and text to speech using deep learning models. The system employs automatic speech recognition (ASR) for transcription and text-to-speech (TTS) synthesis for voice output. It has significant applications in accessibility tools and voice assistants. The proposed approach achieves high accuracy and intelligibility in both speech recognition and synthesis. Compared to Hidden Markov Model (HMM)-based speech recognition systems, which rely on statistical modeling, deep learning-based approaches offer greater robustness to variations in speech patterns. While HMM models are computationally efficient, deep learning techniques significantly improve recognition accuracy and naturalness in synthesized speech.

**5.** J. Seong, W. Lee, and S. Lee, "Multilingual Speech Synthesis for Voice Cloning," 2021.

This paper explores techniques for multilingual speech synthesis, focusing on voice cloning. It highlights advancements in creating natural and personalized synthetic voices across multiple languages using deep learning-based text-to-speech (TTS) models. The proposed method leverages speaker embeddings and transfer learning to generate synthetic speech with high fidelity. Compared to traditional concatenative TTS systems, which rely on pre-recorded speech segments, deep learning-based TTS models achieve greater flexibility and adaptability. While concatenative synthesis produces highly natural speech for specific speakers, the proposed approach enables voice adaptation across multiple languages, enhancing accessibility and user personalization.

**6.** M. Siek and E. S. Setiadi, "Analysis of News Sentiment and Stock Price Using Web Scraping and Vader Sentiment Analysis," 2024.

This study examines the relationship between news sentiment and stock price fluctuations using web scraping and Vader Sentiment Analysis. It collects financial news articles and applies sentiment analysis to predict stock market trends. The findings indicate a correlation between sentiment polarity and stock price movements, demonstrating the potential for sentiment-driven financial predictions. Compared to machine learning-based sentiment analysis models, which require labeled training data, Vader is optimized for real-time applications and does not require extensive training. While machine learning models offer greater accuracy with large datasets, Vader provides a lightweight and efficient solution for financial sentiment analysis.

**7.** S. Kaliappan, L. Natrayan, and A. Rajput, "Sentiment Analysis of News Headlines Based on Sentiment Lexicon and Deep Learning," 2023.

This paper delves into sentiment analysis of news headlines, employing sentiment lexicons and deep learning methodologies. The study improves sentiment classification accuracy by combining lexicon-based techniques with deep learning models such as LSTMs and transformers. Results show a significant improvement in detecting sentiment polarity in news headlines. Compared to rule-based sentiment classification, which relies on predefined lexicons, deep learning-based methods adapt dynamically to contextual variations, leading to higher accuracy. While rule-based systems offer better interpretability, deep learning models excel in handling complex linguistic structures, making them more suitable for large-scale sentiment analysis.

**8.** N. M. K. Varma, S. H. Mattaparty, S. Ismail, J. Thaduri, G. D. Arora, and A. B. AnandKumar, "Sentiment Analysis: A Machine Learning Perspective," 2024.

This work provides a comprehensive analysis of sentiment classification using machine learning techniques. It explores supervised and unsupervised learning models, emphasizing their application in finance, media, and customer feedback analysis. The proposed approach outperforms traditional lexicon-based methods in accuracy and robustness. Compared to frequency-based sentiment analysis techniques, which rely on term occurrences, machine learning-based approaches leverage contextual relationships for improved sentiment classification. While frequency-based methods are computationally efficient, machine learning models provide greater adaptability, making them more effective for complex sentiment analysis tasks.

**CHAPTER 3**

**THEORETICAL BACKGROUND**

**CHAPTER 3**

**THEORETICAL BACKGROUND**

**3.1. IMPLEMENTATION ENVIRONMENT**

The project is built using Flask for backend development, with MongoDB as the database and MongoDB Compass for visualization. NLP and sentiment analysis are powered by spaCy and nltk, while pytesseract handles OCR for PDF text extraction. Moviepy processes YouTube videos, and AssemblyAI transcribes audio. Twilio enables real-time notifications. Secure authentication is implemented within Flask, ensuring efficient, scalable, and AI-driven content analysis.

**3.1.1 HARDWARE REQUIREMENTS**

* Hard Disk: 500GB and Above
* RAM: 8GB and Above
* Processor: Intel i5 and Above

**3.1.2 SOFTWARE REQUIREMENTS**

* OperatingSystem: Windows 10 and above
* Pythonenvironment**:** FLASK
* Programming Language: Python
* Database: MongoDB
* IntegratedDevelopmentEnvironment **(**IDE**)**: Visual Studio Code
* Natural Language Processing Libraries: spaCy, NLTK
* WebScrapingTools: BeautifulSoup, Scrapy

**3.1.3 TECHNOLOGIES USED**

* Framework: Flask
* TemplateEngine: Thymeleaf
* MachineLearning: TensorFlow, Scikit-learn
* JavaScriptRuntime: Node.js
* Database: MySQL 8.0
* Blockchain: Solidity (for secure data transactions)
* DistributedFileSystem: IPFS
* OCRTool: Tesseract
* Audio**-**to-TextConversion: Google Speech-to-Text API

**3.2. SYSTEM ARCHITECTURE**

The system architecture of this project is designed to analyze YouTube videos, extract text from PDFs, and perform sentiment analysis while ensuring efficient data processing and secure storage. Users upload video links or PDF files, which are processed using Optical Character Recognition (OCR) and Natural Language Processing (NLP) techniques. The extracted text undergoes sentiment analysis to determine the overall tone. MongoDB is used to store extracted text, sentiment scores, and metadata for efficient retrieval. The Flask framework handles user requests, processes data, and interacts with the database. Speech-to-text conversion is performed using AssemblyAI, while MoviePy assists in video processing. Web scraping tools like BeautifulSoup and Scrapy are used to gather additional relevant data. A notification system powered by Twilio ensures users receive timely updates. By integrating these technologies, the system offers a streamlined approach to analyzing video and document content while maintaining data integrity and accessibility.



*FIGURE 3.2.1 Architecture Diagram*

**3.2.1 Input Handling**

The system accepts inputs from two main sources: content contributors and media analysts/users. Contributors upload media files (videos, audio, or text), which are processed for transcription, keyword extraction, sentiment analysis, and tonality assessment. Extracted keywords are stored securely, and a blockchain proof ensures the integrity of the content. Users interact with the system by querying based on keywords or sentiment-based filters. These inputs are matched against stored metadata, and if a match is found, the relevant media insights are retrieved and presented.

**3.2.2. File Upload by Content Contributors**

Content contributors (e.g., news agencies, social media users) upload media files through the interface. The system processes these files for transcription, keyword extraction, and sentiment analysis. The extracted metadata helps in building an index for easy retrieval.

**3.2.3. Keyword Generation and Sentiment Analysis Storage**

Media files undergo processing to extract keywords, sentiments, and tonality metrics. Extracted metadata is stored securely, and a blockchain proof is recorded for integrity verification. Keywords and sentiment insights are encrypted to prevent unauthorized access.

**3.2.3. Query Submission and Secure Retrieval**

Users search using keywords, sentiment-based filters, or tonality scores. The system matches the query with stored metadata and verifies blockchain proof before retrieval. Once validated, relevant insights and analysis are retrieved and presented securely.

**3.3. PROPOSED METHODOLOGY**

The proposed methodology for the Media Monitoring and Feedback Hub is designed to address key challenges in media content analysis, sentiment evaluation, data integrity, and secure retrieval mechanisms. The system leverages natural language processing (NLP), optical character recognition (OCR), and blockchain-based verification to ensure accurate and secure analysis of media content. It integrates automated transcription, sentiment classification, keyword extraction, and tonality assessment to provide insightful feedback. By utilizing secure cloud storage, encrypted keyword matching, and decentralized verification, this methodology ensures a scalable, privacy-preserving, and trustworthy media monitoring system.

**3.3.1. Database Design**

The database is structured to store media content, extracted metadata, and user interactions.

* Content Contributor Table

*Table 3.1: Content Contributor Table*

|  |  |  |  |
| --- | --- | --- | --- |
| **COLUMN** | **DATA TYPE** | **CONTRAINTS** | **DESCRIPTION** |
| id | INT (PK) | Unique, Not Null | Unique ID for content contributor |
| contact\_number | VARCHAR(10) | Unique, Not Null | Contributor’s contact number |
| email | VARCHAR(30) | Unique, Not Null | Contributor’s email |
| name | VARCHAR(100) | Not Null | Contributor’s name |
| password | VARCHAR(100) | Not Null | Secure login credentials |

* Media Content Table

*Table 3.2:Media Content Table*

|  |  |  |  |
| --- | --- | --- | --- |
| **COLUMN** | **DATA TYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| id | INT (PK) | Unique, Not Null | Unique ID for media file |
| file\_name | VARCHAR(255) | Unique, Not Null | Name of uploaded file |
| content | TEXT | Not Null | Transcribed text from media |
| keywords | TEXT | Not Null | Extracted keywords |
| sentiment | VARCHAR(50) | Not Null | Sentiment label (Positive, Neutral, Negative) |
| tonality | DECIMAL(5,2) | Not Null | Tonality percentage score |
| hash\_id | VARCHAR(255) | Unique, Not Null | Blockchain integrity proof |

* User Table

*Table 3.3:User Table*

|  |  |  |  |
| --- | --- | --- | --- |
| **COLUMN** | **DATA TYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| id | INT (PK) | Unique, Not Null | Unique ID for media analyst/user |
| contact\_number | VARCHAR(10) | Unique, Not Null | User’s phone number |
| email | VARCHAR(30) | Unique, Not Null | User’s email |
| name | VARCHAR(100) | Not Null | User’s name |
| password | VARCHAR(100) | Not Null | Secure login credentials |

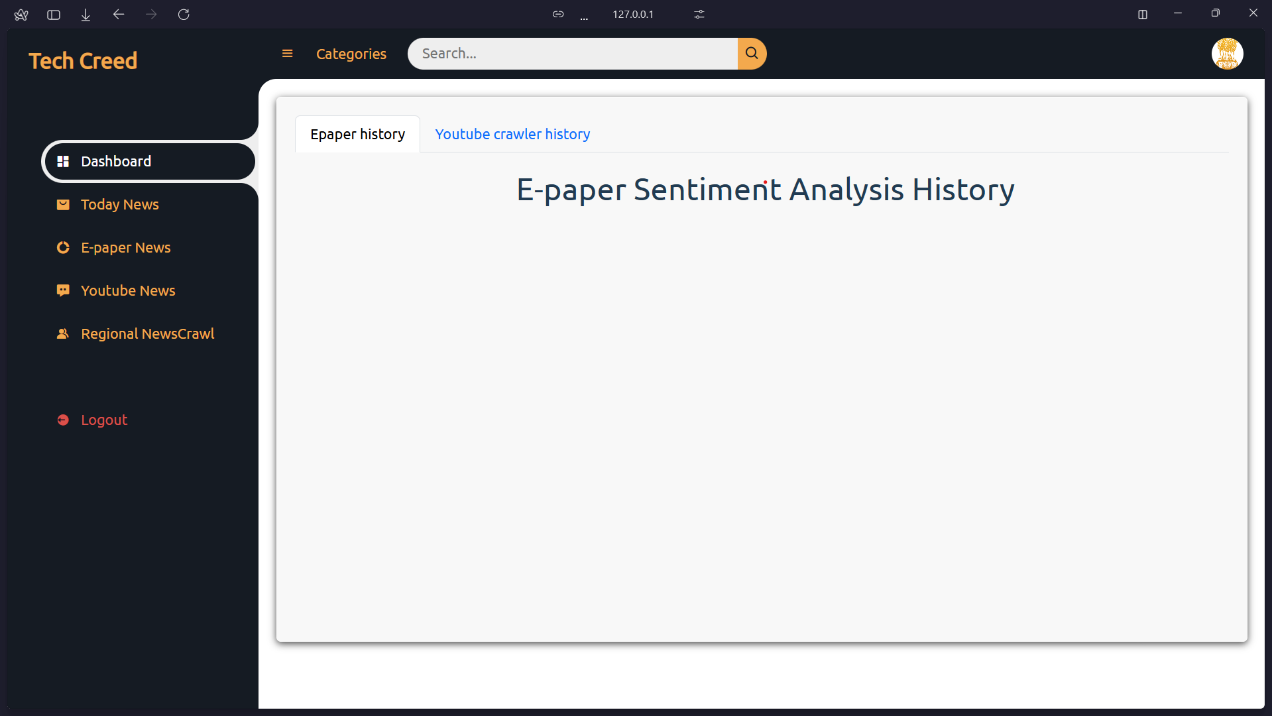
* Search and Feedback Table

*Table 3.4:Search and Feedback Table*

|  |  |  |  |
| --- | --- | --- | --- |
| **COLUMN** | **DATA TYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| id | INT (PK) | Unique, Not Null | Unique ID for query |
| user\_id | INT (FK) | Not Null | User submitting search query |
| search\_query | TEXT | Not Null | User's search query |
| sentiment\_filter | VARCHAR(50) | Not Null | Applied sentiment filter (if any) |
| feedback | TEXT | Nullable | Feedback provided by user |

**3.3.2. Input Design(UI)**

Fig 3.2 demonstrates the input design and dashboard (UI) of the Media Monitoring and Feedback Hub. The UI is designed to provide a user-friendly and interactive experience for both content contributors and analysts. The system includes media upload, real-time analysis, secure search, and feedback submission interfaces.



*Fig 3.2: Search UI*

**CHAPTER 4**

**SYSTEM IMPLEMENTATION**

**CHAPTER 4**

**SYSTEM IMPLEMENTATION**

**4.1 SYSTEM DESIGN:**

**4.1.1 SEQUENCE DIAGRAM:**

A Sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of Message Sequence diagrams are sometimes called event diagrams, event sceneries and timing diagram.

A black screen with white text

AI-generated content may be incorrect.

*Fig 4.1: Sequence diagram*

**4.1.2 ACTIVITY DIAGRAM:**

Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control.

The most important shape types:

* Rounded rectangles represent activities.
* Diamonds represent decisions.
* Bars represent the start or end of concurrent activities.
* A black circle represents the start of the workflow.
* An encircled circle represents the end of the workflow.

A diagram of a process

AI-generated content may be incorrect.

*Fig 4.2: Activity diagram*

**4.1.4. COLLABRATION DIAGRAM**

UML Collaboration Diagrams illustrate the relationship and interaction between software objects. They require use cases, system operation contracts and domain model to already exist. The collaboration diagram illustrates messages being sent between classes and objects.

**A diagram of a software process

AI-generated content may be incorrect.**

*Fig 4.4 Collabration Diagram*

**4.2. MODULE DESCRIPTIONS**

The project is divided into six modules, each constituting specific features and procedures. The modules are as follows:

**4.2.1 CHECKING NEWS WEBSITES**

This module is dedicated to analyzing information hosted on news websites. With the proliferation of online media, numerous websites publish unverified or misleading content that may influence public opinion. The module involves:

* **Data Extraction:** Gathering news articles, headlines, and metadata from multiple websites using web scraping techniques.
* **Verification Mechanisms:** Employing algorithms to cross-check the factual accuracy of news content against reliable databases and verified news sources.
* **Natural Language Processing (NLP):** Utilizing NLP techniques to identify patterns associated with fake news, such as sensational language or lack of credible citations.
* **Output:** Generating a credibility score for each article and categorizing content based on reliability. This module lays the groundwork for broader analysis and flags dubious content for further scrutiny.

**4.2.2 DIGITAL NEWSPAPERS MODULE**

The focus of this module is to assess the veracity of news circulating in digital newspapers. It involves addressing challenges such as the re-sharing of outdated or manipulated articles. Key components include:

* Historical Analysis: Comparing current articles with previous versions to identify edits or discrepancies.
* Image Forensics: Detecting manipulated images or photos accompanying news stories using advanced image analysis tools.
* Contextual Integrity: Ensuring that articles are presented in the correct context, as misrepresentation of information is a common tactic in fake news dissemination.
* Integration with Module 1: Insights generated in this module are cross-referenced with data from Module 1, ensuring a cohesive approach to identifying patterns of misinformation.

**4.2.3 YOUTUBE NEWS CHANNEL ANALYSIS**

YouTube, being a significant platform for news consumption, often becomes a breeding ground for unverified information. This module focuses on detecting fake news spread through YouTube news channels. It incorporates:

* Video Content Analysis: Processing video transcripts to identify discrepancies and ensure alignment with factual information.
* Channel Credibility: Assessing the history and reputation of channels based on their publishing history, subscriber interactions, and compliance with community guidelines.
* Viewer Influence Metrics: Evaluating metrics such as likes, comments, and shares to determine the potential reach and impact of misleading content.
* Deepfake Detection: Using AI tools to spot altered visuals or audio in videos, as these are increasingly used in the propagation of fake news.
* Real-time Monitoring: Providing a mechanism for continuous monitoring of news channels to identify emerging trends in misinformation.

**4.2.4 OCR AND TEXT ANALYSIS**

Optical Character Recognition (OCR) technology is used to convert scanned images of printed text into machine-readable text. This module processes documents, images, and PDFs to extract text, which is then analyzed using NLP for sentiment and categorization. It is particularly useful for handling printed media and documents that are not available in digital formats. The module ensures that all forms of media are included in the monitoring system, providing a comprehensive view of public sentiment. It also supports multi-language text

recognition, making it versatile for use across different regions and languages, and includes error correction features to improve the accuracy of text extraction.

**4.2.5 CENTRALIZED DASHBOARD AND NOTIFICATION**

This module integrates data from all other modules into a centralized dashboard. It provides an intuitive interface for government officials to view and analyze media coverage and sentiment trends. The dashboard includes visualizations, reports, and real-time notifications for negative news content. It allows for easy access to insights and facilitates timely decision-making. The module ensures that all relevant information is available in one place, enhancing transparency and responsiveness. It also supports customizable views and filters, enabling users to tailor the dashboard to their specific needs and focus on the most relevant data for their roles.

**CHAPTER 5**

**RESULT & DISCUSSION**

**CHAPTER 5**

**RESULTS & DISCUSSION**

**5.1. TESTING**

Testing is a crucial phase in the development of the "Media Monitoring and Feedback System for the Government of India." It ensures that the system functions correctly, securely, and meets the specified requirements. Below are detailed descriptions of various testing types:

**Unit Testing**

Unit testing involves testing individual components or modules of the system in isolation to ensure they function as expected. Each unit, such as the web scraping module or sentiment analysis algorithm, is tested for correctness by verifying its output against known inputs. This helps identify and fix bugs early in the development process, ensuring that each part of the system works correctly before integration. Unit tests are typically automated, allowing for frequent and consistent testing throughout the development cycle. This type of testing also helps developers understand the functionality of each module and make necessary adjustments to improve performance and reliability.

**Integration Testing**

Integration testing focuses on verifying the interactions between different modules of the system. After unit testing, modules are combined and tested as a group to ensure they work together seamlessly. This type of testing helps identify issues related to data flow, interface mismatches, and communication between modules, ensuring that the integrated system functions as intended. Integration tests often involve scenarios that mimic real-world use cases, providing a comprehensive assessment of how the system operates in practice. This testing phase is crucial for ensuring that the system's components integrate smoothly and that data is accurately transferred between modules.

**Security Testing**

Security testing aims to identify vulnerabilities and ensure that the system is protected against threats. This includes testing for common security issues such as SQL injection, cross-site scripting (XSS), and unauthorized access. Security testing ensures that sensitive data is protected, user authentication and authorization are correctly implemented, and the system complies with security standards and regulations. Penetration testing, vulnerability scanning, and security audits are common methods used in this phase. Security testing also involves assessing the system's ability to handle potential attacks and ensuring that robust security measures are in place to prevent data breaches and unauthorized access.

**System Testing**

System testing involves testing the complete and integrated system to verify that it meets the specified requirements. This type of testing evaluates the system's overall functionality, performance, and reliability under various conditions. System testing ensures that all components work together as a whole and that the system performs well in real-world scenarios. It includes testing the system's response to different inputs, its ability to handle high loads, and its stability over extended periods. System testing also involves checking the system's compatibility with various hardware and software environments, ensuring that it operates correctly across different platforms.

**Validation Testing**

Validation testing ensures that the system meets the needs and expectations of the end-users. It involves verifying that the system's functionality aligns with the requirements and use cases defined during the planning phase. Validation testing helps confirm that the system delivers the intended value and solves the problems it was designed to address. This phase includes user feedback sessions, where actual users interact with the system and provide input on its usability and effectiveness. Validation testing also involves comparing the system's performance against predefined criteria to ensure that it meets all necessary standards and requirements.

**User Acceptance Testing**

User Acceptance Testing (UAT) is the final phase of testing, where actual users test the system in a real-world environment. UAT ensures that the system is user-friendly, meets user requirements, and is ready for deployment. Feedback from users during this phase is crucial for identifying any last-minute issues and making necessary adjustments before the system goes live. UAT involves testing the system's functionality, performance, and usability from the perspective of the end-users. This phase also includes training sessions to ensure that users are comfortable with the system and can effectively utilize its features. Successful completion of UAT signifies that the system is ready for full-scale implementation.

**CHAPTER 6**

**CONCLUSION**

**CHAPTER 6**

**CONCLUSION**

The proliferation of social media platforms has revolutionized the way information is shared and consumed, but it has also led to the widespread dissemination of fake news. This phenomenon has severe implications for societal trust, political stability, and the credibility of media platforms. As the volume and speed of information sharing grow, traditional manual fact-checking methods have proven insufficient, creating a critical demand for automated and scalable detection techniques. These automated methods not only enhance the accuracy and efficiency of fake news detection but also help curb its far-reaching influence across diverse online communities.

Among the various techniques utilized, Sentiment Analysis (SA) has emerged as a powerful approach for identifying fake news. Sentiment Analysis leverages advanced natural language processing (NLP) and machine learning algorithms to evaluate the emotional tone of content. Fake news often aims to manipulate readers by evoking strong emotions such as anger, fear, or excitement, making sentiment-based insights particularly valuable in distinguishing misleading information from credible sources. Additionally, SA's applicability extends beyond articles to include related user comments, discussions, and social interactions, thereby providing a holistic perspective on the dissemination and impact of fake news.

**6.1 FUTURE ENHANCEMENT**

The proposed system can be significantly enhanced to further optimize its performance, scalability, and security. Several key areas for future enhancement include:

**Advanced Predictive Analysis**:

**Overview**: Use machine learning models to forecast future trends in media coverage and public sentiment.

**Implementation**: Analyze historical data to identify patterns and predict future events. This involves training models on past data and continuously updating them with new information.

**Benefits**: Enables proactive decision-making by anticipating public reactions. Government officials can prepare for potential issues and respond more effectively.

**Challenges**: Requires significant historical data and computational resources. Ensuring the accuracy of predictions and avoiding biases in the models are critical challenges.

**Multilingual Support Extension**:

**Overview**: Enhance the system to process more regional languages and dialects.

**Implementation**: Integrate additional language processing models and translation tools. This involves training NLP models on diverse datasets to improve their accuracy.

**Benefits**: Provides a more inclusive view of public sentiment across different regions. Ensures comprehensive coverage and analysis of media content from all parts of the country.

**Challenges**: Developing accurate models for less common languages due to limited data. Ensuring the quality and consistency of translations is also crucial.

**Integration with Social Media Platforms**:

**Overview**: Include data from emerging and niche social media platforms.

**Implementation**: Develop connectors and APIs to gather data from these platforms. Integrate this data into the existing analysis framework. **Benefits**: Provides a comprehensive view of public opinion and engagement. Helps understand the sentiments of different demographic groups and communities.

**Challenges**: Maintaining connectors and ensuring data accuracy and relevance. Developing and maintaining these integrations can be resource-intensive.

**APPENDICES**

**APPENDICES**

**A.1 SDG GOALS**

**1) SDG 17: Technology**

Sustainable Development Goal 17 aims to strengthen the means of implementation and revitalize the global partnership for sustainable development. It focuses on bringing together governments, private sectors, civil society, and global institutions to collaborate and support all 17 SDGs by enhancing financial, technological, and capacity-building support - especially in developing countries.

**Impact:**

* Global Cooperation: Encourages strong international partnerships to address global challenges like poverty, climate change, and inequality.
* Resource Mobilization: Improves access to financial resources and investments in underdeveloped regions.
* Technology Sharing: Promotes the transfer of technology and innovation across countries for inclusive growth.
* Data and Monitoring: Strengthens systems to track SDG progress effectively through quality data and transparent reporting.

**A.2 SOURCE CODE**

from flask import Flask, request, render\_template, jsonify, redirect, flash, session, url\_for

import pytesseract

import fitz # PyMuPDF

import cv2

import numpy as np

from PIL import Image

import concurrent.futures

import os

import spacy

import nltk

from nltk.sentiment.vader import SentimentIntensityAnalyzer

from werkzeug.utils import secure\_filename

from youtube\_transcript\_api import YouTubeTranscriptApi

from pytube import YouTube

import os

import requests

from flask import Flask, request, render\_template

from pytube import YouTube

import moviepy.editor as mp

import assemblyai as aai

import nltk

from nltk.sentiment.vader import SentimentIntensityAnalyzer

import yt\_dlp

from twilio.rest import Client

import easygui

from flask\_pymongo import PyMongo

from bson.objectid import ObjectId

import bcrypt

from werkzeug.security import generate\_password\_hash, check\_password\_hash

from moviepy.editor import AudioFileClip

ASSEMBLYAI\_API\_KEY = "80095427ee394316aea00ead3b58c51c"

aai.settings.api\_key = ASSEMBLYAI\_API\_KEY

app = Flask(\_\_name\_\_, static\_url\_path='/static')

app.secret\_key = '1234567890qwertyuiopasdfghjklzxcvbnm'

users = {}

sentiment\_history = [] # Initialize sentiment analysis history

# Download NLTK resources (if not already downloaded)

nltk.download('vader\_lexicon')

# Initialize the NLTK Sentiment Intensity Analyzer

sia = SentimentIntensityAnalyzer()

# Create a directory for uploaded files

upload\_dir = 'uploads'

os.makedirs(upload\_dir, exist\_ok=True)

# MongoDB Configuration

app.config["MONGO\_URI"] = "mongodb://localhost:27017/mydatabase"

mongo = PyMongo(app)

# Collections

users\_collection = mongo.db.users

sentiment\_collection = mongo.db.sentiments

def download\_and\_convert\_to\_mp3(youtube\_url):

try:

print(f"Downloading audio from: {youtube\_url}")

# Define output filename

mp4\_file\_path = "temp\_audio.mp4"

mp3\_file\_path = "temp\_audio.mp3"

# yt-dlp options to download audio only

ydl\_opts = {

'format': 'bestaudio/best',

'ffmpeg\_location': 'C:/ffmpeg/bin', # Change this to your actual path

'outtmpl': mp4\_file\_path,

'postprocessors': [{

'key': 'FFmpegExtractAudio',

'preferredcodec': 'mp3',

'preferredquality': '192',

}],

}

# Download audio using yt-dlp

with yt\_dlp.YoutubeDL(ydl\_opts) as ydl:

ydl.download([youtube\_url])

# Convert to MP3 (if yt-dlp didn't do it already)

if not os.path.exists(mp3\_file\_path):

audio\_clip = AudioFileClip(mp4\_file\_path)

audio\_clip.write\_audiofile(mp3\_file\_path)

audio\_clip.close()

os.remove(mp4\_file\_path) # Cleanup MP4 file

print(f"MP3 file created at: {mp3\_file\_path}")

return mp3\_file\_path

except Exception as e:

print(f"Error downloading YouTube audio: {e}")

return None

def p\_perform(text):

sentiment\_scores = sia.polarity\_scores(text)

sentiment = "Neutral"

if sentiment\_scores["compound"] > 0.05:

sentiment = "Positive"

elif sentiment\_scores["compound"] < -0.05:

sentiment = "Negative"

return sentiment, sentiment\_scores

def calculate\_tonality\_percentages(sentiment\_scores):

total = sum(sentiment\_scores.values())

positive\_percentage = (sentiment\_scores["pos"] / total) \* 100

neutral\_percentage = (sentiment\_scores["neu"] / total) \* 100

negative\_percentage = (sentiment\_scores["neg"] / total) \* 100

return positive\_percentage, neutral\_percentage, negative\_percentage

# Function to extract text from a page

def extract\_text\_from\_page(page):

try:

# Export the page as an image

pix = page.get\_pixmap()

img = Image.frombytes("RGB", [pix.width, pix.height], pix.samples)

# Convert PIL image to OpenCV format (BGR)

img\_cv2 = np.array(img)

img\_cv2 = cv2.cvtColor(img\_cv2, cv2.COLOR\_RGB2BGR)

# Perform OCR on the image

text = pytesseract.image\_to\_string(img\_cv2)

return text

except Exception as e:

return f"An error occurred: {e}"

# Function to extract text from a scanned PDF

def extract\_text\_from\_scanned\_pdf(pdf\_path):

text\_data = ''

try:

pdf\_document = fitz.open(pdf\_path)

pages = [pdf\_document.load\_page(page\_number) for page\_number in range(pdf\_document.page\_count)]

# Process pages concurrently

with concurrent.futures.ThreadPoolExecutor() as executor:

results = list(executor.map(extract\_text\_from\_page, pages))

# Join and clean up the extracted text

text\_data = ' '.join(results)

text\_data = text\_data.replace('\n', ' ').replace('\r', '') # Remove line breaks and carriage returns

text\_data = ' '.join(text\_data.split()) # Remove extra whitespace

except Exception as e:

print(f"An error occurred: {e}")

finally:

pdf\_document.close()

return text\_data

# Function to extract government-related news articles

def extract\_news\_related\_to\_government(text):

nlp = spacy.load("en\_core\_web\_sm")

doc = nlp(text)

government\_keywords = ["government", "politics", "public policy"]

government\_articles = []

current\_article = ""

for sentence in doc.sents:

if any(keyword in sentence.text.lower() for keyword in government\_keywords):

current\_article = sentence.text

elif current\_article:

current\_article += "\n" + sentence.text

if current\_article and (sentence.text.endswith(".") or sentence.text.endswith("?")):

government\_articles.append(current\_article)

current\_article = ""

return government\_articles

# Function to perform sentiment analysis on government-related news articles

def perform\_sentiment\_analysis(pdf\_name, government\_articles):

positive\_count = 0

negative\_count = 0

neutral\_count = 0

negative\_sentences = [] # Add this list to store negative sentences

for article in government\_articles:

sentiment\_scores = sia.polarity\_scores(article)

compound\_score = sentiment\_scores['compound']

if compound\_score >= 0.05:

positive\_count += 1

elif compound\_score <= -0.05:

negative\_count += 1

# Store the negative sentence

negative\_sentences.append(article)

else:

neutral\_count += 1

total\_articles = len(government\_articles)

positive\_percentage = (positive\_count / total\_articles) \* 100

negative\_percentage = (negative\_count / total\_articles) \* 100

neutral\_percentage = (neutral\_count / total\_articles) \* 100

formatted\_positive\_percentage = "{:.2f}".format(positive\_percentage)

formatted\_negative\_percentage = "{:.2f}".format(negative\_percentage)

formatted\_neutral\_percentage = "{:.2f}".format(neutral\_percentage)

sentiment\_result = {

"pdf\_name": pdf\_name,

"positive": formatted\_positive\_percentage,

"negative": formatted\_negative\_percentage,

"neutral": formatted\_neutral\_percentage,

"negative\_sentences": negative\_sentences # Include negative sentences in the result

}

sentiment\_history.append(sentiment\_result)

return sentiment\_result

# Routes for rendering templates

@app.route('/')

def home():

return render\_template('main.html')

@app.route('/dashboard')

def dashboard():

utube\_sentiment\_history = []

return render\_template('dashboardContent.html', sentiment\_history=sentiment\_history, utube\_sentiment\_history=utube\_sentiment\_history)

@app.route('/signin', methods=['GET', 'POST'])

def signin():

if request.method == 'POST':

username = request.form.get('username') # Get username

password = request.form.get('password') # Get password

# Check if user exists

user = users\_collection.find\_one({"username": username})

if user and check\_password\_hash(user["password"], password):

session["username"] = username # Store username in session

return redirect(url\_for('dashboard'))

else:

return "Invalid credentials."

return render\_template('signin.html')

@app.route('/signup', methods=['GET', 'POST'])

def signup():

if request.method == 'POST':

username = request.form['username']

password = request.form['password']

confirm\_password = request.form['confirm\_password']

# Check if user exists

existing\_user = users\_collection.find\_one({"username": username})

if existing\_user:

return "Username already exists. Choose a different one."

# Password confirmation

if password != confirm\_password:

return "Passwords do not match."

# Hash the password before storing

hashed\_password = generate\_password\_hash(password)

# Insert new user into MongoDB

users\_collection.insert\_one({

"username": username,

"password": hashed\_password

})

return redirect(url\_for('signin')) # Redirect to login page

return render\_template('signup.html')

@app.route('/today')

def today\_news():

return render\_template('today.html')

@app.route('/epaper')

def epaper\_news():

return render\_template('epaper.html')

# @app.route('/Utube')

# def youtube\_news():

# return render\_template('Utube.html')

# Route to display the Utube.html page

@app.route('/Utube', methods=['GET'])

def display\_utube():

return render\_template('Utube.html')

@app.route("/analyze\_youtube", methods=["GET", "POST"])

def index():

transcript = None

sentiment = None

sentiment\_scores = None

positive\_percentage = None

neutral\_percentage = None

negative\_percentage = None

negative\_timestamps\_minutes = [] # Initialize list for negative timestamps

negative\_sentences = []

video\_title = None

utube\_sentiment\_history = [] # Initialize list for negative sentences

if request.method == "POST":

youtube\_url = request.form["youtube\_url"]

if youtube\_url:

mp3\_file\_path = download\_and\_convert\_to\_mp3(youtube\_url)

yt = YouTube(youtube\_url)

video\_title = yt.title

# Send the MP3 to AssemblyAI for transcription

transcriber = aai.Transcriber()

transcript = transcriber.transcribe(mp3\_file\_path)

os.remove(mp3\_file\_path) # Clean up - delete the temporary MP3 file

# Perform sentiment analysis on the transcript

sentiment, sentiment\_scores = p\_perform(transcript.text)

# Calculate tonality percentages

positive\_percentage, neutral\_percentage, negative\_percentage = calculate\_tonality\_percentages(sentiment\_scores)

# Split the transcript into sentences for analysis

sentences = transcript.text.split('. ')

# Iterate through sentences and check for negative sentiment

for i, sentence in enumerate(sentences):

# Perform sentiment analysis on each sentence

sentence\_sentiment, \_ = p\_perform(sentence)

if sentence\_sentiment == "Negative":

seconds = i \* 4 # Assuming each sentence takes 4 seconds

negative\_timestamps\_minutes.append(seconds)

negative\_sentences.append(sentence) # Add negative sentence to the list

formatted\_positive\_percentage = "{:.2f}".format(positive\_percentage)

formatted\_negative\_percentage = "{:.2f}".format(negative\_percentage)

formatted\_neutral\_percentage = "{:.2f}".format(neutral\_percentage)

utube\_sentiment\_result = {

"video\_title": video\_title,

"positive\_percentage": formatted\_positive\_percentage,

"negative\_percentage": formatted\_negative\_percentage,

"neutral\_percentage": formatted\_neutral\_percentage,

"negative\_timestamps\_minutes": negative\_timestamps\_minutes,

"negative\_sentences": negative\_sentences

}

utube\_sentiment\_history.append(utube\_sentiment\_result)

# Convert formatted\_negative\_percentage to a float for comparison

formatted\_negative\_percentage\_float = float(formatted\_negative\_percentage)

if formatted\_negative\_percentage\_float > 1.0:

account\_sid = 'ACdefa7df4e5786818bb3a419b514cc9d8'

auth\_token = '4150b604c118d5c19ffb0ecf77886c78'

client = Client(account\_sid, auth\_token)

message\_body = f'\n Title: {video\_title}, \n Negative Percentage: {formatted\_negative\_percentage}%, \n URL: {youtube\_url}'

message = client.messages.create(

from\_='+12622879688',

body=message\_body,

to='+919360364123'

)

print(message.sid)

easygui.msgbox("Notification has been sent succesfully")

return render\_template(

'Utube.html',

video\_title=video\_title ,

positive\_percentage=formatted\_positive\_percentage,

negative\_percentage=formatted\_negative\_percentage,

neutral\_percentage=formatted\_neutral\_percentage,

negative\_timestamps\_minutes=negative\_timestamps\_minutes,

negative\_sentences=negative\_sentences,

utube\_sentiment\_history=utube\_sentiment\_history

)

@app.route('/upload\_and\_analyze', methods=['POST'])

def upload\_and\_analyze():

if 'file' not in request.files:

return jsonify({"error": "No file uploaded"})

file = request.files['file']

if file.filename.endswith('.pdf'):

pdf\_filename = os.path.join(upload\_dir, secure\_filename(file.filename))

file.save(pdf\_filename)

text = extract\_text\_from\_scanned\_pdf(pdf\_filename)

if text:

government\_articles = extract\_news\_related\_to\_government(text)

if government\_articles:

pdf\_name = secure\_filename(file.filename)

sentiment\_result = perform\_sentiment\_analysis(pdf\_name, government\_articles)

return render\_template("epaper.html", \*\*sentiment\_result) # Render the result template with sentiment analysis results

return jsonify({"error": "No text or articles found"})

# Route to view sentiment analysis results for a specific PDF

@app.route('/dashboard')

def view\_sentiment():

pdf\_name = request.args.get('pdf\_name')

youtube\_video\_title = request.args.get('youtube\_video\_title')

sentiment\_result = None

utube\_sentiment\_result = None

# Search for the PDF sentiment result

for result in sentiment\_history:

if result["pdf\_name"] == pdf\_name:

sentiment\_result = result

break

# Search for the YouTube sentiment result

for result in utube\_sentiment\_result:

if result["video\_title"] == youtube\_video\_title:

utube\_sentiment\_result = result

break

return render\_template(

'dashboardContent.html',

sentiment\_result=sentiment\_result,

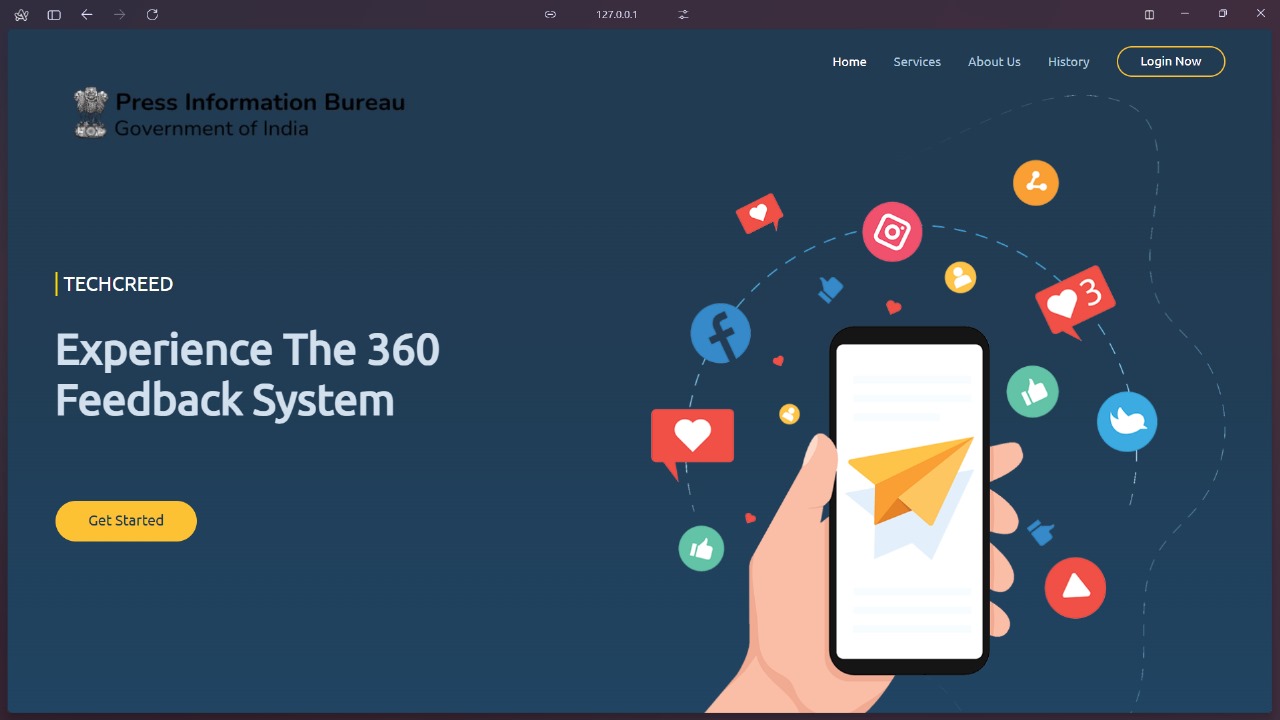
utube\_sentiment\_result=utube\_sentiment\_result

)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**A.3 SCREEN SHOTS**

****

***Fig A.3.1 Landing Page***

**A person sitting on a couch with a computer

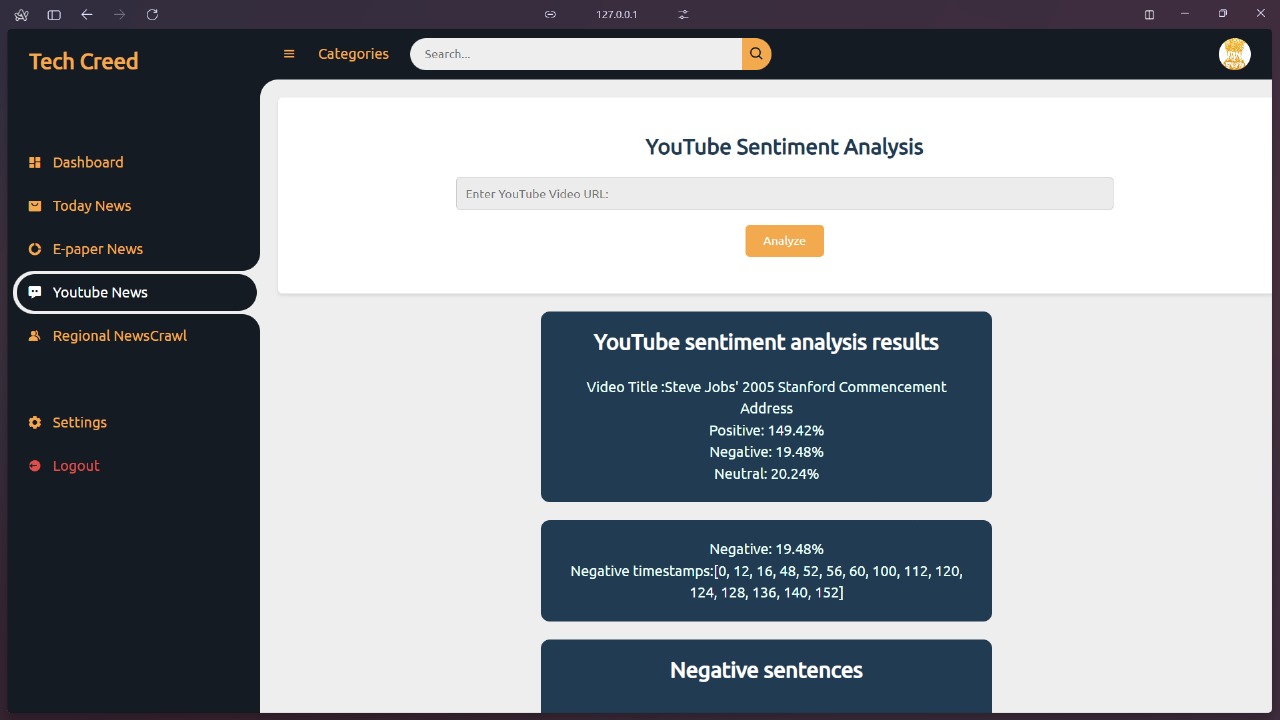
AI-generated content may be incorrect.**

***Fig A.3.2 Login Page***

**A screenshot of a computer

AI-generated content may be incorrect.**

***Fig A.3.3 Dashboard UI***

****

***Fig A.3.4 Youtube Sentiment Analysis UI***

**A screenshot of a computer

AI-generated content may be incorrect.**

***Fig A.3.***

***5 Youtube sentiment analysis result UI***

**A screenshot of a computer

AI-generated content may be incorrect.**

***Fig A.3.6 PDF Sentiment analysis UI***

**A screenshot of a computer

AI-generated content may be incorrect.**

***Fig A.3.7 PDF Sentiment analysis result UI***

**A.4 PLAGIARISM REPORT**

**A screenshot of a computer

AI-generated content may be incorrect.**

**FIG A.4.1 Plagiarism Report**

**A.5 PAPER PUBLICATION**

**A screenshot of a computer

AI-generated content may be incorrect.**

**FIG A.5.1 Paper Publication**

**REFERENCES**

**REFERENCES**

1. Shearer, E.; Mitchell, A. News Use Across Social Media Platforms in 2020. 2021. Available online: [https://www.journalism.org/2](https://www.journalism.org/2021/01/12/news-use-across-social-media-platforms-in-2020/) [021/01/12/news-use-across-social-](https://www.journalism.org/2021/01/12/news-use-across-social-media-platforms-in-2020/) [media-platforms-in-2020/](https://www.journalism.org/2021/01/12/news-use-across-social-media-platforms-in-2020/) (accessed on 3 June 2021).
2. Hernon, P. Disinformation and misinformation through the internet: Findings of an exploratory study. *Gov. Inf. Q.* **1995**, *12*, 133–139. [[CrossRef](http://doi.org/10.1016/0740-624X(95)90052-7)]
3. Zubiaga, A.; Aker, A.; Bontcheva, K.; Liakata, M.; Procter, R. Detection and Resolution of Rumours in Social Media: A Survey. *ACM Comput. Surv.* **2018**, *51*, 32:1–32:36. [[CrossRef](http://dx.doi.org/10.1145/3161603)]
4. Meel, P.; Vishwakarma, D.K. Fake news, rumor, information pollution in social media and web: A contemporary survey of state-of-the-arts, challenges and opportunities. *Expert Syst. Appl.* **2020**, *153*, 112986. [[CrossRef](http://dx.doi.org/10.1016/j.eswa.2019.112986)]
5. Zannettou, S.; Sirivianos, M.; Blackburn, J.; Kourtellis, N. The Web of False Information: Rumors, Fake News, Hoaxes, Clickbait, and Various Other Shenanigans. *ACM J. Data Inf.*

*Qual.* **2019**, *11*, 10:1–10:37. [[CrossRef]](http://dx.doi.org/10.1145/3309699)

1. Sharma, K.; Qian, F.; Jiang, H.; Ruchansky, N.; Zhang, M.; Liu, Y. Combating Fake News: A Survey on Identification and Mitigation Techniques. *ACM Trans. Intell. Syst. Technol.* **2019**, *10*, 21:1–21:42. [[CrossRef](http://dx.doi.org/10.1145/3305260)]
2. Guerini, M.; Staiano, J. Deep Feelings: A Massive Cross- Lingual Study on the Relation between Emotions and Virality. In Proceedings of the 24th International Conference on World Wide Web Companion, WWW 2015, Florence, Italy, 18–22 May 2015; Companion Volume; Gangemi, A., Leonardi, S., Panconesi, A., Eds.; ACM: New York, NY, USA, 2015; pp. 299–305. [[CrossRef](http://dx.doi.org/10.1145/2740908.2743058)]
3. Dickerson, J.P.; Kagan, V.; Subrahmanian, V.S. Using sentiment to detect bots on Twitter: Are humans more opinionated than bots? In Proceedings of the 2014

IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2014, Beijing, China, 17–20 August 2014; Wu, X., Ester, M., Xu, G., Eds.; IEEE Computer Society: Washington, DC, USA, 2014; pp. 620–627. [[CrossRef](http://dx.doi.org/10.1109/ASONAM.2014.6921650)]

1. Chen, Y.; Conroy, N.J.; Rubin, V.L. Misleading Online Content: Recognizing Clickbait as “False News”. In Proceedings of the 2015 ACM Workshop on Multimodal Deception Detection, WMDD@ICMI 2015, Seattle, WA, USA, 13 November 2015; Abouelenien, M., Burzo, M., Mihalcea, R., Pérez-Rosas, V., Eds.; ACM: New York, NY, USA, 2015; pp. 15–19. [[CrossRef](http://dx.doi.org/10.1145/2823465.2823467)]
2. Conroy, N.J.; Rubin, V.L.; Chen, Y. Automatic deception detection: Methods for finding fake news. In *Information Science with Impact: Research in and for the*

*Community— Proceedings of the 78th ASIS&T Annual Meeting, ASIST 2015, St. Louis, MO, USA, 6–10 October 2015*; Wiley: Hoboken, NJ, USA, 2015; Volume 52, pp. 1–4. [[CrossRef](http://dx.doi.org/10.1002/pra2.2015.145052010082)]

1. Shu, K.; Sliva, A.; Wang, S.; Tang, J.; Liu, H. Fake News Detection on Social Media: A Data Mining Perspective. *SIGKDD Explor.*

**2017**, *19*, 22–36. [[CrossRef]](http://dx.doi.org/10.1145/3137597.3137600)

1. Shu, K.; Wang, S.; Lee, D.; Liu, H. Mining Disinformation and Fake News: Concepts, Methods, and Recent Advancements. In *Disinformation, Misinformation, and Fake News in Social Media: Emerging Research Challenges and Opportunities*; Shu, K., Wang, S., Lee, D., Liu, H., Eds.; Springer International Publishing: Cham, Switzerland, 2020;

pp. 1–19. [[CrossRef]](http://dx.doi.org/10.1007/978-3-030-42699-6_1)

1. Shu, K.; Liu, H. Detecting Fake News on Social Media. In *Synthesis Lectures on Data Mining and Knowledge Discovery*; Morgan & Claypool Publishers: San Rafael, CA, USA, 2019; Volume 18.
2. Hussein, D.M.E.D.M. A survey on sentiment analysis challenges. *J. King Saud Univ. Eng. Sci.* **2018**, *30*, 330–338. [[CrossRef](http://dx.doi.org/10.1016/j.jksues.2016.04.002)]
3. Thorne, J.; Vlachos, A. Automated Fact Checking: Task Elhadad, M.K.; Li, K.F.; Gebali, F. Fake News Detection on Social Media: A Systematic Survey. In Proceedings of the IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, ACRIM 2019, Victoria, BC, Canada, 21–23 August 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–8. [[CrossRef](http://dx.doi.org/10.1109/PACRIM47961.2019.8985062)]
4. Bondielli, A.; Marcelloni, F. A survey on fake news and rumour detection techniques. *Inf. Sci.* **2019**, *497*, 38–55. [[CrossRef](http://dx.doi.org/10.1016/j.ins.2019.05.035)]
5. Zhou, X.; Zafarani, R. A Survey of Fake News: Fundamental

Theories, Detection Methods, and Opportunities. *ACM Comput. Surv.* **2020**, *53*, 109:1–109:40. [[CrossRef]](http://dx.doi.org/10.1145/3395046)

1. Zhang, X.; Ghorbani, A.A. An overview of online fake news: Characterization, detection, and discussion. *Inf. Process. Manag.* **2020**, *57*, 102025. [[CrossRef](http://dx.doi.org/10.1016/j.ipm.2019.03.004)]
2. Antonakaki, D.; Fragopoulou, P.; Ioannidis, S. A survey of Twitter research: Data model, graph structure, sentiment analysis and attacks. *Expert Syst. Appl.* **2021**, *164*, 114006. [[CrossRef](http://dx.doi.org/10.1016/j.eswa.2020.114006)]
3. Allcott, H.; Gentzkow, M. Social Media and Fake News in the 2016 Election. *J. Econ. Perspect.* **2017**, *31*, 211–36. [[CrossRef](http://dx.doi.org/10.1257/jep.31.2.211)]
4. Wardle, C. Fake News. It’s Complicated. 2017. Available online: <https://firstdraftnews.org/articles/fake-news-complicated/> (accessed on 3 June 2021).
5. Tandoc, E.C., Jr.; Lim, Z.W.; Ling, R. Defining “Fake News”.

*Digit. J.* **2018**, *6*, 137–153. [[CrossRef]](http://dx.doi.org/10.1080/21670811.2017.1360143)

1. 8. J. Seong, W. Lee, and S. Lee, "Multilingual Speech Synthesis for Voice Cloning," *2021 IEEE International Conference on Big Data and Smart Computing (BigComp)*, IEEE, 2021. (Cited by 7 papers)
2. M. Siek and E. S. Setiadi, "Analysis of News Sentiment and Stock Price Using Web Scraping and Vader Sentiment Analysis," *2024 International Conference on Information Management and Technology (ICIMTech)*, IEEE, 2024.
3. S. Kaliappan, L. Natrayan, and A. Rajput, "Sentiment Analysis of News Headlines Based on Sentiment Lexicon and Deep Learning," *2023 4th International Conference on Smart Electronics and Communication (ICOSEC)*, IEEE, 2023. (Cited by 43 papers)
4. N. M. K. Varma, S. H. Mattaparty, S. Ismail, J. Thaduri, G. D. Arora, and A. B. AnandKumar, "Sentiment Analysis: A Machine Learning Perspective," *2024 First International Conference on Electronics, Communication and Signal Processing (ICECSP)*, IEEE, 2024.
5. S. S. Korti and S. G. Kanakaraddi, "Depression detection from Twitter posts using NLP and Machine learning techniques," *2022 Fourth International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT)*, IEEE, 2022.
6. A. Dixit, S. Sharma, P. D. Rao, V. Reddy, M. Janaki, R. Thirumalaivasan, and M. M. Subashini, "Audio to Indian and American Sign Language Converter using Machine Translation and NLP Technique," *2022 Third International Conference on Intelligent Computing Instrumentation and Control Technologies (ICICICT)*, IEEE, 2022.
7. K. Kavita and H. Singh, "Utilizing Mixture Methods for Classifier in NLP: An Essential Consideration," *2023 International Conference on Artificial Intelligence and Smart Communication (AISC)*, IEEE, 2023.
8. V. M. Reddy, T. Vaishnavi, and K. P. Kumar, "Speech-to-Text and Text-to-Speech Recognition Using Deep Learning," *2023 2nd International Conference on Edge Computing and Applications (ICECAA)*, IEEE, 2023. (Cited by 11 papers)