Chapter 1

Introduction

1.1 Project Introduction

Cyberbullying is one of the most serious issues emerging across digital platforms such as social media, messaging apps, and online communities. Offensive comments, threats, identity-based insults, and harassment severely impact mental health and online safety. Identifying such harmful content manually is extremely difficult due to the large volume of online messages.

This project, Cyberbullying & Hate Speech Detector, is built using HTML, CSS, JavaScript, TensorFlow.js, and the Toxicity Model. The system allows users to input any text and instantly detect whether it contains cyberbullying or hate speech. It provides predictions for categories such as Insult, Threat, Identity Attack, Obscene, Toxic, and Severe Toxic, along with graphical analysis using bar and pie charts. A dark/light mode toggle and history log enhance usability.

1.2 Problem Description

Online platforms generate millions of text messages every minute, making it nearly impossible to monitor harmful content manually. Cyberbullying often goes unnoticed until it causes emotional or psychological damage. Existing systems lack real-time detection, multi-label toxicity classification, and user-friendly interfaces.

The traditional approach relies on human moderators, making detection slow, inconsistent, and prone to bias. There is a strong need for an automated, real-time system that can detect multiple types of toxicity instantly and accurately.

1.3 Objectives of the Study

The main objectives are:

1. To develop a system that detects cyberbullying using a pre-trained toxicity model.

2. To classify text into multiple labels such as insult, threat, identity attack, and more.

3. To provide visual analysis using bar charts and pie charts.

4. To implement an easy-to-use interface with dark/light mode.

5. To maintain a history of previous analyses for quick reference.

1.4 Scope of the Project

The system detects six major categories of toxic content. It can be used in:

Educational platforms

Social media content moderation

Online chat monitoring

Mental health awareness applications

Real-time message checking for safe communication

1.5 Advantages

Real-time cyberbullying detection

Multi-label prediction with confidence scoring

Graph-based representation of toxicity

User-friendly UI with theme toggle

Stores history automatically

No backend required (fully client-side)

Chapter 2

Literature Review

2.1 Literature Survey

In the paper titled "Detecting Cyberbullying on Social Media using Machine Learning Techniques" [1], the authors Nahar, Li and Pang proposed a system that identifies bullying behaviour in online text using classification algorithms such as Naive Bayes, Support Vector Machine (SVM), and Decision Trees. The study highlights that SVM performs better due to its capability to handle high-dimensional text data and sparse feature vectors.

In "Automated Hate Speech Detection and the Problem of Offensive Language" [2], Davidson et al. developed a model that classifies text into three categories: hate speech, offensive language, and non-offensive content. Their approach uses Logistic Regression with TF-IDF features. Although effective, the model struggles to understand context and sarcasm, which limits accuracy in real-world scenarios.

In the paper titled "Convolutional Neural Networks for Cyberbullying Detection" [3], Park and Fung introduced a deep learning-based approach using CNNs to automatically learn linguistic features from raw text. Their results show that CNN models outperform traditional ML algorithms, especially when trained on large datasets from social platforms such as Twitter and Instagram.

These papers together provide a foundation for understanding both classical and modern approaches for cyberbullying and hate speech detection. They highlight the progression from traditional machine learning methods toward deep learning and pretrained models capable of handling multi-label classification with higher accuracy..

2.2 Comparative Analysis of the Related Work

The table 2.1 discusses the comparative analysis of the existing cyberbullying and toxicity detection systems.

Table 2.1: Comparative Analysis

Sl. No	Author(s)	Algorithms/Techniques	Performance Measures
1.	Nahar, Li & Pang	Naive Bayes, Support Vector Machine (SVM), Decision Tree	Accuracy, Precision
2.	Davidson, Warmsley, Macy & Weber	Logistic Regression with TF-IDF	F1 Score
3.	Park & Fung	Convolutional Neutral Networks (CNN)	Accuracy

2.3 Summary

The research papers discussed above provide a strong foundation for understanding the various techniques used in detecting cyberbullying and hate speech. Earlier studies relied heavily on traditional machine learning algorithms such as Naive Bayes, Support Vector Machines, and Logistic Regression, which showed reasonable performance but lacked contextual understanding. With the advancement of deep learning, models such as CNNs and LSTMs demonstrated improved accuracy by capturing linguistic patterns more effectively.

More recent developments, such as Google's Toxicity Model, use transfer learning and multi-label classification to accurately detect different categories of harmful content. These modern approaches address limitations found in traditional techniques, including the inability to understand context, sequential patterns, and subtle forms of online aggression.

Therefore, based on the existing literature, deep learning and pretrained models offer more accurate and scalable solutions for cyberbullying detection. This project builds upon these advancements by integrating the TensorFlow.js Toxicity Model to provide real-time, browser-based classification of toxic text.

Chapter 3

Problem Formulation

3.1 Problem Statement

With the rapid growth of social media and online communication, a large volume of harmful and abusive content is being generated, making it difficult to monitor cyberbullying manually. Traditional detection methods lack accuracy, as they depend on simple keyword matching and fail to understand the context or intent behind messages. This leads to undetected harmful content and false alerts. Therefore, there is a need for an automated system that can accurately identify different categories of toxic behaviour such as insults, threats, identity attacks, obscene language, and general toxicity. To address these challenges, a real-time, browser-based cyberbullying detection system using the TensorFlow.js Toxicity Model is proposed.

3.2 Objectives of the Present Study

The objectives of the proposed project are as follows:

1. To design and develop a platform capable of detecting cyberbullying and hate speech in online text.
2. To classify user input into categories such as toxic, offensive or safe using AI/ML models.
3. To implement NLP preprocessing techniques for better text analysis.
4. To apply deep learning/transformer models for accurate detection of abusive language.
5. To provide user-friendly interface.

3.3 Summary

The proposed system aims to provide an effective solution for detecting cyberbullying and hate speech by utilizing the TensorFlow.js Toxicity Model. The problem statement highlighted the need for an automated and accurate method to identify various types of toxic behaviour in online text. The objectives of the study establish a clear direction for implementing real-time detection, multi-label classification, graphical analysis, and a user-friendly interface. This chapter outlines the foundation for developing a reliable, browser-based cyberbullying detection system.

Chapter 4

Requirements and Methodology

4.1 Hardware Requirements

The hardware requirements for the proposed project are depicted in Table 4.1.

Table 4.1: Hardware requirements

Sl. No	Hardware/Equipment	Specification
1.	Processor	Intel i3 or above
2.	RAM	4GB or above
3.	Storage	Minimum 1 GB free
4.	Display	720p resolution or higher

4.2 Software Requirements

The software requirements for the proposed project are depicted in Table 4.2.

Table 4.2: Software requirements

Sl. No	Software	Specification
1.	Web Browser	Google Chrome/Edge/Firefox
2.	Programming Languages	HTML, CSS, JavaScript
3.	Framework	TensorFlow.js
4.	Library	Chart.js
5.	Storage	LocalStorage API

4.3 Methodology Used

The proposed stroke prediction system is implemented using the following steps:

1. Input Collection:

The user enters a text message in the input box provided in the web interface.

2. Model Loading:

The TensorFlow.js Toxicity Model is loaded in the browser with a predefined threshold to classify text into multiple toxicity labels.

3. Pre-processing:

The entered text is trimmed and passed to the model in the required array format for prediction.

4. Classification:

The model analyses the text and predicts whether each label (Toxic, Insult, Threat, Obscene, Identity Attack, Severe Toxic) matches, along with probability scores.

5. Result Display:

The system displays the final prediction, confidence score, and generates bar and pie charts for visual representation.

6. History Storage:

The last five analyses are saved using localStorage and displayed in the Past Analyses section.

7. Theme Toggle & UI Rendering:

The interface updates dynamically with light/dark mode, charts rendering, and result formatting.

Chapter 5

System Architecture

5.1 Architecture of the Proposed System

Figure 5.1 shows the architecture of the proposed system.

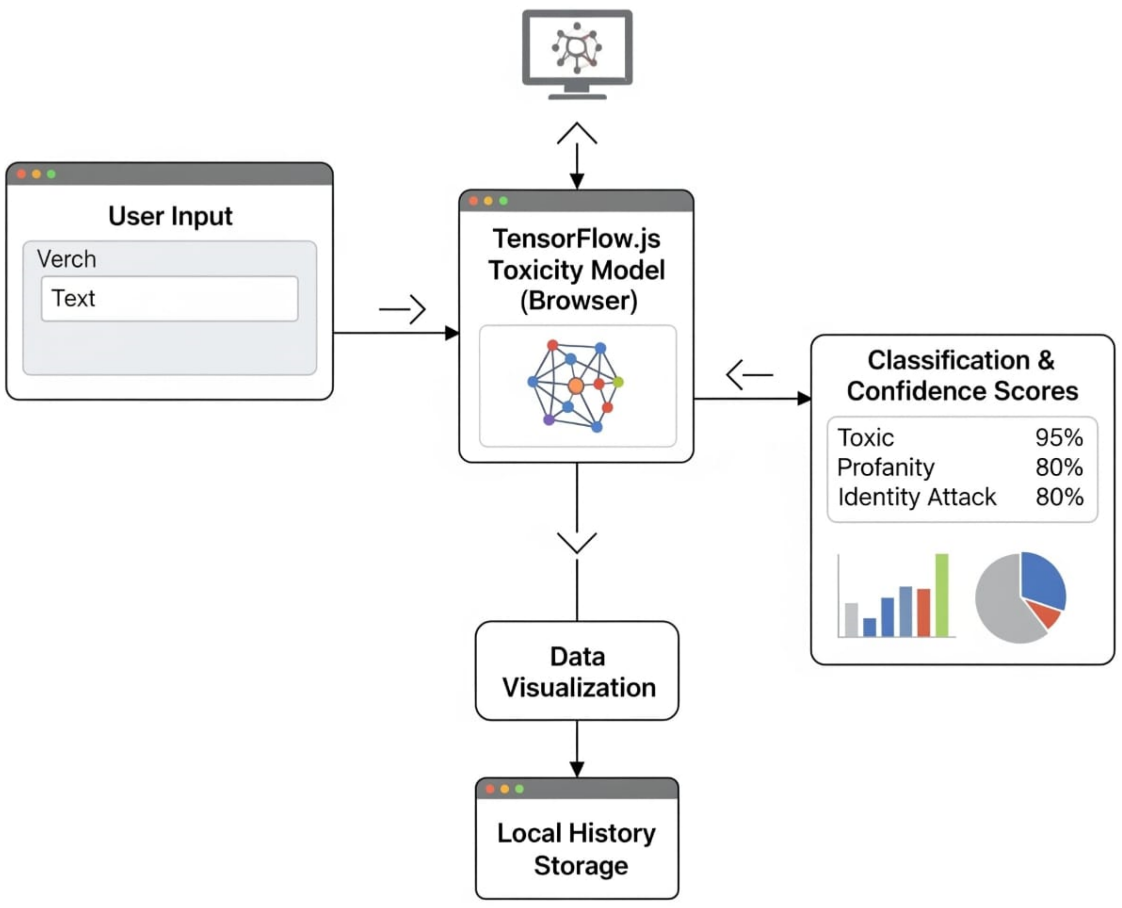


Figure 5.1: Architecture of the proposed system

The first step involves accepting the input text from the user through the interface. The text is then passed to the TensorFlow.js Toxicity Model, which is loaded in the browser. The model processes the input and classifies it into multiple toxicity categories such as insult, threat, identity attack, obscene, toxic, and severe toxic. Based on the probability values returned by the model, the system displays the final prediction along with the confidence percentage. The results are then visualized using bar and pie charts, and the analyzed text is stored in the local history for future reference.

5.2 Datasets Used

Overview

The proposed system uses the pretrained Toxicity Dataset released by Jigsaw/Google under the Perspective API project. This dataset is widely used for training online toxicity and cyberbullying detection models.

Dataset Description

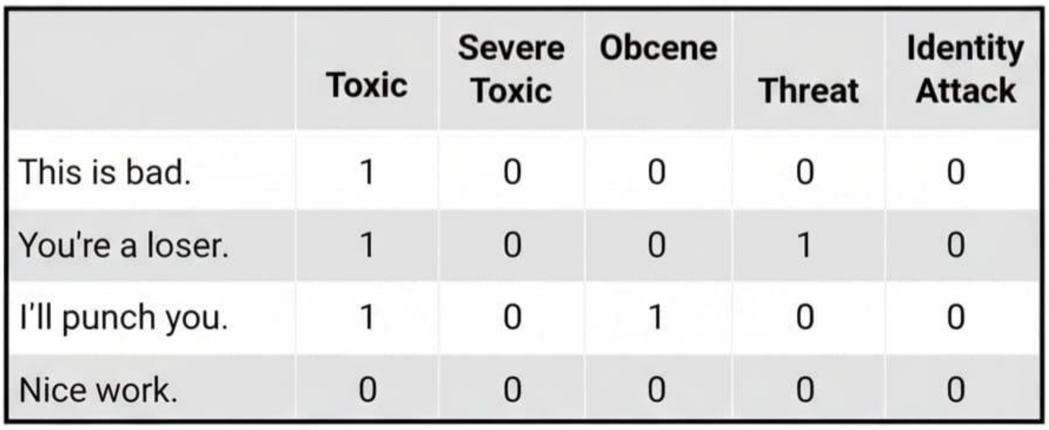
The dataset contains thousands of user comments collected from online discussion forums.

Each comment is manually labelled into multiple categories such as:
Toxic, Severe Toxic, Obscene, Threat, Insult, Identity Attack.

The dataset is used to train the TensorFlow.js Toxicity Model, which is integrated into the project for real-time predictions.

It is a multi-label classification dataset, meaning a comment can belong to more than one toxicity category at the same time.

Sample Dataset Snapshot



	Toxic	Severe Toxic	Obcene	Threat	Identity Attack
This is bad.	1	0	0	0	0
You're a loser.	1	0	0	1	0
I'll punch you.	1	0	1	0	0
Nice work.	0	0	0	0	0

5.3 Algorithms Used

The proposed system uses the TensorFlow.js Toxicity Model, which is a pre-trained deep learning model based on transfer learning. It identifies multiple toxicity categories such as Insult, Threat, Identity Attack, Obscene, Toxic, and Severe Toxic. The model works by analyzing input text and returning probability scores for each label.

Steps of the algorithm:

1. The user enters the text to be analyzed.
2. The text is passed to the Toxicity Model.
3. The model extracts the features and computes probability scores for each label.
4. Each label is marked as toxic or non-toxic based on the threshold value.
5. The predicted labels and confidence values are displayed to the user.

5.4 User Interface Design

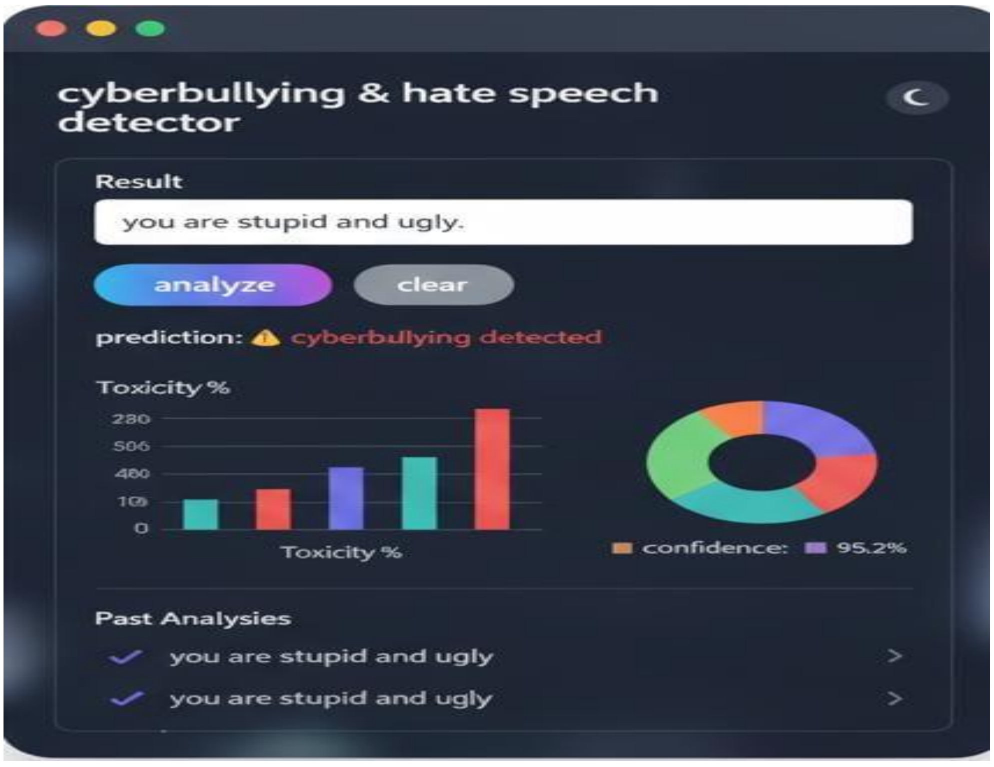
- **Overview of User Interface (UI):**

  The user interface is designed as a single-page web application that provides a clean and intuitive layout for analyzing text-based cyberbullying. It includes the input area, model status, prediction section, graphical representation, and history log.

- **Sketches of UI:**

  The User Interface (UI) of the system is designed to be simple, clean, and easy to use. The interface contains a text input area, buttons for analysis and clearing input, a result display section, and graphical output using bar and pie charts. A theme toggle is provided for switching between light mode and dark mode. The UI layout is centered and responsive to ensure proper visibility on all devices.

Figure 5.1:UI Sketch of Homepage in Dark Mode



you are stupid and ugly.

prediction: ⚠ cyberbullying detected

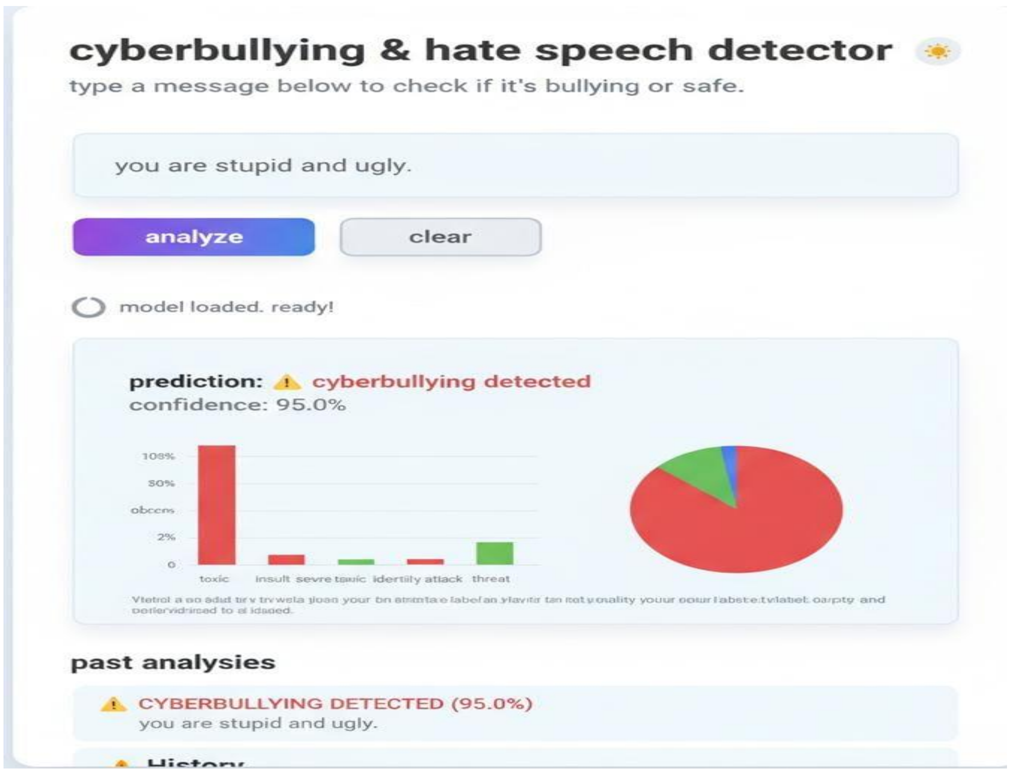
🟧 confidence: 🟪 95.2%

Past Analysies

✓ you are stupid and ugly

✓ you are stupid and ugly

Figure 5.2:UI Sketch of Homepage in Light Mode



cyberbullying & hate speech detector

type a message below to check if it's bullying or safe.

you are stupid and ugly.

model loaded. ready!

prediction: ⚠ cyberbullying detected
confidence: 95.0%

Vtetrol a so adul tirv trvwela jloan your bn atmtae label an ylavitr tan not y onality youur oour labste tvlatet oarpty and oetlervidnised to al idaged.

past analysies

⚠ CYBERBULLYING DETECTED (95.0%)
you are stupid and ugly.

⚠ History

- **User Interaction Flow:**
  1. User enters a message in the text box.

  2. The "Analyze" button processes the text using the toxicity model.

  3. Output is displayed along with confidence scores.

  4. Bar and pie charts provide visual interpretation.

  5. Past analyses are stored and displayed in the history section.

  6. User may toggle between dark and light themes.

- **UI Component Description:**

Component Name	Purpose	Layout	User Inputs
Textarea(Input Box)	Allows user to type message	Center of the page	Text message
Analyze Button	Starts toxicity analysis	Below input box	Click
Clear Button	Clears input text	Next to Analyze	Click

Output Section	Displays prediction & confidence	Below buttons	None
Bar & Pie Charts	Visual representation	Bottom section	None
Theme Toggle	Switch between dark & light modes	Top-right	Click
History Section	Shows upto last 5 analysis	Bottom area	None

5.5 Summary

This chapter discussed the implementation aspects of the system, including how the TensorFlow.js Toxicity Model is loaded, how user input is processed, and how the predictions and confidence values are generated. It also explained the features such as graph visualization, theme switching, and history storage. The steps involved in integrating the model with the user interface were described, along with the overall flow of how the system performs toxicity detection.

Chapter 6

Implementation

6.1 Pseudocode

```
// Pseudocode for Toxicity Classification

Input: User text input

1. Load the TensorFlow.js Toxicity Model with threshold T
2. Read the input text from the user
3. Preprocess the text and send it to the model
4. For each toxicity label Li
    a. Retrieve probability score Pi
    b. Check if $Pi \geq T$
    c. If yes, mark label as matched
5. Collect all matched labels
6. Display prediction result and confidence score
7. Generate bar chart and pie chart for label probabilities
8. Store text and prediction in history
```

Classification Label Decision

```
1. If any label ≥ threshold, classify as:
   Cyberbullying Detected
2. If no label meets threshold, classify as:
   Non-Cyberbullying
3. Highlight the highest probability label as the primary toxicity category.
```

Chapter 7

System Testing, Results and Discussion

7.1 System Testing

Table 7.1: Unit test cases

Test Case No.	**Input**	**Expected behavior**	**Observed behavior**	**Status P=Pass F=Fail**
1	Empty input text	System should display warning or disable analyze button	Warning message displayed	P
2	“Hello, how are you?”	Should classify as non-toxic	Displayed “Non-cyberbullying”	P
3	“You are stupid and ugly”	Should detect insult/toxic	Toxic category detected with high confidence	P
4	Very long paragraph s	System should not crash and should analyze smoothly	Processed without errors	P
5	“Have a nice day”	Should show non-toxic	Classified as safe	P

7.2 Result Analysis

The main objective of the project was to detect cyberbullying and hate speech in user-entered text using the TensorFlow.js Toxicity Model. Table 7.2 shows the Analysis of Toxicity Labels for Sample Input it analyses the amount of Toxic, Severe Toxic, Threat, Insult, Identity Attack, and Obscene.

Table 7.2: Analysis of Toxicity Labels for Sample Inputs

Input Text	Label	Confidence (%)	Detected (Yes/No)
You are stupid and ugly	Insult	92.4%	Yes
You are stupid and ugly	Toxic	88.1%	Yes
You are stupid and ugly	Identity Attack	12.5%	No
You are stupid and ugly	Severe Toxic	6.2%	No
You are stupid and ugly	Obscene	41.3%	No
You are stupid and ugly	Threat	1.8%	No

Figure 7.1 shows the bar graph generated for toxicity percentage(Insult, Threat, Toxic, etc)

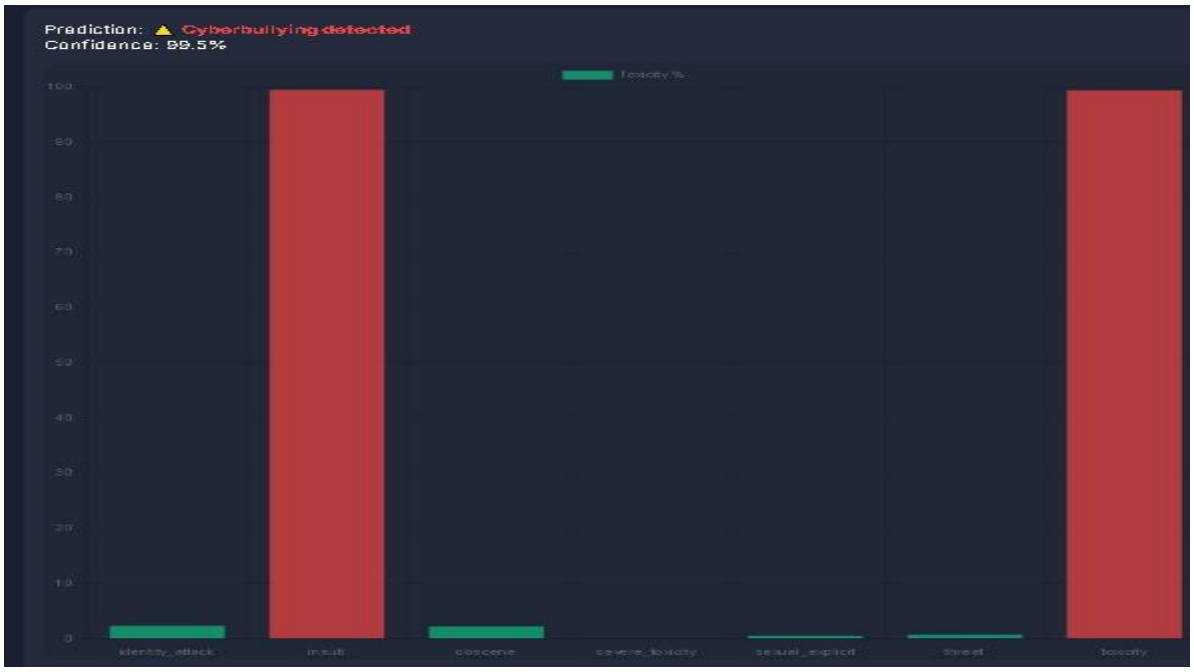


Figure 7.1: Bar graph analysis of sample text

Figure 7.2 shows the pie chart generated based on toxicity distribution.

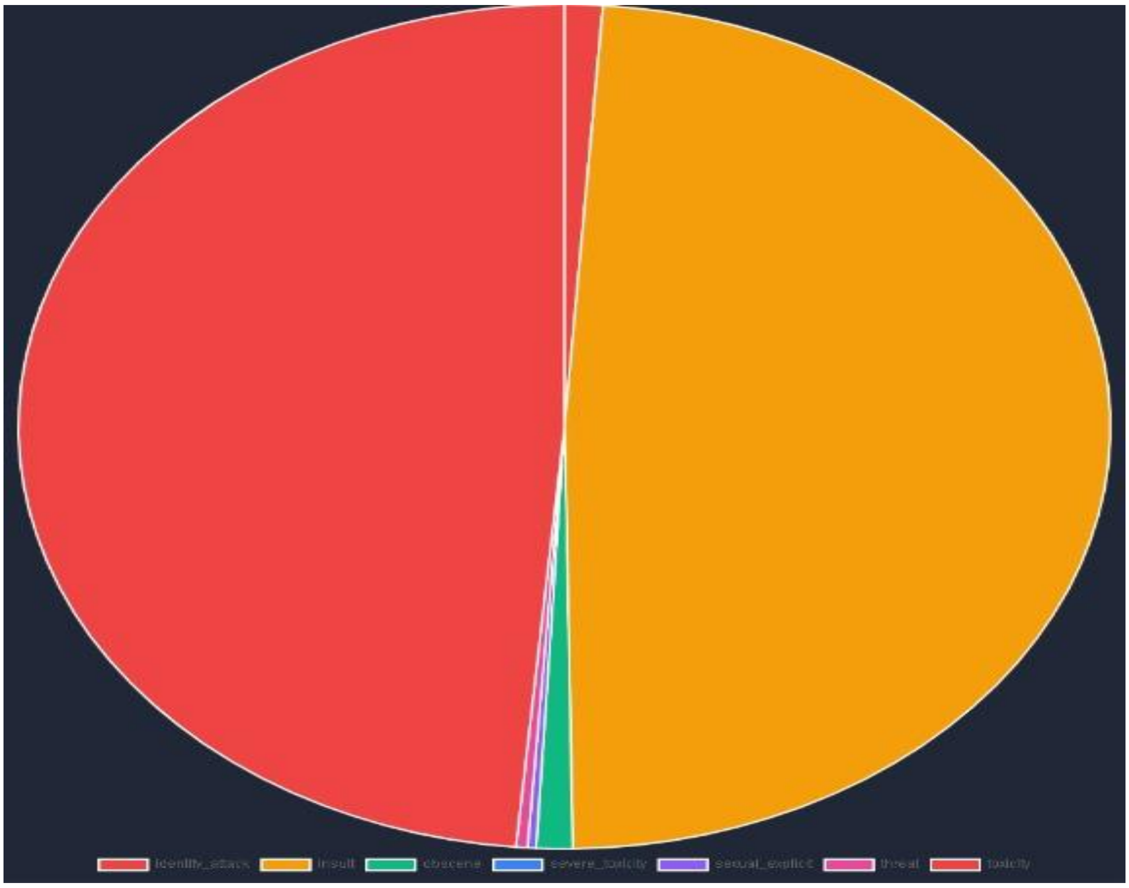
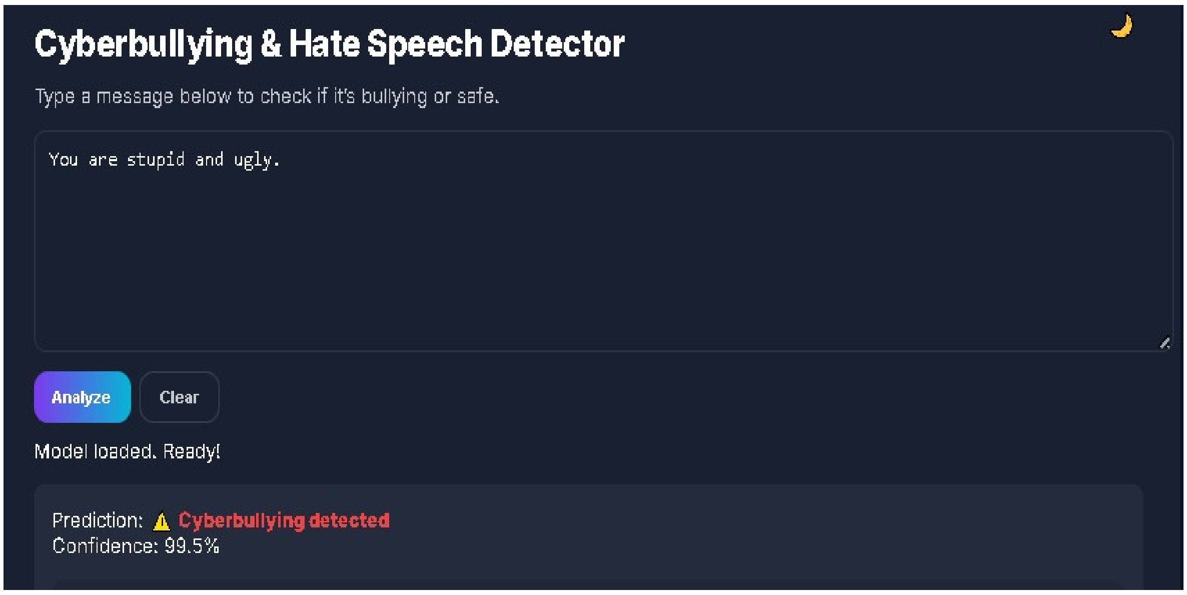


Figure 7.2: Pie chart analysis of the sample text

Figure 7.3 shows the input and prediction result page(Cyberbullying/Non-Cyberbullying)



Cyberbullying & Hate Speech Detector

Type a message below to check if it's bullying or safe.

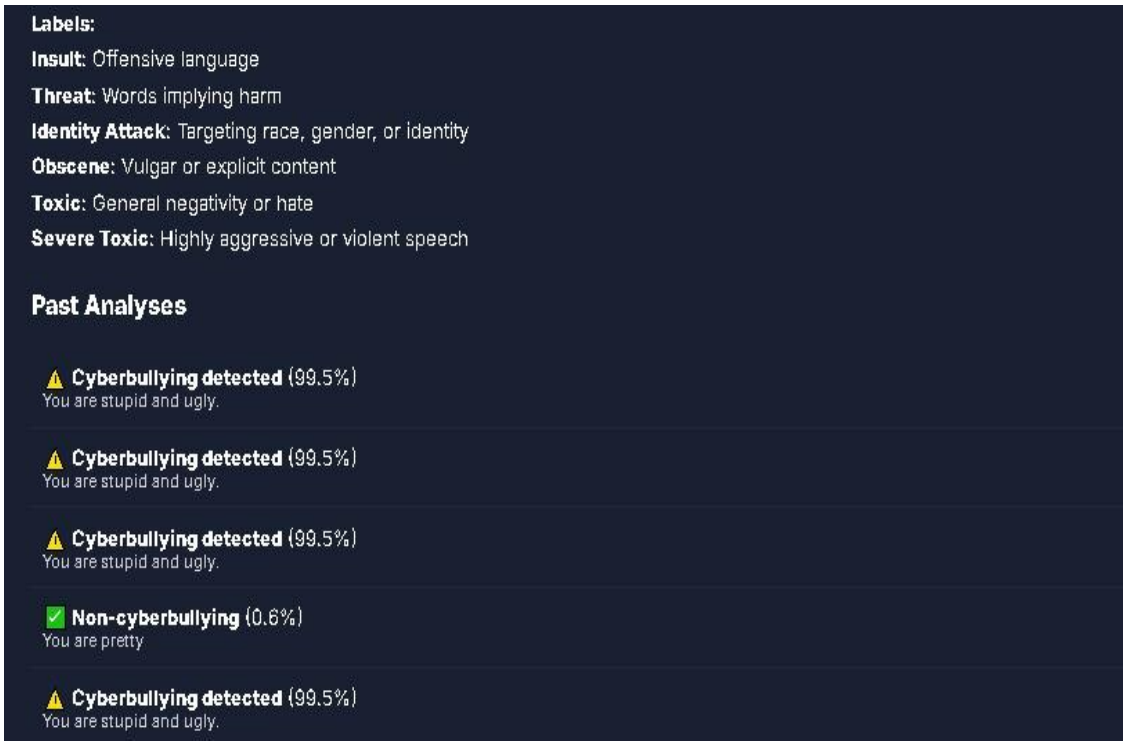
```
You are stupid and ugly.
```

Model loaded. Ready!

Prediction: ⚠ Cyberbullying detected
Confidence: 99.5%

Figure 7.3: Prediction of the sample input

Figure 7.4 shows the history page recording all the past inputs given by the user.



Labels:
Insult: Offensive language
Threat: Words implying harm
Identity Attack: Targeting race, gender, or identity
Obscene: Vulgar or explicit content
Toxic: General negativity or hate
Severe Toxic: Highly aggressive or violent speech

Past Analyses

⚠ **Cyberbullying detected** (99.5%)
You are stupid and ugly.

⚠ **Cyberbullying detected** (99.5%)
You are stupid and ugly.

⚠ **Cyberbullying detected** (99.5%)
You are stupid and ugly.

✅ **Non-cyberbullying** (0.6%)
You are pretty

⚠ **Cyberbullying detected** (99.5%)
You are stupid and ugly.

Figure 7.4: Past Analysis of User Inputs

Figure 7.5 is the login page for the admin to use the page.

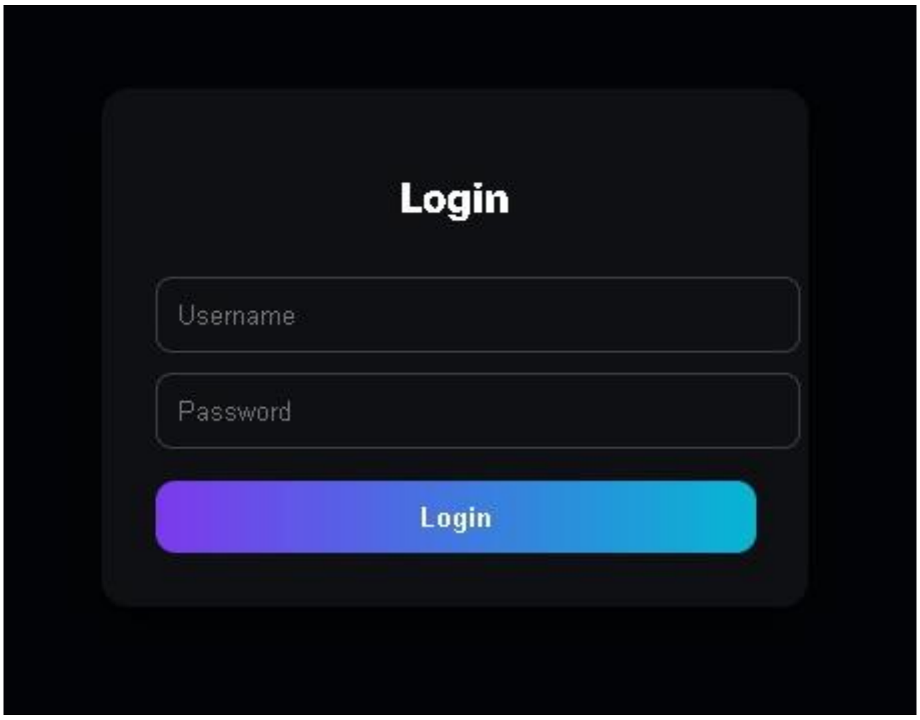


Figure 7.5: Login Page

7.3 Summary

In this chapter, the system was tested using different types of input text to analyze its performance in identifying multiple toxicity labels. For a single user input, the model generated separate probability values for each label such as Insult, Threat, Identity Attack, Obscene, Toxic, and Severe Toxic. Based on these scores, the system accurately highlighted the dominant label and classified whether the content was bullying or non-bullying. The graphical output clearly represented the distribution of toxicity levels through bar and pie charts, making it easier to understand how the model interprets the input. Overall, the results confirm that the system works effectively for multi-label toxicity detection and provides clear visual and textual interpretation for each prediction.

Chapter 8
Conclusion and Scope for Future Work

8.1 Conclusion

The proposed system provides an efficient way to detect cyberbullying and hate speech based on user-entered text using the TensorFlow.js Toxicity Model. The system accurately identifies different categories of toxic behaviour such as insults, threats, obscene language, identity attacks, and general toxicity. The implementation is simple, user-friendly, and works entirely in the browser without requiring any backend support. Features such as confidence scoring, graphical analysis, theme toggling, and history storage enhance its usability. Thus, the developed system is effective for real-time detection of harmful online content and contributes towards creating a safer digital environment.

8.2 Scope for Future Work

The system can be further improved by integrating multilingual support to analyse text in different Indian and foreign languages. Future enhancements may include real-time monitoring of social media platforms, addition of a login-based dashboard for administrators, and storing predictions in a database for large-scale analysis. Incorporating audio and image-based toxicity detection, as well as using advanced transformer models such as BERT or GPT-based classifiers, can further increase accuracy. These improvements would make the system more robust, scalable, and suitable for deployment in real-world applications.