JIKO (Jurnal Informatika dan Komputer) Accredited KEMENDIKTI SAINSTEK, No.0173/C3/DT.05.00/2025
Vol. 8, No. 1, April 2025, pp. 38-47 p-ISSN: 2614-8897
DOI: 10.33387/jiko.v8i1.9490 e-ISSN: 2656-1948

EVALUATING POST-DIVORCE WOMEN'S AND CHILDREN'S RIGHTS FUNDING APPLICATIONS USING OWASP TOP TEN AND ISO 25010:2023

Deta Oktariani[1], Ema Utami [2*]

[1,2]Magister of Informatics, Universitas Amikom Yogyakarta
Email: [1]detaoktariani@students.amikom.ac.id, [*2]ema.u@amikom.ac.id

(Received: 23 January 2025, Revised: 17 February 2025, Accepted: 03 March 2025)

Abstract

Evaluating an information system from both performance and security aspects is crucial for anticipating and improving the quality of the information system. In collaboration with the provincial government, a high religious court developed a web-based application to support one of its services: monitor court decisions regarding alimony payments from former husbands to former wives and children in civil servants' divorce cases. This is certainly very important because before this application existed, many complaints were filed due to the non-payment of alimony. A comprehensive system evaluation is required to ensure that the application runs according to its purpose and that the data is secure. The main objective of this evaluation is to identify vulnerabilities and their mitigations, as well as to ensure that the functions in the application work as expected so that the application's goals are achieved. This study uses the ISO 25010:2023 information system standard integrated with OWASP Top Ten to evaluate its security to achieve this goal. This study uses five ISO 25010:2023 characteristics selected according to the system's goals. The results show that combining ISO 25010:2023 and OWASP Top Ten effectively comprehensively identifies vulnerabilities in the application's functions and security. Overall, the functions in the application have run as expected, although there are still several things that need to be improved to enhance the quality and secure its data.

Keywords: *IT Audit, OWASP Top Ten, ISO, Security, Information System*

**Corresponding Author: Ema Utami*

1. INTRODUCTION

The rapid pace of digital transformation has led to an increasing adoption of information systems across various sectors, especially those closely tied to public services. The High Religious Court has leveraged information technology to improve its services by developing an electronic application that monitors the execution of funding for women's and children's rights post-divorce, specifically designed for civil servants. The application was designed to function as a monitoring mechanism that ensures divorced male civil servants comply with their obligations as stipulated in the religious court's ruling in response to the numerous complaints filed with relevant authorities concerning former husbands who default on their responsibilities.

There has been a growing trend of study on information system audits, driven by the escalating incidence of cyberattacks. A study conducted by [1] leveraged a combined OWASP Testing Guide and ISO 31000 approach for information system auditing. The study concentrated on assessing security risks via the OWASP Testing Guide, whereas ISO 31000 was applied for risk mitigation purposes.

Study [2] focuses on detecting Cross-Site Scripting (XSS) vulnerabilities using OWASP Security Shepherd, while research [3] targeted SQL Injection (SQLi) attacks using OWASP ZAP. The key difference between OWASP Security Shepherd and OWASP ZAP lies in their penetration testing approaches. OWASP Security Shepherd facilitates manual testing and OWASP ZAP conducts automated audits.

From a functional perspective, studies [4], [5], [6] have conducted evaluations using ISO 25010:2011. Study [4] utilized ISO 25010:2011 security characteristics to compare the security of paid password manager applications. In contrast, study [5], [6] employed ISO 25010:2011 for evaluating application quality. However, the difference lies in the methodology used: study [5] relied solely on questionnaires as primary data, whereas study [6] utilized a combination of questionnaires, black box testing, and stress testing.

Study [7] detected the security level of two e-government websites using NIST SP 800-115 and OWASP guidelines, along with a web vulnerability scanner. The results showed numerous medium-level vulnerabilities due to server configuration errors. Study [8] combined PTES and OWASP to evaluate an academic information system, which could detect vulnerabilities fairly well but fell short in detecting human errors.

While the application aims to provide a convenient and effective solution, certain limitations and vulnerabilities must be acknowledged and addressed to minimize their impact. In addition to functionality, the security dimension is equally important to address, as risks like data breaches can erode public confidence. Conducting an IT audit is a crucial step in ensuring that performance and security operate as intended by assessing and verifying the adequacy of system management, thereby facilitating focused improvements within a continuous improvement framework.[9]

This study aims to evaluate the application's security and performance by leveraging a combined framework of the OWASP Top Ten and ISO/IEC 25010:2023 standards. It also seeks to assess the application's efficacy in facilitating the reporting and documentation of post-divorce funding for women's and children's rights. Furthermore, the study aims to investigate the integration of OWASP and ISO/IEC 25010:2023 standards to identify potential vulnerabilities in the application and provide recommendations for improving its security and performance.

Previous studies generally do the evaluation separated between security and application performance. But in this study, both aspects are evaluated equally well, so application vulnerabilities can be identified and spread all over so that they can give recommendations to enhance the application's quality and security.

2. RESEARCH METHOD

This study utilizes a combination of ISO 25050:2023 and OWASP Top Ten standards to evaluate the application's security, performance, and reliability. This combined approach aims to provide a comprehensive evaluation framework for the application, which manages sensitive and confidential data.

The OWASP Top Ten is an annual list published by the Open Web Application Security Project (OWASP). It contains the most critical vulnerabilities of web applications. This list aims to raise awareness about the major risks that can impact web applications and assist developers and security professionals in identifying and addressing significant vulnerabilities. [10] Concurrently, ISO/IEC 25010:2023, consisting of nine characteristics [11], provides criteria for evaluating software quality, including security and performance aspects. Implementing these two standards makes it possible to guarantee that the application is secure from security threats, performs optimally, and appropriate software quality standards. The study methodology is outlined as follows Figure 1.

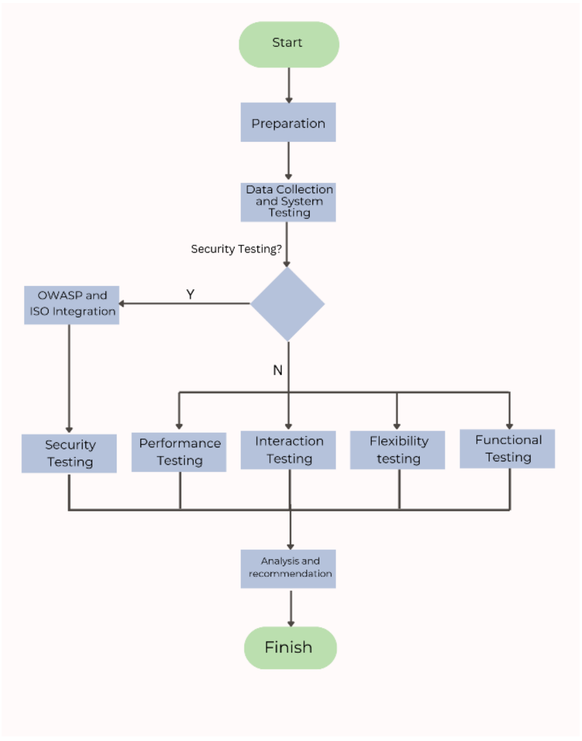


Figure 1. Research flow

2.1. Preparation

This preparatory phase aims to collect the requisite information for undertaking the study. During this phase, a literature review was conducted utilizing accredited journals from SINTA and Scopus. Furthermore, this study also examined the application's workflow and prepared the requisite hardware and software for testing purposes. The tools utilized in this study include OWASP ZAP, Apache JMeter, and PuTTY for server monitoring activities.

2.2. Data Collection and System Testing

In this data collection phase, a comprehensive evaluation of the application's performance and security was undertaken. The assessment followed the ISO 25010:2023 standard, supplemented by the OWASP Top Ten framework for security characteristics. A comprehensive testing approach comprised four distinct tests: black box functional testing, penetration testing utilizing OWASP ZAP, performance testing with Apache JMeter, and manual testing.

Table 1. Selection of ISO Characteristics

Characteristic	Sub Characteristic	Reason
Performance Efficiency	*Time behaviour, resource utilization*	To guarantee the application's ability to efficiently utilize resources and withstand high usage demands.
Functional Suitability	*Functional completeness, functional correctness*	To ensure the application meets its functional requirements and performs as expected.
Interaction Capability	*User error protection*	To ensure that the application has addressed and minimized the impact of human error
Security	*Semua karakteristik*	Due to sensitive and critical data within the application, it is paramount to ensure the data's security and safeguard it against potential threats.
Flexibility	*Adaptability, installability*	As the application requires ubiquitous accessibility, assessing whether it has optimized user access and usability across various devices is crucial.

Table 1 illustrates the chosen characteristics and sub-characteristics from ISO 25010:2023 that will be the central focus of this investigation. Five primary characteristics were selected out of the nine most pertinent to the study goals. The table also justifies selecting these characteristics and sub-characteristics, detailing their relevance to the study context.

2.3. Performance efficiency

Performance efficiency refers to the degree to which a product executes its functions within a predetermined time and throughput parameters while optimizing resource utilization under specified conditions.

a) Time behaviour evaluates the average response time and system throughput.
b) Resource utilization, assesses the system's resource usage and allocation.

Testing was conducted with load testing and stress testing with Apache JMeter while monitoring the server.

2.4. Functional Suitability

Functional suitability assesses the degree to which a product or system fulfils its intended functions and meets the specified requirements. This is achieved through scenario-based testing employing a black-box methodology.

a) Functional completeness, verifies whether the functions cover the specified objectives
b) Functional correctness, Verifies whether the system provides accurate results to the user.

2.5. *Interaction Capability*

Interaction Capability Refers to the extent to a system can effectively interact with several users to exchange information through the user interface to achieve specific goals in diverse usage contexts. Evaluation is conducted through manual testing and observational analysis of the application. The sub-characteristic of user error protection assesses the system's ability to anticipate and mitigate operational errors caused by user actions.

2.6. *Flexibility*

Flexibility assesses the application's ability to accommodate changes in its requirements, usage scenarios, or system environment, ensuring its continued relevance and effectiveness.

a) Adaptability: Evaluating the application's ability to maintain functionality and usability when accessed across various devices and browsers.
b) Installability: Evaluate the application's installability and portability across various server environments.

2.7. Security

Security assesses the application's ability to resist and defend against various attack vectors, safeguarding sensitive information and data and ensuring access is granted under users' roles, permissions, and authorization levels. The security characteristic evaluation incorporates not only the sub-characteristics outlined in ISO 25010:2023 but also integrates the OWASP Top Ten, resulting in a comprehensive set of evaluation criteria presented in Table 2.

Table 2. Security Testing Standards

Kode	OWASP Top Ten	ISO 25010:2023	Goals
P1	*A01 – Broken Access Control*	*Confidentiality Authenticity Accountability*	Verifying that application data is accessible solely to authorized users with appropriate access privileges and that all access activities are properly logged and recorded.
P2	*A02 – Cryptographic failures A08 – Software and data integrity failures*	*Integrity*	Verifying that all data is encrypted correctly to safeguard against unauthorized tampering and modification.
P3	*A03 – Injection*	*Integrity, resistance*	Verifying the application's resilience against SQL injection attacks that could potentially compromise its integrity and disrupt its functionality.
P4	*A04 – Insecure Design, A05 – Security*	*Resistance*	Validating the application's design and architecture to ensure freedom from security vulnerabilities and verifying that the server, application, and database

	Misconfiguration		configurations are secure and effectively mitigate potential threats and attacks.
P5	*A06 – Vulnerable and outdated components*	*Resistance Integrity*	Verifying that all components are updated to the latest versions mitigates potential vulnerabilities and reduces the risk of exploitation by attackers targeting outdated components.
P6	*A07 – Identification and authentication failures*	*Authenticity*	Verifying the integrity of authentication and identification mechanisms to ensure accurate and secure user verification.
P7	*A09 – Security logging and monitoring failures*	*Non-repudiation, accountability Resistance*	Verifying that comprehensive logging mechanism are in place, and that log monitoring processes effectively detect and respond to security incidents.
P8	*A010 – Server-side request forgery*	*Resistance Confidentiality*	Verifying that the server is protected against unauthorized access, tampering, and manipulation to prevent unauthorized access to sensitive resources and data

2.8. Analysis and Recommendations

This stage involves analyzing the collected data by conducting tests and interpreting the results. Finally, recommendations are compiled for improving the quality and security of the system.

3. RESULT AND DISCUSSION

Data analysis is performed after collecting data from the test results for each characteristic and sub-characteristic, providing a comprehensive understanding of the system’s performance.

3.1. Performance Efficiency

The performance efficiency testing was carried out by using Apache JMeter, concurrently monitoring server resources. The testing involved multiple experiments, progressively increasing the thread count, starting with 50 threads executing lightweight tasks, specifically HTTP requests to the two application pages.

In the first experiment, testing was conducted to determine the maximum number of threads the application could process without encountering a bottleneck. The testing was conducted five times with 50 threads, 100 threads, 150 threads, 200 threads, and 300 threads, each with two activities. The five tests ran smoothly with a maximum CPU usage of 48.5% and 400MB RAM. However, when the testing was increased to 350 threads (700 samples), the application experienced a bottleneck.

After experiencing a bottleneck, the server was restarted, and testing was conducted with a smaller number, 325 threads (considering the resource usage at 300 threads was not excessively high), resulting in a bottleneck.

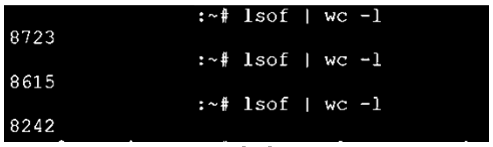
Since the usage of resources such as CPU and RAM was still within normal limits, further checks were conducted on other resources such as network and disk I/O. However, the results showed that both were still in normal condition. After further investigation, the check was extended to file descriptor usage. Upon inspection, it was found that file descriptor usage was relatively high, reaching 7596. In the subsequent testing, after the server had been idle for five hours, periodic checks were again conducted on the resources to observe whether the file descriptor caused the issue.

Table 3. Performance testing

***Sample* (*thread* x activity)**	***AVERAGE RESPONSE TIME* (MS)**	***Throughput***	**CPU**	***Memory***
200	337	3.3/s	16.6%	426M
300	333	5/s	21.2%	435M
400	75208.51	0.013/s		

The result of performance testing is shown in Table 3. The application's performance data, including average response time, throughput, CPU utilization, and memory consumption, are displayed in Table 3. Remarkably, the average response time for 200 samples was only 337 ms, and when the sample number was extended to 300, it unexpectedly dropped to 333 ms. This indicates the application’s response time stability in handling user requests. In comparison to the website tested by [12] The application performs relatively well, with an average response time of 329 ms at 100 samples, a 91% increase from 50 samples.

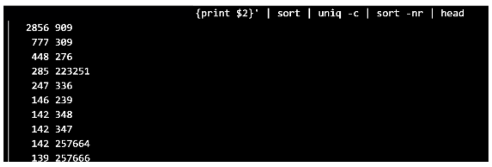
However, the average response time, which was initially relatively low, increased significantly during the test with 400 samples, indicating a bottleneck had occurred. Previously, testing could be performed up to 300 threads with 2 activities (600 samples). The CPU and RAM conditions were also normal. After further investigation, it was found that the file descriptor was high (reaching 8723), while the set limit was only 1024, as shown in Figure 2.



```
                :~# lsof | wc -l
8723
                :~# lsof | wc -l
8615
                :~# lsof | wc -l
8242
```

Figure 2. File descriptor

After examining the large number of file descriptors, a check was performed to determine which activity used the most file descriptors, shown in Figure 3.



```
                              {print $2}' | sort | uniq -c | sort -nr | head
   2856 909
    777 309
    448 276
    285 223251
    247 336
    146 239
    142 348
    142 347
    142 257664
    139 257666
```

Figure 3. File descriptor usage

After checking, the most significant activity using file descriptors was PID 909, which turned out to be Fluent Bit, responsible for log management. Therefore, the bottleneck was caused by the file descriptor resource capacity exceeding its limit, mostly utilized by Fluent Bit.

Therefore, for the sub-characteristic of resource utilization, a configuration change is necessary to maximize resource usage. Before that, it is recommended to investigate the cause of the extensive file descriptor usage even when no activities or tests are being conducted. To conserve usage, limiting the logs managed by Fluent Bit is also possible so that not all logs are recorded. Meanwhile, the application's performance is good for the sub-characteristic of time behaviour, except for the third experiment with a sample size of 400, where a bottleneck occurred.

3.2. Functional Suitability

Functional suitability testing was conducted to determine whether the functions in the application align with its intended goals and are accurate. To achieve this, goal mapping was performed, followed by black box testing. Functional suitability testing was conducted to determine whether the functions in the application align with its intended goals and are accurate. To achieve this, goal mapping was performed, followed by black box testing [13]. This is consistent with the findings of [14] users who prioritize ease of access prefer the black box method because the results are easier to understand.

Table 4. Blackbox testing

Scenario	INPUT	Expected results	Status
Case registration	Unregistered NIK	The system fails to display data, issuing a notification that the ex-husband's information has not been incorporated into the system.	Failed
Entering a verdict	Unregistered NIK	Ex-Husband's data not found in system, input cannot proceed.	Failed
Follow-up process (financing process)	Data input with incorrect format, leading to selective field emptying	Data storage unsuccessful due to formatting errors and incomplete required fields.	Failed .
Follow-up process (non-financing process)	field emptying	Notification appears stating that the file is required, and data is not saved.	Failed

Out of 37 functional scenario tests, 4 scenarios failed, while 33 scenarios were successful as expected. The four failed scenarios that failed to produce the desired results during testing are displayed in Table 4. During the case registration process, when entering the registered ex-husband's NIK, the application functions normally and displays the ex-husband's data. However, when entering an incorrect NIK, the application fails to respond and does not provide notification that the entered NIK is incorrect. Additionally, the program still does not react and display the relevant data even after the correct NIK has been entered. Even after refreshing, the application takes a long time to respond, resulting in a 504 Gateway Timeout error message. This is due to the Largest Contentful Paint (LCP) value reaching 269.14 seconds. This indicates a configuration error in the source code, which fails to anticipate human error in the case registration input. Since there are no issues with the Provincial Government Civil Servants case registration', the configuration error lies in the data source (API).

When entering a verdict, the husband's personal data should be automatically filled in (autofill) if the correct NIK/NIP is entered. When the right input is entered, the application reacts appropriately. However, when the user enters an incorrect NIK, the data can still be filled in by clicking the 'tab' button and pasting, or by clicking on the browser's autocomplete suggestions. Although the data can be saved in the application, it cannot be processed because it is considered invalid. This is probably the result of a source code problem that prevents input from being restricted in the disabled field.

During the case follow-up process (financing process), input fields requiring specific formats such as PDF, can still be filled with other formats, like PNG. Although the application restricts file selection to PDF only, a vulnerability remains when the file type is changed to 'all files' during selection. Also, this process lacks validation to prevent empty file uploads, allowing partially incomplete data to be saved.

When following up on a case (non-financing process), even when all input fields are left empty, the data is still saved and moved to the 'not followed up' menu. All of that could result in inaccurate data, which would affect the application's functionality and goal. In cases of unprocessed divorce, this will hinder updates to population documents due to incomplete data and prevent the divorce status from being monitored by the BKD (Civil Service Agency).

This user-centered testing approach, focusing on input-output, has proven effective in evaluating the application's overall functionality, and identifying non-functional features from the user's perspective. Similarly, [6] employed this method to assess several ISO characteristics, including functional suitability, security, and reliability. The test results show that black box testing is effective, achieving a score of 5 for functional suitability.

The four identified findings suggest that the developer made errors in defining these functions. Consequently, a thorough discussion and mapping of each function's objectives are required. Following this, the source code can be revised by the mapped objectives. The results of this blackbox testing can serve as a reference for mapping functional improvements.

3.3. Interaction Capability

The interaction capability testing with the sub-characteristic of user error protection was conducted by performing input/process activities and checking if there were any error messages on the application.
From the results of entering data into the system, the following conclusions can be made :

1. The "name" input field can still be filled with numbers, and the system does not warn. This, of course, results in invalid data.
2. In some menus, fields that should be disabled and not editable can still be filled by tabbing the field from the previous one and pasting data. Although the data cannot be typed manually, it can still be entered through copy-paste. There is no warning from the system, and data can still be entered but not processed. This greatly affects the data, as integrated data from other systems should have automatically filled it. If entered manually, it will cause data inconsistencies.
3. There is already a warning for mobile phone number, NIK (National Identity Number), and NIP (Employee Identity Number) inputs, which must be in numeric format and meet the minimum digit requirements (mobile phone number 11 digits, NIK 16 digits, and NIP 18 digits), and cannot be entered with non-numeric characters.
4. There is already a notification for verification to confirm whether to perform a process (button) such as data processing, data deletion, and data acceptance. This is to anticipate user errors in case a button is accidentally pressed so that the process is not automatically executed.
5. The type of file to be uploaded has been filtered according to the needs (the file extension, when attaching, automatically adjusts to the needs). However, when the user changes the extension to "all files" and selects another file with a different extension, the system accepts the file without warning. Although such cases may be rare, it does not rule out the possibility that the data becomes incompatible with the requirements.

3.4. Flexibility

The first testing of flexibility characteristics was conducted by testing the application's responsiveness on various devices and different browsers to test adaptability characteristics. The following are the test results.

Table 5. Flexibility testing

Devices	Browser	OS	Results
Asus vivobook K6501Z	Chrome	Win 11	All features and UI appear normal
P5	Edge	Win 11	All features and UI appear normal
Lenovo ThinkPad	Chrome	Win 8.1	All features appear normal, UI slightly distorted (1 element)
Samsung A52s	Chrome	Android	All features appear normal, but UI is slightly distorted (5 elements)
iPhone 14 PM	Chrome	iOs	All features appear normal, but UI is slightly distorted (3 elements)
Asus Zenfone M2	Chrome	Android	All features appear normal, but UI is slightly distorted (5 elements)
iPhone 13	Safari	iOs	All features appear normal, UI is slightly distorted (5 elements)
Macbook air	Safari	macOS	All features and UI appear normal

As shown in Table 5, all features function properly on various devices and browsers. However, regarding display, the website is not yet fully responsive for smaller screens, particularly those under 13 inches, especially on mobile phones. It is apparent from the table that browser and OS do not affect the application, but an impact occurs when there is a significant difference in screen size. Although all functions can run smoothly, UI/UX remains crucial as it also affects the ease of use for users when interacting with the application. The study evidences this [15] which shows that UX design elements, such as intuitive layout, easy navigation, and application responsiveness, significantly influence users ease of use when interacting with the application. This indicates the need for improvement to make the application responsive on all devices, making it easier for users.

To evaluate the installability subcharacteristic, the application was installed on three servers: localhost, cloud-CPU-15, and cloud VPS.

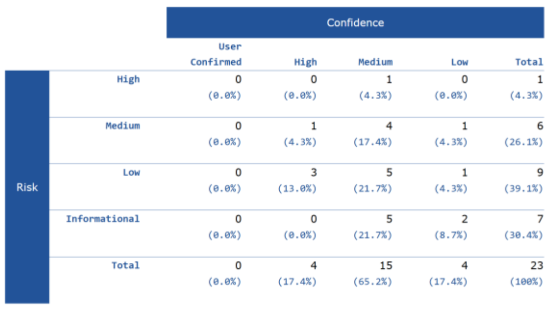
Table 6. Installability testing

Server	WEBSERVER	Database	Results
Localhost	Apache 2.4.46	10.4.18-MariaDB	Sucess
cloud-cpu-15	linux	10.3.39-MariaDB-cll-lve	Sucess
VPS	Nginx	0.013/s	Sucess
Localhost	Apache 2.4.46	10.4.18-MariaDB	Sucess

From the test results in Table 6, the application can be successfully installed on various types of servers but with configuration changes. Of the three servers, the longest installation time was when installing to VPS due to the numerous configurations and third-party tools needed, such as changing the PHP version, database management with additional tools, and a GitLab connection to the server. The importance of a website that can be applied to various types of servers is so that it can choose a server with the best performance, as done by [16] who compared the performance of various web servers. The results showed significant differences in throughput and scalability among the six web servers tested. Knowing how to find a web server with the best performance is undoubtedly essential.

3.5. Security

OWASP ZAP and manual auditing were used for security testing, with reference to ISO 25010:2023 and the OWASP Top Ten. Study conducted by [17] shows that OWASP-ZAP has a high detection rate among open-source tools, although Acunetix and NetSparker show lower false positive rates. Study [18] comparing Burp Suite and OWASP also shows that OWASP has a higher medium confidence level than Burp Suite. Therefore, testing using OWASP ZAP was performed using manual scanning by opening each website one by one using Chrome. A spider attack was also conducted to identify endpoints requiring more in-depth testing and active attacks per endpoint. From OWASP ZAP, 23 alerts were obtained, consisting of 1 high-severity alert, 6 medium-severity alerts, 9 low-severity alerts, and 7 informational alerts.



		Confidence				
		User Confirmed	High	Medium	Low	Total
Risk	High	0 (0.0%)	0 (0.0%)	1 (4.3%)	0 (0.0%)	1 (4.3%)
	Medium	0 (0.0%)	1 (4.3%)	4 (17.4%)	1 (4.3%)	6 (26.1%)
	Low	0 (0.0%)	3 (13.0%)	5 (21.7%)	1 (4.3%)	9 (39.1%)
	Informational	0 (0.0%)	0 (0.0%)	5 (21.7%)	2 (8.7%)	7 (30.4%)
	Total	0 (0.0%)	4 (17.4%)	15 (65.2%)	4 (17.4%)	23 (100%)

Figure 4. Penetration testing with OWASP ZAP

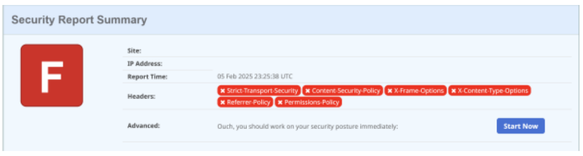
As shown in Figure 4, the confidence level of the 23 alerts is 4 high, 15 medium, and 4 low. However, only alerts with low, medium, and high risks are considered in this discussion, excluding informational findings, so only 16 alerts are discussed.

Besides testing with OWASP ZAP, manual testing was also conducted referring to standards (P1-P8), resulting in 4 high-risk findings. The findings are presented in Table 7

Table 7. Security findings

Code	Findings	Risk	Confidence	Total
P3	*Cross-site scripting*	High	Medium	1
P2, P5	*Weak Cryptographic Hash Algorithm*	High	-	1
P7	*Lack of Logging and Monitoring*	High	-	1
P6, P7	*Lack of Bot and Brute Force Protection*	High	-	1
P6	*Weak passwords*	High	-	1
P1	*Absence of Anti-CSRF Tokens*	Medium	Low	30
P7	*Application Error Disclosure*	Medium	Medium	4
P4, P3	*Content Security Policy (CSP) Header Not Set*	Medium	High	77
P4, P1	*Cross-Domain misconfiguration*	Medium	Medium	26
P4	*Missing anti-clickjacking header*	Medium	Medium	46
P5	*Vulnerable JS Library*	Medium	Medium	5
P7	*Application error disclosure*	Low	Medium	1
P4	*Cookie without secure flag*	Low	Medium	15
P4	*Cross-domain javascript source file inclusion*	Low	Medium	32
P7	*Information Disclosure - Debug Error Messages*	Low	Medium	5
P4	*Server Leaks Version Information via "Server" HTTP Response Header Field*	Low	High	142
P4	*Strict-Transport-Security Disabled*	Low	High	21
P4	*Strict-Transport-Security Header Not Set*	Low	High	147
P6	*Timestamp Disclosure – Unix*	Low	Low	10
P4	*X-Content-Type-Options Header Missing*	Low	Medium	97

Study conducted by [7] concluded that OWASP-ZAP has a high detection rate among open-source tools but a higher false positive rate than Acunetix and NetSparker. To address this, each finding detected by OWASP ZAP undergoes re-examination, both manually and through source code review, and using other specific tools related to the finding. For example, for findings related to unset headers, Online resources like https://securityheaders.com/, for instance, were used to verify discoveries pertaining to unset headers, and it was established that the results ZAP had found were correct. (shown in figure 5)



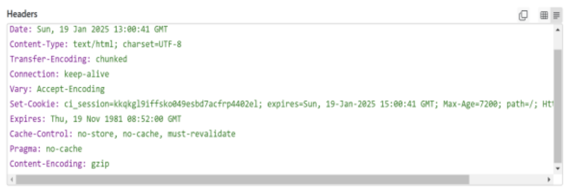
Security Report Summary

F

Site:	
IP Address:	
Report Time:	05 Feb 2025 23:25:38 UTC
Headers:	✖ Strict-Transport-Security ✖ Content-Security-Policy ✖ X-Frame-Options ✖ X-Content-Type-Options ✖ Referrer-Policy ✖ Permissions-Policy
Advanced:	Ouch, you should work on your security posture immediately:

Figure 5. Re-testing of Unset Header Findings

Furthermore, Figure 6 shows that verification was performed using online curl tools regarding server leaks of version information and timestamp disclosure.



```
Date: Sun, 19 Jan 2025 13:00:41 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
Vary: Accept-Encoding
Set-Cookie: ci_session=kkqkgl9iffsko049esbd7acfrp4402el; expires=Sun, 19-Jan-2025 15:00:41 GMT; Max-Age=7200; path=/; Ht
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Encoding: gzip
```

Figure 6. Re-testing with cURL

Out of 20 findings, there was one finding, Strict-Transport-Security Disabled, was indeed true but unrelated to the application. This finding was related to a third-party application that served only as a plugin in the application and could not be managed by the developers. Access to the third-party application also uses HTTPS, ensuring its security. Therefore, ZAP false positive is only 6.25 because just one from 16 alerts detected by ZAP was inaccurate.

Additionally, one finding, application error disclosure, was mentioned twice, with different risk levels. This was due to the difference in the location of the findings, where the medium-risk finding was located in highly sensitive data. Both results were deemed valid after being manually reexamined and subjected to a source code review. Therefore, minus one false-positive result and two duplicate discoveries, there were 18 real security findings out of 20.

Out of these 18 findings, the most common was the discovery of Cross-Site Scripting (XSS) vulnerabilities (6 out of 18 findings) due to the absence of anti-CSRF tokens, unvalidated user input, calling JavaScript files from third-party URLs, lack of Content Security Policy (CSP) and X-Content-Type-Options headers, and the use of outdated JavaScript libraries. XSS (Cross-Site Scripting) is the biggest threat to website application security as it can steal data, hijack user sessions, and compromise system integrity [19].

Configuration errors, such as unset headers, inactive secure flags, and the disclosure of sensitive information due to inadequate access restrictions, were also frequently found. This indicates that even small configuration mistakes can significantly impact the vulnerability of an application.

The vulnerability of user passwords, which are supported by antiquated or inadequate hashing methods, is another significant problem that jeopardizes system security. The application's lack of logging and monitoring features also makes it unable to identify malicious activity taking place on the system.

Several measures need to be taken to address the 18 security findings, tailored to each specific finding. To mitigate cross-site scripting vulnerabilities, input validation should be implemented to restrict characters that are not allowed, preventing attackers from injecting malicious scripts. Outdated hash algorithms and JavaScript libraries should be replaced with newer versions to enhance application security. Several findings caused by the lack of headers, such as CSP, HSTS, X-Content-Type-Options, and X-Frame-Options, should be addressed by adding these headers. Additionally, other security-enhancing headers should also be implemented to strengthen application security further.

Application information that could attract attackers and facilitate attacks, if displayed, should be hidden. Furthermore, for weak passwords, it is necessary to conduct socialization or warn users to change their passwords periodically.

This study successfully evaluated the effectiveness of the functionality and security of the application. The research results show that the application generally performs well, proven to respond to numerous requests at once. However, server configuration changes are necessary, suspected to be related to file descriptors, to prevent bottlenecks. Nevertheless, during normal daily activities, the application performs stably.

Functionally, the application meets the standards to achieve its purpose, which is to monitor alimony payments by former husbands who are civil servants. However, some functions fail to address errors caused by human error. Therefore, it is necessary to re-adjust the input validation performed by users to prevent compromising data integrity.

This application is sufficiently flexible to be used on various platforms, as evidenced by the fact that all its features function properly on different devices and browsers. However, the application's responsiveness is not yet optimal on small screens. A responsive UI/UX is essential to enhance the user experience, making it easier to utilize the application.

In installability testing, the application can be installed on various servers (localhost, cloud-cpu-15, and VPS) with some configuration adjustments, although installation on VPS takes longer due to additional configuration requirements.

This combination was also employed in research [1], which combined the Owasp Testing Guide and ISO 31000. The difference lies in the fact that the Owasp Testing Guide was used to evaluate the system, whereas ISO 31000 was focused on mitigating vulnerabilities.

This study excluded user's feedback while evaluating the application. Future research can take user's feedback to support the evaluation, so the application can improve more. It can be facilitated by using a combination of user-centered standards, which will ultimately guide efforts to enhance quality and develop applications.

4. CONCLUSION

The combination of ISO 25010:2023 standards and OWASP Top Ten can broaden the scope of evaluation against the effectiveness and security functions of the application. The use of tools such as JMeter and OWASP ZAP greatly aids the audit process, but manual audits are still necessary for more in-depth examination. Using OWASP ZAP for penetration testing is highly beneficial in detecting vulnerabilities and taking steps for mitigation. The selection of characteristics and sub-characteristics should be aligned with the audit's objectives, and characteristics or sub-characteristics that may lead to repetitive findings should be avoided. Integrating OWASP Top Ten and security characteristics in ISO 25010:2023 facilitates the identification of findings.

The result of this evaluation is that the application's functions generally work well and align with the expected goals. However, the restrictions implemented to prevent invalid data due to human error in several scenarios are not well-configured. Furthermore, greater attention is needed regarding server configuration, which is vulnerable to bottlenecks when handling many requests or activities simultaneously.

Another crucial aspect is that the data security of this application needs to be improved, given the numerous security findings discovered. There are 27 findings, although 7 of them are merely informational. As many as five high-risk findings must be prioritized for immediate attention, considering the importance of the integrity of the stored data.

The high confidence level from the penetration testing results using OWASP ZAP was only 17.39%, the same as the low confidence level. Meanwhile, the medium confidence level reached 65.21%. To avoid false positives from the findings, manually and with other tools that are more specific to those findings, re-examining is necessary. After re-examination was conducted manually and with other tools, only one of the 16 findings (excluding informational categories) was found to be incorrect. Therefore, the false positive rate of OWASP ZAP was only 6.25%.

5. REFERENCES

[1] A. A. B. A. Wiradarm and G. M. A. Sasmit, “IT Risk Management Based on ISO 31000 and OWASP Framework using OSINT at the Information Gathering Stage (Case Study: X Company),” *International Journal of Computer Network and Information Security*, vol. 11, no. 12, 2019, doi: 10.5815/ijcnis.2019.12.03.

[2] R. M. Wibowo and A. Sulaksono, “Web Vulnerability Through Cross Site Scripting (XSS) Detection with OWASP Security Shepherd,” *Indonesian Journal of Information Systems*, 2021, doi: 10.24002/ijis.v3i2.4192.

[3] I. Riadi, A. Fadlil, and M. A. Mu’min, “OWASP Framework-based Network Forensics to Analyze the SQLi Attacks on Web Servers,” *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 22, no. 3, 2023, doi: 10.30812/matrik.v22i3.3018.

[4] W. Y. Aditama, I. R. Hikmah, and D. F. Priambodo, “Analisis Komparatif Keamanan Aplikasi Pengelola Kata Sandi Berbayar Lastpass, 1Password, dan Keeper Berdasarkan ISO/IEC 25010,” *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 10, no. 4, 2023, doi: 10.25126/jtiik.20231036544.

[5] B. I. Rumabar and E. Maria, “Evaluasi Kualitas Shopeepay Menggunakan ISO/IEC 25010,” *Jurnal Sistem Informasi Bisnis*, vol. 14, no. 1, 2024, doi: 10.21456/vol14iss1pp54-61.

[6] A. A. Pratama and A. B. Mutiara, “Software Quality Analysis for Halodoc Application using ISO 25010:2011,” *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 8, 2021, doi: 10.14569/IJACSA.2021.0120844.

[7] E. Z. Darojat, E. Sediyono, and I. Sembiring, “Vulnerability Assessment Website E-Government dengan NIST SP 800-115 dan OWASP Menggunakan Web Vulnerability Scanner,” *JURNAL SISTEM INFORMASI BISNIS*, vol. 12, no. 1, 2022, doi: 10.21456/vol12iss1pp36-44.

[8] F. Putra Utama, R. Muhamad, and H. Nurhadi, “Uncovering the Risk of Academic Information System Vulnerability through PTES and OWASP Method,” *CommIT Journal*, vol. 18, no. 1, 2024, doi: https://doi.org/10.21512/commit.v18i1.9384.

[9] R. Sarno, *Audit sistem dan teknologi informasi*. Surabaya: ITSPress, 2009.

[10] OWASP, “OWASP Top Ten | OWASP Foundation,” https://owasp.org/www-project-top-ten/. [Accessed 6 Agustus 2024].

[11] ISO, “ISO 25010,” https://iso25000.com/index.php/en/iso-25000-standards/iso-25010. [Accessed 5 Agustus 2024]

[12] Indrianto, “PERFORMANCE TESTING ON WEB INFORMATION SYSTEM USING APACHE JMETER AND BLAZEMETER,” *Jurnal Ilmiah Ilmu Terapan Universitas Jambi*, vol. 7, no. 2, 2023, doi: 10.22437/jiituj.v7i2.28440.

[13] P. Ammann and J. Offutt, *Introduction to Software Testing 2nd Edition*. New York: Cambridge University Press, 2017.

[14] K. Hartwig and C. Reuter, “Nudging users towards better security decisions in password creation using whitebox-based multidimensional visualisations,” *Behaviour and Information Technology*, vol. 41, no. 7, 2022, doi: 10.1080/0144929X.2021.1876167.

[15] M. Satya Fadzana and D. Agus Diartono, “Pengaruh User Experience (UX) Design Terhadap Kemudahan Pengguna dalam Menggunakan Aplikasi TIX ID,” *Jurnal Teknologi Informasi dan Komunikasi)*, vol. 8, no. 3, p. 2024, 2024, doi: 10.35870/jti.

[16] S. Pragestu, H. Sujaini, and E. F. Ripanti, “Analisis Skalabilitas Web Server Apache Tomcat, Node.Js Dan Go Pada Protokol Hypertext Transfer Protocol (HTTP) Dan Message Queue Telemetry Transport (MQTT),” *Jurnal Sistem dan Teknologi Informasi (JustIN)*, vol. 11, no. 4, p. 605, Oct. 2023, doi: 10.26418/justin.v11i4.71607.

[17] J. Shahid, M. K. Hameed, I. T. Javed, K. N. Qureshi, M. Ali, and N. Crespi, “A Comparative Study of Web Application Security Parameters: Current Trends and Future Directions,” *Applied*

Sciences (Switzerland), vol. 12, no. 8, 2022, doi: 10.3390/app12084077.

[18] P. Jarupunphol, S. Seatun, and W. Buathong, "Measuring Vulnerability Assessment Tools' Performance on the University Web Application," *Pertanika J Sci Technol*, vol. 31, no. 6, pp. 2973–2993, Oct. 2023, doi: 10.47836/pjst.31.6.19.

[19] J. Grossman, R. Hansen, P. D. Petkov, A. Rager, and S. Fogie, *XSS attacks: Cross site scripting exploits and defense*. 2007. doi: 10.1016/B978-1-59749-154-9.X5000-8.