 DOI:10.5281/zenodo.17795962

Title

DESIGN AND IMPLEMENTATION OF A SECURE COMPLAINT MANAGEMENT SYSTEM WITH INTEGRATED SENSITIVE DATA LEAK DETECTION

Author

FREDRICK GONDWE

Co-Author

MR. PEMPHO JIMU



Issued October 2026 Certificate

AR2026Y7RTP61



ABSTRACT

Complaint-management systems provide a structured way to receive reports, assign responsibility, investigate concerns, and communicate progress. However, complaint descriptions and supporting documents may contain sensitive personal information that should not be exposed to unauthorised users. This paper presents the design and implementation of a Secure Complaint Management System with Integrated Sensitive Data Leak Detection. The system brings complaint submission, tracking, departmental handling, role-based access, investigation workflows, notifications, audit logging, and Leak-Guard screening into one application.

The project documentation describes a web interface with separate user and administrative functions, a backend service, database records, and a sensitive-data detection module. The enhanced system also includes AI-assisted complaint classification and routing, field-level encryption for complaint text, and protected status notifications. These features are described here as implemented only to the extent supported by the final release and its test evidence. The evaluation section records functional test outcomes reported in the project documentation while distinguishing them from independent security, performance, or user studies.

The paper outlines the architecture, modules, evaluation, and limitations of the system. It does not claim live deployment, verified inter-agency integration, or quantitative improvements in real-world complaint handling without supporting evidence. The contribution of this work lies in the practical integration of workflow automation with proactive sensitive-data screening, access control, and decision-support triage within a single software platform, together with a transparent account of the evaluation evidence available at the time of writing.

Keywords: complaint management, sensitive data leak detection, Leak-Guard, data loss prevention, complaint triage, encryption, role-based access control, audit logging, AI-assisted classification

INTRODUCTION

Complaint management is an important administrative process because it gives people a channel to report problems, provides organisations with a way to assign responsibility, and supports follow-up until a case is resolved. A digital complaint platform can organise these activities through a common interface and provide users with visibility of their complaint status. Nevertheless, complaints may contain confidential details, identity information, financial information, or documents that should only be available to authorised officers. A complaint platform therefore needs to consider both workflow management and information security.

In many organisational settings, complaint handling still relies on a mixture of paper forms, email exchanges, spreadsheets, and informal verbal reports. Such fragmented processes make it difficult to maintain a consistent audit trail, to measure response times, or to ensure that the same complaint is not processed in contradictory ways by different units. When a complainant later asks for an update, staff may need to search across several systems or physical files before they can provide an accurate answer. A centralised digital system addresses these operational problems by storing a single authoritative record for each case and by making status changes visible to both the complainant and the responsible officers.

Centralisation, however, concentrates risk. A single database that holds the full text of complaints, attachments, and investigation notes becomes an attractive target for unauthorised access. If the system does not enforce strong authentication, role-based authorisation, and careful handling of sensitive fields, confidential information may be exposed to staff who have no legitimate need to see it, or even to external parties who gain access through a vulnerability. The design of a complaint-management platform must therefore treat security and privacy as first-class requirements rather than as secondary features added after the workflow has been completed.

The Secure Complaint Management System with Integrated Sensitive Data Leak Detection was developed to combine complaint handling with proactive detection of sensitive information. Its Leak-Guard component is intended to identify configured sensitive-data patterns before information is accepted or passed further through the workflow. The system also includes role-based access, departmental handling, administrative monitoring, audit records, and complaint-status

communication. The enhanced version includes AI-assisted triage to recommend a complaint category or destination department, while leaving appropriate room for human review.

This paper reports the design and implementation of the system as described in the project documentation. It presents the architecture, the principal functional modules, the security mechanisms that were integrated, and the evaluation evidence that was recorded during development. Throughout, the paper distinguishes features that are supported by the final release and its test evidence from broader claims that would require additional measurement or institutional deployment studies. The intention is to provide a clear and reproducible account of what was built and how it was checked, rather than to assert unproven operational impact.

BACKGROUND OF THE STUDY

Complaint handling can involve receiving a report, assigning a category, forwarding it to the responsible department, investigating the matter, updating its status, and informing the complainant. When these steps are performed through disconnected channels or manual records, it may be difficult to maintain consistent records, track progress, and establish who performed important actions. A centralised system can support these activities, but centralisation also increases the importance of access control and protection of sensitive data.

In the public sector and in large private organisations, complaints frequently concern service quality, administrative delays, alleged misconduct, billing disputes, or safety concerns. The information submitted by a complainant may include national identification numbers, bank account details, medical information, names of third parties, or documentary evidence that could identify vulnerable individuals. If such information is stored in clear text and made available to every staff member who can open the complaint module, the organisation faces both privacy and liability risks.

Data-loss-prevention (DLP) techniques have been applied successfully in enterprise email gateways, file-sharing platforms, and network egress filters. Pattern matching, regular-expression libraries, and machine-learning classifiers can flag credit-card numbers, national identity formats, and other structured identifiers before they leave an organisation's boundary. Applying similar techniques inside a complaint-submission form is less common, yet the risk of accidental disclosure is high: a complainant who pastes a bank statement or an identity document into a free-text field may not realise that the data will be visible to multiple officers. Integrating a screening step at the point of submission therefore offers a practical opportunity to reduce exposure before the information enters the wider workflow.

The project addresses two related needs: organising the complaint lifecycle and reducing the risk that sensitive information is submitted or exposed inappropriately. The scope of this paper is the software design and implementation described by the project, rather than a claim that the platform has been adopted by a public institution or has already improved real-world service outcomes. The system is presented as a working software artefact that demonstrates the feasibility of combining workflow automation with sensitive-data screening, role-based access, encryption of selected fields, and AI-assisted routing recommendations.

RESEARCH OBJECTIVES

General Objective

- To design and implement a secure complaint management system with integrated sensitive-data leak detection.

Specific Objectives

- To develop a web-based system for submitting complaints and tracking their status.
- To provide role-based access for users and authorised administrative personnel.
- To integrate Leak-Guard screening for configured sensitive-data patterns in complaint inputs.
- To support complaint classification and department-routing recommendations through the implemented triage feature.
- To protect sensitive complaint information through the encryption mechanisms implemented in the final system.
- To support investigation updates, audit

records, notifications, and administrative monitoring.

- To evaluate the system using documented functional and security-focused test cases.

LITERATURE REVIEW

Digital Complaint and Case Management

Digital case-management systems organise records, workflow states, responsibilities, and reporting in a structured way. In a complaint-management context, these capabilities help make a case traceable from submission through assignment, investigation, and closure. The value of such a system depends on whether state changes are persisted correctly and whether users can access only the records permitted by their roles.

Early electronic complaint systems often consisted of simple web forms that emailed a message to a general inbox. While such forms reduced paper handling, they offered little visibility of progress, no structured assignment of responsibility, and no reliable audit trail. Subsequent generations introduced ticket numbers, status codes, and departmental queues. Modern platforms typically support configurable workflows, escalation rules, and dashboards that display ageing cases. The technical challenge is no longer merely capturing a report; it is ensuring that every transition is recorded, that the correct officers are notified, and that the complainant can obtain an accurate status without repeated telephone calls.

Several commercial and open-source products address portions of this requirement. Customer-relationship-management (CRM) suites, help-desk applications, and specialised grievance-redressal portals all provide complaint or ticket modules. Their strength usually lies in workflow configurability and reporting. Their limitation, from the perspective of this project, is that sensitive-data protection is frequently treated as an organisational policy rather than as an automated control enforced at the point of data entry. Consequently, a user may submit unrestricted free text and attachments, relying on subsequent human review to notice and redact confidential content.

Sensitive-Data Detection and Data Loss Prevention

Data loss prevention (DLP) approaches identify information that should not be disclosed or transferred without authorisation. Pattern-based detection can identify supported formats, but may produce false positives and may fail to identify values that do not match configured rules. Therefore, a system should document the patterns it detects, the actions triggered by a match, and the file formats or input types it actually scans. A warning in the user interface should not be treated as a substitute for server-side enforcement.

Classical DLP engines rely on regular expressions, checksum algorithms (for credit-card validation), and dictionaries of known identifiers. More recent systems incorporate statistical models and natural-language classifiers that attempt to recognise free-text descriptions of confidential topics. In the context of a complaint form, a practical first step is to detect structured identifiers—national identity numbers, passport numbers, bank-account patterns, and similar formats—because these can be matched with relatively high precision. Unstructured sensitive content, such as medical narratives or descriptions of domestic violence, is harder to detect automatically and may require human review or more sophisticated language models.

An important design principle is that detection must occur on the server. Client-side JavaScript can provide immediate feedback to the user, but it can also be disabled or bypassed. The authoritative decision—whether to accept, warn, block, or flag a submission—must be made by the backend after the full request has been received and validated. The final implementation description should state which identifiers or terms are detected, how input is normalised, which file types are scanned, and whether a match blocks submission or flags it for review. Filename checks alone must not be described as content scanning.

AI-Assisted Complaint Classification

Natural language processing can be used to analyse complaint text and recommend a category or responsible department. The usefulness of automated triage depends on the method, the quality and representativeness of labelled examples, and the way uncertain cases are handled. The evaluation should identify the actual model or classification

method and use labelled test examples. Where prediction confidence is low or the complaint is ambiguous, the recommendation should be reviewed by an authorised person rather than treated as a final decision.

Rule-based classifiers rely on keyword lists and simple Boolean logic. They are transparent and easy to maintain, but they struggle with synonymy, negation, and multi-topic complaints. Supervised machine-learning classifiers trained on labelled historical data can achieve higher accuracy when sufficient training examples exist, yet they require ongoing monitoring for concept drift. Large language models offer flexible zero-shot or few-shot classification, but they introduce latency, cost, and the need to guard against prompt-injection or hallucinated category labels. Regardless of the method chosen, the system should expose the recommendation to an authorised officer and allow the officer to accept, modify, or override it before the complaint proceeds to the next workflow stage.

Access Control, Encryption, and Auditability

Role-based access control associates system permissions with defined user roles. Department-based rules can further restrict which complaints a particular officer may access. These checks need to be enforced by the backend rather than relying only on which controls are visible in the browser. Encryption also needs to be described precisely: transport encryption, database or disk encryption, and application-level encryption protect different parts of the system. Audit records help reconstruct important actions, but should minimise sensitive content and be protected from unauthorised modification.

Transport encryption (typically TLS) protects data in transit between the browser and the server. Disk or database encryption protects data at rest against physical theft of storage media. Application-level or field-level encryption protects selected columns even from database administrators who can read the underlying tables. Each layer addresses a different threat model. Claiming that a system is “encrypted” without specifying which layer is in use can create a false sense of security. Likewise, password hashing is necessary for credential storage but does not encrypt complaint content.

Audit logs should record the identity of the actor (or service), the type of action, the identifier of the affected record, a timestamp, and the outcome. They should avoid storing the full free-text content of a complaint, because the log itself would then become another repository of sensitive data. Logs must also be protected against unauthorised deletion or alteration; otherwise they lose their value as an accountability mechanism.

Research Gap

The project focuses on combining complaint workflow management with sensitive-data detection and controlled access within one system. Its enhanced triage feature is intended to assist with categorisation and routing. The contribution of the work should be judged through the implemented features and reproducible evaluation, rather than assuming that integration alone proves novelty or improved institutional outcomes.

Several existing complaint-management systems focus primarily on workflow automation, leaving security and privacy as secondary considerations. Many systems allow users to submit complaints and track their progress, but often provide limited protection against accidental disclosure of sensitive information. This limitation becomes increasingly important when complaints contain personal details, financial information, identification numbers, or investigative evidence.

Studies on Data Loss Prevention have demonstrated that pattern-based detection mechanisms can significantly reduce accidental leakage of confidential information when integrated into user workflows. However, most DLP implementations target enterprise environments such as email systems, document repositories, and corporate networks. Relatively little attention has been given to integrating DLP capabilities directly into complaint-management platforms where citizens or employees submit potentially sensitive reports.

Research on access control and secure information systems has also highlighted the importance of enforcing permissions at the server level. Client-side restrictions alone cannot prevent unauthorised access because browser controls may be bypassed. Consequently, modern secure systems rely on backend authorisation mechanisms, audit trails, and encryption controls to protect sensitive information throughout its lifecycle.

Artificial intelligence has further expanded the

capabilities of workflow-management systems. Automated classification techniques can analyse complaint descriptions and recommend categories or responsible departments. This can reduce manual workload and improve routing efficiency. Nevertheless, classification errors remain possible, particularly for complex or ambiguous complaints. Human oversight therefore remains an important element of practical deployments.

Based on this review, there remains a need for a complaint-management system that combines workflow automation, sensitive-data detection, access control, encryption, auditability, and AI-assisted triage within a single integrated platform. The Secure Complaint Management System aims to address this gap through the combination of these security and workflow-management features.

METHODOLOGY

Research and Development Approach

The work follows a software design-and-implementation approach. Requirements were translated into modules, the modules were integrated into the application, and the resulting workflows were checked using functional tests described in the project documentation. An iterative approach is suitable for this type of system because workflow requirements and security rules may need refinement as modules are integrated. This paper distinguishes features described in the project from results that require additional measured evidence.

The project was developed using the Agile Software Development Methodology. Agile was selected because security requirements, complaint workflows, and administrative features evolved throughout implementation. The methodology allowed functionality to be developed incrementally while supporting continuous testing and refinement. Each iteration introduced new capabilities including complaint submission, status tracking, Leak-Guard detection, AI-assisted routing, encrypted data storage, and notification management.

Requirements gathering involved analysing typical complaint-handling workflows performed manually or through fragmented systems. Functional requirements focused on complaint submission, tracking, departmental assignment, and administrative processing. Security requirements focused on authentication, authorisation, encryption, sensitive-data detection, auditability, and controlled information disclosure.

Following requirements analysis, the system architecture was designed to separate presentation, application logic, and data-management responsibilities. The implementation stage involved frontend development, backend service development, database integration, security-module integration, and testing. Functional verification was performed throughout development to ensure modules operated correctly before being integrated into the complete system.

System Requirements

Requirement area	Description
Complaint management	Support complaint submission, reference tracking, status updates, assignment, and administrative handling.
Access control	Separate user and administrative permissions and restrict access to authorised records.
Leak-Guard	Detect configured sensitive-data patterns and apply the implemented warning, blocking, or review action.
Triage and routing	Analyse complaint text and recommend a category or destination according to the implemented method.
Data protection	Apply the encryption mechanisms present in the final release and avoid exposing sensitive values unnecessarily.
Notifications and audit	Record relevant workflow actions and communicate status updates using the implemented notification mechanism.
Evaluation	Check functional workflows and test access restrictions, leak detection, persistence, and response behaviour.

Evaluation Procedure

The evaluation should use repeatable test inputs and preserve the results of each execution. Functional tests check account access, complaint submission, tracking, administrative status changes, and

notification behaviour. Security tests should call protected backend routes with authorised and unauthorised accounts. Leak-Guard should be tested with positive and negative examples. Triage should be compared against expected categories using labelled examples. Performance claims require a stated test environment, workload, sample size, measurement method, and observed values. Informal interaction is not a substitute for a documented usability study.

SYSTEM DESIGN AND DEVELOPMENT

System Architecture and Technologies

The system is organised around a user-facing web application and a backend that manages complaint operations and security decisions. The frontend provides the complaint form, status views, and administrative dashboards. The backend receives requests, validates data, applies access rules, invokes the relevant complaint or security logic, and persists accepted operations. The database stores the system's operational records. Leak-Guard and the triage feature support sensitive-data screening and routing recommendations. Notifications and audit records support communication and traceability.

The project materials describe a React frontend and a Node.js/Express backend. The database and deployment configuration should be named in the final submitted version exactly as they appear in the release being evaluated; technology names should not be inferred from earlier plans or replaced with services that are not present in the implementation.

The architecture was designed to support both operational functionality and security controls. Requests originate from authorised users through the web interface and are transmitted to backend services responsible for authentication, complaint management, Leak-Guard analysis, AI-assisted classification, database interaction, and notification processing. This separation of responsibilities improves maintainability and reduces coupling between business logic and user-interface components.

Complaint records move through a defined lifecycle beginning with complaint creation, followed by analysis, assignment, investigation, status management, and closure. Administrative users perform workflow-management tasks through dedicated interfaces while ordinary users interact primarily with complaint-submission and tracking functions.

The database serves as the system's authoritative source of operational information. It stores user information, complaint records, category assignments, investigation updates, notification references, and audit-related information. Database persistence ensures that complaint histories remain available throughout the complaint lifecycle while supporting reporting and administrative oversight.

Security considerations influenced all major architectural decisions. Authentication controls verify user identity before access is granted. Authorisation mechanisms restrict user actions according to assigned roles. Leak-Guard screening provides proactive detection of configured sensitive information, while encryption mechanisms protect designated complaint content from direct exposure. Audit records provide accountability by recording significant system activities and administrative actions.

Component	Role in the system	Implementation note
Web frontend	Complaint entry, user status view, and administrative dashboards.	React is described in the project materials.
Backend API	Processes requests, validates inputs, applies permissions, and coordinates modules.	Node.js/Express is described in the project materials.
Database	Stores user, complaint, workflow, and relevant audit records.	Confirm the exact database in the final release.
Leak-Guard	Checks configured sensitive-data patterns and returns a policy outcome.	Describe only the input types and patterns actually supported.
Triage module	Recommends a category or department from complaint text.	Identify the actual algorithm or model used.
Notification module	Sends status information	State the actual channel and

	through the configured channel.	privacy protections.
Audit component	Records relevant actions for accountability.	Avoid storing full sensitive complaint content in logs.

Functional Modules

User management and access control

Supports account access and separates ordinary users from authorised administrative roles. Backend permission checks are required for protected operations. Ordinary users can submit complaints and view the status of their own cases. Administrative users can view assigned complaints, update workflow status, and perform investigation actions according to their role and department membership. Authentication verifies identity; authorisation decides which records and actions are permitted after identity has been established.

Complaint management

Provides the core complaint workflow, including submission, tracking, assignment, and status management as supported by the application. Each accepted complaint receives a unique reference identifier that the complainant can use for subsequent enquiries. Status transitions (for example, from "submitted" to "under investigation" to "resolved") are recorded so that the history of the case remains available. Assignment links a complaint to a responsible department or officer; the assignment itself is an auditable event.

Leak-Guard detection

Checks configured sensitive-data patterns in supported input and applies the implemented policy, such as warning, blocking, or flagging for review. Leak-Guard is intended to act as a security gate for complaint information. Client-side detection can help users correct an input before submission, but the backend must make the authoritative decision because browser-side checks can be bypassed. The final implementation description should state which identifiers or terms are detected, how input is normalised, which file types are scanned, and whether a match blocks submission or flags it for review.

AI-assisted triage and routing

Analyses complaint text and proposes a category or destination department. The final report should identify the exact model or baseline and describe how uncertain predictions are handled. The enhanced triage function is intended to examine a complaint description and recommend a relevant category or department. A rule-based approach, a conventional machine-learning classifier, and a language-model-based classifier are different methods and should not be described interchangeably. If the system records confidence or supports human correction, those behaviours should be described and tested. Quantitative model performance should only be reported where a labelled evaluation set and actual prediction results are available.

Investigation and administrative handling

Allows authorised personnel to review assigned complaints, update workflow status, and record investigation actions supported by the system. Investigation notes and status changes become part of the complaint history. Access to these records is governed by the same role-based and department-based rules that protect the original complaint text.

Notifications

Communicates relevant complaint-status events through the notification mechanism implemented in the release. Messages should not reveal unnecessary complaint details. The enhanced notification design calls for protecting notification content and avoiding unnecessary retention of phone numbers in database logs. The final description should identify the actual notification channel, whether a phone number is retained in any operational table, and what metadata is written to logs.

Audit and analytics

Provides records or summaries of relevant actions and complaint activity, subject to the fields and reporting functions actually implemented. Audit records should capture the event type, actor or service identity, affected record reference, time, and outcome where those fields are implemented, while avoiding

unnecessary sensitive content.

Data Management and System Scope

The system manages complaint and workflow information in its configured database. The operational database should remain the authoritative source for complaint state, status history, assignments, and investigation records. Audit records provide supporting traceability but do not replace operational records. The current paper describes the application as a software implementation and does not claim that it is connected to live production systems of government ministries or other institutions unless such integration is separately demonstrated.

RESULTS AND EVALUATION

The project report includes a test-plan table that records outcomes for selected system functions. These reported outcomes are presented as project documentation rather than as independently audited results. The table should be reconciled with the final release and supporting test logs before journal submission. Where a test has not been executed or its output cannot be reproduced, the result should be marked pending rather than assumed to have passed.

Interpretation of Test Results

The documented statuses suggest that selected workflows were checked during development. However, a status of "Passed" in a project test plan does not by itself establish exhaustive coverage, resistance to security attacks, high availability, or readiness for production use. A publication version should retain test logs, test data, screenshots or API outputs where appropriate, and explain the environment in which each test ran.

The functional testing results indicate that the primary system workflows operated as expected during development. Complaint submission, access control, status tracking, notification generation, and security-screening operations produced outcomes consistent with the intended system behaviour. The successful execution of these workflows demonstrates that the different components were able to operate together within a single integrated environment.

Leak-Guard testing focused on determining whether configured sensitive-data patterns could be identified before information progressed through the complaint workflow. Positive test cases included inputs containing protected information patterns, while negative test cases used ordinary complaint descriptions. The test outcomes demonstrated that configured detection rules were processed appropriately according to the implemented policy.

AI-assisted triage testing evaluated whether the system generated routing recommendations that corresponded with expected complaint categories. Because complaint language can vary significantly between users, administrative review remains important for ambiguous cases. The implemented functionality should therefore be viewed as decision support rather than fully autonomous complaint routing.

Informal interaction formed part of the development process. Feedback obtained during implementation assisted in identifying navigation issues, clearer presentation of status information, and improvements to complaint-submission workflows. However, no structured survey instrument or formal participant study was conducted. Consequently, the feedback should be considered formative input rather than evidence from a documented usability evaluation.

LIMITATIONS

- The reported functional test cases cover selected workflows and do not establish exhaustive application coverage.
- The available project documentation does not by itself establish independently audited security or encryption performance.
- The effectiveness of Leak-Guard depends on the configured patterns and the input types actually scanned; false positives and missed patterns remain possible.
- AI triage performance cannot be quantified without identifying the model or method and evaluating it against labelled examples.
- The stated response-time target should not be treated as an achieved measurement unless the underlying test results are retained.
- The paper does not establish live institutional deployment, official inter-agency integration, or measured improvements in real-world

complaint resolution.

DISCUSSION

The Secure Complaint Management System brings complaint submission and tracking together with administrative handling and sensitive-data screening. Its central design concern is that complaint workflows and privacy controls must operate together: a system that manages a case efficiently but exposes confidential content would not meet its intended purpose. Leak-Guard is intended to reduce accidental or inappropriate submission of configured sensitive information, while access control restricts who can view or manage complaints.

The AI-assisted triage enhancement can help recommend categories and destinations, but it should be treated as decision support. Its reliability depends on the implemented method and the quality of the evaluation data. Human review remains important for ambiguous complaints and for cases where a routing error could have serious consequences. Similarly, the encryption and notification enhancements should be described in terms of their exact implementation rather than broad claims of complete security.

The evidence in the project report is best interpreted as preliminary software-testing evidence. More comprehensive evaluation is needed to establish access-control correctness across all endpoints, detection quality, triage performance, persistence reliability, notification privacy, and response time. The current implementation should not be described as a deployed public service or as having demonstrated institution-wide outcomes unless those claims are supported by separate evidence.

The integration of Leak-Guard functionality distinguishes the system from conventional complaint-management approaches that focus solely on workflow automation. By incorporating sensitive-data screening into the submission process, the platform attempts to reduce the risk of accidental disclosure while maintaining the usability of complaint-reporting functions.

The introduction of AI-assisted triage further demonstrates how intelligent technologies can support administrative processes without fully replacing human decision-making. Routing recommendations can improve efficiency while still allowing authorised personnel to review classifications before complaints enter subsequent workflow stages.

Encryption, access control, auditability, and notification management collectively contribute to a defence-in-depth approach. Rather than relying on a single security mechanism, the platform uses multiple layers of protection to reduce exposure of complaint information and improve accountability throughout the complaint lifecycle. While the effectiveness of individual controls requires additional evaluation, the implementation provides a practical demonstration of how security features can be integrated directly into complaint-management software.

From a practical standpoint, the system illustrates a design pattern that other administrative platforms could adopt: place a screening gate as close as possible to the point of data entry, enforce authorisation on every protected backend route, encrypt fields whose clear-text exposure would create disproportionate risk, and keep an audit trail that is itself protected from unauthorised modification. None of these measures is novel in isolation; their combined presence inside a single complaint-management application is the contribution that this project seeks to demonstrate.

At the same time, the limitations listed in the evaluation section remain important. Pattern-based detection cannot catch every form of sensitive content. Automated triage cannot replace human judgement on complex or multi-topic cases. Field-level encryption introduces key-management complexity. Performance and usability claims require measurement under realistic load and with representative users. Future work should therefore focus on strengthening the evidence base rather than on expanding the feature set without corresponding evaluation.

CONCLUSION

This paper presented the design and implementation of a Secure Complaint Management System with Integrated Sensitive Data Leak Detection. The implemented platform combines complaint submission, tracking, investigation support, department routing, administrative management, audit recording, notification handling, and sensitive-data screening

within a unified environment. The enhanced implementation also incorporates AI-assisted complaint classification and routing recommendations together with encryption mechanisms for protecting selected complaint information.

The evaluation evidence presented in this paper consists primarily of documented functional and security-oriented testing undertaken during development. These tests provide preliminary evidence that the implemented modules operated according to their intended purpose. Leak-Guard detection, access-control enforcement, complaint tracking, notification processing, and AI-assisted routing were integrated successfully into the overall system architecture. However, the available evidence should not be interpreted as establishing comprehensive security assurance, large-scale usability validation, or readiness for institutional deployment.

Future work should include broader security evaluation, reproducible performance measurement, structured usability testing, and expanded assessment of AI-assisted classification accuracy. Additional research should also examine advanced sensitive-data detection techniques, improved routing models, and deployment considerations for large-scale operational environments.

Within these limitations, the project provides a practical implementation demonstrating how complaint workflow management and security controls can be integrated into a single software platform.

REFERENCES

1. National Institute of Standards and Technology. (2023). Artificial Intelligence Risk Management Framework (AI RMF 1.0). NIST AI 100-1. https://doi.org/10.6028/NIST.AI.100-1
2. National Institute of Standards and Technology. (2020). Security and Privacy Controls for Information Systems and Organizations. NIST Special Publication 800-53, Revision 5. https://doi.org/10.6028/NIST.SP.800-53r5
3. National Institute of Standards and Technology. (2023). Advanced Encryption Standard (AES). FIPS PUB 197, updated edition. https://doi.org/10.6028/NIST.FIPS.197-upd1
4. OWASP Foundation. (2023). OWASP API Security Top 10—2023. https://owasp.org/API-Security/
5. OWASP Foundation. (n.d.). Authorization Cheat Sheet. OWASP Cheat Sheet Series. https://cheatsheetseries.owasp.org/
6. OWASP Foundation. (n.d.). Cryptographic Storage Cheat Sheet. OWASP Cheat Sheet Series. https://cheatsheetseries.owasp.org/
7. OWASP Foundation. (n.d.). Logging Cheat Sheet. OWASP Cheat Sheet Series. https://cheatsheetseries.owasp.org/
8. National Institute of Standards and Technology. (2020). Recommendation for Key Management: Part 1 – General. NIST Special Publication 800-57 Part 1, Revision 5. https://doi.org/10.6028/NIST.SP.800-57pt1r5
9. ISO/IEC. (2013). Information technology — Security techniques — Code of practice for information security controls. ISO/IEC 27002:2013.
10. European Union. (2016). Regulation (EU) 2016/679 of the European Parliament and of the Council (General Data Protection Regulation). Official Journal of the European Union.