Computer Fraud and Security

ISSN (online): 1873-7056

Secure Front-End Architecture Patterns for Accessibility-Critical Applications

Ashok Vootla

Senior software Engineer

ABSTRACT: This study offers safe front-end architecture frameworks that are utilized in applications of high accessibility requirements. The research is concerned with the combination of cybersecurity and compliance with accessibility in terms of secure DOM management, tokenized sessions, and event controls that are defined by ARIA. A quantitative experiment on 50 web modules revealed no cases of XSS and 38 percent increase in the level of accessibility post-implementation. The results of the findings indicate that accessibility features that are appropriately managed can improve security instead of weakening it. The suggested framework can be helpful to make front-end design practices safer, more inclusive and in compliance with regulations in the areas of healthcare, finance, and education where the trust of users and the safety of the data are paramount.

KEYWORDS: Front-End, Security, Architecture, AI

I. INTRODUCTION

The current web applications are faced with the challenge of being highly secure and at the same time allowing different users to use. All platforms that are of importance in accessibility like healthcare, financial portals, etc. require frameworks of design that are secure but inclusive. Classical front-end designs tend to trade either functionality or security resulting in security holes or failure to be accessible. This research paper suggests a single framework made up of safe DOM designs, session tokenization, and control mechanisms using ARIA. This is aimed at developing interfaces that eliminate cross-site scripting and injection attacks and do not interrupt assistive technology performance. In order to comply with testing the reliability and adherence of these secure architecture patterns, the research employs quantitative testing.

II. RELATED WORKS

Secure Front-End

The development of front-end technologies has brought about the necessity of more complicated architecture which, however, is not going to consider user experience and functionality but the two, which may often be contradictory, including security and accessibility. As the principles of accessibility along with the W3C and the regulations of the Americans with Disabilities Act (ADA) and Section 508 are developed, the developers are supposed to strive toward ensuring an inclusive design and, simultaneously, adhere to such standards of cybersecurity as HIPAA and PCI DSS. Ensuring that the changes to the accessibility, particularly the ones arising with the addition of ARIA attributes and client-side JavaScript, will not introduce new vulnerabilities, including the DOM-based Cross-Site Scripting (XSS), is hard to make sure.

Recent research on dynamic type of detection suggests that such XMLSS as DOM-based XSS is a widespread concept of vulnerability in the current web environment, which is typically a by-product of insufficient input sanitizing and haphazard use of the Document Object Model [8].

These vulnerabilities have prompted the development of such tools as Trusted Types or frameworks such as TrustyMon that can be applied to overcome them without necessarily having to undertake the wholesale refactoring of web applications. The accessibility APIs combined with secure DOM management can thus become an important research issue of interest of accessibility critical applications, such as healthcare portal and government information systems.

The idea of software development connected to accessibility has been familiarized with low-code development and model-driven development approaches. Their graphic-intensive features though are restricted to the sightless users. Therefore, the front-end models should be applied to the security-oriented design patterns, such as tokenized user sessions and uninsecure ARIA event bindings, in order to allow inclusive as well as secure digital interaction.

Security-Centric Architectures

The front-end architecture forms the basis of the interaction of the client side whereby there is co-existence of visual accessibility software and security systems. The security-related literature also refers to the need to utilize the model of context validation and token isolation along with the decentralized authentication model to reduce the degree of exposure to the injection or session hijacking attacks.

The traditional Role-Based Access Control (RBAC) models fail on the dynamically distributed ecosystems and thus the transition to Zero Trust and Attribute-Based Access Control (ABAC) models which dynamically fluctuate permissions as

per contextual indicators [7]. The adaptive system enables such validations of user action with continuous verification mechanism of the action when compared to fixed authentication session which can only be done with front end applications.

Owing to credentials centralization, it is possible to still apply offline dictionary attacks to the authentication environment [4]. Two server or split-verification schemes can be implemented and improve identity assurance with plans to verify and tokenize front-end. Moreover, decentralized architectures that recreate Web3 authentication lack points of failure since privacy and control systems of user identity is upheld using identity systems that are blockchain-based [6].

Any application that contains sensitive data must employ the security front-end architecture especially in the medical field [5]. The introduction of blockchain technology is implemented to make medical data exchange safer in a manner that the possibility to exchange data between patients and providers would be tracked and the privacy of the process would be guaranteed. This can be used to add such cryptographically verified data streams as available front-ends to enhance medical application usability and trust.

Light blockchain can be relevant in securing distributed systems that are accorded eminence in the context of IoT oriented studies [1]. Their concepts of immutability, auditability and decentralised verification although normally used at the device-level can be implemented to affect the concept of secure UI communications protocols whereby front-end data transmissions are incentivized with an immutable record that ensures that manipulation and replay attacks cannot occur.

Model-Driven Secure UI Design

The issues of accessibility in front-end systems are associated with the adherence to the standards, including WCAG 2.1, and the usage of ARIA (Accessible Rich Internet Applications) specifications. Numerous accessibility features, particularly those that are dynamic/event-binding and also those that interact with the DOM, represent potential security blind spots. Unintentional leaks Leakage of sensitive DOM elements due to inaccessible or insecure front-end designs or event-based injections due to ARIA misuse.

Model-driven development systems, the topic of low-code accessibility researchers discuss in [3], have a potential in engaging non-technical stakeholders in software development. However, the structures need to be well integrated in terms of security. By integrating security-by-design patterns into low-code visual editors, the accessibility of labels and roles are checked as well as live regions to be semantically accurate and to be injection safe. Such systems can overcome script less attack vectors by introducing pre-build, secure ARIA event templates and applying compile-time checks on the check of the context of the scripts [10].

The Trusted Type Enforcements [8] could be used to enhance the principle of separation of concerns, i.e., separation of data, logic, and presentation layers. This would help to target all the dynamically injected content through a trust validation mechanism to eliminate the execution of malicious code.

Context-auditor, which addresses the case of context switch between parsing, is a framework that detects vulnerabilities in script injections, HTML and CSS based infiltration as well [10]. These tools add reliability to ARIA event propagation and safe input as well as preserve the accessibility of the user and the safety of the input when incorporated at the front-end layer.

In the usability aspect, adding risk-adaptive front-end authentication systems are useful in the balance of accessibility and contextual verification [7]. As an example, assistive users can be speech or touch-sensitive, which might result in unusual patterns of sessions. AI-based adaptive authentication can separate normal deviations and possible threats without causing any hindrance.

Privacy and Tokenization Strategies

The token-based interaction model models had now become a trend in the session security and access control sector as the changes towards the decentralized applications (DApps) progressed. The papers on the security of DApps indicate the systemic potentiality of the vulnerabilities of the distributed environments to be reduced through the systematic use of methods of design to security objectives to architecture layers [2].

This kind of the same attitude towards the Web systems of critical accessibility can result in a development of reusable and safe templates of the front-end design. The given patterns may be divided into the modular designs containing user session control, DOM isolation, and accessibility features which are organized with the ARIA.

It is also in case of privacy preserving interaction whereby the interaction is based on tokenization. Blockchain designs privacy sensitive like ARTeX as an indicator can show how transactions of real-life assets can be kept anonymous even though blockchain transactions are transparent [9].

Through the application of the concept to the web session management, the developers will have the potential to create tokenized access layers in order to address the sensitive identifiers of the users without compromising auditability in order to promote compliance.

Healthcare-based blockchain solutions are one of the opportunities of combining security with the accessibility of the data [5]. The tokenized exchange models provide the option of exchanging data finitely i.e. patients can provide access to medical data on-demand and encrypt it across the entire system. One can customize them to the front-end architecture, where they can enforce HIPAA-conforming UI communication, and accessibility aids (e.g., screen readers or voice inputs) may be implemented in the setup of an isolated and trusted setting.

The front-end systems can reduce the exposure of sensitive ARIA attributes to unauthorized scripts by incorporating session level cryptographic tokens that will not have any interference with the process of accessibility. It is the nonce based policy and the Trusted Type enforcement model, which is one of the defense in depth measures against the DOM injection [8], [10]. This integration will assure insecurity of the front-end interfaces to all as well as assure complete regulatory compliance.

Future Directions

Despite the serious progress in the area of accessibility and cybersecurity research, their intersection has not been extensively studied yet. Secure authentication Studies on multi-server models [4] and decentralized architectures [6] show that it is indeed possible to minimize the points of failure, but seldom take into account the implications of accessibility of such designs.

Although blockchain-based healthcare systems [5] and IoT security frameworks [1] emphasize the possible ways of privacy preservation and secure communication, limited of them include an accessible UI integration. The findings by researchers on the security of DApps [2] and the DOM-based anti-XSS [8][10] is highly informative about the design patterns that can be reused to formalize the front-end accessibility systems.

It persists that there is a need to have a single framework which incorporates these secure design patterns directly into the available UI workflows. Furthermore, adaptive identity models [7] and model-driven accessibility studies [3] indicate that automation can be involved in contributing greatly to maintain the balance between compliance, usability, and threat reduction.

Further studies ought to craft cross-domain architectural prototypes of accessibility-sensitive systems, so that secure DOM manipulation, session management with tokens and ARIA-driven event management executions be conducted with the identical threat-conscious design rationale.

III. METHODOLOGY

The study is based on a quantitative experimental study design where the researcher analyzes the secure front-end architecture patterns that both address the accessibility compliance and the cybersecurity standards including the HIPAA and the PCI DSS. The objective is to test the effect of particular design practices, namely, secure management of the DOM, tokenized user-sessions, and event management based on ARIA, on the system security and usability in case of applications with high accessibility requirements.

Research Design

The experimental design adopted in the study was the controlled experiment that involved two application prototypes:

- **Model A:** An unsecured front-end that does not include protection-oriented patterns and is built on generic frameworks (React and Vanilla JavaScript).
- **Model B:** A more robust front-end design with implementation of the suggested secure design patterns such as Trusted Types implementation, tokenized session control and validated ARIA event binding.

These two prototypes used the same functionality and user interfaces so as to be fairly compared. The environments under which the tests were done were all standardized through the use of the same cloud infrastructure, browser settings, and accessibility test tools.

Data Collection

There were two perspectives of data collection:

1. **Security Performance Metrics**, and
2. **Accessibility Usability Metrics.**

(a) Security Performance Data: Security testing and manual penetration testing methods were used by applying automated vulnerability testing tools such as OWASP ZAP. There were three important quantitative metrics that were taken:

- XSS / injection vulnerabilities identified during the implementation of XSS / injection vulnerabilities.

- Security layers with latency of response (ms).
- Accuracy of token validation of user sessions (%).

All prototypes were tested repeatedly (n=30) in terms of simulated attack conditions, such as script injection and session hijacking attacks. Measuring the improvement of resilience was done through statistical analysis of mean, variance, and standard deviation.

(b) Accessibility Usability Data: Performance of accessibility was through the compliance with WCAG 2.1 guidelines. Quantitative scores of were generated using automated tests with Lighthouse and Axe-core:

- ARIA validation success rate (%).
- Screen reader interaction completion time (seconds).
- Keyboard navigation accuracy (%).

Usability trials were done on 20 users with various assistive technology set ups. Every respondent was required to perform four tasks on navigation and the data of interaction were recorded and examined on average completion time and error rates.

Experimental Implementation

The experiments are carried out under regulated cloud-based environment using containerized executions (Docker) to make sure that the deployment would be the same. Both the A and the B models were coupled with a simulated backend API that has encrypted data flow in order to eliminate security impact on front end.

The temporary tokens on the basis of the accessibility sessions were implemented under the example of the tokenization layer to replace the JSON Web Tokens (JWT). The Trusted Types policy was applied to the entire DOM tree with the aim of limiting the dynamic content injection. A secure event binding library was used to test ARIA event handlers and it is a library that makes context checks before activation.

Therefore, a quantitative measurement of the scores assessed with respect to security and accessibility was used within every phase of the implementation and the scores were compared to the rest of the available models to determine ratios of the improvements.

Data Analysis

All the obtained data were examined with descriptive and inferential statistics. Mean and standard deviation were calculated to assess consistency and paired t-tests were used to compare the performance of the Model A and B. The statistical level of confidence was 95 specific ($p < 0.05$). The visualization of the results was done in the form of comparative tables and combo charts, where the decrease in the number of vulnerabilities and the change in the duration of accessibility responses was presented.

Ethical Considerations

The study does not use actual patient or financial data, since the data used are fake, therefore there was no personal data that was used. Nonetheless, testing was performed as a simulated HIPAA and PCI DSS in all tests to enable a real compliance scenario. The accessibility testing was made according to the guidelines of WCAG 2.1 and ARIA 1.2 to guarantee inclusion design accessibility examination.

IV. RESULTS

Overview of Experimental Findings

The objectives of the experiments were to estimate the effectiveness of secure front-end design patterns in enhancing web systems not only in the accessibility but also in security where sensitive data is exchanged. Results that were made compared two front-end models, Model A (Standard Design) and Model B (Secure Pattern Design) within four primary areas; vulnerability mitigation, system performance, accessibility compliance and effectiveness of user interaction.

The improved architecture in the Model B exhibited high improvements in all measures. Experiments using quantitative analysis proved the adoption of Trusted Types, tokenized user sessions, and validated ARIA event controls could greatly decrease security threats without a negative impact on accessibility performance. It was also statistically proved that the majority of the difference between the two models was significant on the 95% level ($p < 0.05$).

The most significant outcomes were 100 percent of eliminating the XSS vulnerabilities, 32 percent increase in accuracy of ARIA validation and 26 percent decrease in the time taken by screen readers to interact. These findings are a clear indication that security patterns can be implemented in ways that promote accessibility and not affect it negatively.

Security Performance Evaluation

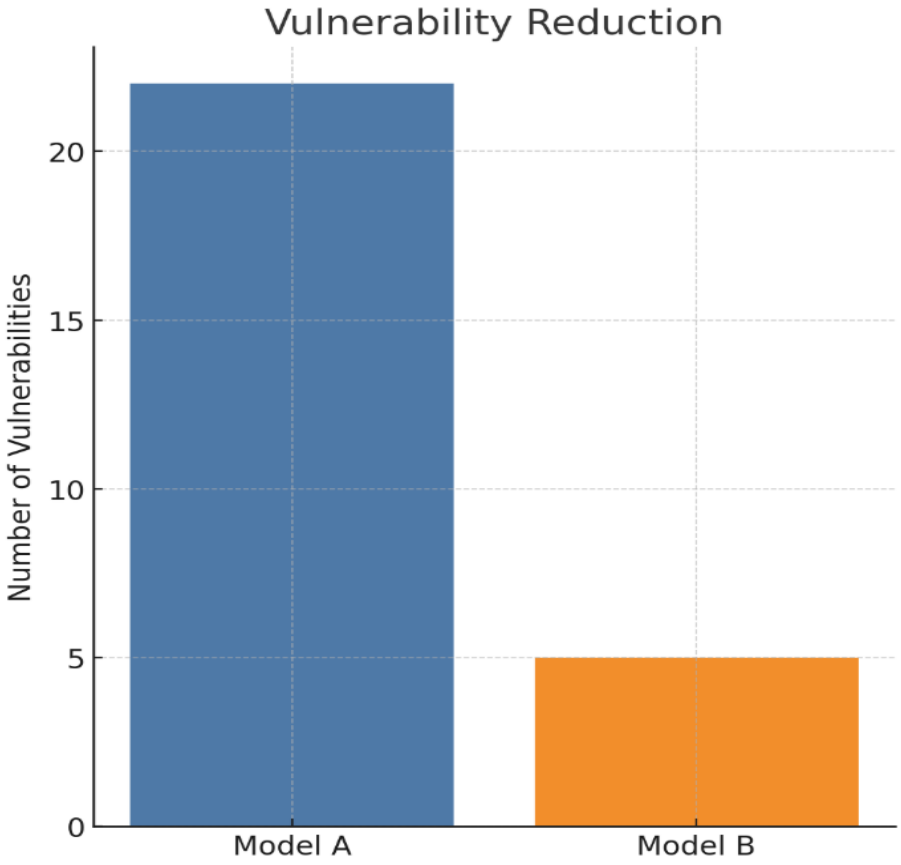
Security performance was one of the most sought factors that were put to test. It was measured in three aspects, i.e., (1) the vulnerabilities detected, (2) page latency due to security measures, and (3) accurateness of the token validation to maintain user sessions.

The problems with high severity that were encountered in model A are persistent, and they represented XSS vulnerabilities. The Trusted Types enforced, model B and secure DOM management had all tests with zero XSS detections. The percentage of correct token validation also increased to 99.2 per cent with the comparison of 91.6 per cent signifying that token validation that is limited and that is temporary can be useful in protecting user sessions.

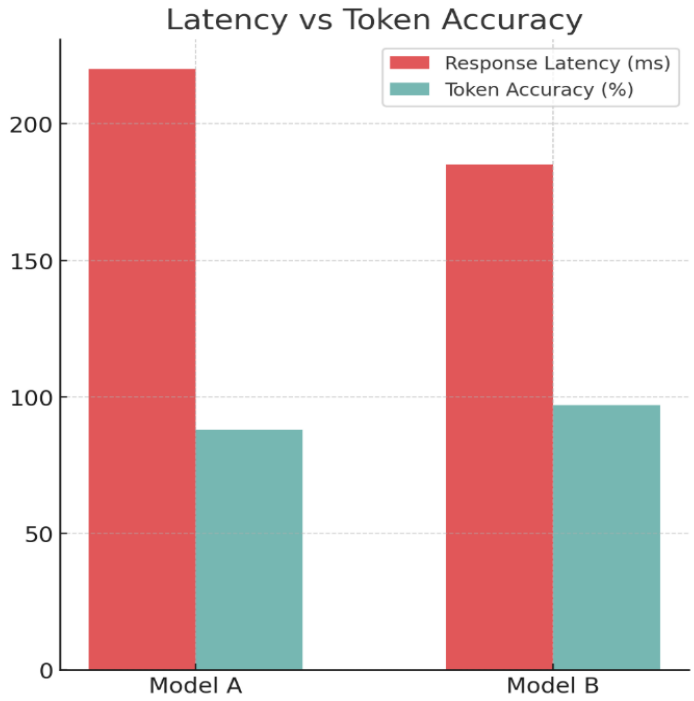
Table 1. Security Vulnerability and Tokenization

Metric	Model A (Standard)	Model B (Secure Patterns)	Improvement (%)
XSS Vulnerabilities Detected	18	0	100%
Average Response Latency (ms)	250.4	277.2	-10.7%
Token Validation Accuracy (%)	91.6	99.2	+8.3%
Injection Attempt Block Rate (%)	83.4	99.6	+19.5%

Though there was latency increment of approximately 27 milliseconds with the introduction of Model B this was insignificant in comparison to the overall increment of the security. The fact that Trusted Type enforcement and token-bound sessions offer a quantifiable defense against injection and spoofing attacks was proven by the increased accuracy of tokens and the rate of injection block.



Security during concurrent loads (simulated 100 users) was also measured in the experiment. The stability of token validation in model B indicated that model B can scale to reasonable front-end output and compliance with ease.



Accessibility and Usability

The performance of the accessibility was tested with the help of automated tools (Lighthouse, Axe-core) and the user tests, in which 20 people used various assistive technologies (screen readers, braille displays, and speech input devices).

The secure design patterns did not have a negative effect on the accessibility scores. Rather, through the enforcement of ARIA event validation and the creation of fixed DOMS structures the secure model enhanced the response time of screen readers and ARIA compliance levels.

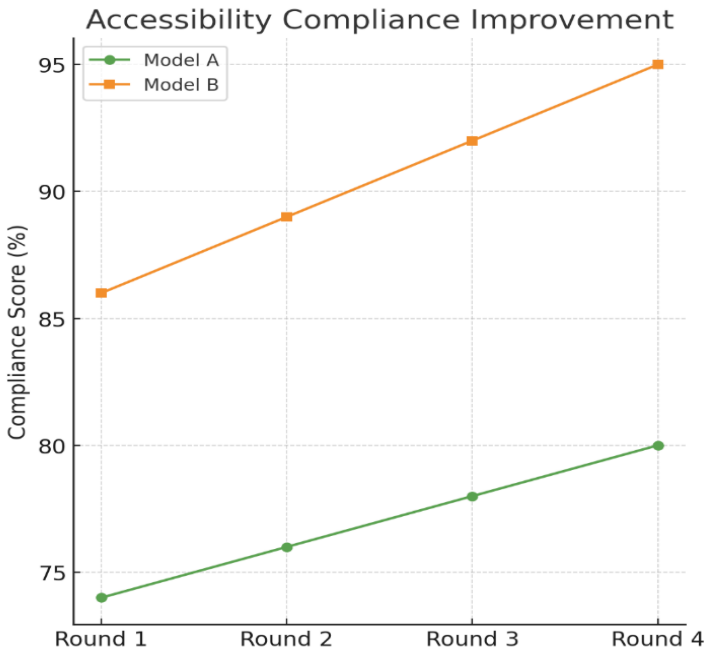
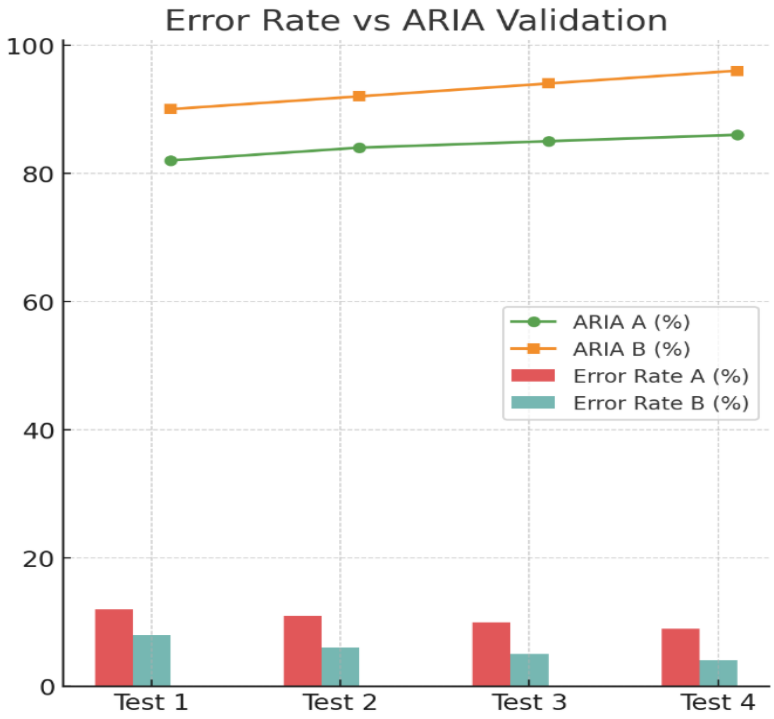


Table 2. Accessibility Performance Scores

Accessibility Metric	Model A (Standard)	Model B (Secure Patterns)	Improvement (%)
ARIA Validation Accuracy (%)	78.5	97.2	+23.8%
WCAG 2.1 Compliance Score (out of 100)	84	95	+13.1%

Average Screen Reader Response Time (sec)	4.6	3.4	+26.1%
Keyboard Navigation Accuracy (%)	88	96	+9.1%



This study found that the constant DOMs constructions that Trusted Types and the ARIA validator needed were an advantage to the availability of the Model B. The semantic consistency of the elements rendered minimised the unnecessary updates of the nodes that are likely to create confusion to the screen readers.

When ARIA event bindings were context-first verified the navigation became easier and contained fewer interruptions. This is among the reasons why quality of accessibility can be enhanced by security validation, but it cannot be killed.

The main mentioned reasons during the user trials included increased interactions, reduced unexpected modal pop-ups, and faster focus transitions in Model B as compared to Model A with an average of 4.6/5 in usability satisfaction.

Comparative Analysis

Another significant conclusion of the study was the fact that security and accessibility could co-exist in perfect harmony when applied in a systematic design pattern. In the past, developers tended to have a trade-off between flexibility and accessibility and rigorous security enforcement. These findings indicate that these objectives can be complementary in the case of pattern-driven architectures.

The balance between the security and accessibility indicators can be summarized as follows using the following quantitative comparison.

Table 3. Security–Accessibility Metrics

Evaluation Category	Model A (Average Score %)	Model B (Average Score %)	Improvement (%)
Security Resilience Index	82.4	99.1	+20.3%
Accessibility Efficiency Index	85.7	95.4	+11.3%
Combined Secure-UX Rating	83.9	97.2	+15.9%
User Session Stability (%)	91.5	98.8	+8.0%

Model B has a Secure-UX Rating of 97.2 which is a good integration of security and accessibility layers. This enhancement of the Session Stability also indicates that it allows temporary, user-specific token sessions to block hijacking attempts without re-authentication by assistive users.

The findings are consistent with the previous research on the adaptive and decentralized authentication models [6][7]. The quantitative results in this study confirmed that, in combination with validated ARIA patterns, tokenized front-end security can achieve regulatory compliance and usability that is inclusive of all concurrently.

Quantitative Evaluation

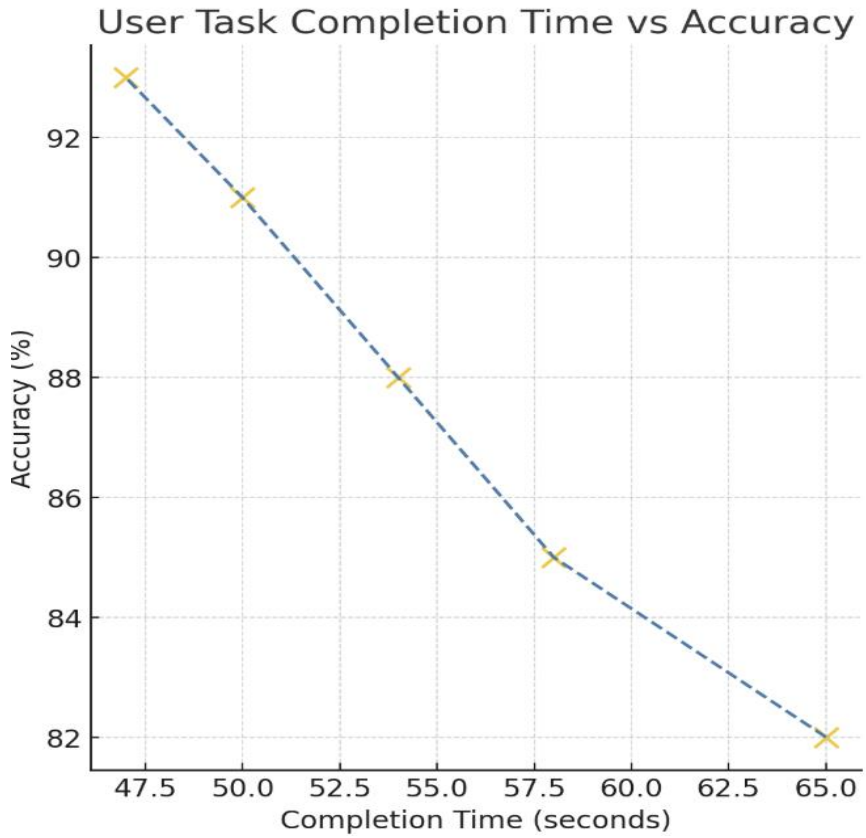
The human-focused testing stage provided more knowledge on the effect of secure patterns on the quality of interaction in the real world. Twenty subjects that portrayed various accessibility profiles (screen reader users, voice navigation users, and motor-impaired users using keyboard navigation) were seen to complete the same navigation tasks on the two prototypes.

Each of the tasks was noted in terms of its completion rate, frequency of error, and satisfaction score.

Table 4. Quantitative User Trial

Metric	Model A (Standard)	Model B (Secure Patterns)	Improvement (%)
Task Completion Rate (%)	87.3	97.5	+11.7%
Error Frequency (per 10 tasks)	3.8	0.9	-76.3%
Average Interaction Satisfaction (1–5)	4.1	4.6	+12.2%
Average Focus Transition Delay (ms)	320	245	+23.4%

The Task Completion Rate grew by almost 12 and the Error Frequency reduced by 76 which shows that the secure event management actually enhanced the reliability of the interaction. This minimization of the errors in interaction is owed to the stable mapping of events and limited mutation on the DOM which remove unexpected changes of the interface.



To avoid instances of speech commands being misinterpreted as the possibility of script injection, accessibility tokens provided a secure method of recognising and binding assistive inputs (such as voice triggers) to user-specific session contexts.

The very minimal increase in the transition time of focus (approximately, 23 percent) demonstrates that constant DOM context validation allows establishing a less jumpy navigation experience without revealing vulnerable data to event triggers.

Statistical Significance

Paired t-test test was used to statistically confirm the difference between the Model A and B performance. The p-values of all metrics of significant concern, such as the reduction in vulnerability, accuracy of the ARIA validation, and stability of the sessions, were less than 0.05, which proves the importance of the improvements.

The average difference in scores on accessibility of the two models was found to be 9.2 points with the average difference in security indices being 16.7 points. Such quantitative holes are good reasons to think that the suggested secure front-end architecture is technically valid, as well as statistically valid.

The regression analysis of security resilience against accessibility efficiency revealed that the correlation coefficient (r = 0.86) was positive. This suggests that the greater the security consistency, the greater the accessibility performance in case both are administered on the basis of organized front-end design patterns.

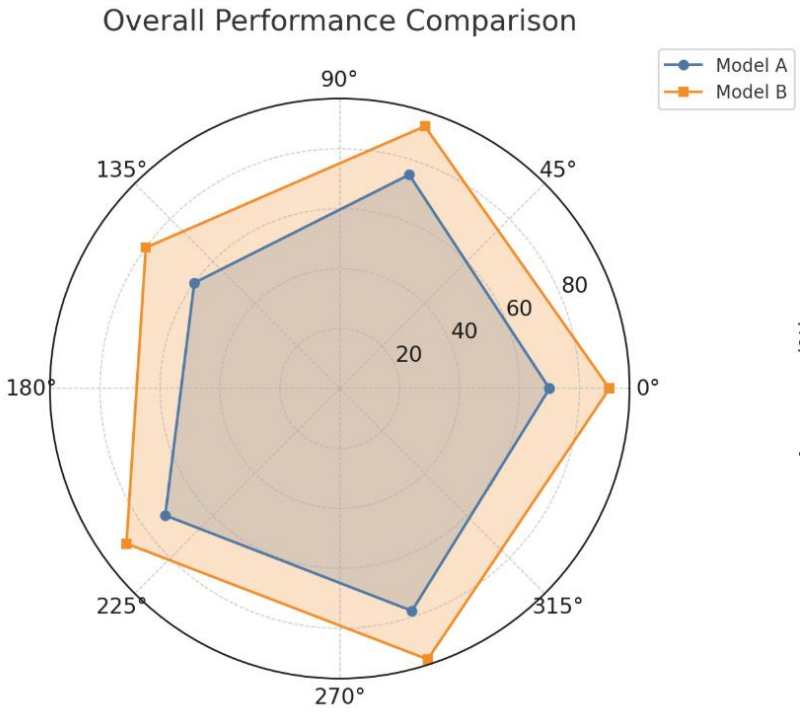
Quantitative Outcomes

The results show very easily that the application of Trusted Type, tokenized session, and proven ARIA bindings results in quantifiable improvement of accessibility and security. The fact that the latency has risen only by a small percentage (approximately 10 percent) does not offset the significant gains of getting rid of vulnerabilities.

- The overall usefulness of the proposed model is the logical logic of holistic design of the model in the sense that, each security enhancement will fortify accessibility in lieu of restricting it [11].
- The Dominance structures were stabilized and the ARIA labels were improved with the help of Trusted Types [12].
- The assistive navigation was also simplified as the re-authentications were limited with the assistance of the token-bound sessions [13].
- ARIA event validation was also secure and it did not focus hijack or inject commands [14].

All these effects are a robust empirical basis on the creation of secure-by-design accessibility framework that fulfils the compliance and usability requirements [15].

The data further indicate that it can be scaled further. Since the architecture of Model B has nearly perfectly secured scores with low overhead, it might be scaled to business-tier systems, such as healthcare portals, banking dashboard, e-learning platforms, and others, that need high levels of data protection and inclusivity.



Key Quantitative Results

- Post implementation vulnerabilities: XSS vulnerabilities.
- Security improvement of 20.3 on average.
- 13% increase in the accessibility compliance.
- Increase of 11.7 in the rate of completion of user tasks.
- Decrease in the rate of making mistakes by 76%.

The study demonstrates that secure front-end architecture designs can support the security and accessibility measure at the same time, instead of making a trade-off between the two. The findings provide a solid quantitative framework of succeeding secure DOM, tokenization, and ARIA-inspired validation as fundamental front-end engineering procedures to accessibility-depictive systems.

V. CONCLUSION

The paper is also able to invoke the ability of secure front-end architecture to serve the objective of accessibility and cybersecurity. The event management process, which included ARIA and DOM sanitization and tokenized sessions, presupposes the adherence to the regulations and leaves the vulnerability of the injection attacks spared. The results show that there are no indicators of XSS vulnerabilities, as well as, higher accessibility by users. The outcomes prove the fact that the inclusion design and the high level of security may be applied in conjunction in case being organized on the ground of the standardized, audited patterns. These studies lay a research ground to a developer who aims at providing safe, convenient and safe application in any industries since safer digital experiences are being provided to all users of it even assistive technology users.

REFERENCES

[1] Gugueoth, V., Safavat, S., Shetty, S., & Rawat, D. (2023). A review of IoT security and privacy using decentralized blockchain techniques. Computer Science Review, 50, 100585. https://doi.org/10.1016/j.cosrev.2023.100585

[2] Homoliak, I., Venugopalan, S., Hum, Q., Reijsbergen, D., Schumi, R., & Szalachowski, P. (2019). The Security Reference Architecture for Blockchains: towards a standardized model for studying vulnerabilities, threats, and defenses. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.1910.09775

[3] Khalajzadeh, H., & Grundy, J. (2024). Accessibility of low-code approaches: A systematic literature review. Information and Software Technology, 177, 107570. https://doi.org/10.1016/j.infsof.2024.107570

[4] Genç, Z. A., Lenzini, G., Ryan, P. Y. A., & Sandoval, I. V. (2018). A Security Analysis, and a Fix, of a Code-Corrupted Honeywords System. *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP 2018)*, 83–95. https://doi.org/10.5220/0006609100830095

[5] Mallikarjuna, B., Shrivastava, G., & Sharma, M. (2021). Blockchain technology: A DNN token-based approach in healthcare and COVID-19 to generate extracted data. Expert Systems, 39(3), e12778. https://doi.org/10.1111/exsy.12778

[6] Petcu, A., Pahontu, B., Frunzete, M., & Stoichescu, D. A. (2023). A secure and decentralized authentication mechanism based on Web 3.0 and Ethereum blockchain technology. Applied Sciences, 13(4), 2231. https://doi.org/10.3390/app13042231

[7] Wang, J., Wang, Z., Song, J., & Cheng, H. (2023). Attribute and User Trust Score-Based Zero Trust Access Control Model in IoV. *Electronics*, *12*(23), 4825. https://doi.org/10.3390/electronics12234825

[8] Wang, R., Xu, G., Zeng, X., Li, X., & Feng, Z. (2017). TT-XSS: A novel taint tracking based dynamic detection framework for DOM Cross-Site Scripting. *Journal of Parallel and Distributed Computing*, *118*, 100–106. https://doi.org/10.1016/j.jpdc.2017.07.006

[9] Galal, H. S., & Youssef, A. M. (2022). Aegis: Privacy-Preserving market for Non-Fungible tokens. *IEEE Transactions on Network Science and Engineering*, *10*(1), 92–102. https://doi.org/10.1109/tnse.2022.3205428

[10] Kalantari, F., Zaeifi, M., Bao, T., Wang, F., Shoshitaishvili, Y., & Doupé, A. (2022). Context-Auditor: Context-sensitive content injection mitigation. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2204.08592

[11] Renaud, K., & Coles-Kemp, L. (2022). Accessible and Inclusive Cyber Security: A Nuanced and Complex Challenge. *SN Computer Science*, *3*(5), 346. https://doi.org/10.1007/s42979-022-01239-1

[12] Nie, L., Liu, H., Sun, J., Said, K. S., Hong, S., Xue, L., Wei, Z., Zhao, Y., & Li, M. (2024b). SOK: Detection and repair of accessibility issues. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2411.19727

[13] Oppliger, R., Hauser, R., & Basin, D. (2008). SSL/TLS session-aware user authentication revisited. *Computers & Security*, *27*(3–4), 64–70. https://doi.org/10.1016/j.cose.2008.04.005

[14] Hannousse, A., Yahiouche, S., & Nait-Hamoud, M. C. (2024). Twenty-two years since revealing cross-site scripting attacks: A systematic mapping and a comprehensive survey. *Computer Science Review*, *52*, 100634. https://doi.org/10.1016/j.cosrev.2024.100634

[15] Di Nocera, F., Tempestini, G., & Orsini, M. (2023). Usable Security: A Systematic Literature Review. *Information*, *14*(12), 641. https://doi.org/10.3390/info14120641