IJARSCT

ISSN (Online) 2581-9429

International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)

International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal

Impact Factor: 7.53

Volume 4, Issue 4, November 2024

Online Abusive Attack Prevention

Prof. C. S. Jaybhaye, Manish More ,Shreyash Mandalik ,Yogita Tarade, Prachi Thakare
Smt. Kashibai Navale College of Engineering, Pune, Maharashtra, India

Abstract: *Our website, "Online Abusive Attack Prevention," is designed to create a safer and more respectful online environment. By leveraging advanced word processing, tokenization techniques, and real-time data analysis, our platform effectively detects and prevents abusive language in online interactions. Built using the MERN stack for the frontend and Python for modular design, the system is both robust and scalable.*

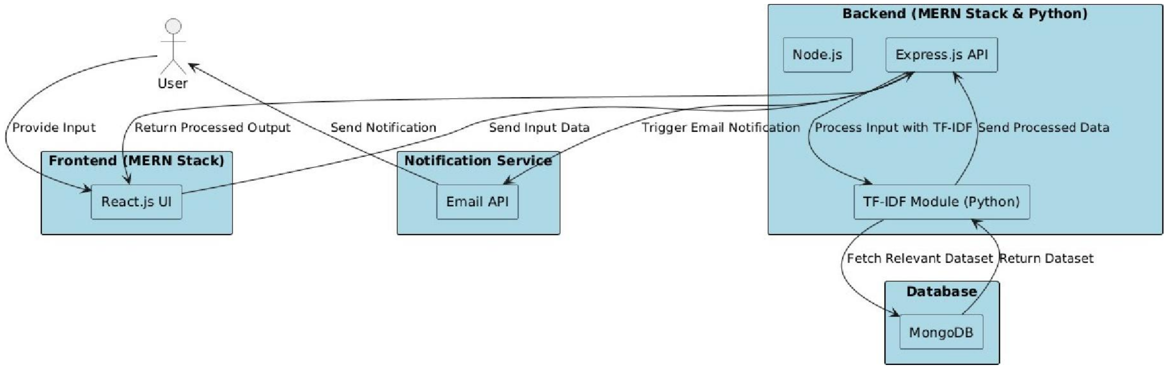
Keywords: Attack Prevention

I. INTRODUCTION

Problem Statement

- Designing an effective and efficient system to detect and prevent online abusive attacks, with a focus on accuracy, real-time processing, and user notifications.
- Our website, "Online Abusive Attack Prevention," is designed to create a safer and more respectful online environment. By leveraging advanced word processing, tokenization techniques, and real-time data analysis, our platform effectively detects and prevents abusive language in online interactions. Built using the MERN stack for the frontend and Python for modular design, the system is both robust and scalable. With a target accuracy of over 90%, our solution not only identifies harmful content but also integrates Email API notifications to alert users and administrators, ensuring proactive management of online safety.

System Architecture



II. LITERATURE SURVEY

Sr.	Title	Author(s)	Journal & Published Year	Algorithm Used	Summary
1	Emotionally Informed Hate Speech Detection: A Multi-target Perspective	Patricia Chiril, Endang Wahyu Pamungkas, Farah Benamara, Véronique Moriceau, Viviana Patti	2021	Neural Models, Multi- task Learning	This study tackles hate speech detection from a multi-target perspective, exploring models that detect both topics and hate speech targets. It emphasizes the importance of affective knowledge in finer-grained detection.

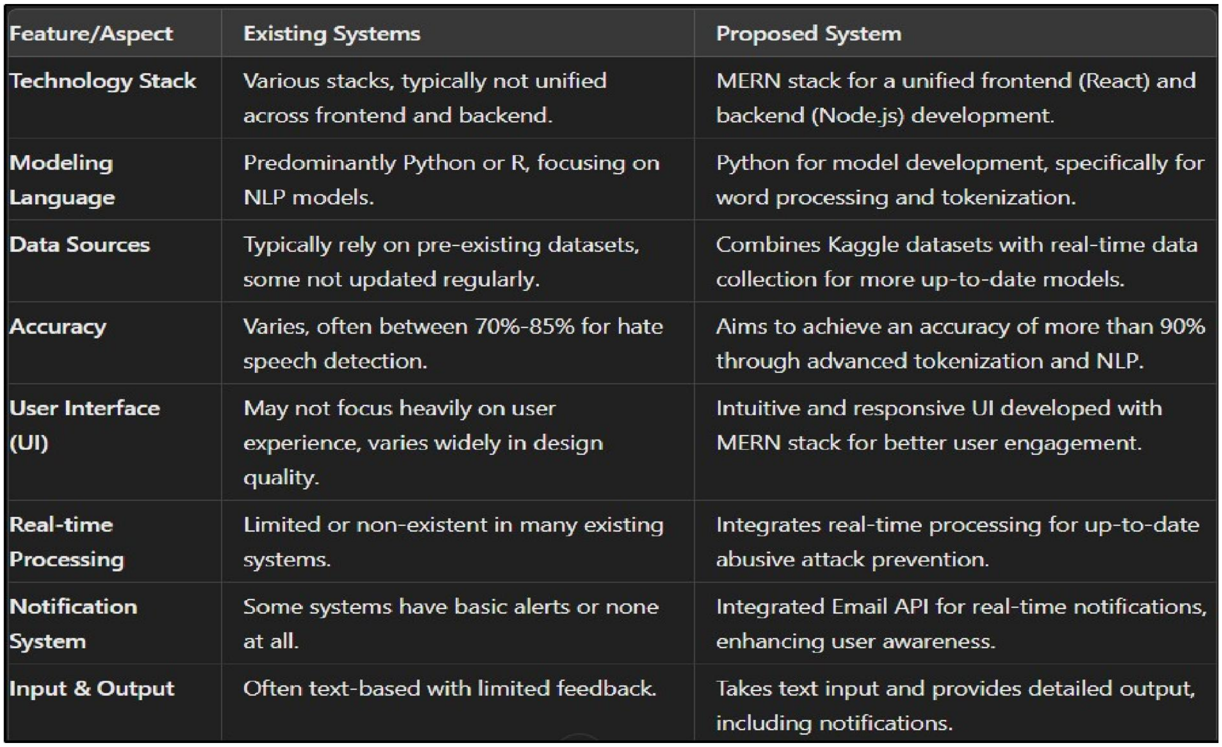
DOI: 10.48175/568

2	Hate Speech Classification Using SVM and Naive Bayes		IOSR Journal of Mobile Computing & Application, 2022	SVM, Naive Bayes	The paper proposes SVM and Naive Bayes for automatic hate speech detection, achieving high accuracy for SVM (99%) and moderate accuracy for Naive Bayes (50%).
3	A survey on hate speech detection and sentiment analysis using machine learning and deep learning models	Malliga Subramanian, Veerappampalayam Easwaramoorthy Sathiskumar, G. Deepalakshmi, Jaehyuk Cho, G. Manikandan	Alexandria Engineering Journal, 2023	Machine learning, Deep learning models	The article offers a brief overview of recent advancements in hate speech detection and sentiment analysis, focusing on methodologies, challenges, and future research opportunities in machine learning and deep learning.
4	Explainable Hate Speech Detection with Step-by-Step Reasoning	Yongjin Yang, Joonkee Kim, Yujin Kim, Namgyu Ho, James Thorne, Se-young Yun	KAIST, 2023	HARE Framework	The paper introduces the HARE framework, which improves hate speech detection by using large language models to enhance explanation quality and model generalization.
5	Hate Speech Detection with Generalizable Target-aware Fairness	Tong Chen, Danny Wang, Xurong Liang, Marten Risius, Gianluca Demartini, Hongzhi Yin	The University of Queensland, 2023	GetFair	This study proposes GetFair, a method for fair hate speech detection across diverse and unseen targets, using a series of adversarially trained filter functions and hypernetworks.
6	An Interpretable Approach to Hateful Meme Detection	Tanvi Deshpande, Nitya Mani	ACM, 2021	Gradient-Boosted Decision Tree, LSTM-based model	This study focuses on detecting hateful memes by using interpretable machine learning models, such as a gradient-boosted decision tree and an LSTM- based model. The models achieve comparable performance to state-of- the-art transformer models and human classification in identifying hateful memes.
7	Combining FastText and Glove Word Embedding for Offensive and Hate Speech Text Detection	Nabil Badria, Ferihane Kboubia, Anja Habacha Chaibia	Elsevier, 2022	BiGRU, Glove, FastText	This paper introduces a method that combines Glove and FastText word embeddings with a BiGRU model for detecting offensive and hate speech in social media texts. The model achieved high performance metrics on the OLID dataset, with accuracy, precision, recall, and F1-score all exceeding 84%.
	A Systematic Review of Hate	Md Saroar Jahan, Mourad Oussalah	Elsevier, 2023	Various NLP and	The paper provides a systematic review of hate speech detection

DOI: 10.48175/568

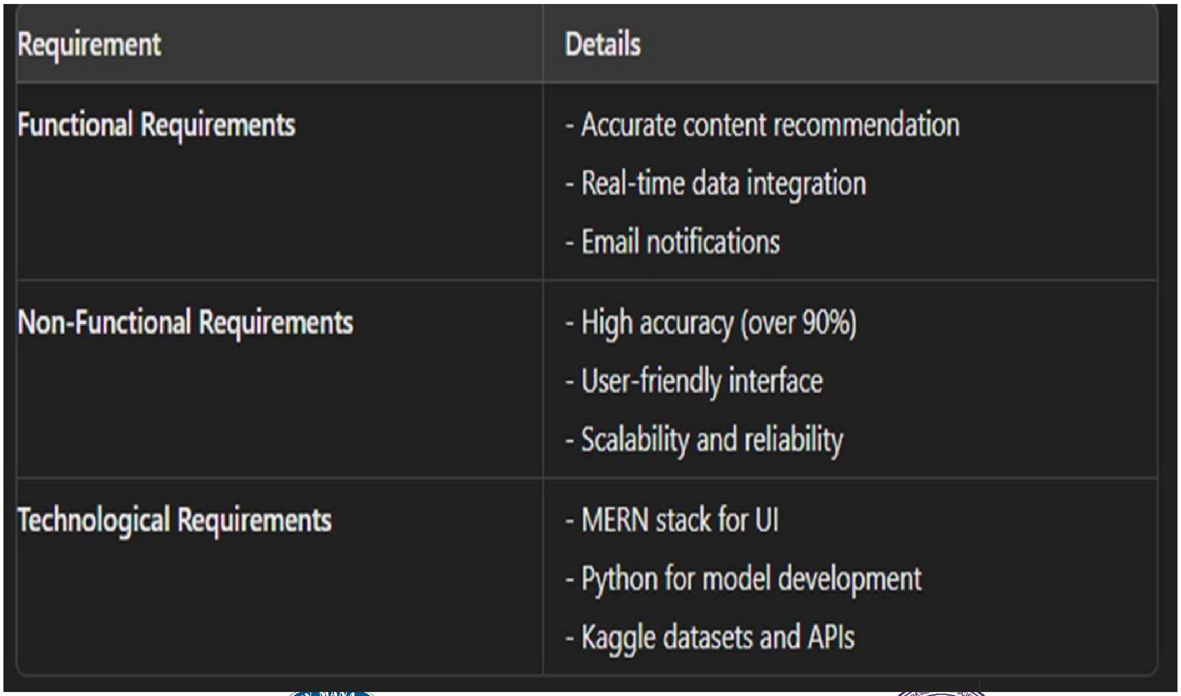
8	Speech Automatic Detection Using Natural Language Processing			Deep Learning Models	using NLP and deep learning technologies. It highlights the processing pipelines, core methods, and deep learning architectures, and discusses the limitations and future research directions in this field.
9	Hate Speech Detection: Challenges and Solutions	Sean MacAvaney, Hao-Ren Yao, Eugene Yang, Katina Russell, Nazli Goharian, Ophir Frieder	Information Retrieval Laboratory, Georgetown University, 2019	Multi-view SVM	This study examines the challenges of hate speech detection, including language subtleties, definitional issues, and data limitations. It proposes a multi-view SVM approach that offers high performance while being interpretable, unlike many neural methods.
10	A Literature Review of Textual Hate Speech Detection Methods and Datasets	Fatimah Alkomah, Xiaogang Ma	University of Idaho, 2022	Various Machine Learning Models	This literature review surveys textual hate speech detection methods and datasets. It highlights the primary datasets, textual features, and machine learning models used in the field. The study emphasizes the challenges with small and inconsistent datasets and calls for more robust research in this area.

Gap Analysis



Feature/Aspect	Existing Systems	Proposed System
Technology Stack	Various stacks, typically not unified across frontend and backend.	MERN stack for a unified frontend (React) and backend (Node.js) development.
Modeling Language	Predominantly Python or R, focusing on NLP models.	Python for model development, specifically for word processing and tokenization.
Data Sources	Typically rely on pre-existing datasets, some not updated regularly.	Combines Kaggle datasets with real-time data collection for more up-to-date models.
Accuracy	Varies, often between 70%-85% for hate speech detection.	Aims to achieve an accuracy of more than 90% through advanced tokenization and NLP.
User Interface (UI)	May not focus heavily on user experience, varies widely in design quality.	Intuitive and responsive UI developed with MERN stack for better user engagement.
Real-time Processing	Limited or non-existent in many existing systems.	Integrates real-time processing for up-to-date abusive attack prevention.
Notification System	Some systems have basic alerts or none at all.	Integrated Email API for real-time notifications, enhancing user awareness.
Input & Output	Often text-based with limited feedback.	Takes text input and provides detailed output, including notifications.

Requirement Analysis



Requirement	Details
Functional Requirements	- Accurate content recommendation - Real-time data integration - Email notifications
Non-Functional Requirements	- High accuracy (over 90%) - User-friendly interface - Scalability and reliability
Technological Requirements	- MERN stack for UI - Python for model development - Kaggle datasets and APIs

III. METHODOLOGY

- Research Phase: Study of existing algorithms and identification of optimal NLP , TF & IDF techniques.
- Design Phase: Designing system architecture, selecting datasets, and planning the UI/UX using MERN stack.
- Development Phase: Implementation of the algorithm using Python, data processing, integration of real-time data, and UI development.
- Testing Phase: Validation of system accuracy, user testing, and integration of Email API.
- Deployment Phase: Final system deployment and user feedback loop for continuous improvement.

Algorithms and Project Features

- Algorithms: Content-Based Filtering: Based on NLP, word processing, and tokenization. Email API Integration: For sending personalized content notifications.
- Project Features: High Accuracy: Targeting above 90%.Real-time Data Processing: Integration with real-time data sources. User-Friendly Interface: Developed using MERN stack. Scalability: System designed to handle large volumes of data.

Basic details of Implementation

Development Tools:

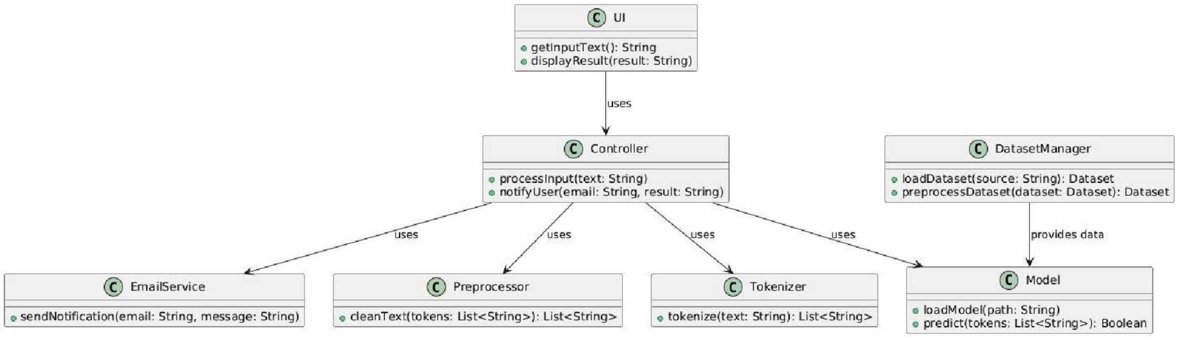
- UI: MERN stack (MongoDB, Express.js, React.js, Node.js). Modeling: Python (with NLP libraries like NLTK or SpaCy). Datasets: Kaggle datasets and real-time data sources.
- Notification System: Email API integration.

Implementation Steps:

- Step 1: Data collection and preprocessing. Step 2: Developing and testing the NLP model.
- Step 3: UI design and integration with the backend.
- Step 4: Real-time data integration and testing.
- Step 5: Final deployment and monitoring.

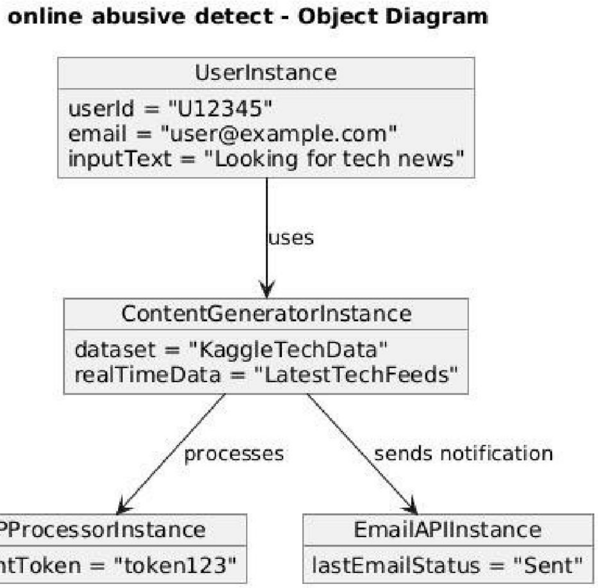
Project Design

Class

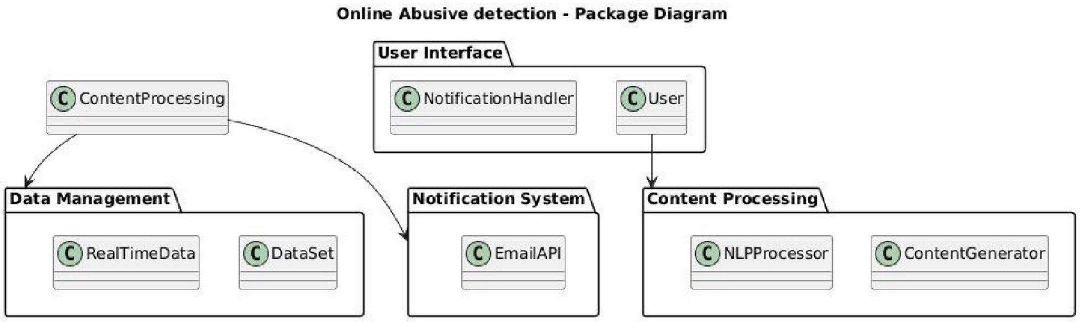

UI
getInputText(): String
displayResult(result: String)
uses
Controller
processInput(text: String)
notifyUser(email: String, result: String)
DatasetManager
loadDataset(source: String): Dataset
preprocessDataset(dataset: Dataset): Dataset
uses
uses
uses
uses
provides data
EmailService
sendNotification(email: String, message: String)
Preprocessor
cleanText(tokens: List<String>): List<String>
Tokenizer
tokenize(text: String): List<String>
Model
loadModel(path: String)
predict(tokens: List<String>): Boolean

DOI: 10.48175/568

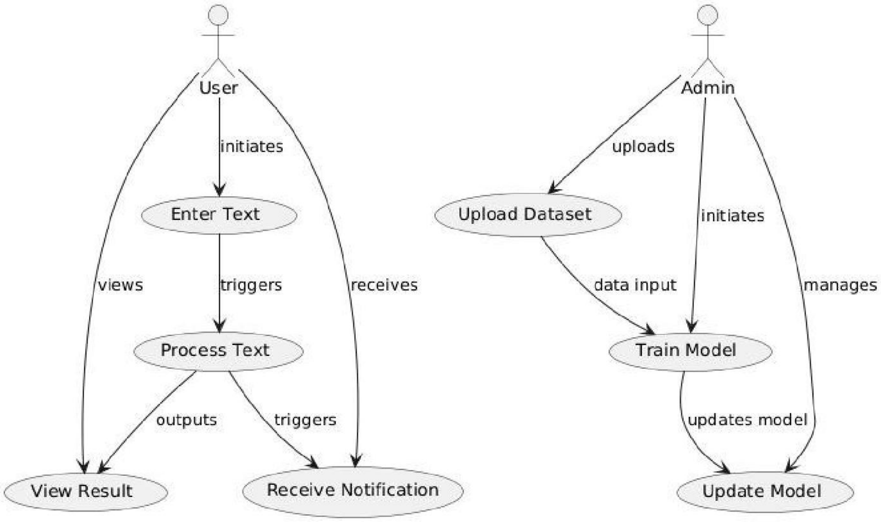
Object Diagram


online abusive detect - Object Diagram
UserInstance
userId = "U12345"
email = "user@example.com"
inputText = "Looking for tech news"
uses
ContentGeneratorInstance
dataset = "KaggleTechData"
realTimeData = "LatestTechFeeds"
processes
sends notification
NLPProcessorInstance
currentToken = "token123"
EmailAPIInstance
lastEmailStatus = "Sent"

Package Diagram

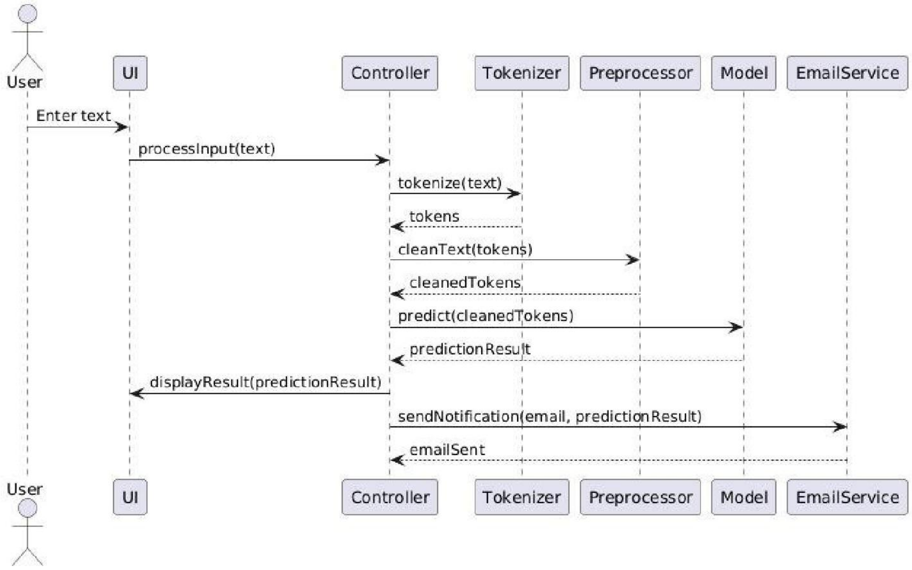


Use-Case

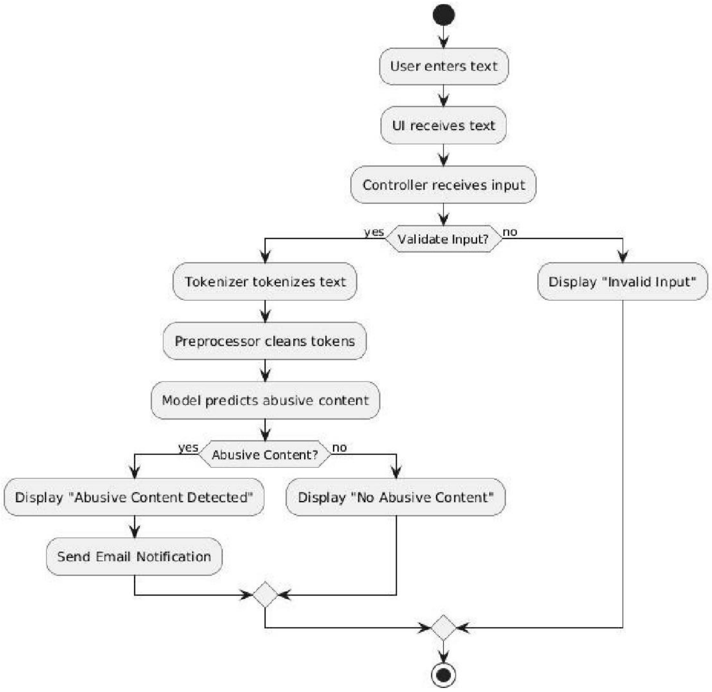


DOI: 10.48175/568

Sequence



Activity Diagram


User enters text
UI receives text
Controller receives input
Validate Input?
yes
no
Tokenizer tokenizes text
Display "Invalid Input"
Preprocessor cleans tokens
Model predicts abusive content
Abusive Content?
yes
no
Display "Abusive Content Detected"
Display "No Abusive Content"
Send Email Notification

IV. SOFTWARE AND HARDWARE REQUIREMENTS

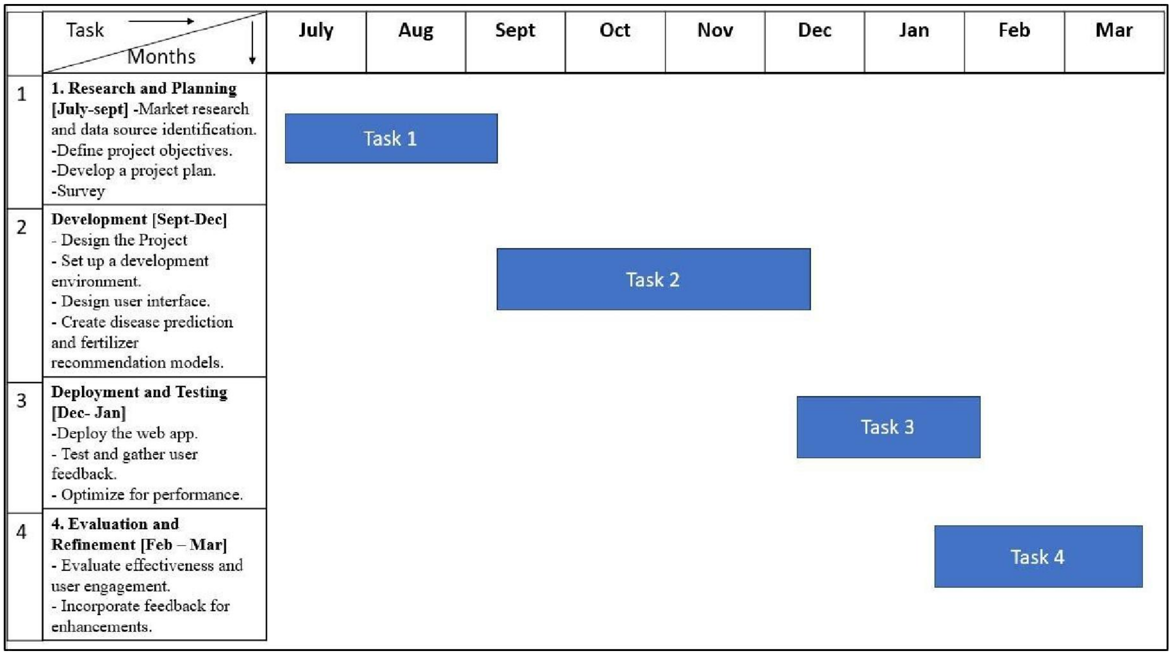
Software:

1. MERN stack for UI development
2. Python for backend processing
3. Email API for notifications

Hardware:

1. Standard development machine with internet connectivity.

Project Timeline



	Task → / Months ↓	July	Aug	Sept	Oct	Nov	Dec	Jan	Feb	Mar
1	**1. Research and Planning [July-sept]** -Market research and data source identification. -Define project objectives. -Develop a project plan. -Survey	Task 1	Task 1							
2	**Development [Sept-Dec]** - Design the Project - Set up a development environment. - Design user interface. - Create disease prediction and fertilizer recommendation models.			Task 2	Task 2	Task 2				
3	**Deployment and Testing [Dec- Jan]** -Deploy the web app. - Test and gather user feedback. - Optimize for performance.						Task 3	Task 3		
4	**4. Evaluation and Refinement [Feb – Mar]** - Evaluate effectiveness and user engagement. - Incorporate feedback for enhancements.								Task 4	Task 4

Motivation

1. Growing Online Threats : With the rise of digital abuse, our system provides a crucial solution to detect and prevent harmful language.
2. Advanced Technology : Utilizing cutting-edge word processing and tokenization, we empower users to combat online abuse effectively.
3. Immediate Response : Real-time analysis and notifications ensure prompt action against abusive behavior, protecting users.
4. Scalable Solution : Built with the MERN stack and Python, our system is robust and adaptable to various online environments.
5. Dedication to Safety : We are committed to fostering a safer online space, continuously enhancing our system for maximum impact.

V. CONCLUSION

• "Online Abusive Attack Prevention" stands as a powerful tool in the fight against online abuse. By integrating advanced word processing and real-time data analysis, our system effectively detects and mitigates harmful language, ensuring a safer and more respectful digital environment. Built on a scalable and robust architecture using the MERN stack and Python, our platform is designed to adapt and evolve with the ever-changing landscape of online interactions. With a commitment to accuracy and proactive protection, we empower users and administrators to take control of online safety, fostering positive and constructive communities.

REFERENCES

[1] Emotionally Informed Hate Speech Detection: A Multi- target Perspective by Patricia Chiril et al. at Springer.

[2] Hate Speech Classification Using SVM and Naive Bayes by IOSR Journal of Mobile Computing & Application.

[3] A survey on hate speech detection and sentiment analysis using machine learning and deep learning models by Malliga Subramanian et al. at Alexandria Engineering Journal.

[4] HARE: Explainable Hate Speech Detection with Step-by- Step Reasoning by Yongjin Yang et al. at KAIST.

[5] Hate Speech Detection with Generalizable Target-aware Fairness by Tong Chen et al. at The University of Queensland.