International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 13, Issue 3, 2025

Ai Based Safe Surfing System

G Sudhakar Raju, M Siri Bhavishya, Aelijerla Sneha, Linga Srilaxmi

[1]Associate Professor, Department of Information Technology, Bhoj Reddy Engineering College for Women

[2.3.4]B,tech students, Department of Information Technology, Bhoj Reddy Engineering College for Women

siribhavishya24@gmail.com

ABSTRACT

As internet usage continues to grow rapidly among children and teenagers, safeguarding their online experience has become increasingly important. The open nature of search engines exposes young users to inappropriate and potentially harmful content that may include explicit language, violent media, or misleading information. This project titled "Safe Search for Kids" addresses this problem by creating a child-friendly web search system that filters and moderates content based on predefined rules and advanced AI moderation techniques.

Our system is built using a layered filtering architecture that starts with a static blocklist check, progresses to pattern recognition for disguised terms, and concludes with a context-aware evaluation powered by the OpenAI API for ambiguous or unknown queries. This multi-layered approach significantly enhances the accuracy and reliability of the moderation system.

By classifying queries into "Safe," "Blocked," and "Under Construction," the system provides a user-friendly experience while maintaining safety. The project combines web development, content moderation techniques, and machine learning APIs to present a practical solution to an important real-world issue.

With potential for integration in homes, schools, and libraries, the system can serve as a first step towards building a safer digital environment for the next generation of internet users.

1. INTRODUCTION

The internet has become an indispensable tool for education, communication, and entertainment. With the rapid growth of digital content and accessibility, children and teenagers are increasingly exposed to a vast repository of information. While this opens up numerous learning opportunities, it also poses a significant risk due to the unfiltered nature of much of the content available online[1]. Unmonitored internet usage can result in children accessing websites or search results that contain explicit, violent, or otherwise inappropriate material[3]. Recognizing this growing issue, our project aims to create a dedicated safe-search platform for children. The system is designed to prevent harmful search results by filtering queries before they are processed by a search engine or public API[4]. Using a hybrid approach that combines basic keyword filtering, pattern recognition, and advanced content evaluation via an AI API, the system ensures a more comprehensive and intelligent content moderation process[2].Our project addresses several concerns that parents, educators, and policy makers have raised. It not only improves the safety of children's digital experiences but also educates them on responsible online behavior by highlighting blocked and unknown queries appropriately. This dual approach makes our system both a technological safeguard and a pedagogical tool.

Existing System:

Current search engines such as Google, Bing, and Yahoo offer options like "SafeSearch" that attempt to restrict adult content or inappropriate material. However, these systems rely heavily on metadata, manual tagging, and surface-level keyword analysis, which can easily be bypassed[3]. Children can disable SafeSearch, or use misspelled, disguised, or contextually ambiguous queries that slip through simple filters. The existing systems do not provide real-time, deep content analysis of queries, nor do they offer user-specific moderation based on age or educational setting. Moreover, these engines are not specialized for child-specific needs. They fail to consider language ambiguity, cultural sensitivity, or evolving slangs that can bypass traditional filters. They lack explainable moderation that informs children or parents why a particular search is blocked or marked. With rapid evolution of slang and digital content, static filtering becomes ineffective over time.There is also minimal integration of artificial intelligence in traditional search safety systems. They lack contextual understanding and the ability to adapt and learn from new threats or inappropriate trends. This results in either over-blocking harmless content or under-blocking harmful content, leading to frustration or risk[5].

In conclusion, while traditional SafeSearch systems offer a basic level of filtering, they are neither comprehensive nor intelligent enough to ensure consistent safety, especially for younger users who may unintentionally or unknowingly access harmful material.

Proposed System:

The proposed system, "Safe Surfing System ," aims to deliver a secure, educational, and responsive online search experience tailored for children. The architecture combines conventional filtering techniques with artificial intelligence to dynamically classify search queries into safe, blocked, or under-review categories[2][5]. The system is designed to be easily deployable in schools, homes, and child-focused institutions.

The core idea is to intercept and analyze search queries before they are passed to any search engine or API. The system begins by matching the query against a blocklist of inappropriate terms and common disguises (e.g., altered spellings like “se*y” or “po*n”). If the term is caught here, the system immediately blocks it. If the query is unclear or not matched directly, it proceeds to an AI-based moderation layer using the OpenAI API, which analyzes the intent and semantics of the input.

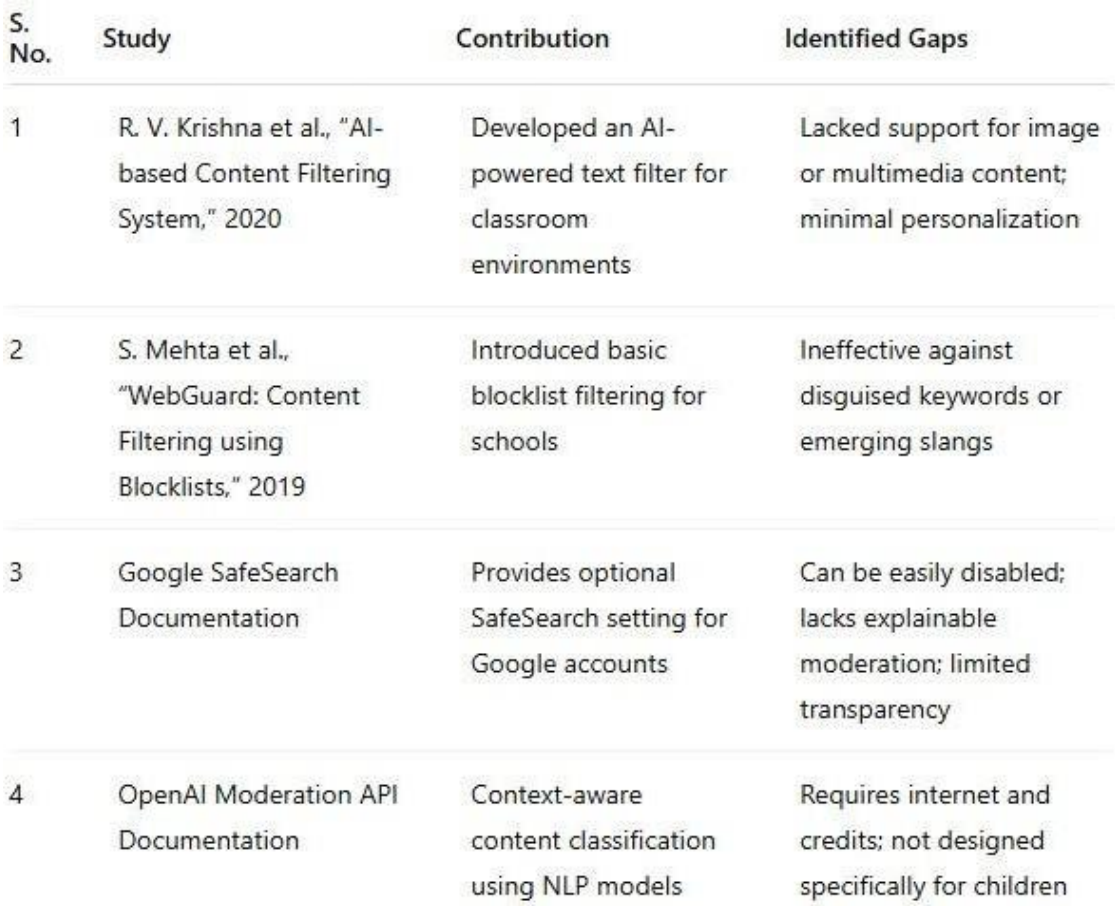
This tiered filtering system offers the advantages of high accuracy, contextual understanding, and extensibility. A lightweight and intuitive frontend ensures usability for children, while the backend manages real-time filtering and query classification efficiently. The architecture supports easy future upgrades such as voice search moderation, multilingual analysis, and image or video content filtering. Thus, the proposed system addresses limitations in current safe search mechanisms and represents an intelligent, user-centric approach to protecting children online.

2. RELATED WORK

The pathway taken into account to develop the Review Model for text-filtering modeling is conduct the organized review of existing techniques for safe search filtering, artificial intelligence-based moderation, and child safety online[1][3]. The purpose of this review is to identify research gaps in current technologies and propose key areas of improvement.

Research questions are the foundation upon which a systematic study is built. They help define the scope of the project, direct the investigation, and ensure that efforts are aligned with relevant and practical outcomes. In this context, our study was guided by the following key questions:

1. What are the existing techniques, tools, and algorithms for content filtering and safe search systems?
2. Which are the key areas of research in the field of AI-driven safe search systems for children?
3. What are the existing gaps in the current literature and practical implementations in search filtering ?



S. No.	Study	Contribution	Identified Gaps
1	R. V. Krishna et al., “AI-based Content Filtering System,” 2020	Developed an AI-powered text filter for classroom environments	Lacked support for image or multimedia content; minimal personalization
2	S. Mehta et al., “WebGuard: Content Filtering using Blocklists,” 2019	Introduced basic blocklist filtering for schools	Ineffective against disguised keywords or emerging slangs
3	Google SafeSearch Documentation	Provides optional SafeSearch setting for Google accounts	Can be easily disabled; lacks explainable moderation; limited transparency
4	OpenAI Moderation API Documentation	Context-aware content classification using NLP models	Requires internet and credits; not designed specifically for children

3. REQUIREMENT ANALYSIS

3.1 *Functional Requirements:*

Functional requirement should include function performed by a specific screen outline work-flows performed by the system and other business or compliance requirement the system must meet. Functional requirements specify which output file should be produced from the given file they describe the relationship between the input and output of the system, for each functional requirement a detailed description of all data inputs and their source and the range of valid inputs must be specified.

The functional specification describes what the system must do, how the system does it is described in the design specification. If a user requirement specification was written, all requirements outlined in the user requirements specifications should be addressed in the functional requirements.

1. Search Interface: Provide a query input box designed for children along with a search button.
2. Content Filtering Logic: The backend must analyze submitted queries using blocklists, regex filters, and AI moderation[2].
3. Result Rendering: If a query is classified as safe, the system should render search results including titles, images, and source links from a preloaded safe dataset[4].
4. Alert System: Show a message banner or popup when a query is blocked or under review[1].
5. Event Listeners: Implement logic to respond to both button clicks and Enter key submissions.
6. Fallback Handling: When a query falls under "under construction," the system should auto-reload after a timeout or allow user to retry.
7. Content API Support: Fetch educational results dynamically from a trusted source or internal safe website[4].

3.2 *Non-Functional Requirements:*

Describe user-visible aspects of the system that are not directly related with the functional behavior of the system. Non-Functional requirements include quantitative constraints, such as response time (i.e., how fast the system reacts to user commands.) or accuracy (e. how precise are the systems numerical answer). Non-Functional requirements describe the user- visibility aspects of the system that are not directly related with the functional behavior of the system. There some non-Functional Requirements that define our project they are:

- Portability:
  The ability of the system to run consistently on modern browsers across desktops, tablets, and mobile phones without modification.

- Reliability:
  The ability of the system to handle errors gracefully and maintain consistent behavior under various load conditions.

- Usability:
  The ability of the interface to be intuitive, accessible, and age-appropriate for children aged 8–14, ensuring ease of use and clarity.

3.2.1 Software Requirements:

- Front-End : HTML5, CSS3,Javascript
- Web Framework : Node.js, Express
- IDE : Visual Studio Code
- Operating System : Windows 11
- Application Programming Interface : OpenAI Moderation API

3.2.2 Hardware Requirements:

- Processor : I5
- RAM : 8GB
- Hard Disk :25GB

4. Design

4.1 *Architecture:*

4.1.1 System Architecture:

It describes the structure and behavior of technology infrastructure of an enterprise, solution or system. In other words, System architecture can be described as the flow of application which is represented below in the pictorial form. He purpose of system architecture activities is to define a comprehensive solution based on principles, concepts, and properties logically related to and consistent with each other. The solution architecture has features, properties, and characteristics which satisfy, as far as possible, the problem or opportunity expressed by a set of system requirements (traceable to mission/business and stake holders requirements) .

System architecture is abstract, conceptualization-oriented, global, and focused to achieve the mission and life cycle concepts of the system. It also focuses on high- level structure in systems and system elements. It addresses the architectural principles, concepts, properties, and characteristics of the system-of-interest. It may also applied to more than one system, in some cases forming the common structure, pattern, and set of requirements for classes or families of similar or related systems.

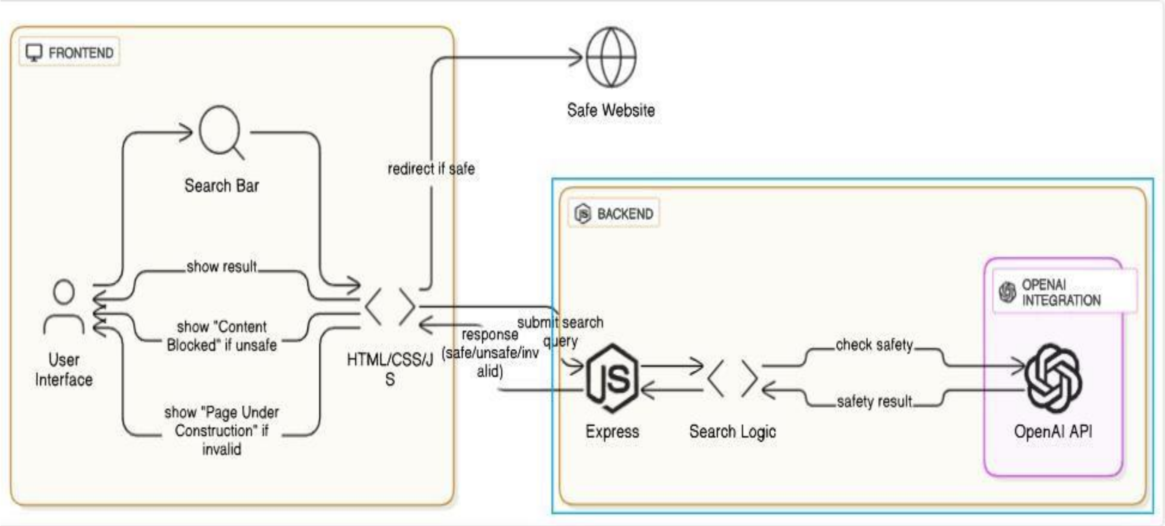


Fig. 4.1.1 System Architecture

4.1.2 Technical Architecture:

Technical Architecture refers to the structural process of designing and building system's architecture with focus on the users and sponsors view of the environment. Technology architecture associates application components from application architecture with technology components representing software and hardware components. Its components are generally acquired in the market place and can be assembled and configured to constitute the enterprise's technological infrastructure. A technical architecture diagram provides a bird's eye view of the infrastructure of our project. The diagram illustrates how components in a system interact with one another in the large scale of things. Technical Architecture (TA) is a form of IT architecture that is used to design computer systems. It involves the development of a technical blueprint with regard to the arrangement, interaction, and interdependence of all elements so that system-relevant requirements are met.

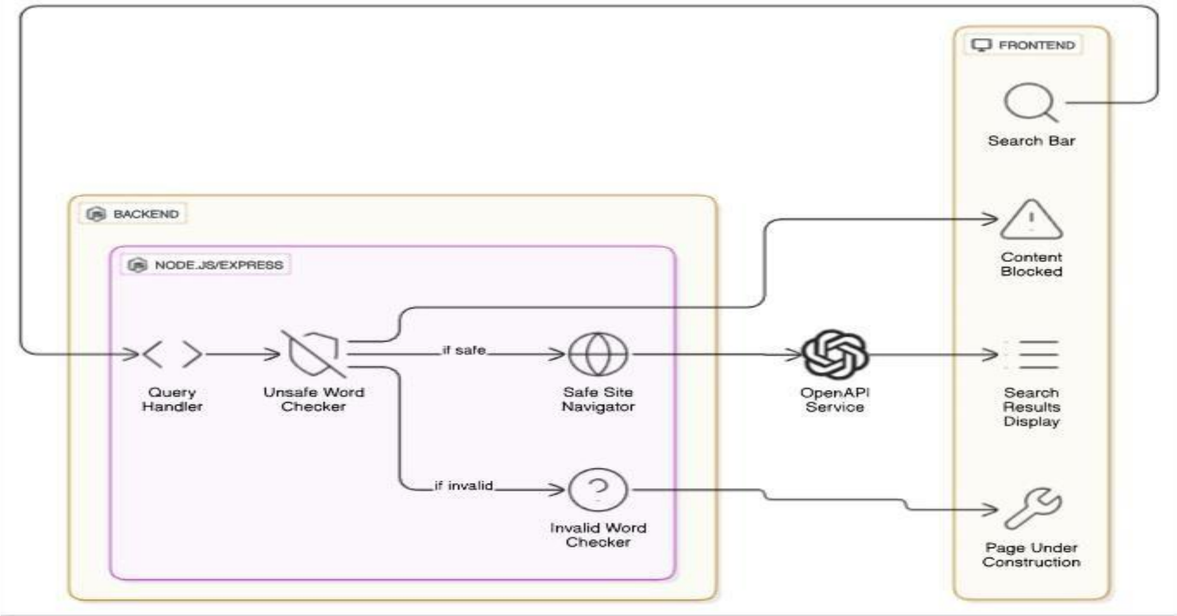


Fig. 4.1.2 Technical Architecture

5. IMPLEMENTATION

5.1 *Technologies*

The development of the "Safe Surfing System " platform involves a combination of front-end and back-end technologies along with artificial intelligence APIs. Below is a detailed description of the technologies used:

1. **HTML5**

HTML5 is used for structuring the content on the web pages. It ensures compatibility across devices and provides semantic elements that enhance accessibility and maintainability of the code. HTML5 forms the backbone of the user interface, supporting the layout of the search box, result display, and alerts.

2. **CSS3**

CSS3 is utilized to style the HTML components. It enhances the appearance of the UI elements, including font, color schemes, spacing, and layout alignment. CSS ensures the frontend is visually appealing and suitable for children through bright, clear, and simple designs.

3. **JavaScript (Vanilla JS)**

JavaScript handles the dynamic behavior of the application on the client side. It is used for validating input queries, capturing button clicks, triggering events on keypresses, and handling auto-reload features for "Under Construction" messages. It ensures an interactive user experience.

4. **Bootstrap 5**

Bootstrap is a responsive front-end framework that provides ready-to-use CSS and JavaScript components. It is used to quickly prototype the UI with consistent styling and mobile responsiveness. Components like buttons, cards, and grids simplify layout creation.

5. **Node.js**

Node.js is a server-side JavaScript runtime used to build the backend of the application. It allows for asynchronous handling of requests, integration of APIs, and routing logic. Node.js forms the core of the filtering engine and response management.

6. **Express.js**

Express is a minimalist web application framework for Node.js. It simplifies server- side routing, middleware management, and API creation. In the project, it is used to handle incoming search queries, integrate blocklist checks, and process AI moderation logic.

7. **OpenAI API**

The OpenAI Moderation API is used to evaluate the context and intent of ambiguous queries. It classifies them into safe, blocked, or unknown categories. This adds a layer of intelligent, context-aware filtering beyond static keyword detection.

8. **dotenv**

This module is used to manage environment variables securely. It allows developers to store API keys and configuration values in a separate .env file, keeping sensitive information out of the codebase and enhancing security.

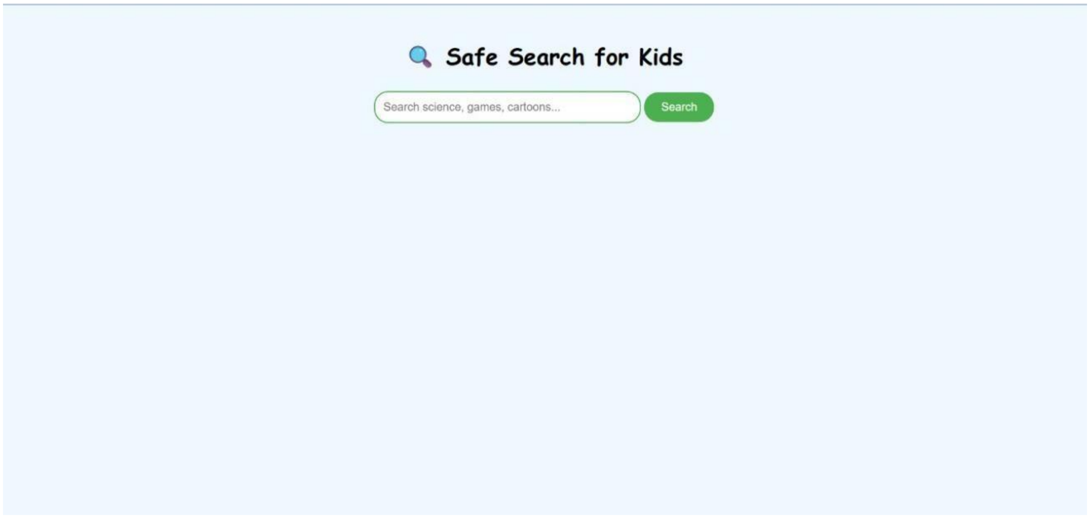
9. **CORS (Cross-Origin Resource Sharing)**

CORS middleware is used to allow frontend and backend running on different origins (e.g., localhost ports) to communicate without access issues. It is essential for seamless interaction between client-side and server-side components.

10. **nodemon**

Nodemon is a utility tool that automatically restarts the Node.js server whenever file changes are detected. It improves developer productivity by eliminating the need for manual server reboots during testing.

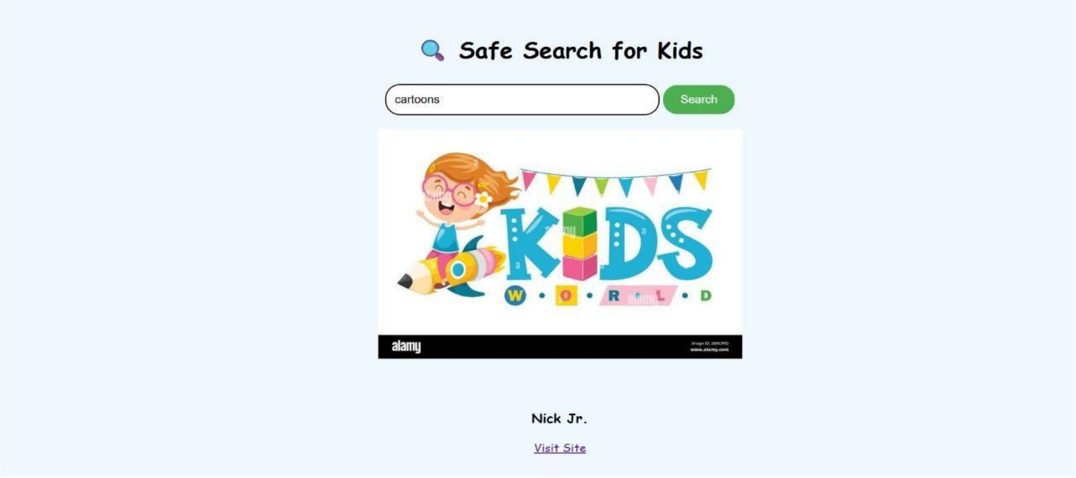
6-SCREENSHOTS



Safe Search for Kids

Search science, games, cartoons...

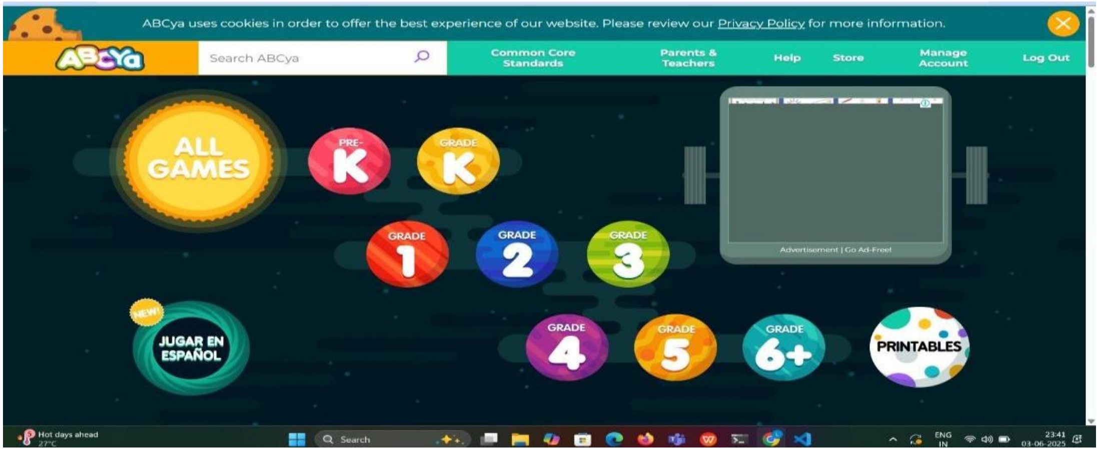
Screenshot 7.1: Safe Search Portal Home



Safe Search for Kids

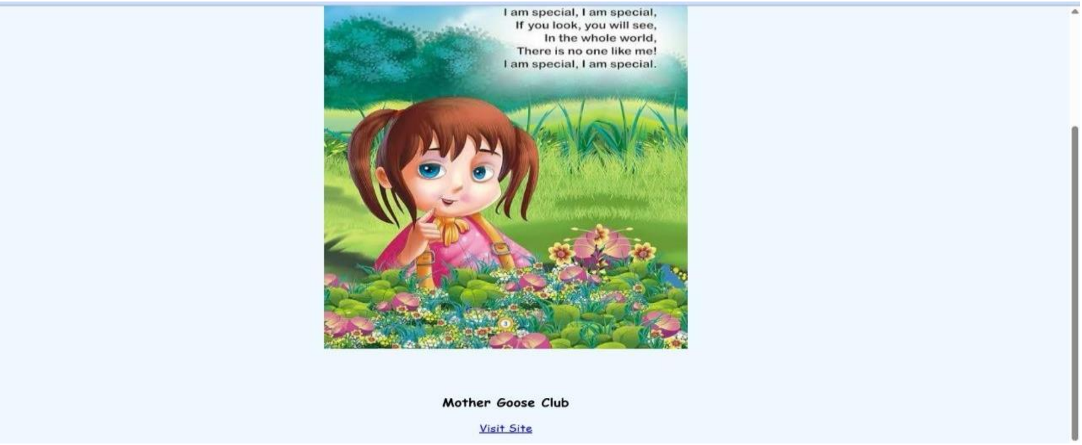
cartoons

Screenshot 7.2: Safe Search Home with Safe content searched



ABCya uses cookies in order to offer the best experience of our website. Please review our Privacy Policy for more information.

Screenshot 7.3: Navigation to Safe Website as Search result

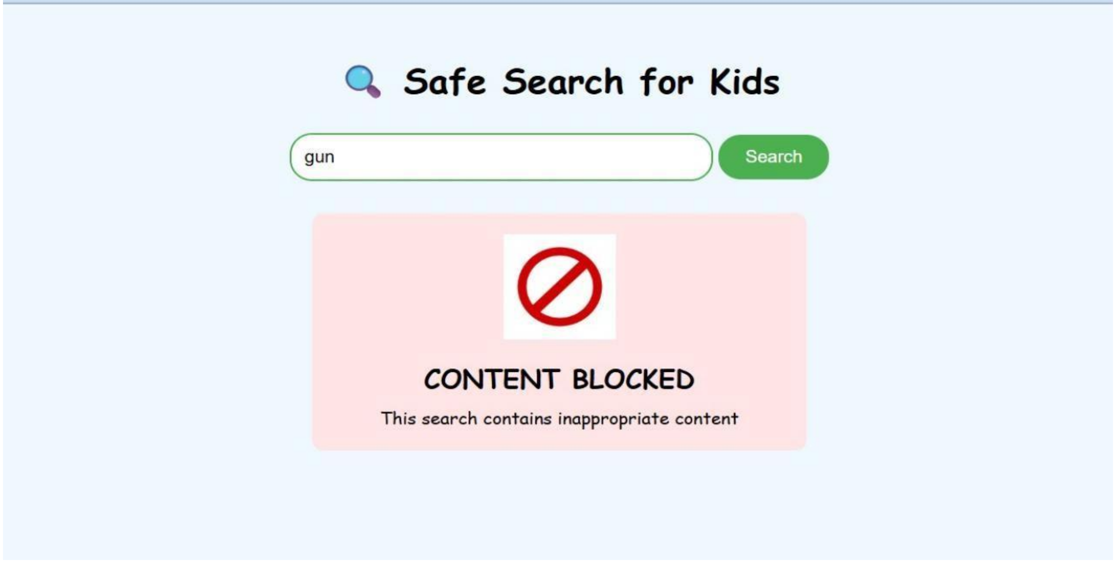


I am special, I am special,
If you look, you will see,
In the whole world,
There is no one like me!
I am special, I am special.

Mother Goose Club

Visit Site

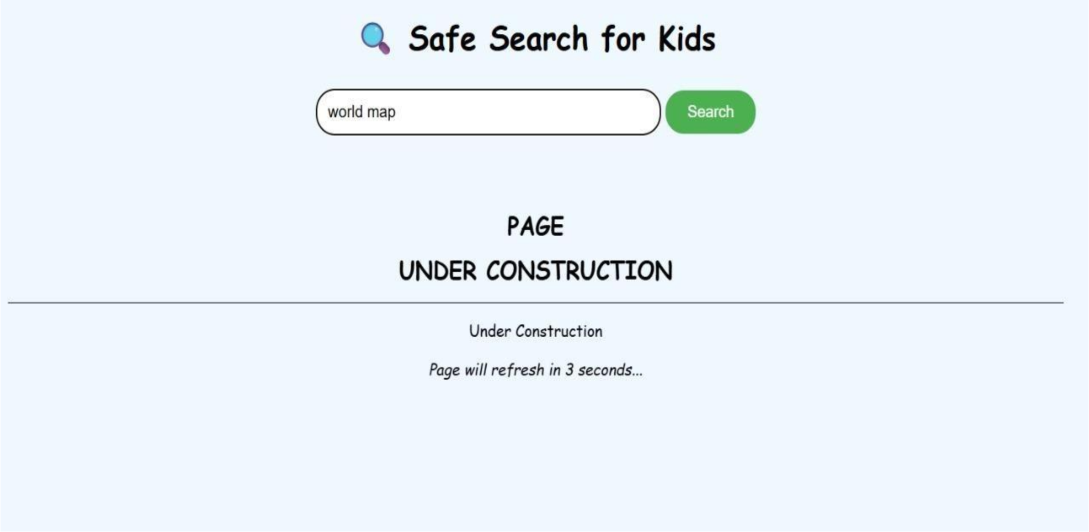
Screenshot 7.4: Rhymes Search



CONTENT BLOCKED

This search contains inappropriate content

Screenshot 7.5: Blocked Unsafe Content for kids



PAGE
UNDER CONSTRUCTION

Under Construction

Page will refresh in 3 seconds...

Screenshot 7.6:Under Construction page with auto-refresh for unknown results

7-CONCLUSION

This project is designed to provide children with a secure and educational online browsing experience by enabling safe, filtered access to web content. It employs a “hybrid content moderation approach” that combines a manual blocklist for known harmful queries, a whitelist mapping system that redirects safe searches to approved educational websites, and AI-based moderation using external services to analyze and filter inappropriate images, videos, or text. Instead of showing standard search engine results, the system prioritizes and displays child-friendly educational resources. The platform features an intuitive, colorful interface that engages children while ensuring clarity and safety through friendly alerts and fallback pages. Overall, the system ensures protection from unsafe or explicit internet content while promoting safe digital exploration for young users.

REFERENCES

[1] A. Jevremovic, M. Đuro Veinović, M. Cabarkapa, M. Krstić, et al., "Keeping Children Safe Online With Limited Resources: Analyzing What is Seen and Heard," *IEEE Access*, vol. 9, pp. 128822-128839, 2021, doi: 10.1109/ACCESS.2021.3116644.

[2] J. Liu, H. Li, X. Zhao, and Y. Sun, “A keyword detection and context filtering method for document level relation extraction,” *Applied Sciences*, vol. 12, no. 3, p. 1599, Mar. 2022.

[3] Y. Chen, S. Zhu, Y. Zhou, and H. Xu, "Detecting offensive language in social media to protect adolescent online safety," in *Proc. 2012 Int. Conf. Social Computing (SocialCom)*, Amsterdam, Netherlands, Sep. 2012, pp. 71–80.

[4] R. Du, R. Safavi-Naini, and W. Susilo, "Web filtering using text classification," *Faculty of Informatics - Papers*, Univ. of Wollongong, Wollongong, NSW, Australia, Standard, 2003, p. 166.

[5] D. Patel, V. Khan, R. K. Shukla, and M. Kherajani, “A customized children friendly and secure search engine,” in *Proc. 2nd Int. Conf. Data, Eng. and Commun. Technol. (ICDECT)*, **IEEE**, Feb. 2020. [Online]. Available: https://www.researchgate.net/publication/343751394_A_Customized_Children_Friendly_and_Secure_Search_Engine. [Accessed: Jun. 7, 2025].

[6] J. Žufić, T. Žajgar, and S. Prkić, “Children online safety,” in *Proc. 40th Int. Conv. Inf. Commun. Technol., Electron. Microelectron. (MIPRO)*, Opatija, Croatia, May 2017, doi: 10.23919/MIPRO.2017.7973563.

[7] National Geographic Kids.[Online].Available: https://kids.nationalgeographic.com.

[8] "Unified Modeling Language (UML) Diagrams," *GeeksforGeeks*, [Online]. Available:https://www.geeksforgeeks.org/unified-modeling-language-uml- introduction/.

[9] DeepSeek. [Online]. Available: https://www.deepseek.com