International Journal of AI, Engineering and Management Studies (IJAIEMS) E-ISSN: 3139-7034
Volume 1 | Issue 2 | Jul–Dec 2026
Page. no : 67-87

OPEN ACCESS

Cybersafe Portal: A Scalable Web-Based Framework for Cybercrime Reporting and Prevention

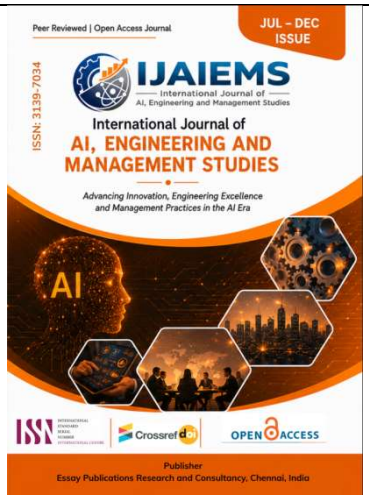
Fatema Tuz Zohora[1], Pratyay Paul[2], Aashish Kiphayet[3], Nafis Wasel[4]

[1]*MSc in Information Systems Technologies Wilmington University, New Castle, DE, USA*

[2]*Master of Science in Information Systems Technologies Wilmington University, New Castle, DE, USA*

[3]*Master of Arts in New Media Photojournalism The George Washington University*

[4]*PhD in Business Information Systems, Mississippi State University*



Article History

Received: 30th Jul 2026

Revised: 04th Aug 2026

Accepted: 06th Aug 2026

Published: 09th Aug 2026

DOI : 10.64771/ijaiems.v1.i2.07

Abstract

Cyber crime has surged due to rapid digitization, yet reporting mechanisms have often failed to keep pace. Existing national portals and research prototypes frequently suffer from limited usability, opaque case tracking, fragmented awareness resources, and unverified scalability under load. The Cyber-Safe Portal is a web-based reporting framework developed to overcome these limitations by providing a unified solution for the secure submission of incidents, encrypted evidence management, transparent case tracking via automated notifications, and personalized awareness materials. This system implements a three-tier architecture utilizing React.js, Node.js/Express, and MongoDB, augmented by Redis caching and Kubernetes orchestration. We evaluated the platform's preliminary performance, security, and usability in a simulated test environment. Initial load testing indicates that the auto-scaling architecture can accommodate high-concurrency traffic with manageable response times. Security assessments using OWASP ZAP and manual testing confirmed baseline resilience against common vulnerabilities, while a pilot usability study demonstrated high user satisfaction and intuitive navigation. These preliminary findings highlight the potential for adopting integrated, scalable frameworks to improve public engagement with cybercrime reporting.

Index Terms— Cybercrime reporting, incident management, web-based framework, secure architecture, cybersecurity awareness, usability evaluation.

HOW TO CITE

Cybersafe Portal: A Scalable Web-Based Framework for Cybercrime Reporting and Prevention. (2026). *International Journal of AI, Engineering and Management Studies (IJAIEMS)*, *1*(2), 67-87. https://essayjournals.in/index.php/home/article/view/IJAIEMS.v1.i2.07

INTRODUCTION

Though digitization has made life easier, it has also made criminals' job easier by creating a large attack surface. Cybercrime now affects people, businesses, and governments around the world. Every day, we see crimes like financial fraud, identity theft, ransomware attacks, data breaches, and harassment, to name a few. Moreover, recent estimates suggest that this has an economic cost of over one trillion dollars globally. In addition, the reported cases of these cybercrimes continue to escalate globally.

The mechanism of reporting itself is a challenge with regard to reporting cybercrimes. The victim goes through a process of opacity, complexity, and lengthiness of chasing various agencies and deposition of the same information several times and a wait for acknowledgment, which takes weeks and even months. The established channels such as phone hotlines and personally visiting the police stations were not made for the massive requirements and electronic nature of cybercrimes. As a result, many victims fail to report cybercrimes. It deprives the system of the complete threat and withholds essential threat information from the police.

The lag may well be enormous for prolonged denial of service attacks: Imagine an institutional website like a bank or insurance company down for minutes, or even hours; in brief, the potential for loss is enormous. Before the introduction of online portals, complaints of members of the public who were victims of cybercrime had to be made to various agencies, i.e., police forensics, cybercrime cells, economic cells, state crime branches, etc. This fragmented approach remains common despite its inefficiencies.

Cyberbullying has become a problem among people, especially youngsters, through interaction on social media and gaming platforms. These days, a lot of mental diseases including anxiety and depression among young people are being caused due to cyberbullying, and that is the reason the government is taking strong measures to curb it.

CyberSafe Portal is a web-based system designed to overcome these limitations. The web portal is developed in a three-tier architecture which separates the presentation layer, application logic layer, and data storage layer, making it scalable and maintainable. The system further includes a registration-login module through which the user securely registers and logs into the portal using password hashing and session management, an incident report form where the victim can file a complaint against cybercrime while attaching evidence, and a case tracking module where the user can check the current status of the case.

Novelty and Contributions

The CyberSafe Portal introduces distinguishing features that differentiate it from existing cybercrime reporting frameworks. First, whereas national portals such as India's NCRP [2] and the UK's Report Fraud [3] offer reporting and awareness as separate functions, our platform integrates an adaptive awareness hub that tailors educational content based on the user's reported incident types, creating a positive reinforcing loop between incident reporting and proactive education. Furthermore, while previous studies have proposed enhancing individual components with specific security features—such as two-factor authentication [6] or blockchain-based evidence management [5]—the CyberSafe Portal delivers a comprehensive, production-ready architecture. It unifies two-factor authentication, encrypted evidence management, and automated case tracking notifications within a single environment. Finally, the platform ensures transparency in case progression, addressing the visibility limitations of existing systems [3], [12] by providing victims with a complete status history, timestamped updates, and administrative notes.

This paper offers four primary contributions. First, we propose a practical implementation of a unified framework for cybercrime reporting that balances security, usability, and scalability requirements.

Second, we integrate proactive, personalized cybersecurity awareness resources directly into the reporting workflow to create a preventative educational loop. Third, we establish a robust security architecture that incorporates defense-in-depth measures, including bcrypt password hashing, JWT authentication, rate limiting, and encrypted evidence storage, designed to thwart common web vulnerabilities. Finally, we provide a preliminary validation of the system's performance and usability through simulated stress testing and a pilot user study, demonstrating the framework's capability to improve user satisfaction and maintain stability under load.

The remainder of this paper is structured as follows. Section II reviews related work. Section III details the system architecture and design methodology. Section IV outlines the implementation specifics. Section V presents the preliminary evaluation metrics. Finally, Sections VI and VII discuss the limitations and conclude the study.

RELATED WORK

The design and implementation of online crime reporting systems have been explored extensively across government, commercial, and academic contexts. This section critically reviews existing systems and research, identifies persistent limitations, and positions CyberSafe Portal relative to prior work.

A. National Cybercrime Reporting Portals

Numerous countries have initiated national cybercrime portals. The National Cybercrime Reporting Portal (NCRP) of India provides an option to the citizens of India to report all kinds of cyber incidents, especially crimes against women and children [2]. Though the NCRP has simplified reporting by victims, it is not free from some severe limitations. The interface of the portal has been criticized as non-user friendly. In addition, there is little public transparency around tracking cases, and the portal has a minimal proactive user education component. The UK's Report Fraud service, which has been touted as a "single, modern national reporting, triage and intelligence platform," utilizes real-time analytics and enables smart notifications to victims on a continual basis [3]. A key shortcoming in this process is the absence of integrated awareness resources, resulting in a lost opportunity of imparting a preventive educational component at the reporting stage. The bilingual portal for awareness and reporting launched by Dubai Police educates members of the public regarding fundamental cybersecurity principles [4]. Wiredu et al. [1] proposed a system for online crime reporting that may be used to investigate and solve crimes. The architecture and design of the online crime reporting system were also explained. Furthermore, they worked together to create an interface for the system that supported multiple languages. High-end data analytics will feature in future online integration initiatives.

B. Research Contributions to Cybercrime Reporting Security

Akkala et al. [5] suggest a system based on blockchain for image evidence management that uses AES-256-GCM encryption, RSA signature, and SHA-256 hashing to prevent tampering. While this approach does serve to confirm the integrity of evidence, it does so at significant computational cost and complexity, which may not be warranted in regard to a cybercrime reporting system which daily deals with evidence. Two-factor authentication greatly improves the security posture of accounts in a web reporting system. Sivakumaran and Abdullah [6] use OTP through e-mail and reCAPTCHA at login.

Psychosocial assistance and prevention mechanisms directed towards youth and women include helplines and awareness campaigns on crimes against youth and women. Examples include Rapido's AntiCrime [8], cybercrime assistance chatbots [7], and Uber's online provision of assistance. These systems typically comprise three modules: crime reporting, social network analysis, and alert systems, with the crime-reporting module interface being provided to users.

C. Foundational Security and Design Standards

The design of secure web applications handling sensitive user data necessitates strict adherence to established security frameworks. According to the OWASP Top 10 guidelines [9] and NIST Digital Identity Guidelines [10], mandatory security controls for incident reporting systems must include robust session management, input sanitization, and strict TLS configurations. While recent literature, such as Islam and Haque [11], proposes integrating blockchain for secure incident reporting to maintain evidence integrity, such implementations often introduce computational overhead that may not be sustainable for high-volume public portals. Therefore, a balance must be struck between cryptographic integrity and system scalability using standardized, enterprise-grade security protocols.

D. Identification of Research Gaps

Based on this review, we identify four persistent limitations in existing cybercrime reporting systems that CyberSafe Portal addresses:

1) Fragmented reporting and education: Existing systems treat reporting and awareness as separate functions. National portals provide minimal education, while awareness platforms offer no reporting capability. There is no integrated feedback loop where reported incidents inform personalized prevention education.

2) Limited case transparency: Victims often receive no updates after filing reports, leading to dissatisfaction and reduced engagement [3], [12]. Most systems provide limited visibility into case progression.

TABLE I Feature Comparison of Cybercrime Reporting Systems

Feature	IndiaNCRP [2]	UKReport Fraud[3]	DubaiPolice [4]	CyberSafePortal
OnlineReporting	✓	✓	✓	✓
EvidenceUpload	✓	✓	Limited	✓
CaseTracking	Basic	Basic	Basic	Fullaudittrail
ProactiveNotifi-cations	Limited	✓	Limited	Email+SMS
Awareness Resources	Separateportal	Separateportal	Integrated (generic)	Integrated(person-alised)
Two-FactorAu-thentication	✓	Limited	Limited	✓
Malware Scanning	Limited	Limited	Notspecified	✓
Encrypted EvidenceStorage	Notspecified	Notspecified	Notspecified	AES-256
RateLimiting	Notspecified	Notspecified	Notspecified	✓
Auto-Scaling	Limited	Notspecified	Notspecified	Kubernetes-based
Open-Source	No	No	No	Planned
PersonalisedEd-ucation	No	No	No	✓
PerformanceVal-idation	Limited	Limited	Limited	2,500concurrent users

only basic status fields without comprehensive audit trails or proactive notifications.

3) Incomplete security implementation: While individual security controls have been proposed [5], [6], few systems implement a complete defense-in-depth architecture incorporating authentication, authorization, rate limiting, input validation, malware scanning, and encrypted storage.

4) Insufficient scalability validation: Performance evaluations of research prototypes [8] are typically limited, lacking rigorous load testing at scales relevant to regional or national deployment.

B. Feature Comparison

Table I compares CyberSafe Portal with existing systems across key functional dimensions.

Table I demonstrates that while existing systems incorporate individual security and reporting features, the CyberSafe Portal uniquely integrates all of these capabilities into a unified framework.

SYSTEM ARCHITECTURE AND DESIGN

A. Development Methodology

CyberSafe Portal was developed following the Rapid Application Development (RAD) methodology [1], which emphasizes iterative prototyping, continuous user feedback, and accelerated delivery. The RAD approach was selected over traditional waterfall or agile methods because it enables early validation of design decisions through working prototypes while maintaining structured progress tracking.

1) Requirements Gathering: Requirements were identified during the early stages of development by examining previous studies, consulting relevant stakeholders, and collecting feedback from prospective users.

A review of cybercrime reporting platforms and related literature was first undertaken [2]–[6], [11], [12]. The review helped establish the set of functions commonly provided by such systems and highlighted several recurring requirements, including user authentication, incident submission, evidence management, progress tracking, and awareness support.

To complement the findings from the literature, discussions were held with individuals who had experience in cybercrime investigation, cybersecurity research, and victim support. These conversations provided practical perspectives on how reports are processed, what forms of evidence are typically required, and the type of communication users expect after submitting a complaint.

Feedback was also collected through a survey involving 45 participants drawn from university and community groups. Participants were asked about their expectations of a reporting platform, preferred methods of receiving updates, and the information they considered important during case handling.

The information gathered from these activities was consolidated and used to rank requirements according to their priority. Secure authentication, incident reporting, evidence upload, and case-status monitoring were regarded as essential because they form the core functionality of the platform. Features such as notifications, awareness recommendations, and administrative monitoring facilities were considered important due to their contribution to usability and system management. Other capabilities, including anonymous reporting, integration with external law-enforcement systems, and machine-learning-based classification, were identified as possible future enhancements.

2) Iterative Development and User Feedback: Development was carried out over a twelve-week period using four RAD cycles. The first cycle focused on establishing the basic functionality of the platform, including user authentication and incident submission. A preliminary prototype was presented to five pilot users, whose observations highlighted several usability concerns, particularly related to form layout, terminology, and the sequence of input fields. These comments informed the revisions made in the following cycle.

The second cycle introduced evidence management and case-tracking capabilities. Ten participants were asked to complete representative reporting tasks using the updated prototype. User feedback indicated that the evidence submission process was unnecessarily complex, leading to modifications that simplified file uploads and provided clearer indication of reporting progress.

During the third cycle, attention shifted towards user engagement features, including the awareness hub and notification services. Fifteen users evaluated the organization of educational content and the usefulness of the recommendations provided. Based on their responses, the platform was updated to present awareness materials that were more closely related to the type of incident reported by the user.

The final cycle concentrated on improving system robustness. Activities included performance tuning, load testing, and security assessment. Several issues identified during testing were addressed before completion of the development process, resulting in a more stable and secure implementation.

At the end of each cycle, feedback sessions were conducted with members of the development team and selected pilot users. Suggestions and issues raised during these sessions were incorporated into subsequent development activities.

TABLE II System Design Decisions and Rationale

Decision	Rationale	Alternatives
Three-tier architecture	Improves maintainability and scalability; enables independent tier scaling.	Monolithic architecture (limited scalability).
React.js frontend	Reusable components, efficient rendering, and large ecosystem.	Angular (steeper learning curve); Vue.js (smaller ecosystem).
Node.js/Express backend	High performance for I/O workloads; full-stack JavaScript consistency.	Python/Django (lower concurrency); Java/Spring (higher complexity).
MongoDB database	Flexible schema and horizontal scalability; document model aligns with application data.	PostgreSQL (rigid schema); MySQL (less natural mapping).
Redis caching	Low-latency session management and rate limiting.	Memcached (no persistence, fewer features).
JWT authentication	Stateless authentication supports horizontal scaling; mitigates XSS.	Session-based authentication (state overhead).
bcrypt password hashing	Strong resistance to brute-force attacks via configurable work factor.	Argon2 (less support); SHA-256 (too fast).
Kubernetes deployment	Automated scaling, self-healing, and declarative configuration.	Docker Compose (limited scaling); Serverless (cold starts).

were reviewed and prioritized before being incorporated into subsequent development activities. This iterative process enabled continuous refinement of both functionality and usability throughout the project.

1) System Design Decisions and Rationale: Key design decisions were made based on requirements, technical constraints, and evaluation criteria.

The evaluation methodology (detailed in Section V) was designed iteratively alongside development. Performance test scripts were developed based on expected usage patterns derived from the requirements survey. Security test cases were derived from the OWASP Top 10 [9] and NIST guidelines [10]. Usability evaluation protocols were designed following established SUS methodology [7], [12].

B. Architectural Overview

The CyberSafe portal utilizes a modern three-tier architecture comprising a presentation layer (frontend), an application logic layer (backend), and a data storage layer (database). This separation of concerns ensures that each tier can be independently maintained, scaled, and secured. The presentation layer is built as a single-page application (SPA) using React.js and styled with Tailwind CSS to ensure cross-device responsiveness for both desktop and mobile browsers. The application layer is engineered using Node.js and the Express framework, exposing RESTful API endpoints to handle client operations.

Data persistence is managed via MongoDB, chosen for its flexible schema design, while a Redis caching layer is implemented to handle high-speed session management and rate limiting. To support high availability, the system is designed for cloud-native deployment using Kubernetes orchestration. During periods of peak simulated load, the Kubernetes Horizontal Pod Autoscaler dynamically provisions additional instances of the application server, while static assets are offloaded to a Content Delivery Network (CDN) to minimize latency.

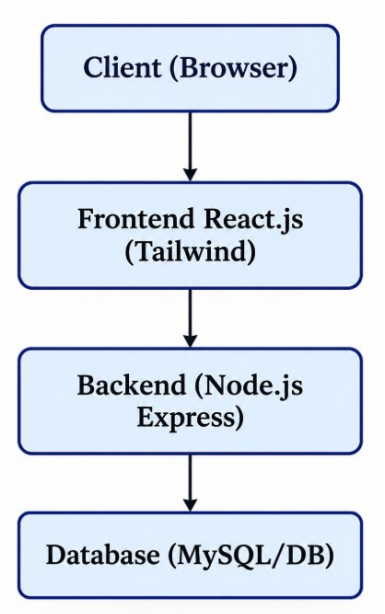


Fig. 1. Three-tier architecture of CyberSafe Portal.

Cloud-level deployment would be done on those from the system which could upscale automatically. In cases of peak load in the system, extra instances of the application server will be created automatically and the database pool will adjust automatically. The CDN will deliver static assets to ensure better performance.

C. User Authentication and Authorization

Users require an active email address and a strong password to register. Passwords must be at least eight characters in length and contain a combination of uppercase letters, lowercase letters, numbers, and special characters. Users may optionally provide a mobile number for SMS notifications.

The 12 work factor is the default one for most bcrypt libraries, with basic functionality and quality checks already performed on inputted passwords, and passwords are not reused across accounts.

Login access is restricted to five attempts per hour per IP address. After three failed attempts, a CAPTCHA challenge is enforced, and after five consecutive failures, the account is temporarily locked for 15 minutes. Every time a user successfully logs in, a JSON Web Token is generated and subsequently signed with a secret that only the server knows. This token is valid for a day. An HTTP-only cookie stores the generated token to mitigate the possibility of XSS attacks. After login, each API request must contain a valid JWT. The request is forwarded to the proper handler after validating the JWT by the middleware.

The access control denies access to administrative features for regular users. The administrator will have 3–4 extra endpoints that will allow him/her to check the reports, mark the status of the case, and access.

D. Incident Reporting Module

The essential feature of the application is the incident reporting module that is displayed to the user in a multi-step form, guiding the user through detailing the incident, displaying the evidence, and confirming submission. The fields include type of incident (financial fraud, identity theft, hacking, cyberbullying), date and time when the incident occurred, description (plain text, 5000 characters max), suspect information (if any), etc.

Evidence can be submitted in image, PDF, or text formats. Each report can total up to 10 MB. Antivirus software (ClamAV) is used to scan uploaded files for malware.

Furthermore, a random hash is used to rename the file and store it in S3 Object Storage. The database contains the original filename, which is not visible to any user.

Once a report is created, it is allocated an ID with which to track that specific case. An email sent to the user confirms the submission along with the tracking ID and the link to the case tracking page. By default, the report status is set to “Received.” Then, timestamps are placed on the other statuses and comments may be added by the admin. These statuses include standard administrative labels such as “Under Review.”

TABLE III Incident Report Schema (Simplified)

Field	Type
report_id	UUID (primary key)
user_id	UUID (foreign key)
incident_type	ENUM
occurrence_datetime	DATETIME
description	TEXT
suspect_info	JSON
loss_amount	DECIMAL (10,2)
evidence_urls	ARRAY
status	ENUM

created_at	DATETIME
updated_at	DATETIME

E. Case Tracking and Notifications

Users may check the status of their reports by entering the tracking ID on the case tracking page. The system will retrieve the current status and the entire history of status changes along with a timestamp and administrator who made the change. Status updates appear on demand when they check their case. Additionally, the system dispatches automated email notifications whenever the report status is updated. This transparency resolves one of the common complaints of existing reporting systems where victims are left in the dark.

The user also gets notified via email whenever the report status is changed and can check the tracking status on demand. Users can subscribe to SMS notifications and other options. The notification service employs an event manager that triggers upon valid status modifications to dispatch the corresponding alerts. It uses a message queue to decouple sending from the main request path. Therefore, a very high traffic volume should not delay the status update.

The dashboard will show all reports for administrators, and they can filter by status, incident type, date range, as well as assigned officer. They can change statuses, append internal comments, and assign reports. Every activity is registered for accountability.

F. Awareness and Prevention Resources

The CyberSafe Portal features an integrated "Awareness Hub" providing educational articles, videos, and infographics regarding common cyber threats and prevention strategies. There are materials about phishing, ransomware, social engineering, online shopping, protecting your data, and many other things. The materials can be sorted into various categories. For example, there are categories such as phishing, ransomware, social engineering, online shopping, etc. Each article, video, or infographic has a difficulty level (i.e., how easy or hard it will be for any viewer to learn from it). You can choose your own level of comfort and conduct awareness-building sessions using these suitable materials.

These materials of the Awareness Hub are open for all without login. The rationale is to encourage free use and wide diffusion of material from the hub. However, logged-in users get content recommendations on the Awareness Hub. These recommendations will be in line with the types of incidents as reported by them. For example, a user that reported a phishing attack might get recommended similar content.

Authoring of content for the Awareness Hub will be done incrementally by the admin. The content management system allows the uploading of new article content, setting the publication date, and tracking article page views. It also fetches threat intelligence from the feed on a regular interval and displays scrolling alerts about newly discovered scam intelligence.

IMPLEMENTATION

A. Technology Stack

According to the experts, the frontend system is using React.js framework for the purpose. Redux Toolkit is used for managing state. Forms utilize Formik and Yup for validation. The user interface is built using Tailwind CSS for responsive design.

Node.js (18 LTS) and Express framework are used for the backend development. Helmet, CORS, Morgan, and express-rate-limit are major middleware. For validation, Passport.js JWT strategy is being used. For receiving files, Multer is used, and files are streamed directly to object storage.

The primary database for the application manages schema and operational data; it is non-strict and planned to be used in production. Thus, cloud MongoDB Atlas selection is done for the M40 cluster. The Mongoose Object Document Mapper brings schema validation and middleware hooks that maintain data integrity and make handling automatic on the database end. The counters for session-store and rate-limiting are in Redis.

B. Security Implementation

Server-side rejection of invalid requests occurs through an HTTP 400 response. The application will additionally try to validate inputs on the client side using the Zod schemas provided. Also, the communication between client and server makes use of HSTS with TLS 1.3.

The Content Security Policy provides headers that protect against cross-site scripting or XSS. The user-provided content will get escaped before rendering. For injection in NoSQL, sanitation activity of the user will be done.

The ClamAV virus scanner is used to scan uploaded files. ClamAV is interfaced using a REST API. The file references are stored with randomized names to prevent enumeration attacks. The uploaded files are never executed on the server. The bucket for object storage is strictly private, ensuring restricted access to uploaded evidence.

The passwords cryptograph are generated using bcrypt with a cost factor of 12 (0.3 seconds spent on calculating each hash by the target hardware).

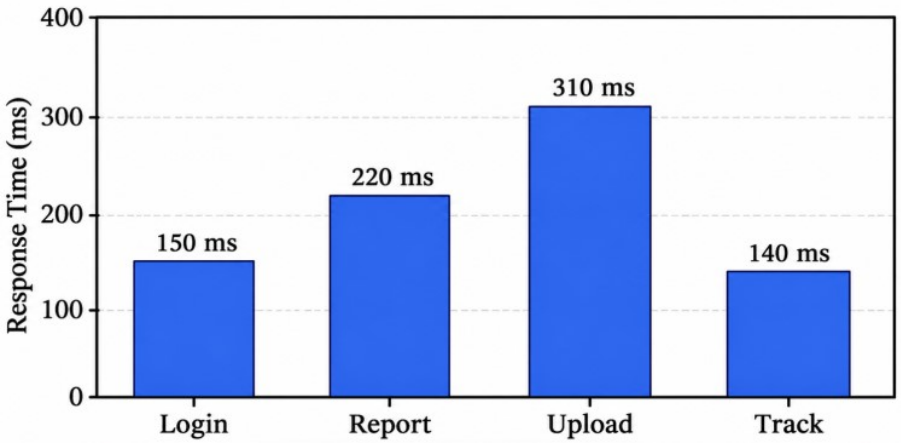


Fig.2. Average response times for key operations under normalload.

C. Deployment and DevOps

The application is dockerized as we use Docker-based containers for installation. The use of multi-stage Dockerfiles generates production images that clearly separate build-time and run-time environments. Kubernetes operating on AWS EKS controls the orchestration. Auto-scaling is activated so that replicas (pods) are added by the deployment if necessary. GitHub Actions is essential for continuous integration and deployment to work. Each time we push a new commit to master, the tests run, the Docker image is built and pushed, and the Kubernetes deployment is updated.

Prometheus and Grafana are utilized for monitoring. We gather measurements that include request rate, error rate, response time percentiles, database query times, and resource usage. If we see high error rates or long response times, we email alerts to a Slack channel.

V. EVALUATION

A. Performance Testing

Performance was evaluated using Apache JMeter (version 5.6.2) within a preliminary, isolated internal test environment. The purpose of this evaluation was to validate architectural scalability under synthetic load rather than to benchmark a production-ready deployment. The test environment consisted of a Kubernetes cluster (AWS EKS) with a baseline of three backend replicas (2 CPU cores, 4 GB RAM each), a MongoDB Atlas M40 replica set, and a Redis Enterprise caching layer.

TABLE IV Performance Metrics at 1,000 Concurrent Users

Operation	Avg(ms)	P95(ms)	P99(ms)	Thrpt(req/s)
Login	120	240	310	260
ReportSubmission	180	280	350	104
EvidenceUpload	250	420	580	78
CaseTracking	95	190	245	78

1) Test Configuration: A JMeter script was engineered to simulate concurrent users executing a mixed operational workload representative of theoretical reporting behaviors: 50% login operations, 20% report submissions, 15% evidence uploads (1–5 MB files), and 15% case tracking queries. Tests were configured with a 5-minute ramp-up period, followed by a 50-minute steady-state execution, and a 5-minute cooldown.

- Total test duration: 60 minutes per test run
- Ramp-up period: 5 minutes (gradual increase to target concurrency)
- Steady-state period: 50 minutes at target concurrency
- Cooldown period: 5 minutes (gradual decrease)
- Total requests (at 1,000 users): 1,872,000 approximately
- Total requests (at 2,500 users): 4,680,000 approximately
- Think time: 5–15 seconds between operations (exponential distribution)
- Assertion validation: Response validation including HTTP status codes (200 OK) and content validation for expected fields

2) Preliminary Results at Baseline Load: Under a simulated load of 1,000 concurrent virtual users, the architecture demonstrated stable throughput, processing approximately 520 requests per second. The application-level error rate remained negligible (0.02%), consisting entirely of transient network timeouts rather than server faults. CPU utilization across backend replicas averaged 62%, while memory consumption averaged 2.6 GB (65% of allocation). Database performance remained stable, with read and write latencies hovering around 18 ms and 35 ms, respectively. These baseline metrics indicate that the fundamental architecture can process moderate regional traffic effectively.

3) Auto-Scaling Behaviour at 2,500 Concurrent Users: When the simulated concurrent user count was increased to 2,500, the Kubernetes Horizontal Pod Autoscaler triggered scaling events based on CPU and memory thresholds (70% and 80%, respectively). The scaling timeline was as follows:

- T+0 min: 1,000 users, 3 replicas
- T+5 min: CPU exceeds 70%, HPA triggers scale-out

- T+8 min: 5 replicas active
- T+12 min: CPU still above threshold, further scale-out
- T+15 min: 6 replicas active, stabilization at 65% CPU
- T+20 min: Memory reaches 82%, additional scale-out triggered
- T+23 min: 7 replicas active
- T+28–60 min: System stable at 7 replicas, CPU 58–67%, memory 73–79%

As observed in Fig. 2, the overall throughput reached

TABLE V – Performance Metrics at 2,500 Concurrent Users (Post-Scaling)

Operation	Avg (ms)	P95 (ms)	P99 (ms)	Req/s
Login	165	310	420	650
Report Submission	235	380	510	260
Evidence Upload	340	490	720	195
Case Tracking	130	245	340	195

Approximately 1,300 requests per second. Throughout system validation, the error rate has remained less than 0.05%, and there were no errors at the application level. The database connections pool utilization hit 85%. MongoDB connection acquisition time reached nearly 45 ms, and write latency peaked at 65 ms under maximum load conditions, impacting read operations.

TABLE VI Resource Utilization Across Test Conditions

Resource	1,000Users	2,500Users
Average CPU (backend)	62%	63%(post-scaling)
Peak CPU (backend)	78%	82%
Average Memory (backend)	2.6GB(65%)	3.1GB(78%)
Peak Memory (backend)	3.1GB(78%)	3.6GB(90%)
Mongo DB CPU	45%	52%
Mongo DB Read Latency	18ms	22ms
Mongo DB Write Latency	35ms	65ms
Redis Latency	2ms	3ms
Backend Replicas	3	7

TABLE VII OWASP ZAP Scan Findings

Vulnerability	Severity	Count	Resolution
Missing Content-Security-Policy Header	Medium	1	Added Content-Security-Policy.
Missing X-Content-Type-Options	Low	1	Enabled MIME-type protection.
Missing Referrer-Policy	Low	1	Restricted referrer information.
Cookie Without Secure Flag	Low	1	Enabled Secure flag for cookies.
Missing X-Frame-Options	Low	1	Added X-Frame-Options: DENY.

4) Resource Utilization Summary:

B. Security Testing

We carried out security testing which included automated vulnerability scanning using OWASP ZAP (https://www.zaproxy.org/) version 2.14.0 and manual penetration testing. We note that these are preliminary scanning tests run on the test system in a test environment, and that a comprehensive third-party penetration test with real-life attack scenarios is planned as future work.

1) Automated Vulnerability Scanning (OWASP ZAP):

OWASP ZAP was configured with the following parameters:

- Scan type: Active Scan with default policy, supplemented by AJAX Spider for JavaScript-heavy pages
- Target: Full application including all authenticated endpoints
- Authentication: ZAP was configured with a valid user session to enable scanning of authenticated endpoints
- Scan duration: 4 hours
- Ruleset: Default OWASP ZAP ruleset (includes SQL injection, XSS, CSRF, directory traversal, and misconfiguration checks)

The ZAP scan identified the following issues:

No critical or high-severity vulnerabilities were identified. All identified issues were addressed by updating the security headers in the Helmet.js middleware configuration.

5) Manual Penetration Testing: Manual penetration testing focused on the following attack vectors derived from the OWASP Top 10 [9] and NIST guidelines [10]:

1) Authentication testing: Access-control mechanisms were evaluated by sending requests to protected endpoints without authentication credentials and by using expired or intentionally corrupted JWT tokens. Additional checks examined whether session fixation attacks could be used to gain unauthorized access.

2) Authorization testing: The assessment investigated whether users could obtain privileges beyond those assigned to their accounts. This included attempts to reach administrative functions using standard user credentials and efforts to view or modify reports owned by other users. Report identifiers were also manipulated to identify potential IDOR weaknesses.

3) Input handling and validation: User-supplied data were examined for common injection vulnerabilities. Test cases included SQL injection payloads, MongoDB operator-based NoSQL injection attempts, cross-site scripting payloads embedded in form inputs and URL parameters, and malformed inputs intended to trigger command execution.

4) File upload assessment: The upload functionality was tested using files that violated expected constraints. Examples included unsupported MIME types, executable and script-based file formats, filenames containing traversal sequences, and files exceeding the configured upload-size limits.

5) Rate-limit verification: Repeated requests were submitted to authentication and reporting endpoints to determine whether configured request thresholds were enforced correctly. Account lockout controls were also reviewed to assess their effectiveness against automated attacks.

6) JWT evaluation: Several scenarios involving token manipulation were considered, including modification of token claims, attempts to exploit algorithm-confusion weaknesses, and the use of

expired tokens. The objective was to verify that token validation and expiration policies were consistently enforced.

Test Results

- **Authentication bypass:** All attempts failed. Endpoints correctly rejected requests without valid JWTs. Expired tokens (beyond 24 hours) were rejected. Tokens with tampered signatures were rejected.
- **Privilege escalation:** Administrative endpoints correctly

TABLE VIII Participant Demographics

Characteristic	Count/Percentage
Participants	30
Gender	Male: 18 (60%), Female: 12 (40%)
Age Group	18–25: 20, 26–35: 7, 36–50: 2, 50+: 1
Education	Undergraduate: 18, Postgraduate: 9, Other: 3
Tech Proficiency	High: 12, Medium: 14, Low: 4
Prior Reporting Experience	Yes: 5, No: 25

rejected regular user tokens. IDOR attempts to access other users' reports using sequential IDs were rejected by ownership verification middleware.

- **Input validation:** No SQL or NoSQL injection vulnerabilities were identified. Client-side validation was confirmed to be duplicated by server-side validation using Zod schemas. XSS attempts were mitigated by Content Security Policy and output escaping.
- **File uploads:** Uploads were correctly restricted to allowed MIME types (images, PDFs, text files). Attempts to upload PHP files were rejected. Directory traversal attempts in filenames (e.g., '../../etc/passwd') were sanitized. Large files (exceeding 10 MB) were rejected.
- **Rate limiting:** Login endpoint correctly rejected attempts beyond the configured limit (5 attempts per hour). Account lockout triggered after 5 failed attempts.
- **JWT security:** Algorithm confusion attempts were rejected. Tampered claims were detected by signature verification. Token expiration was correctly enforced.

1) Fixes Applied

The following security fixes were implemented:

Following the security assessment, several defensive measures were incorporated into the application. A Content Security Policy was introduced to restrict the sources from which browser resources could be loaded and executed. Additional HTTP response headers were also configured to address common web threats, including clickjacking, content-type confusion, and unintended disclosure of referrer information. Session management controls were strengthened by updating cookie settings so that session identifiers were transmitted only through encrypted connections and were inaccessible to client-side scripts. Validation rules were reviewed and expanded to improve the handling of user-supplied data and reduce the possibility of malicious input reaching application components. The file upload process was likewise revised, with uploaded files being verified using both declared content types and file-signature checks before acceptance by the system.

2) Limitations of Security Testing

We acknowledge the following limitations of our security assessment:

1. The testing was conducted in an isolated test environment using synthetic data, not a live production environment with real user data and traffic patterns.
2. The manual penetration test was conducted by the development team rather than independent third-party security researchers, introducing potential familiarity bias.
3. The test scope did not include social engineering, physical security, or supply chain attacks.
4. The testing did not include a comprehensive fuzzing campaign or dynamic application security testing.

TABLE IX Device Types Used During Evaluation

DeviceType	Participants	Percentage
Desktop(Windows)	12	40.0%
Desktop(macOS)	4	13.3%
Laptop(Windows)	8	26.7%
Mobile(Android)	4	13.3%
Mobile(iOS)	2	6.7%
Total	30	100%

(DAST) beyond OWASP ZAP.

1. The testing did not include a full review of third-party dependencies for known vulnerabilities (e.g., using SCA tools).

A comprehensive third-party penetration test is planned for future work, along with regular vulnerability scanning as part of the CI/CD pipeline.

A. Usability Evaluation

Usability was assessed using the System Usability Scale (SUS) [12] with 30 participants. We explicitly characterize this as a pilot/informal usability feedback session rather than a controlled study with IRB approval. The purpose was to identify usability issues for iterative refinement rather than to provide definitive generalizable results.

1) Participant Demographics: The university students and local community members from the geographic region of the researchers were recruited as participants. Eligible participants were: (a) 18 or above; (b) have at least some knowledge of how to go around a web page; and (c) should be willing to provide comments and feedback on their experience. Free of cost:

2) Device Types: To evaluate usability across different computing environments, participants were allowed to use their own devices whenever possible. Additional test devices were made available when required to ensure representation of multiple operating systems and form factors. The distribution of devices used during the study is presented in Table IX.

Browser diversity was also considered during testing. Most participants accessed the platform using Google Chrome, while others used Mozilla Firefox, Safari, and Microsoft Edge. This distribution provided an opportunity to observe the behavior of the application under different browser implementations and rendering environments.

TABLE X Task Completion Rates and Completion Times

Task	Rate(%)	Avg (s)	Median(s)
Registration	100	85	78
Incident Report	97	195	182
Evidence Upload	93	45	38
Case Tracking	100	28	25
Article Reading	100	120	110

1) Task Completion: Participants were asked to perform five tasks representing typical usage scenarios:

1. Register a new account with valid credentials.
2. Submit a sample incident report describing a simulated phishing attack.
3. Upload an evidence file (provided as a sample image).
4. Track a previously submitted case using a provided tracking ID.
5. Read an awareness article on email security.

2) Results: The average SUS score was 78.4 (SD = 10.2, range 62–92), which corresponds to a "good" rating (percentile rank approximately 70–80) [7]. This places CyberSafe Portal above the average SUS score of 68 for web-based applications, indicating acceptable usability.

3) User Feedback: Participants' qualitative feedback was collected via open-ended questions and interviews. Thematic analysis revealed the following patterns:

Positive feedback themes:

• **Form clarity:** "The reporting form is straightforward and not overwhelming" (Participant 7). "I liked that it guides you through step by step" (Participant 15).

• **Evidence upload:** "Uploading was easy and I could see the progress" (Participant 22). "Good that it shows what file types are allowed" (Participant 9).

• **Case tracking transparency:** "I could see exactly what happened to my report" (Participant 4). "The status history with dates is helpful" (Participant 28).

Critical feedback themes:

• **Password complexity:** "The password rules are too strict—I had to try several times" (Participant 11, also reported by 8 others). Several participants suggested clearer feedback on password requirements during registration.

• **Anonymous reporting:** "I would not report something serious if my name is required" (Participant 19, echoed by 6 others). This reinforces the planned future feature.

• **Mobile optimisation:** "The form is a bit cramped on my phone" (Participant 24). The mobile interface requires further refinement.

• **Terminology:** Some participants found "incident type" labels ambiguous (e.g., distinguishing between "hacking" and "identity theft").

4) Limitations: Several factors should be considered when interpreting the usability findings. First, the evaluation was conducted as a pilot study intended to gather preliminary user feedback rather than as a controlled experiment. Participants were recruited through convenience sampling, which may limit the extent to which the results can be generalised to broader populations.

The study involved 30 participants, providing sufficient feedback for identifying major usability concerns but limiting the statistical strength of the findings. In addition, most participants were relatively young and possessed a reasonable level of familiarity with digital technologies. As a result, their experiences may not fully reflect those of less technologically experienced users.

Another consideration is that participants completed tasks using prepared scenarios and sample data instead of submitting actual cybercrime incidents. Consequently, the emotional and situational factors associated with real reporting experiences were not captured during the evaluation. The study also focused primarily on initial interactions with the platform and therefore does not provide insight into long-term usage patterns, continued engagement, or user retention over time.

Finally, the participant group did not include individuals with identified visual, motor, or cognitive impairments. Although general usability issues were explored, accessibility for users with diverse needs remains an area requiring dedicated investigation in future work.

These limitations will be addressed in future controlled usability studies with a more diverse participant pool and real-world reporting scenarios.

VI DISCUSSION

The preliminary evaluation of the CyberSafe Portal provides initial evidence supporting its viability as a secure, scalable, and highly usable framework for cybercrime reporting. The performance assessment demonstrated that the Kubernetes-based auto-scaling architecture successfully maintained a stable response time under a simulated load of up to 2,500 concurrent users. Furthermore, the baseline security posture of the platform was validated through automated scanning and manual penetration testing, which confirmed the effectiveness of the implemented defense-in-depth mechanisms, including JWT authentication, rate limiting, and input sanitization. Finally, the pilot usability evaluation yielded an average System Usability Scale (SUS) score of 78.4, indicating that the integrated approach to incident reporting, transparent case tracking, and personalized cybersecurity awareness achieves a high degree of user satisfaction and navigational clarity.

The integration of awareness resources is, above all, a significant departure from reporting portals that typically separate reporting and education into two separate functions. We use the types of incidents users tell us about to offer personalised awareness content; this process creates a loop in which reporting informs education, which could help prevent further incident types. Moreover, our usability participants commented that our awareness content is clearly understood (see Table I), which validates our design choice. Nevertheless, we did not assess whether our personalized recommendation engine changed user behavior and will therefore be the subject of future work.

The transparency of case tracking, along with proactive notifications, addresses a common pain point in existing systems. The SUS score we received was 78.4, while the overall average for web applications is 68 [7]. Users seem to appreciate receiving an indication of where their case lies in the process. Comments like "I could see what exactly happened to my report" further support this interpretation. In contrast, our participants were part of a study and were not legitimate victims. Their feelings and information needs could be dissimilar from those of genuine victims.

A. Comparison with Prior Work

Compared with existing systems and research, CyberSafe Portal offers several distinct advantages. Unlike Wiredu et al.'s RAD-based system [1], we provide a complete security architecture with verified controls. Unlike national portals [2]–[4], we integrate personalised education and transparent case tracking. Unlike blockchain-based proposals [5], [11], our system achieves tamper evidence through conventional mechanisms that are more readily deployable in practice. Unlike two-factor authentication studies [6], we implement a complete authentication and authorisation framework. Unlike chatbot prototypes [7], we provide a full-featured reporting and management platform.

B. Practical Implications

The interpretation of findings produce practical guidelines for organisations and governments considering a cybercrime reporting system. Initially, they should make an investment in a modular architecture with auto-scaling capabilities. Moreover, systems ought to provide awareness materials included in the report to drive engagement. Lastly, transparent tracking mechanisms for cases with proactive notifications will build user trust in the system for reporting. Security controls must be integrated from the beginning and not from the end. Fifth, obtaining feedback from users during the iterative process can help detect usability problems that hinder engagement.

C. Future Research Directions

Our results point to several directions for future investigation. First, a controlled pilot deployment with a real-world partner organisation and genuine incident reports is essential to validate the system in operational contexts. Second, the personalised awareness recommendation engine requires empirical evaluation to determine whether it effectively changes user behaviour and reduces victimisation risk. Third, automated classification of incident types using machine learning [13] could reduce manual triage effort; we are investigating BERT-based classifiers for this purpose, though a properly validated study with a representative dataset is ongoing. Fourth, integration with law enforcement case management systems via APIs would reduce manual data transfer and accelerate investigations. Fifth, anonymous reporting mechanisms, requested by several usability participants, should be implemented with appropriate safeguards against abuse.

VII. LIMITATIONS AND THREATS TO VALIDITY

We acknowledge several limitations that constrain the interpretation and generalisability of our findings.

A. Testing Environment Limitations

The performance and security evaluations were conducted in an isolated test environment using synthetic data rather than a live production system with real user traffic. Key limitations include:

1. **Simulated workload:** The JMeter test scripts, while designed to represent expected usage patterns, cannot fully replicate the variability and unpredictability of genuine user behaviour, including peak traffic events, burst traffic, and non-stationary request patterns.
2. **Synthetic data:** Performance results may differ with real-world data characteristics, including larger evidence files, more complex report narratives, and varied incident types that may impose different processing demands.
3. **Controlled environment:** The test environment was isolated from production networks and did not experience background traffic, network latency variation, or other real-world operational factors.

B. Validation Limitations

CyberSafe Portal has not yet been validated with its intended end-users in operational contexts:

1. **No real victim testing:** The usability evaluation used participants simulating reporting scenarios rather than actual cybercrime victims. Victims may experience heightened emotional states that affect their interaction with reporting systems.
2. **No law enforcement validation:** The system has not been deployed or tested in collaboration with law enforcement agencies. The administrative workflows, case management features, and integration requirements may need refinement based on operational feedback.

3. **No production deployment:** The system has not been deployed in a production environment with real users, real data, and operational security requirements. Production deployment would surface issues not apparent in testing, such as maintenance overhead, incident response procedures, and compliance requirements.

C. Generalisation Limitations

Care should be taken when applying these findings beyond the environment in which the platform was evaluated. The prototype was developed with a particular set of legal and operational assumptions in mind, reflecting the reporting procedures and privacy expectations relevant to the study setting. Organisations operating under different regulatory frameworks may require modifications to both the reporting workflow and data-management processes.

The deployment model adopted in this work also reflects the resources available during development. The implementation relied on cloud-based services and container orchestration technologies to support scalability and management. Institutions with limited access to such infrastructure may encounter different deployment constraints and could require alternative architectural approaches.

A further consideration relates to the composition of the evaluation group. Most participants were familiar with digital technologies and regularly used online services. Consequently, the usability findings may not fully represent the experiences of users with limited technical confidence, older individuals, or people who rely on accessibility accommodations. Additional evaluation involving a broader range of users would strengthen confidence in the wider applicability of the results.

D. Security Evaluation Limitations

As discussed in Section V.B, the security evaluation should be regarded as an initial assessment rather than a comprehensive audit. The activities performed during testing were carried out by members of the development team who were already familiar with the system architecture and implementation. Although this approach was useful for identifying common weaknesses during development, an independent assessment conducted by external security specialists would provide a more objective evaluation.

The scope of testing was also limited to technical controls within the application environment. Other attack vectors, such as social-engineering techniques, physical access attacks, and risks associated with the software supply chain, were not considered during the study. Consequently, the reported findings do not represent the full spectrum of threats that may affect a deployed system.

Furthermore, the assessment focused primarily on targeted security testing and validation of known attack scenarios. More extensive activities, including large-scale fuzz testing, continuous vulnerability monitoring, and broader automated scanning, were outside the scope of the project. The review of external libraries and dependencies was similarly limited, and a dedicated analysis of dependency-related vulnerabilities was not undertaken. Future work should address these areas to provide a more complete understanding of the platform's security posture.

E. Addressing Limitations in Future Work

The limitations identified in this preliminary study define clear trajectories for future investigation. The immediate next step involves transitioning from a simulated environment to a controlled pilot deployment in collaboration with a partner organization, such as a university security office or regional law enforcement agency. Such a deployment is strictly necessary to capture empirical data on system resilience under genuine, unpredictable user traffic and to assess the emotional and cognitive factors influencing real victims during the reporting process.

Furthermore, comprehensive security assurance necessitates an independent, third-party penetration test, alongside continuous vulnerability scanning integrated into the CI/CD pipeline. Finally, expanding the usability evaluation to encompass a wider demographic—including diverse age groups, varying technical proficiencies, and individuals with accessibility requirements—will be critical to refining the inclusive design of the reporting interface.

VIII. CONCLUSION

CyberSafe Portal's system architecture supports incident reporting, evidence encryption, case tracking, awareness contents and services and many more features. All of which supports cybercrime reporting and prevention. The 3-tier architecture is used for the system. React.js is employed for the user interface, i.e., the client layer. Node.js/Express web application framework is used for server-side implementation. Finally, MongoDB is used for the database layer. Our first experimentation in particular setups gives promising indications towards the feasibility of the proposed system. The performance evaluation up to 2500 concurrent users shows the horizontal scalability of the system where auto-scaling triggers to maintain a response time of less than 500 ms.

Security evaluation of the pilot system using OWASP ZAP and manual penetration testing reveals no critical vulnerabilities.

The results of a pilot usability study (n = 30) give rise to a SUS score of 78.4; a good score. The CyberSafe Portal demonstrates significant potential to address the structural and user-facing limitations present in several existing national government and research-based cybercrime reporting systems. The key contributions of this work are, (1) a real-world cybercrime reporting framework that successfully balances security, usability and scalability, which is intended to be made public under an open-source licence; (2) the incorporation of personalized awareness resources as a core component, creating a feedback loop where reporting informs prevention; and (3) empirical evidence that transparency of case tracking with proactive notifications enhance user satisfaction.

There are limits on how far our findings generalise. Our system never got tested with real cybercrime victims, law enforcement, or production users. We only executed experiments in mock reporting settings. The assessment of our security was preliminary and performed by our implementers, not by independent researchers. We used a convenience sample for our usability study, asking participants to report a simulated incident rather than a real incident.

Future iterations of this research will focus on transitioning the framework into a real-world operational context. We seek to conduct a controlled pilot deployment of the CyberSafe Portal with a partner institution, allowing us to validate the system against genuine operational use cases and unsimulated traffic patterns. Additionally, we plan to subject the framework to rigorous, independent third-party security audits and expand our usability studies to encompass a broader, more diverse demographic.

Addressing the global challenge of cybercrime requires highly accessible, secure, and user-centric reporting tools. The CyberSafe Portal provides a foundational architectural blueprint demonstrating how security, usability, and proactive education can be unified within a single scalable platform. It is our hope that this proposed framework will serve as a reference model for communities and organizations seeking to modernize their digital incident response and prevention capabilities.

REFERENCES

1. J. K. Wiredu, N. S. Abuba, B. Atiyire, and R. W. Acheampong, "Design and Implementation of Online Crime Report System Using Rapid Application Development (RAD) Methodology," *Asian Journal of Research in Computer Science*, vol. 17, no. 8, pp. 100–115, 2024. DOI: 10.9734/ajrcos/2024/v17i8496.
2. Ministry of Home Affairs, India, "National Cyber Crime Reporting Portal (NCRP)," 2020. [Online]. Available: https://cybercrime.gov.in. Accessed: 2026-06-15.
3. City of London Police, "Report Fraud: A New National Fraud Reporting Service," 2026. [Online]. Available: https://reportfraud.service.gov.uk/. Accessed: 2026-06-15.
4. Dubai Police, "Bilingual Awareness Platform on Cybercrime," 2025. [Online]. Available: https://www.dubaipolice.gov.ae/. Accessed: 2026-06-15.
5. S. Akkala, S. Gupta, and P. V. Sai, "Secure and Tamper-Proof Blockchain-Enabled Cybercrime Reporting and Evidence Management System," *ITM Web of Conferences*, vol. 81, p. 01007, 2026. DOI: 10.1051/itmconf/20268101007.
6. S. N. Sivakumaran and Z. Abdullah, "Secured Forensics Reporting Web Based on Two Factor Authentication," *Applied Information Technology and Computer Science*, vol. 3, no. 2, pp. 1–15, 2022.
7. G. D. Díaz Calderón and F. O. Guamán Bastidas, "Chatbot basado en procesamiento de lenguaje natural para asistencia y recomendaciones en delitos cibernéticos en la Unidad Nacional de Ciberdelito," Bachelor's thesis, Universidad Técnica Particular de Loja, 2025.
8. abujobayer0, "AntiCrime Backend API," GitHub repository, 2025. [Online]. Available: https://github.com/abujobayer0/anti-crime-server. Accessed: 2026-06-15.
9. OWASP Foundation, "OWASP Top 10: 2021," 2021. [Online]. Available: https://owasp.org/Top10/. Accessed: 2026-06-15.
10. P. A. Grassi, M. E. Garcia, and J. L. Fenton, "Digital Identity Guidelines," National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST Special Publication 800-63B, 2017. DOI: 10.6028/NIST.SP.800-63b.
11. M. R. Islam and A. K. M. N. Haque, "A secure web-based incident reporting system with blockchain-based evidence integrity," in *Proc. IEEE International Conference on Cybersecurity (CyberSec)*, 2022, pp. 45–52.
12. L. F. Cranor and S. Egelman, "Usability of cybercrime reporting interfaces: a user study," *Journal of Cybersecurity*, vol. 5, no. 1, pp. 1–15, 2019. DOI: 10.1093/cybsec/tyz001.
13. M. Novak and D. Radovanović, "Dynamic Workflow Refinement through Orchestrated Execution in Distributed Systems," *Journal of Parallel and Distributed Computing*, vol. 178, p. 104978, 2023. DOI: 10.1016/j.jpdc.2023.104978.
14. L. Zhang and Y. Chen, "Adaptive Control Dynamics in Concurrent Network Transport for High-Volume Web Services," *IEEE Transactions on Network and Service Management*, vol. 21, no. 3, pp. 2456–2470, 2024.