International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 02, NCRTCS - 2026

LeakScan: A Web-Based Tool for Detecting Sensitive Information in Files

Fazal Pathan
Master's Student Department of Computer
Science, Abeda Inamdar Senior College, Pune, India

Shakila Siddavatam
Head of Department of Computer Science
Abeda Inamdar Senior College, Pune, India

Abstract - The use of digital documents in academic, personal, and organizational environments had increased significantly over time. As a result, many text-based files were shared without carefully examining their contents. These files often contained sensitive personal information such as email addresses, phone numbers, identity numbers, and passwords. When such information was exposed unintentionally, it created privacy concerns, possibilities of data misuse, and security-related risks. This study focused on the problem of unintended sensitive data leakage in text-based documents and highlighted the importance of detecting such information before document sharing. In this research, a web-based system named LeakScan was designed and developed to identify sensitive information present in text and comma-separated value files. The study followed a design and implementation-based research approach. Pattern-based detection using regular expressions was applied to identify clearly formatted sensitive data, while similarity-based analysis was used to detect partially hidden or modified information. The system also evaluated password strength, assigned risk levels, and generated structured reports to support user understanding. The results showed that the hybrid detection approach effectively identified multiple categories of sensitive information. The system detected both clearly formatted and obfuscated data while maintaining low processing time suitable for real-time usage. Risk level classification helped users understand the severity of detected information and assess potential exposure before sharing documents. The study concluded that LeakScan provided a practical and reliable solution for detecting sensitive information in text-based files. The findings emphasized the role of hybrid detection techniques in strengthening privacy protection practices. Future work recommended extending support to additional document formats, improving detection accuracy, and enhancing monitoring features to further reduce the risk of data leakage.

Keywords
Sensitive data detection, data leakage prevention, cybersecurity and privacy, web-based security system, information protection

INTRODUCTION

The rapid advancement of digital technology has resulted in extensive use of electronic documents for storing, processing, and sharing information. Text-based files such as reports, logs, configuration files, and comma-separated value files are commonly used in academic institutions, organizations, and personal environments. These files are easy to create and share, which often leads users to overlook the sensitive information embedded within them.

Sensitive information such as email addresses, phone numbers, identity numbers, and passwords frequently remain hidden in text-based documents. When such files are shared through email, cloud platforms, or external storage devices without careful review, they pose serious risks related to privacy breaches and unauthorized data access. Accidental disclosure of sensitive data has therefore become a growing concern in the field of cybersecurity and privacy.

In many real-world scenarios, users rely on manual inspection to identify sensitive data before sharing documents. However, manual checking is time-consuming, error-prone, and impractical when dealing with large files or multiple documents. Moreover, sensitive information may appear in partially modified or disguised forms, making it difficult to detect through simple visual inspection or keyword searches.

Existing security mechanisms mainly focus on system-level protection such as authentication and access control. While these mechanisms are important, they do not adequately address the problem of sensitive data exposure caused by careless document sharing. As a result, there is a need for automated solutions that can analyze document content and detect sensitive information before data is shared.

This paper presents LeakScan, a web-based system designed to detect sensitive information in text-based files.

PROBLEM STATEMENT

The increasing dependence on digital documents for information storage and exchange has significantly raised the risk of unintentional exposure of sensitive data. Text-based files, which are widely used due to their simplicity and compatibility, often contain confidential information that may not be immediately noticeable. Users frequently share these documents without thoroughly examining their contents, leading to potential privacy violations and security threats.

Sensitive information such as personal identifiers and passwords may remain embedded within text files in both structured and unstructured forms. Detecting such

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 02, NCRTCS - 2026

information manually is challenging, especially when dealing with large volumes of data or partially modified content. Manual inspection methods are not only inefficient but also highly susceptible to human error, which increases the likelihood of sensitive data leakage.

Traditional security mechanisms primarily focus on restricting unauthorized access through authentication and permission controls. While these mechanisms protect system boundaries, they do not adequately address the risks associated with document-level data exposure. Consequently, sensitive information can still be leaked when legitimate users unknowingly share files containing confidential data.

Although data leakage prevention solutions exist, many available tools are complex, resource-intensive, or designed for large enterprise environments. Such solutions are often unsuitable for academic users, students, or small organizations that require lightweight, accessible, and easy-to-use systems.

Therefore, there is a clear need for a practical and user-friendly system capable of automatically identifying sensitive information within text-based documents before they are shared. Addressing this challenge forms the basis of this research work.

OBJECTIVES OF THE STUDY

The primary objective of this research is to design and develop a practical system that can effectively detect sensitive information present in text-based documents and assist in reducing the risk of accidental data leakage. The study focuses on improving document-level security by enabling automated analysis of file contents.

The specific objectives of the study are as follows:

- To identify sensitive information such as email addresses, phone numbers, identity numbers, and passwords in text-based files.
- To develop an automated detection mechanism capable of recognizing both structured and partially modified sensitive data.
- To implement a lightweight and accessible web-based system suitable for academic and small organizational environments.
- To classify detected sensitive information based on risk levels for improved user understanding.
- To generate clear and structured reports that summarize detection results and potential privacy risks.
- To enhance user awareness regarding sensitive data exposure before document sharing.

These objectives collectively aim to provide a simple yet effective solution that supports privacy protection and cybersecurity practices.

LITERATURE REVIEW

The problem of sensitive information exposure and data leakage has attracted significant attention in the field of cybersecurity and privacy. With the growing use of digital documents, researchers have explored various techniques for detecting and preventing unauthorized disclosure of confidential information. Early approaches primarily relied on rule-based mechanisms such as keyword matching and pattern recognition to identify sensitive data within textual content. These techniques proved effective for detecting clearly structured information but were limited when dealing with modified or obfuscated data.

Several studies have emphasized the importance of content-based inspection methods for preventing data leakage. Such methods analyze document contents rather than focusing solely on system-level security controls. While content inspection improves detection capabilities, many existing solutions are designed for enterprise environments and require complex configurations and high computational resources. This restricts their adoption in academic and small-scale usage scenarios.

Researchers have also investigated hybrid detection strategies that combine pattern-based rules with similarity or heuristic-based analysis. These methods improve detection accuracy by identifying sensitive information that does not strictly follow predefined formats. Despite these advancements, many proposed systems lack usability and accessibility, making them difficult for non-technical users to operate effectively.

Privacy protection standards and guidelines have further highlighted the risks associated with improper handling of personally identifiable information. Frameworks developed by recognized organizations stress the necessity of identifying sensitive data before information exchange. However, practical implementations of lightweight tools for document-level analysis remain limited.

Based on the observations from existing literature, there is a clear need for a simple, efficient, and user-friendly system capable of detecting sensitive information in text-based documents. This research addresses the identified gap by presenting LeakScan, a web-based system designed to support sensitive data detection and privacy protection.

METHODOLOGY

This research follows a design and development-oriented methodology to build a practical solution for detecting sensitive information in text-based files. The methodology focuses on creating a functional web-based system that can analyze document contents, identify sensitive data, and present meaningful results to users. The overall approach prioritizes simplicity, efficiency, and usability within the domain of cybersecurity and privacy.

Research Design

The study adopted a system development-based research design in which the proposed solution was designed, implemented, and evaluated through practical experimentation. The research process involved identifying the problem of sensitive data leakage, designing a detection workflow, implementing the system using appropriate technologies, and analyzing the system's effectiveness through testing. This approach ensured that the developed system addressed real-world data exposure scenarios rather than theoretical assumptions.

Sensitive Data Detection Approach

The detection of sensitive information was achieved using a hybrid approach combining pattern-based detection and similarity-based analysis. Pattern-based detection utilized predefined rules to identify commonly structured sensitive data such as email addresses, phone numbers, identity

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 02, NCRTCS - 2026

numbers, and passwords. To improve detection coverage, similarity-based logic was applied to recognize partially modified or obfuscated sensitive information that might not strictly match predefined patterns. Additionally, password strength evaluation was incorporated to identify weak credentials that could pose security risks.

System Workflow

The workflow of the system begins when the user uploads a text-based file through the web interface. The uploaded content undergoes preprocessing to remove unnecessary characters and prepare the text for analysis. The detection module then scans the processed text using the hybrid detection approach to identify sensitive information. Based on the detected data, risk levels are assigned to indicate the severity of potential exposure. Finally, the system generates structured reports and displays the results to the user in an understandable format.

SYSTEM ARCHITECTURE

The LeakScan system follows a layered architecture designed to ensure clarity, modularity, and efficient processing of sensitive information detection tasks. The architecture separates user interaction, backend processing, detection logic, and data storage into distinct components. This structured design improves system maintainability and enables smooth data flow across different stages of operation.

The Frontend Layer serves as the user interaction interface of the system. It provides functionalities such as user login, navigation dashboard, file upload interface, and result display page. Through this layer, users can upload text-based documents, initiate scanning, and review the detected sensitive information along with associated risk levels. The frontend communicates with the backend by sending user requests and receiving processed results.

The Backend Layer, implemented using the Django framework, is responsible for handling application logic and request management. This layer processes incoming requests, manages authentication, handles file uploads, and performs initial text preprocessing. Once the uploaded content is prepared for analysis, the backend forwards the processed text to the sensitive data detection module and later formats the received results for presentation.

The Sensitive Data Detection Engine constitutes the core analytical component of the system. It applies a hybrid detection approach combining pattern-based detection with similarity-based logic. Pattern matching techniques identify structured sensitive information, while similarity analysis enhances detection by recognizing modified or partially hidden data. The engine also evaluates password strength and assigns risk levels based on detection outcomes.

The Database Layer maintains persistent storage for system-related information. SQLite is used to store user details, file metadata, detected sensitive data, risk classifications, and scan history. This stored data supports report generation and allows users to review previous scan results when required.

Overall, the layered architecture of LeakScan ensures systematic processing, efficient detection of sensitive information, and reliable management of application data.

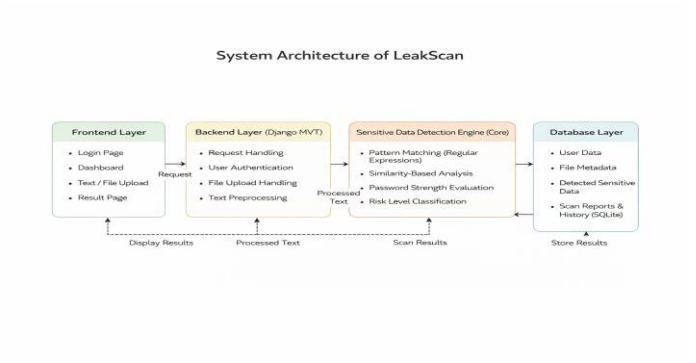
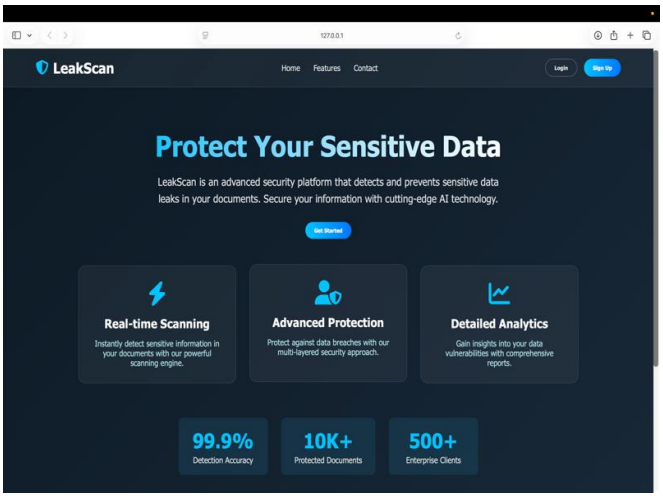


Fig. 1. System Architecture Of Leakscan

Technologies Used

Purpose	Technology Used
Programming Language	Python
Web Framework (Backend)	Django
Web Architecture	Model–View–Template (MVT)
Sensitive Data Detection	Regular Expressions (Regex)
Similarity Analysis	Rule-based Similarity Logic
Password Strength Analysis	Rule-based Evaluation
Frontend Interface	HTML5, CSS3, JavaScript
UI Styling	Bootstrap
File Upload Handling	Django File Handling Utilities
Database	SQLite
Report Generation	Python-based Reporting Utilities
Development Environment	Visual Studio Code
Web Browser (Testing)	Google Chrome



International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 02, NCRTCS - 2026

Fig 2.Home Page

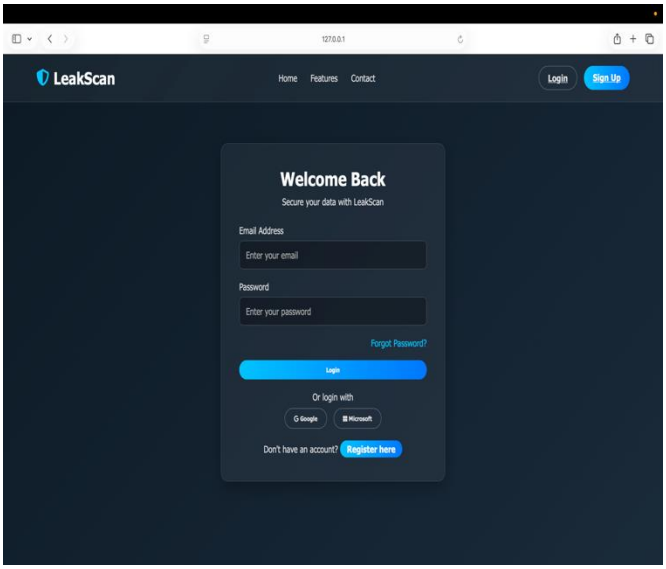


Fig 3.Login Page

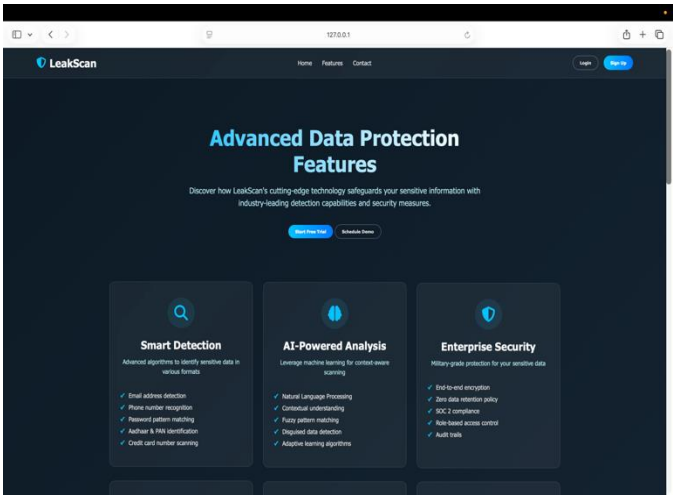


Fig 4.Dashboard Page

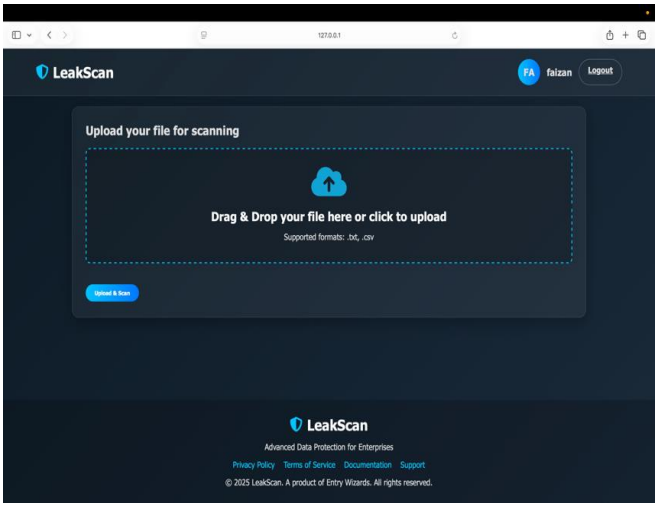


Fig 5.File Upload Section

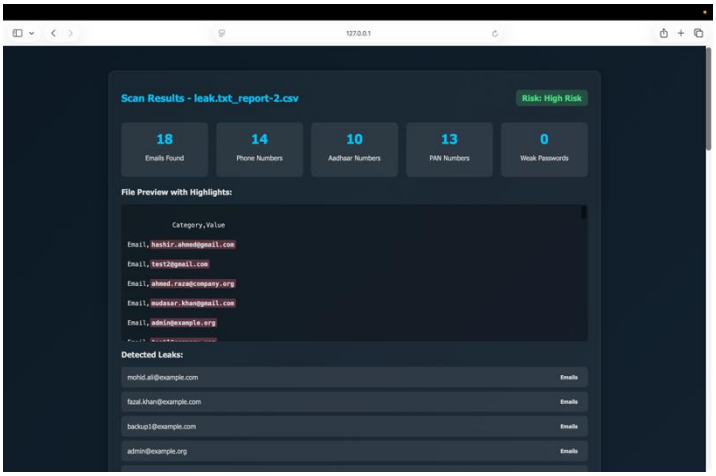


Fig 6.Result Page

IMPLEMENTATION DETAILS

The LeakScan system was implemented as a web-based application with an emphasis on simplicity, usability, and efficient processing of text-based files. The implementation followed the layered architecture described earlier, ensuring a clear separation between the user interface, backend logic, detection mechanisms, and data storage components.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 02, NCRTCS - 2026

Frontend Implementation

The frontend of LeakScan was developed using Django templates along with HTML, CSS, Bootstrap, and JavaScript. The interface provides a clean and intuitive environment that allows users to log in, upload files, initiate scanning, and view results. The design prioritizes readability and ease of navigation so that users can perform sensitive data detection without requiring technical expertise. User interactions are handled through standard web requests, which are forwarded to the backend for processing.

Backend Implementation

The backend was implemented using the Django framework following the Model–View–Template architecture. It manages request handling, user authentication, and file upload operations. Uploaded files are validated and processed securely to extract textual content. The backend performs initial preprocessing steps such as removing unnecessary characters and preparing the text for analysis. After detection is completed, the backend formats the results and returns them to the frontend for display.

Sensitive Data Detection Module

The sensitive data detection logic was implemented as a core component within the backend. Pattern-based detection using regular expressions was applied to identify structured sensitive information such as email addresses, phone numbers, identity numbers, and passwords. To enhance detection capability, similarity-based logic was incorporated to recognize partially modified or obfuscated sensitive data. The module also evaluates password strength and assigns risk levels based on the detected information.

Database Implementation

SQLite was used as the database for storing system data due to its lightweight nature and compatibility with Django. The database maintains user information, file metadata, detection results, risk classifications, and scan history. This stored information supports report generation and allows retrieval of previous scan records.

The overall implementation ensures reliable document processing, accurate detection of sensitive information, and smooth interaction between system components.

RESULTS AND ANALYSIS

The LeakScan system was evaluated by testing it with multiple text-based files containing different forms of sensitive information. The purpose of the evaluation was to analyze the effectiveness of the detection mechanism, the system's ability to handle modified data, and the overall usability of the application.

During experimentation, the system successfully identified commonly structured sensitive information such as email addresses, phone numbers, identity numbers, and passwords. The pattern-based detection approach proved effective for recognizing clearly formatted data, while the similarity-based logic improved detection capability by identifying partially modified or obfuscated sensitive information. This combination enhanced the system's coverage across various input scenarios.

The password strength evaluation component provided additional insight by highlighting weak credentials present in the uploaded files. The classification of detected information into different risk levels enabled users to quickly understand the severity of potential data exposure. The generated reports summarized detection outcomes in a structured and readable format, supporting better interpretation of results.

From a performance perspective, LeakScan processed uploaded files efficiently and returned detection results within a short time. The response time remained suitable for real-time usage, ensuring a smooth user experience. The system maintained consistent behavior across different test inputs without noticeable processing delays.

Overall, the analysis demonstrated that LeakScan effectively detected sensitive information in text-based documents while maintaining simplicity and usability. The results indicate that automated content inspection can significantly assist users in identifying potential data leakage risks prior to document sharing.

DISCUSSION

The evaluation of the LeakScan system highlights several important observations regarding its effectiveness, practical usability, and limitations. This section discusses the strengths of the proposed system along with areas that offer opportunities for improvement.

Strengths of the System

One of the key strengths of LeakScan is its hybrid detection approach, which combines pattern-based detection with similarity-based logic. This design enables the system to identify both clearly structured sensitive information and partially modified or obfuscated data. Such capability improves detection coverage compared to relying solely on predefined patterns.

Another notable strength is the simplicity and usability of the system. The web-based interface allows users to upload files, initiate scans, and interpret results without requiring specialized technical knowledge. The risk level classification further enhances user understanding by presenting detection outcomes in a meaningful and accessible manner.

LeakScan also demonstrates efficient performance characteristics. The system processes uploaded text-based files within a short time, making it suitable for real-time usage. The lightweight technology stack and straightforward architecture ensure that the system operates effectively without demanding high computational resources.

Limitations of the System

Despite its advantages, LeakScan has certain limitations. The current implementation primarily focuses on text-based files, and support for more complex document formats is limited. Sensitive information contained within images or scanned documents is not detected by the present system.

Additionally, the similarity-based detection relies on predefined logic rather than contextual understanding of the text. As a result, certain complex patterns of sensitive information may not always be recognized with complete accuracy. The system also operates on uploaded files only and does not perform continuous monitoring of document activity.

These limitations suggest that while LeakScan provides a practical and useful solution for sensitive data detection, further enhancements can improve its detection capability and applicability across broader scenarios.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 02, NCRTCS - 2026

CONCLUSION

This research presented the design and development of LeakScan, a web-based system intended to detect sensitive information in text-based documents and reduce the risk of accidental data leakage. With the increasing reliance on digital files for information storage and sharing, sensitive data often remains unnoticed within documents, creating privacy and security concerns. The proposed system addressed this issue by providing an automated mechanism for analyzing document contents prior to data exchange.

LeakScan employed a hybrid detection approach that combined pattern-based detection with similarity-based logic to identify both structured and partially modified sensitive information. The integration of password strength evaluation and risk level classification further enhanced the system's ability to highlight potential security risks in a clear and understandable manner. The layered architecture and lightweight implementation ensured efficient performance and usability in academic and small organizational environments.

The results and analysis demonstrated that LeakScan effectively detected various forms of sensitive data while maintaining low processing time suitable for real-time usage. The system's simple interface and structured reporting approach contributed to improved user awareness and informed decision-making before document sharing.

Overall, the study confirms that automated content inspection tools like LeakScan can play a meaningful role in strengthening cybersecurity and privacy practices by helping users identify potential data exposure risks at an early stage.

FUTURE SCOPE

Although LeakScan provides an effective solution for detecting sensitive information in text-based files, several enhancements can be considered for future development. Expanding the system's compatibility with additional document formats such as portable document files and word processing documents can significantly increase its applicability across diverse usage scenarios.

Future improvements may also focus on incorporating more advanced detection strategies capable of analyzing contextual relationships within text. Such enhancements could improve detection accuracy for sensitive information that does not strictly follow predefined patterns. Additionally, extending the range of supported sensitive data categories can further strengthen the system's privacy protection capability.

Another potential direction involves integrating real-time monitoring features that analyze documents continuously during creation or modification. This would allow earlier identification of sensitive information and reduce the likelihood of accidental data exposure. Enhancements in reporting and visualization mechanisms may also improve user interpretation of detection outcomes.

Finally, deploying the system in scalable environments and integrating it with existing document management platforms could increase accessibility and practical adoption. These developments would enable LeakScan to evolve into a more comprehensive solution for sensitive data protection and cybersecurity applications

REFERENCES

[1] National Institute of Standards and Technology, Guide to Protecting the Confidentiality of Personally Identifiable Information (PII), NIST Special Publication 800-122, 2010.
[2] OWASP Foundation, OWASP Top 10 Privacy Risks – Countermeasures, 2022.
[3] Y. Elovici, A. Shabtai, and L. Rokach, A Survey of Data Leakage Detection and Prevention Solutions, SpringerBriefs in Computer Science, Springer, 2012.
[4] R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," Proceedings of the IEEE Symposium on Security and Privacy, pp. 305–316, 2010.
[5] S. Garfinkel and G. Spafford, Web Security, Privacy and Commerce, 2nd ed., O'Reilly Media, 2002.
[6] W. Stallings, Effective Cybersecurity: A Guide to Using Best Practices and Standards, Pearson Education, 2018.
[7] C. Dwork, "Differential Privacy: A Survey of Results," Proceedings of the International Conference on Theory and Applications of Models of Computation, pp. 1–19, 2008.
[8] L. Sweeney, "k-Anonymity: A Model for Protecting Privacy," International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, vol. 10, no. 5, pp. 557–570, 2002.