**A PROJECT SYNOPSIS**

# on

# Hate Speech Detection

## Submitted By

1. **SAIRAJ ALAVE (01)**
2. **TANISH BODEKAR (08)**
3. **SAMI FODKAR (14)**
4. **OMKAR KADAM (28)**

**Under the Guidance of**

Prof. Maahi Khemchandani

## Department of Information Technology



Saraswati Education Society’s

**SARASWATI COLLEGE OF ENGINEERING**

Kharghar,Navi Mumbai

(Affiliated to University of Mumbai)

Academic Year :-2023-24

# Saraswati College of Engineering, Kharghar

**Vision:**

To be universally accepted as autonomous center of learning in engineering education and research.

**Mission:**

1. To educate Students to become responsible & quality technocrats to fulfill society and industry needs.
2. To nurture student’s creativity and skills for taking up challenges in all facets of life.

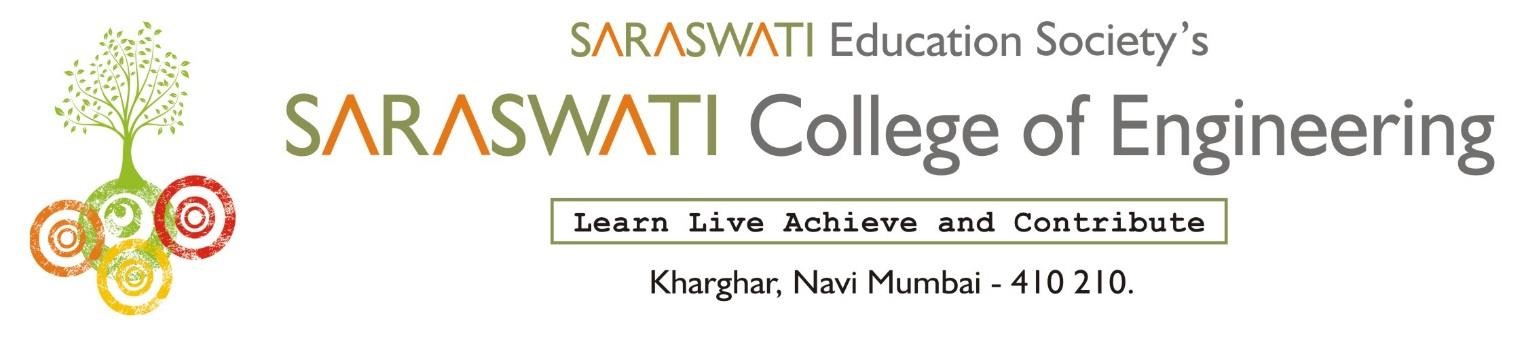
# Department of Information Technology

**Vision:**

To create technically qualified talent through research to take up challenges in industries.

**Mission:**

1. To strengthen theoretical and programming skills through teaching-learning and technical trainings to fulfill society and industry needs.
2. To nurture young minds to participate in research, innovation and entrepreneurship.
3. To encourage participation in co-curricular and extra-curricular activities to equip students with communication, team work and leadership skills.



**(Approved by AICTE, reg. By Maharashtra Govt. DTE, Affiliated to Mumbai University** **)**

**PLOT NO. 46/46A, SECTOR NO 5, BEHIND MSEB SUBSTATION, KHARGHAR,NAVI MUMBAI****-410210**

**Tel. : 022-27743706 to 11 \* Fax : 022-27743712 \* Website: www.sce.edu.in**

# CERTIFICATE

*This is to certify that the requirements for the synopsis entitled ” Hate Speech Detection ”* *have been successfully completed by the following students:*

Roll numbers Name

|  |  |  |
| --- | --- | --- |
| 01 |  | SAIRAJ ALAVE |
| 08 |  | TANISH BODEKAR |
| 14 |  | SAMI FODKAR |
| 28 |  | OMKAR KADAM |

In partial fulfillment of Sem – VI **Bachelor of Engineering of Mumbai University in Information Technology** of Saraswati College of Engineering, Kharghar during the academic year 2023-24.

**Internal Guide**  **External Examiner**

Prof. Oyendrilla Samanta

**Project coordinator Head of Department**

Prof. Oyendrilla SamantaProf. Maahi Khemchandani

# Acknowledgement

A project is something that could not have been materialized without cooperation of many people. This project shall be incomplete if I do not convey my heartfelt gratitude to those people from whom I have got considerable support and encouragement.

It is a matter of great pleasure for us to have a respected Prof. Supriya Kamble as my project guide. We are thankful to her for being constant source of inspiration.

We would also like to give our sincere thanks to **Prof. Maahi Khemchandani,** **H.O.D, Information Technology Department, Prof. Supriya Kamble, Project Co-ordinator** for their kind support.

We would like to express our deepest gratitude to **Dr. Manjusha Deshmukh, our Principal of**

**Saraswati college of Engineering, Kharghar**, Navi Mumbai.

Last but not the least I would also like to thank all the staffs of Saraswati college of Engineering (Information Technology Department) for their valuable guidance with their interest and valuable suggestions brightened us.

1. **SAIRAJ ALAVE (01)**
2. **TANISH BODEKAR (08)**
3. **SAMI FODKAR (13)**
4. **OMKAR KADAM (29)**

## HATE SPEECH DETECTION

### ABSTRACT

Hate speech online has become an increasingly pervasive and harmful issue in today's digital age.

To address this problem, this project proposes a solution of using a machine learning algorithm integrated into a Chrome extension for real-time hate speech detection. The proposed solution aims

to provide users with a safer and more inclusive online environment by detecting and flagging hate speech in real-time, while also increasing awareness about the prevalence and impact of hate speech online. In addition, this project emphasizes the importance of non-technical solutions, such as education and awareness-raising campaigns, clear policies and guidelines by online platforms, and collaboration across different sectors of society. In today's era hate speech emerge as a serious problem in our country.

This project aims to how to use artificial intelligence to identify hate speech and censor it. In this project, we will provide a chrome extension to identify certain keywords and sentences which are inappropriate and abusive.

This aims to classify textual content into non-hate or hate speech, in which case the method may

also identify the targeting characteristics (i.e., types of hate, such as race, and religion) in the hate speech. The aftermath of using this solution can lead to increased online safety, improved reporting, increased awareness, and a positive social impact. Ultimately, the use of this solution can contribute to a more tolerant and inclusive online community.

## INDEX

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Topic** | **Page No.** |
| **1** | **Introduction** | **01** |
| **2** | **Literature Survey** | **03** |
| **3** | **Existing System** | **07** |
| **4** | **Problem Statement** | **08** |
| **5** | **Proposed System**  **Flowchart / Block Diagram / Architecture** | **09** |
| **6** | **Implementation** | **12** |
| **7** | **Results** | **20** |
| **8** | **Conclusion** | **23** |
| **9** | **References** | **24** |

**Program Educational Objectives (PEO)**

1. To embed a strong foundation of Information Technology Engineering fundamentals to identify, solve, analyze & design real time engineering problem as a professional or an entrepreneur for the benefit of society.

2. To motivate & prepare students for lifelong learning & research to manifest global competitiveness.

3. To equip students with communication, team work & leadership skills to accept challenges in all facets of life ethically.

**Program Outcomes (PO)**

**At the end of the program, a student will be able to:**

1. Apply the knowledge of Mathematics, Science, Engineering fundamentals to solve complex Information Technology Engineering Problems.
2. Identify, formulate and analyze Information Technology Engineering problems to derive conclusion using first principles of mathematics and Computer Science.
3. Investigate complex Information Technology engineering problems and find appropriate solution leading to valid conclusion.
4. Design IT systems, components or processes to meet specified needs with appropriate attention to health, safety, standards, environmental and societal consideration.
5. To create select & apply appropriate techniques, resources advance engineering & software tools necessary to analyze & design Information Technology Problems.
6. Understand the impact of IT Solutions on society and environment for sustainable development.
7. Understand social, safety, cultural and legal issues and responsibilities relevant to engineering profession.
8. Apply professional ethics, accountability and equity in engineering profession.
9. Work effectively as a member and leader in multidisciplinary team for a common goal.
10. Communicate effectively within a profession and with society at large.
11. Appropriately incorporate principles of Management & Finance to one’s own work.
12. To identity Educational needs & engage in lifelong learning in a changing word of technology.

**Lab Objectives:**

Students will try to:

1. To acquaint with the process of identifying the needs and converting it into the problem.
2. To familiarize the process of solving the problem in a group.
3. To acquaint with the process of applying basic engineering fundamentals to attempt solutions to the problems.
4. To inculcate the process of self-learning and research.

**Lab Outcomes:**

Student will be able to:

1. Identify problems based on societal /research needs.

2. Apply Knowledge and skill to solve societal problems in a group.

3. Develop interpersonal skills to work as member of a group or leader.

4. Draw the proper inferences from available results through theoretical/ experimental/simulations.

5. Analyse the impact of solutions in societal and environmental context for sustainable development.

6. Use standard norms of engineering practices

7. Excel in written and oral communication.

8. Demonstrate capabilities of self-learning in a group, which leads to lifelong learning.

9. Demonstrate project management principles during project work.

## Chapter 1-Introduction

### Nowadays hate speech, through various medium is very common to see and read about it. It has become one of the biggest problems seen by the society and Identifying hate speech on social media is a crucial first step toward addressing this problem head-on and preventing it from spreading even further. Due to some of these event takes happens in the day-to-day life there is serious need to counter this problem to protect the society from hate.

### In today's world even young people are also on social media and many other media sites through which they consume content. So, it’s very for them hear and see hate speech and exchange inappropriate words. There is a strong need to select this topic in order to stop hate speech and inappropriate words because through this project we can identify this inappropriate word in the content.

### There are also already build product which are like this product the key difference is the efficiency. It is going benefit the entire society Building a Chrome extension for detecting hate speech is a complex task that requires careful planning and research. To start, it's important to look at existing hate speech detection tools and algorithms to get a better understanding of the current state of the field. Next, you'll need to determine the scope of the extension and decide which types of hate speech you want to detect. Then, you'll need to identify potential data sources to build your detection model. After that, you'll need to design an interface for your extension and test it to ensure its accurate and reliable.

Finally, you'll need to address any privacy concerns and ensure that you're not collecting or storing any sensitive user data without their explicit consent. Overall, building an effective hate speech detection tool using a Chrome extension will require a multidisciplinary team with expertise in Python, JavaScript, CSS and JSON.

Hate speech detection is an important research area in natural language processing and machine learning, which involves identifying and classifying hate speech in text data. The goal is to develop automated systems that can accurately detect hate speech, enabling social media platforms and other online communities to effectively moderate user-generated content and maintain a safe and inclusive environment for all users.

We found out that java script is the suitable language for our project. Then we discussed with our mentor to confirm that we would perform the creation of Chrome extension using JSON file. Hate speech detection using a Chrome extension focuses on identifying and flagging user-generated text that contains hateful or discriminatory language. The goal of the extension is to provide users with a tool to help them avoid encountering hate speech online. The way it works is it scans or checks every word on the web page, AI recognize whether that piece of information promotes hate speech or inappropriate content.

Further we will classify the toxicity rate of the web page.

Python is widely used in hate speech detection due to its versatility, extensive libraries, and ease of use. Python is a go-to language for building machine learning models. Libraries like Scikit-learn, TensorFlow, PyTorch, and Keras offer a wide range of algorithms and tools for training and deploying models. Commonly used models for hate speech detection include Support Vector Machines (SVM), Naive Bayes, Random Forests, and deep learning models like LSTM or Transformer-based architectures. Overall, Python's rich ecosystem of libraries and frameworks makes it a powerful tool for developing, deploying, and maintaining hate speech detection solutions, contributing to efforts aimed at promoting a safer and more inclusive online environment.

AI can help identify hate speech by looking at a piece of text and applying an algorithm to detect features that might indicate the content is malicious.

Type of hate speech: The scope of the project can be determined by the type of hate speech that the extension will be detecting. This could include racism, sexism, homophobia, transphobia, or any other type of discrimination. The scope of our hate speech detection Chrome extension could also be limited by technical constraints, such as the accuracy of machine learning model, the amount of data that can be processed in real-time, and the computational resources available.

Overall, hate speech detection is a rapidly evolving research area, with significant potential for impact in social media platforms, online communities, and other applications. By developing accurate and effective hate speech detection systems, we can help promote a safer and more inclusive online environment for all users.

## Chapter 2 - Literature Survey

Hate speech has been an ongoing problem on the Internet for many years. Besides, social media, especially Facebook, and Twitter have given it a global stage where those hate speeches can spread far more rapidly. Every social media platform needs to implement an effective hate speech detection system to remove offensive content in real-time. There are various approaches to identify hate speech, such as Rule-Based, Machine Learning based, deep learning based and hybrid approach.

Social networking sites are the most efficient way to meet new people. However, as social networking sites have grown in popularity, people have discovered an illegal and immoral way to use them. The most commonly encountered and most dangerous misuses of online social media are the expression of hate and harassment. Hate speech may be characterized as violence, hate, intimidation, racism, threats, harassment, insults, provocation, or sexism. These are some of the biggest threats to a social media site online. Several studies have already been worked into the identification of hateful messages in social media platforms, along with the dissemination of hateful messages on the dark web. Certain studies have implemented the domain of detection of hate speech but are primarily focused on supervised learning approaches instruction set.

A Literature Review of Textual Hate Speech Detection Methods and Datasets

by Fatimah Alkomah and Xiaogang Ma:

Hate speech is a form of writing that disparages and is likely to cause damage or danger to the victim on social media. It is a partial, aggressive and malicious speech that targets an individual or a group of people because of their conscious or unconscious intrinsic characteristics. It is a type of speech that shows a strong intent to cause harm, provoke violence, or encourage hate.

The social media environment and collaborative worldwide web offer a conducive environment for hate messages against an alleged enemy group to be created, shared, and exchanged.

The Internet is inherently open and dynamic, but various communities have their own rules to define the limits of speech. These boundaries differ from one culture to the next and are shaped by historical events and cultural norms. The manual method of detecting and eliminating hate speech posts or comments is time-consuming and computationally expensive. Because of these issues and the prevalence of hateful content on social media, there is a strong case for automated hate speech identification. Since hate speech, abusive language, and offensive language have recently become subjects of general concern, detecting hate speech has grown to be a major topic by the community of natural language processing (NLP), as demonstrated by the creation of datasets in a variety of languages. The implementation of systems for automatically detecting abusive and offensive language has followed a general pattern in NLP. Feature-based linear classifiers, fine-tuning pre-trained language models and neural network architectures. There are many approaches by which hate speech detection can be carried out, such as Machine learning, Deep learning, and the Rule-based approach.

In the literature review conducted by Alkomah and Ma in 2022, the authors discussed various approaches to hate speech detection, including:

1. Rule-Based Approach: This approach involves creating a set of rules or patterns to identify hate speech. The authors discussed the works of authors who have used this approach, including those who have used lexicons, gazetteers, and n-grams to identify hate speech.
2. Machine Learning-Based Approach: This approach involves training machine learning models to identify hate speech. The authors discussed various machine learning algorithms used for hate speech detection, including Naive Bayes, Support Vector Machines, and Random Forests.
3. Deep Learning-Based Approach: This approach involves using deep learning models to identify hate speech. The authors discussed various deep learning architectures used for hate speech detection, including Convolutional Neural Networks, Recurrent Neural Networks, and Long Short-Term Memory Networks.
4. Hybrid Approach: This approach involves combining rule-based and machine learning-based approaches to identify hate speech. The authors discussed the works of authors who have used this approach, including those who have used a combination of lexicons and machine learning models.

The authors also discussed various datasets used for hate speech detection, including the Hate Speech and Offensive Language Dataset (HateOffensive), the Kaggle Hate Speech Dataset, and the Wikipedia Talk Page dataset.

In a separate literature review conducted by Md Saroar Jahan and Mourad Oussalah in 2020, the authors discussed various natural language processing and deep learning technologies used for hate speech detection. The authors discussed the terminology used in hate speech detection, including hate speech, offensive language, and cyberbullying. The authors also discussed the processing pipeline used in hate speech detection, including tokenization, part-of-speech tagging, and parsing.

The authors discussed various core methods employed in hate speech detection, including feature extraction, feature engineering, and classification. The authors also discussed various deep learning architectures used for hate speech detection, including Convolutional Neural Networks, Recurrent Neural Networks, and Long Short-Term Memory Networks.

The authors also discussed various challenges in hate speech detection, including the subjectivity of hate speech, the lack of labelled data, and the need for context-aware hate speech detection.

Overall, these literature reviews provide a comprehensive overview of the current state of research in hate speech detection, highlighting the various approaches used and the challenges that remain to be addressed.

Automatic Hate Speech Detection: A Literature Review

By Mr. Mohiyaddeen and Sifatullah Siddiqi - Integral University Lucknow

In this paper, we carried out a comprehensive review of various approaches to detect hate speech on social media platforms that have been employed in recent years, along with a brief description of comparative analysis. Taking limited and public datasets for training hate speech detection model is one of the limitations found, and the model can be improved by using real-time big data sets. We have also found that the hate speech is not limited with texts only, but other modes of interactions, such as image and video detection, can also focus on the future.

Ongoing research efforts continue to advance hate speech detection technologies, aiming to mitigate online hate speech and promote a safer digital environment.

This literature survey provides a snapshot of the diverse approaches, challenges, and future directions in hate speech detection research.

Fighting Hate and Bigotry on the Internet

Raphael Cohen-Almagor

Internet hate speech is a specific type of online content that is designed to threaten certain groups publicly and act as propaganda for offline organizations. Hate groups use websites for sharing ideology, propaganda, linking to similar sites, recruiting new converts, advocating violence and threatening others. This article analyzes the ways hate speech on the Internet can be countered. From the perspective of applied ethics, it discusses the issue in the context of moral and social responsibility, a neglected perspective in the New Media literature. The study discusses the targets of hate on the Internet, provides a framework within which problems can be identified and resolved by accentuating moral and social responsibility, and articulates possible policy solutions to combat this increasing problem.

The advent of social media and online forums has revolutionized the way people communicate and express their opinions. However, this newfound freedom of expression has also given rise to the proliferation of hate speech, cyberbullying, and offensive content, which can have severe implications on individuals and society as a whole. Identifying and curbing such harmful contents has become a critical task for maintaining a respectful and safe online space. For instance, Modha et. dealt with the identification of the aggression types of texts in the online platforms and divided the texts into aggressive and non– aggressive depicts the percentage of hate speech texts posted in Instagram during the four quarters of the years 2020 and 2021. Kaur et. al. mentions the concepts of abusive content detection based on four categories of features namely, activity based, user based, context-based, and network-based features. This survey has also mentioned many parameters to identify the abusive contents such as posts per day, age, gender, etc and helps to build the researchers with fundamental concepts and key insight areas including the recent trends and techniques.

**Chapter 3 - Existing System**

Several existing systems for hate speech detection have been developed, employing various techniques ranging from rule-based approaches to machine learning and deep learning models. Here are some notable systems:

1. Google Perspective API: Developed by Google's Jigsaw, this API uses machine learning models to identify toxic language, including hate speech, in text inputs. It provides a toxicity score for each input text.
2. IBM Watson Content Moderator: IBM's Content Moderator uses machine learning algorithms to detect and filter out inappropriate content, including hate speech, from text, images, and videos. It can be integrated into various platforms for content moderation.
3. HateSonar: HateSonar is a commercial hate speech detection API that uses deep learning models to analyze text and identify hate speech. It offers features such as sentiment analysis and context understanding to improve accuracy.
4. Twitter's Automated Systems: Twitter employs automated systems to detect and take action against accounts and content that violate its hate speech policies. These systems use a combination of machine learning algorithms and human review to flag and remove hateful content.
5. DeepMoji: DeepMoji is a deep learning model developed by MIT that can recognize emotion and sentiment in text, including detecting hateful or offensive language. It uses a large dataset of emojis to understand the emotional context of text.
6. HateXplain: HateXplain is a system that aims to explain why certain content is classified as hate speech. It uses natural language processing techniques to generate explanations for hate speech detection results, improving transparency and interpretability.
7. HateCheck: HateCheck is a platform that combines machine learning models with human moderation to detect and combat hate speech on social media. It uses a collaborative approach to ensure accurate and timely detection of hateful content.

These systems utilize a combination of techniques such as natural language processing, machine learning, deep learning, and human moderation to effectively identify and mitigate hate speech in various online platforms and content types.**Top of Form**

## Chapter 4 -Problem Statement

The proliferation of hate speech and offensive language in online platforms has become a significant societal concern, leading to harmful effects on individuals and communities. Despite efforts to combat hate speech, automated detection systems face challenges in accurately identifying and categorizing such content due to the complex and evolving nature of language use on the internet. This necessitates the development of robust hate speech detection systems that can effectively distinguish between free expression and harmful speech, thereby fostering a safer and more inclusive online environment.

Key Components of the Problem Statement:

1. Prevalence of Hate Speech: Acknowledge the widespread existence of hate speech and offensive language across various online platforms, highlighting its negative impact on individuals and society as a whole.
2. Challenges in Detection: Recognize the difficulties faced by automated systems in accurately detecting and categorizing hate speech due to the dynamic nature of language, context dependency, sarcasm, and cultural nuances.
3. Need for Robust Solutions: Emphasize the necessity of developing robust hate speech detection systems that can effectively differentiate between protected speech and harmful content, ensuring a balance between free expression and preventing harm.
4. Societal Impact: Highlight the importance of addressing hate speech to promote a safer and more inclusive online environment, thereby mitigating the negative consequences of online toxicity on individuals' well-being and social cohesion.

By framing the problem statement in this manner, researchers and developers can focus on creating innovative solutions that leverage advanced technologies such as natural language processing, machine learning, and contextual understanding to tackle the challenges associated with hate speech detection effectively.

## Chapter 5 – Proposed System

Hate speech detection using a Chrome extension focuses on identifying and flagging user-generated text that contains hateful or discriminatory language. The goal of the extension is to provide users with a tool to help them avoid encountering hate speech online.

The way it works is it scans or checks every word on the web page, the Extension recognize whether that piece of information promotes hate speech or inappropriate content.

The Extension can help identify hate speech by looking at a piece of text and applying an algorithm to detect features that might indicate the content is malicious.

Type of hate speech: The scope of the project can be determined by the type of hate speech that the extension will be detecting. This could include racism, sexism, homophobia, transphobia, or any other type of discrimination. The scope of our hate speech detection Chrome extension could also be limited by technical constraints, such as the accuracy of the machine learning model, the amount of data that can be processed in real-time, and the computational resources available.

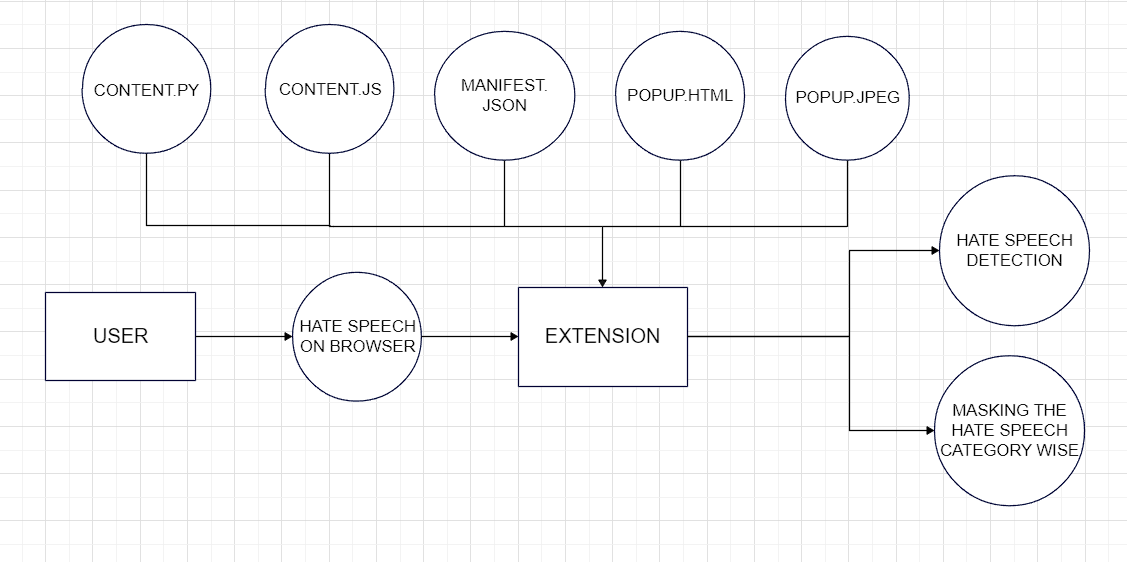


Fig.5.1 Block diagram of Hate Speech Detection

**Content.py :**  This Python script is a utility for modifying HTML content by replacing specific words with colored spans and restoring input values based on defined rules.

The example usage provided at the end of the script demonstrates how these functions can be applied. It initializes an HTML string with some content, defines lists of words categorized by color (e.g., black, red, purple), specifies colors for each category, and then uses the replace\_words\_with\_color function to colorize specific words in the HTML. After that, it uses the restore\_inputs function to restore specific input values.

Overall, this script provides a way to dynamically modify and colorize content in HTML based on predefined rules, which can be useful for tasks like highlighting sensitive words or restoring modified input values in a web application.

**Content.js** : This code is a JavaScript snippet that performs text replacement with HTML markup to highlight or colorize specific words based on predefined categories such as racism, harassment, sexism, etc. It is the main file which also masked the hateful words in each category with specific color codes.

This code provides a more efficient and structured approach to dynamically highlighting words in specific categories within HTML content and safely restoring input field values to their original states. Overall, the code is attempting to perform dynamic text replacement to highlight or colorize words in specific categories across the entire HTML content.

**Manifest.json** : This file is used to create the extension for the browser, it acts as the main structure for the Extension. A manifest.json file is a crucial component of a browser extension. It contains metadata about the extension, such as its name, version, permissions, and other important details.

**Popup.html** : This HTML file sets up the fundamental structure of an HTML document, provides essential meta-information for browser rendering and responsiveness, and attempts to display an image without any additional content in the <body> section. The overall purpose of the above HTML code is to create a basic web page structure with essential meta-information and an image displayed on the page.

**Popup.jpeg** : It is the actual image which is linked in the popup.html file, it showcases the various hate speech categories and also the color used to masked the hateful words belonging from specific categories to user.

Hence, the Extension works by using the above components to detect the hate speech whenever a user encounters such information on the browser and also masks it, in order to prevent the spreading of hateful content.

**Chapter 6 – Implementation**

**manifest.json:**

{

    "manifest\_version": 3,

    "name": "Hate Speech Detector",

    "version": "1.0",

    "action": {

        "default\_popup": "popup.html"

    },

    "content\_scripts": [{

        "matches": [

            "<all\_urls>"

        ],

        "js": ["content.js"]

    }]

}

**content.js:**

var black = ['freak','in shit','special','highlight']; // harassment/bullying

var red = ['nigga','javascript'];  // racism

var purple = ['batsman']; // sexism

var yellow = ['bhangi','dhobi']; //casteism

var pink = ['pansy','queer','gay','lesbian']; //sexual-oritentation

var blue = ['crippled']; //disability

var blackColor = "#000000";

var purpleColor = "#A020F0";

var redColor = "#FF0000";

var yellowColor = "#FFFF00";

var pinkColor = "#FFC0CB";

var blueColor = "#0000FF";

function restoreInputs(oldText,newText) {

    // Select all <input> tags

    const inputs = document.querySelectorAll('input');

    // Loop through all <input> tags and replace the text in their value attribute

    inputs.forEach(input => {

        const oldValue = input.getAttribute('value');

        const newValue = oldValue.replace(oldText, newText);

        input.setAttribute('value', newValue);

        alert(newValue);

    });

}

for (let i = 0; i < red.length; i++) {

    console.log(red[i]);

    let wordToChange1 = red[i];

    var re = new RegExp(wordToChange1, "gi");

    var replaceWith = "<span style='color:" + redColor + ";'>$&</span>";

    var replaceWith1 = "<span style='background-color:" + redColor + ";'>$&</span>";

    document.body.innerHTML = document.body.innerHTML.replace(re, replaceWith);

    document.body.innerHTML = document.body.innerHTML.replace(re, replaceWith1);

searchInInput = "value='<span style='color:" + redColor + ";'><span style='background-color:" + redColor + ";'>" + wordToChange1 + "</span></span>'";

searchInInput2 = "value=\"<span style='color:" + redColor + ";'><span style='background-color:" + redColor + ";'>" + wordToChange1 + "</span></span>\"";

document.body.innerHTML=document.body.innerHTML.replace(searchInInput,

"value='" + wordToChange1 + "'");

document.body.innerHTML=document.body.innerHTML.replace(searchInInput2,

"value='" + wordToChange1 + "'");

    // restoreInputs(searchInInput, wordToChange1);

}

for (let i = 0; i < black.length; i++) {

    console.log(black[i])

    let wordToChange1 = black[i];

    var re = new RegExp(wordToChange1, "gi");

    var replaceWith = "<span style='color:" + blackColor + ";'>$&</span>";

    var replaceWith1 = "<span style='background-color:" + blackColor + ";'>$&</span>";

    document.body.innerHTML = document.body.innerHTML.replace(re, replaceWith);

    document.body.innerHTML = document.body.innerHTML.replace(re, replaceWith1);

searchInInput = "value='<span style='color:" + blackColor + ";'><span style='background-color:" + blackColor + ";'>" + wordToChange1 + "</span></span>'";

searchInInput2 = "value=\"<span style='color:" + blackColor + ";'><span style='background-color:" + blackColor + ";'>" + wordToChange1 + "</span></span>\"";

document.body.innerHTML=document.body.innerHTML.replace(searchInInput,

"value='" + wordToChange1 + "'");

document.body.innerHTML=document.body.innerHTML.replace(searchInInput2,

"value='" + wordToChange1 + "'");

}

for (let i = 0; i < purple.length; i++) {

    console.log(purple[i])

    let wordToChange1 = purple[i];

    var re = new RegExp(wordToChange1, "gi");

    var replaceWith = "<span style='color:" + purpleColor + ";'>$&</span>";

    var replaceWith1 = "<span style='background-color:" + purpleColor +";'>$&</span>";

    document.body.innerHTML = document.body.innerHTML.replace(re, replaceWith);

    document.body.innerHTML = document.body.innerHTML.replace(re, replaceWith1);

searchInInput = "value='<span style='color:" + purpleColor + ";'><span style='background-color:" + purpleColor + ";'>" + wordToChange1 + "</span></span>'";

searchInInput2 = "value=\"<span style='color:" +purpleColor + ";'><span style='background-color:" + purpleColor + ";'>" + wordToChange1 + "</span></span>\"";

document.body.innerHTML=document.body.innerHTML.replace(searchInInput,

"value='" + wordToChange1 + "'");

document.body.innerHTML=document.body.innerHTML.replace(searchInInput2,

"value='" + wordToChange1 + "'");

}

for (let i = 0; i < yellow.length; i++) {

    console.log(yellow[i])

    let wordToChange1 = yellow[i];

    var re = new RegExp(wordToChange1, "gi");

    var replaceWith = "<span style='color:" + yellowColor + ";'>$&</span>";

    var replaceWith1 = "<span style='background-color:" + yellowColor +";'>$&</span>";

    document.body.innerHTML = document.body.innerHTML.replace(re, replaceWith);

    document.body.innerHTML = document.body.innerHTML.replace(re, replaceWith1);

searchInInput = "value='<span style='color:" + yellowColor + ";'><span style='background-color:" + yellowColor + ";'>" + wordToChange1 + "</span></span>'";

searchInInput2 = "value=\"<span style='color:" +yellowColor + ";'><span style='background-color:" +  + yellowColor + ";'>" + wordToChange1 + "</span></span>\"";

document.body.innerHTML=document.body.innerHTML.replace(searchInInput,

"value='" + wordToChange1 + "'");

document.body.innerHTML=document.body.innerHTML.replace(searchInInput2,

"value='" + wordToChange1 + "'");

}

for (let i = 0; i < pink.length; i++) {

    console.log(pink[i])

    let wordToChange1 = pink[i];

    var re = new RegExp(wordToChange1, "gi");

    var replaceWith = "<span style='color:" + pinkColor + ";'>$&</span>";

    var replaceWith1 = "<span style='background-color:" + pinkColor + ";'>$&</span>";

    document.body.innerHTML = document.body.innerHTML.replace(re, replaceWith);

    document.body.innerHTML = document.body.innerHTML.replace(re, replaceWith1);

searchInInput = "value='<span style='color:" + pinkColor + ";'><span style='background-color:" + pinkColor + ";'>" + wordToChange1 + "</span></span>'";

searchInInput2 = "value=\"<span style='color:" + pinkColor + ";'><span style='background-color:" + pinkColor + ";'>" + wordToChange1 + "</span></span>\"";

document.body.innerHTML=document.body.innerHTML.replace(searchInInput,

"value='" + wordToChange1 + "'");

document.body.innerHTML=document.body.innerHTML.replace(searchInInput2,

"value='" + wordToChange1 + "'");

}

for (let i = 0; i < blue.length; i++) {

    console.log(blue[i])

    let wordToChange1 = blue[i];

    var re = new RegExp(wordToChange1, "gi");

    var replaceWith = "<span style='color:" + blueColor + ";'>$&</span>";

    var replaceWith1 = "<span style='background-color:" + blueColor + ";'>$&</span>";

    document.body.innerHTML = document.body.innerHTML.replace(re, replaceWith);

    document.body.innerHTML = document.body.innerHTML.replace(re, replaceWith1);

searchInInput = "value='<span style='color:" + blueColor + ";'><span style='background-color:" + blueColor + ";'>" + wordToChange1 + "</span></span>'";

searchInInput2 = "value=\"<span style='color:" +blueColor + ";'><span style='background-color:" + blueColor + ";'>" + wordToChange1 + "</span></span>\"";

document.body.innerHTML=document.body.innerHTML.replace(searchInInput,

"value='" + wordToChange1 + "'");

document.body.innerHTML=document.body.innerHTML.replace(searchInInput2,

"value='" + wordToChange1 + "'");

}

**content.py:**

import re

from bs4 import BeautifulSoup # type: ignore

def replace\_words\_with\_color(html, words, color):

    soup = BeautifulSoup(html, "html.parser")

    for word in words:

        re\_pattern = re.compile(re.escape(word), re.IGNORECASE)

        replace\_with = f'<span style="color: {color};">{word}</span>'

        soup.body.decodeContents().replace(re\_pattern, replace\_with)

    return str(soup)

def restore\_inputs(html, old\_text, new\_text):

    soup = BeautifulSoup(html, "html.parser")

    inputs = soup.find\_all("input")

    for input\_elem in inputs:

        old\_value = input\_elem.get("value", "")

        new\_value = old\_value.replace(old\_text, new\_text)

        input\_elem["value"] = new\_value

    return str(soup)

# Example usage

html = """

<div class="box">

  <div><strong>Log:</strong></div>

  <div class="log">

    <time>12:34:56</time>: <message>Initial content</message>

  </div>

</div>

"""

black = ['freak', 'in shit', 'special', 'highlight']  # harassment/bullying

red = ['nigga', 'javascript']  # racism

purple = ['batsman']  # sexism

yellow = ['bhangi', 'dhobi']  # casteism

pink = ['pansy', 'queer', 'gay', 'lesbian']  # sexual-orientation

blue = ['crippled']  # disability

black\_color = "#000000"

purple\_color = "#A020F0"

red\_color = "#FF0000"

yellow\_color = "#FFFF00"

pink\_color = "#FFC0CB"

blue\_color = "#0000FF"

html = replace\_words\_with\_color(html, red, red\_color)

html = replace\_words\_with\_color(html, black, black\_color)

html = replace\_words\_with\_color(html, purple, purple\_color)

html = replace\_words\_with\_color(html, yellow, yellow\_color)

html = replace\_words\_with\_color(html, pink, pink\_color)

html = replace\_words\_with\_color(html, blue, blue\_color)

html = restore\_inputs(html, f'<span style="color:{red\_color};"><span style="background-color:{red\_color};">nigga</span></span>', 'nigga')

html = restore\_inputs(html, f'<span style="color:{red\_color};"><span style="background-color:{red\_color};">javascript</span></span>', 'javascript')

print(html)

**popup.html:**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

    <img src="popup.jpeg" alt="Information" width="300px" height="200">

</head>

<body>

</body>

</html>

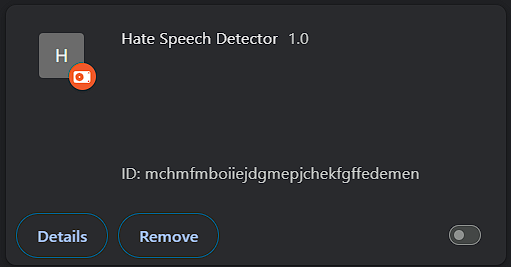
**popup.jpeg:**



Fig 6.1 Categories of Hate Speech

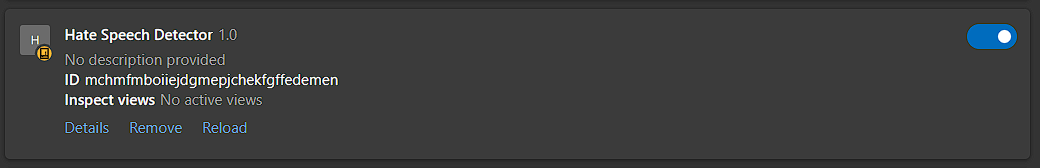
## Chapter 7 – Results

**7.1 EXTENSION ON CHROME BROWSER**

****

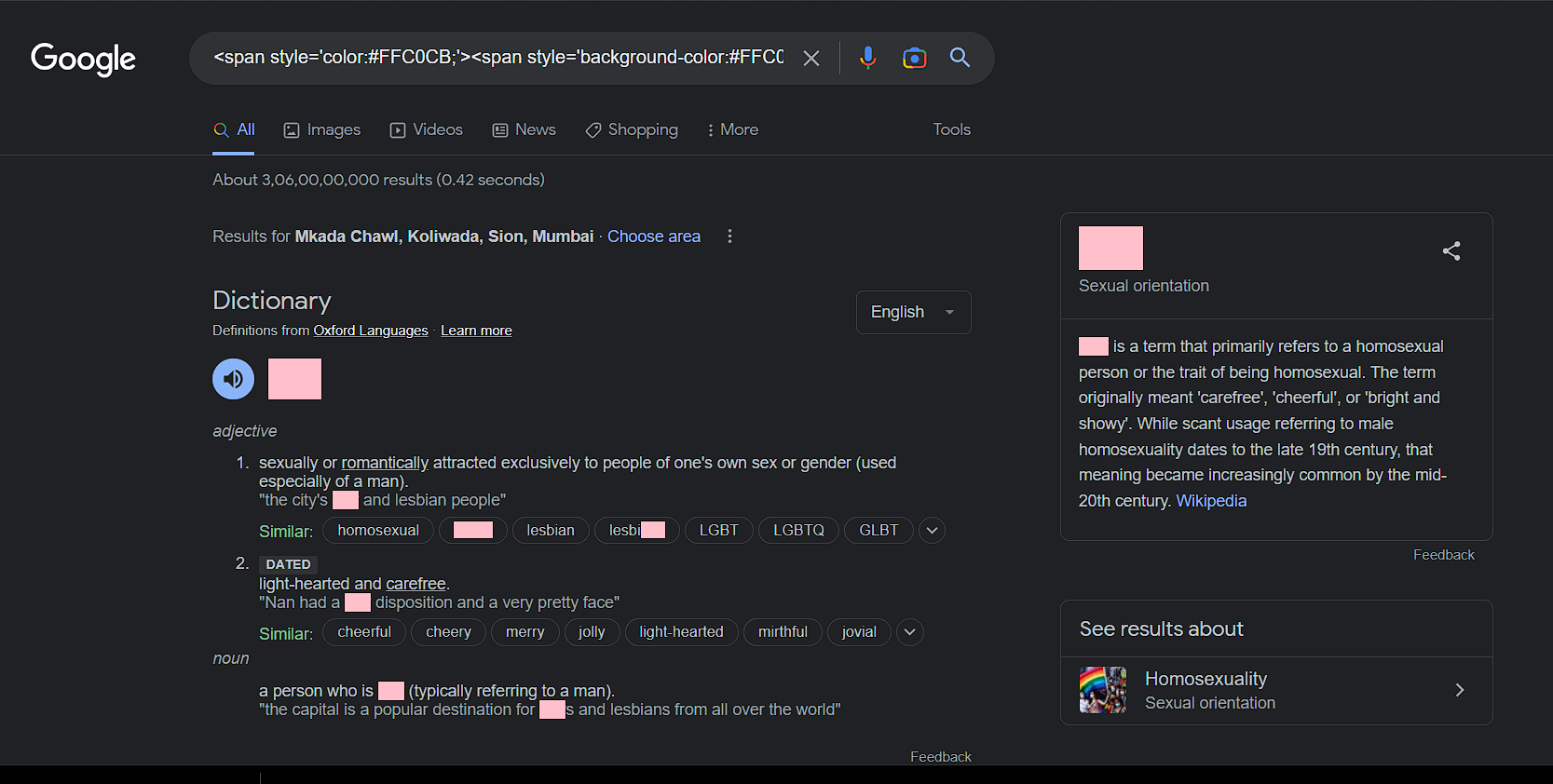
**Fig.7.1 Extension**

**7.2 EXTENSION ON MICROSOFT EDGE**

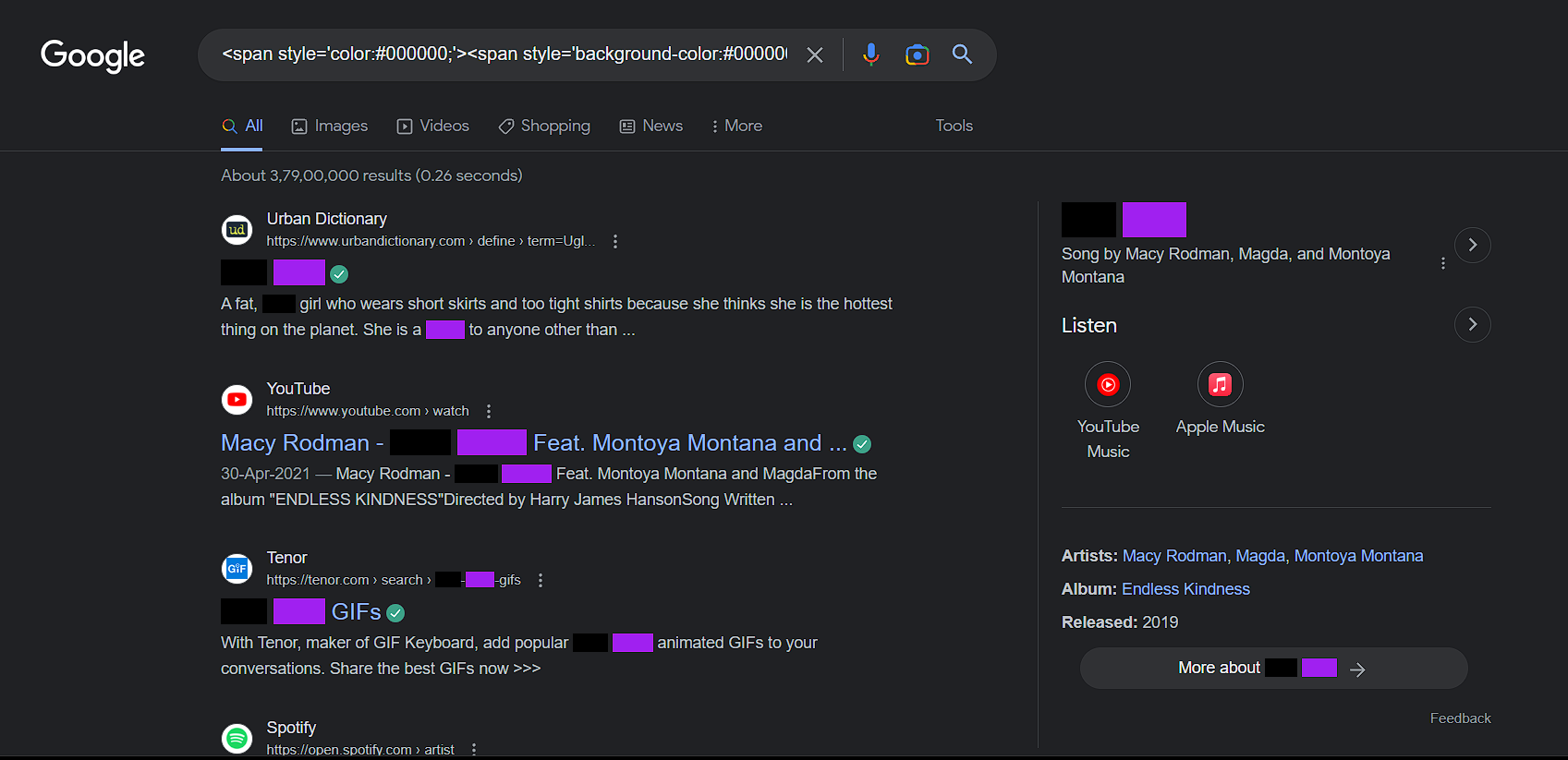
****

**Fig.7.2 Extension**

**7.3 GOOGLE PAGE**

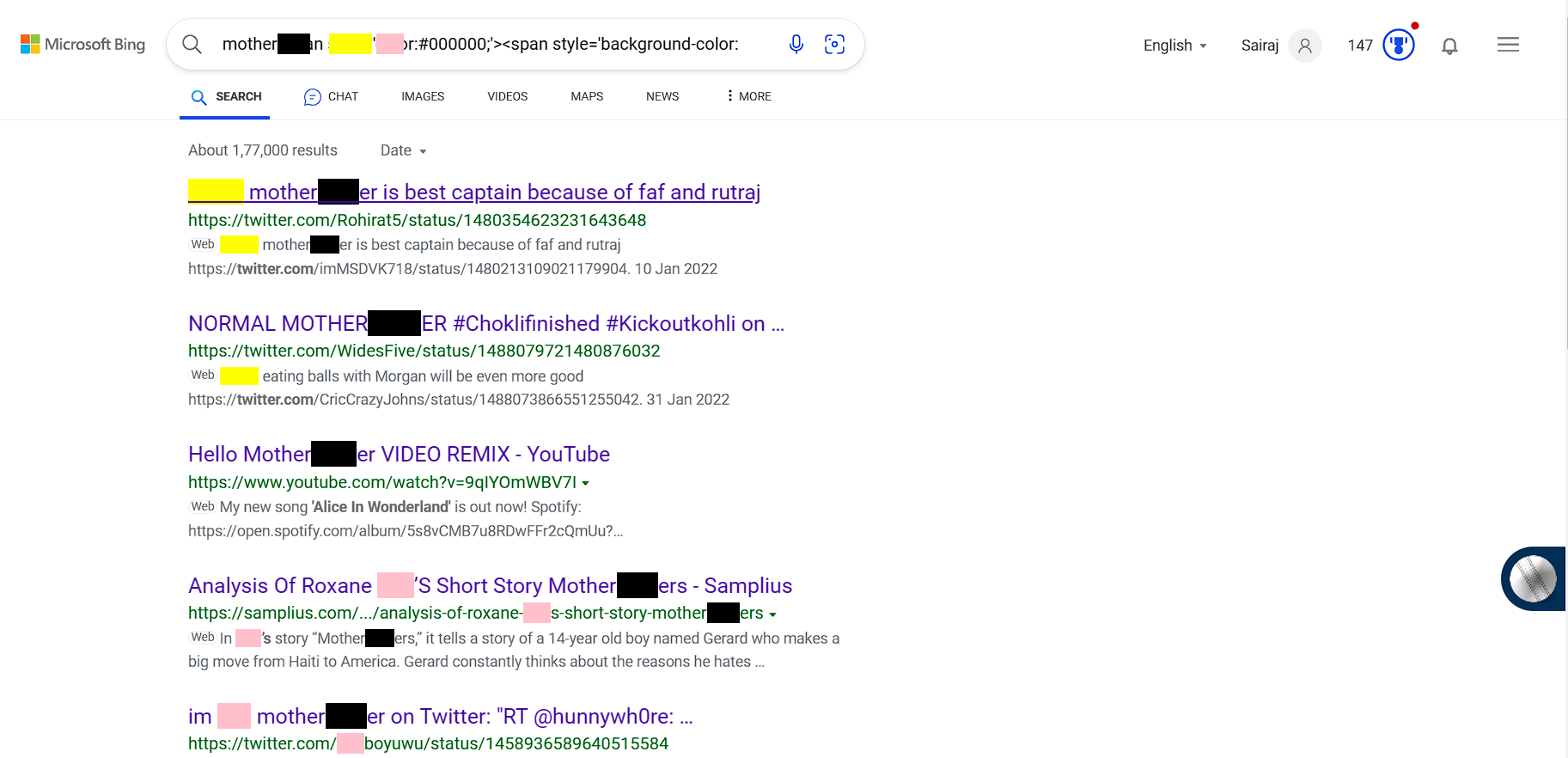
****

**Fig.7.3 Hate Speech Detection on Google**

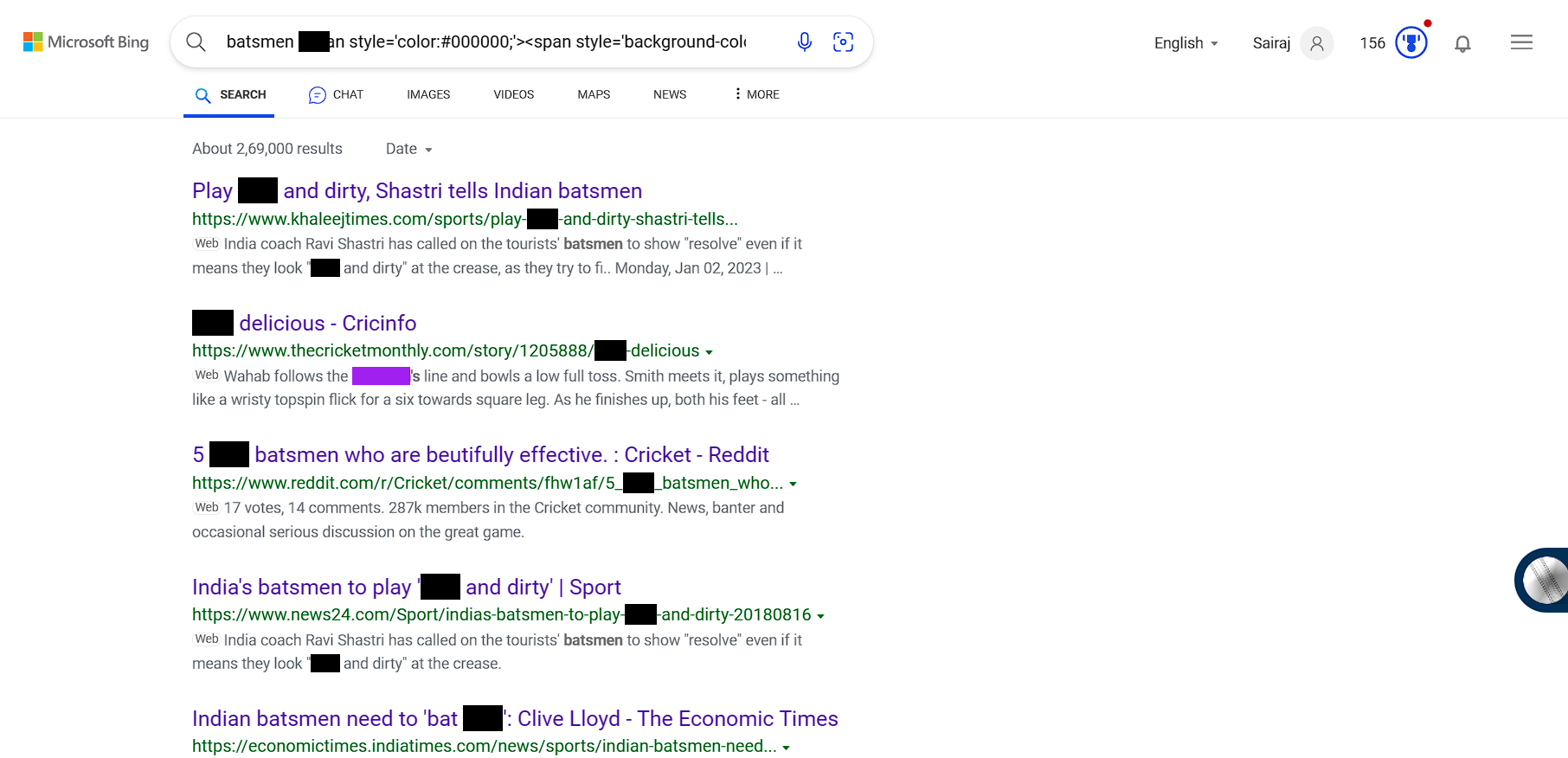
****

**Fig.7.4 Hate Speech Detection on Google**

**7.4 MICROSOFT BING’S PAGE**

****

**Fig.7.5 Hate Speech Detection on Google**

****

**Fig.7.6 Hate Speech Detection on Google**

## Chapter 8 – Conclusion

In conclusion, hate speech detection using a Chrome extension has the potential to be a useful tool in the fight against hate speech online. By leveraging modules and algorithms, the extension can help to identify and flag hate speech in real-time, empowering users to take action and promote a more inclusive online environment.

However, there are also potential drawbacks to consider, such as false positives, limited scope, resource consumption, and privacy concerns. It is important to carefully design and test the extension to minimize these drawbacks and ensure that it is an effective and ethical tool for combating hate speech online.

This project aims at applying AI technologies to identify hate speech and fake news online.

Hence in this project we have discuss the latest techniques for detecting hate speech using our Chrome Extension. This project is expected to contribute to the development of unbiased, fact-based conversations on the internet.

Overall, hate speech detection using a Chrome extension has the potential to make a positive impact on the online community by promoting respect, inclusivity, and safety for all users.

## Chapter 9 – References

1. Smokey: automatic recognition of hostile messages Sánchez-Compaña, M.T.; Sánchez-Cruzado, C.; García-Ruiz, C.R. An interdisciplinary scientific and mathematic education, addressing relevant social problems such as sexist hate speech. Information 2020, 11, 543

1. Hate Speech Blocker:

Mondal, M.; Silva, L.A.; Benevenuto, F. A measurement study of hate speech in social media. In Proceedings of the HT 2017—28th ACM Conference on Hypertext and Social Media, Prague, Czech Republic, 4–7 July 2017; pp. 85–94.

1. Sanoussi, M.S.A.; Xiaohua, C.; Agordzo, G.K.; Guindo, M.L.; al Omari, A.M.M.A.; Issa, B.M. Detection of Hate Speech Texts Using Machine Learning Algorithm. In Proceedings of the 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 26–29 January 2022; pp. 266–273.
2. A Abbasi , H Chen Affect intensity analysis of dark web forums

Ayo, F.E.; Folorunso, O.; Ibharalu, F.T.; Osinuga, I.A. Machine learning techniques for hate speech classification of twitter data: State-of-The-Art, future challenges and research directions.

1. Chetty, N.; Alathur, S. Hate speech review in the context of online social networks. Aggress. Violent Behav. 2018, 40, 108–118.
2. Matamoros-Fernández, A.; Farkas, J. Racism, Hate Speech, and Social Media: A Systematic Review and Critique. Telev. New Media 2021, 2, 205–224.
3. Zampieri, M.; Malmasi, S.; Nakov, P.; Rosenthal, S.; Farra, N.; Kumar, R. SemEval-2019 Task 6: Identifying and Categorizing Offensive Language in Social Media (OffensEval). In Proceedings of the 13th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6–7 July 2019; pp. 75–86.
4. Wikipedia Toxic Comments Dataset: <https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data>
5. "Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network" by K. Badjatiya, S. Gupta, et al. (2017[): https://arxiv.org/abs/1708.00023](file:///C:\Users\saira\AppData\Local\Microsoft\Windows\INetCache\IE\4OMX3YC2\):%20https:\arxiv.org\abs\1708.00023)
6. "Automated Hate Speech Detection and the Problem of Offensive Language" by T. Davidson, D. Warmsley, et al. (2017): <https://aaai.org/ocs/index.php/ICWSM/ICWSM17/paper/view/15665>
7. HateSonar: A Python library for hate speech detection: <https://github.com/HateSonar/HateSonar>
8. Google Perspective API: Offers hate speech detection and toxicity analysis: <https://www.perspectiveapi.com/>
9. "A Comprehensive Guide to Detecting Hate Speech using Natural Language Processing" by Analytics Vidhya: <https://www.analyticsvidhya.com/blog/2022/01/a-comprehensive-guide-to-detecting-hate-speech-using-natural-language-processing/>
10. "Detecting Hate Speech with Machine Learning" by Towards Data Science: <https://towardsdatascience.com/detecting-hate-speech-with-machine-learning-950b178b3e8>
11. Facebook AI's Datasets and Models for Detecting Hate Speech: <https://github.com/facebookresearch/detectron2>