International Journal of Emerging Trends in Computer Science and Information Technology
ISSN: 3050-9246 | https://doi.org/10.63282/3050-9246.IJETCSIT-V7I4P101
Eureka Vision Publication | Volume 7, Issue 4, 1-18, 2026

Original Article

AI-Assisted UX Engineering for Cybersecurity Products: A Design Governance Framework for Design-System Compliance, User-Flow Validation, and Pattern-Based Prototyping

Aravind Rangaswamy
Product Design Architect, Sonicwall, Manteca, USA.

Received On: 19/08/2026 ***Revised On: 21/09/2026*** ***Accepted On: 28/09/2026*** ***Published On: 04/10/2026***

Abstract - In modern cybersecurity products, user experience, security, consistency, and user trust are closely interconnected, while the increasing adoption of generative artificial intelligence (GenAI) introduces new opportunities for improving early-stage product design and prototyping. AI-driven tools can help product managers convert product requirements and early concepts into initial sketches of user interfaces before involving UX designers and software developers. Even so, unchecked AI designs can stray from existing systems of organizational design, fail to capture relevant journeys of users, and create new interaction patterns that are incompatible with current product experiences. This paper proposes a conceptual approach to AI-Assisted UX Engineering for Cybersecurity Products that combines design-system governance, user-flow validation, and validated interaction patterns with generative AI-based prototyping. The proposed approach includes three complementary AI-assisted capabilities: design-system compliance, which helps prototypes adhere to the organization's design system; user-flow validation, which examines proposed experiences for completeness, alternative flows, exception flows, and security-sensitive flows; and pattern-based prototyping, which encourages the reuse of known patterns rather than creating redundant and unvalidated interaction solutions. The framework connects product ideation, AI-aided prototyping, UX design, and software development, allowing product teams to test whether initial ideas satisfy specific design and workflow requirements before moving toward formal design and implementation. An iterative validation process is integrated, where AI-generated prototypes, design-system validations, user-flow reviews, and user experience feedback are used to continuously inform further design decisions. The conceptual framework offers a structured method for leveraging GenAI as a design support tool while adhering to established UX principles and requirements of security-sensitive products. The framework needs to be empirically tested in the future to assess usability, design-system compliance, workflow coverage, prototyping efficiency, and inter-team csollaboration among product, design, and engineering teams. The proposed approach provides organizations with a foundation for integrating AI-assisted prototyping into cybersecurity product development without treating generative AI as an unconstrained replacement for established UX engineering and design governance practices.

Keywords - Artificial Intelligence, Generative AI, Cybersecurity, Product Experience, AI-Assisted Prototyping, Design Systems, User-Flow Validation, Design Governance, Usability Engineering.

1. Introduction

1.1. Why Security Products Cannot Be Designed Like Ordinary Products

Cybersecurity products sit at a nexus of usability and risk: their user interfaces need to enable effective decision-making under time constraints, and consequential actions should not jeopardize an organization's defensive posture. Security operations centres (SOCs) are an example of this challenge. Though 94% of large companies have a SOC, analysts spend over 25% of their time dealing with false positives, 51% of teams are overwhelmed by alert volume, and 63% of practitioners suffer from some level of burnout [13]. In a companion study, about 40% of alerts were ignored completely, 49% were assumed to be false positives, and analysts depended on fellow analysts to deal with the alerts in half of the cases that they did observe [24]. The results indicate that, besides technical security protection mechanisms, interfaces that help the human operator to pay attention, interpret and make decisions under operation pressure are also a key factor in the effectiveness of security products.

The usable-security literature reinforces this relationship between interaction design and security outcomes. The need to redesign security mechanisms was also suggested by Adams and Sasse, who showed that users will be tempted to bypass security mechanisms when their cognitive and procedural needs differ from those required for normal working [28]. Browser-warning studies show the impact of interface manipulation on security related behaviour. After the redesign, the reported rate of compliance with Chrome's warning on HTTPS was 70% of the population, and was

about 30% for an earlier version of that warning [15]. Large-scale field measurements also showed the following bypass rates: Chrome malware warnings (23.2%), Chrome phishing warnings (18.0%), Firefox SSL warning (33.0%), and Chrome SSL warning (70.2%). The worst-performing Chrome variant, involving an untrusted issuer, was bypassed in 81.8% of cases [26]. Laboratory findings provide further evidence of the challenge: 97% of 60 participants fell for at least one phishing message, while 79% followed an active warning compared with only 13% who followed a passive warning; moreover, 6 of the 10 participants in the passive-warning condition never noticed the warning [23]. After progressively removing security indicators, a study involving 67 bank customers found that 23 of the 25 participants who used their own accounts entered their passwords even when the security indicators had been removed [14]. Together, these results indicate that specific aspects of interface visibility, wording, sequencing and affordances can impact people's awareness and compliance with protective measures.

Security-specific research emphasizes the importance of studying usability and security in combination instead of as separate issues. In a systematic review of 55 usable-security studies, 34 papers (56% of the corpus) were identified as focusing on authentication usability, seven on security developers enhancing authentication usability, ten on authentication method-related strategies for influencing security behaviour and four addressing formal models for assessing usable-security [9]. The authors of the previous review, cited in the same paper, reported that only 6% of articles made it to the final analytical stage [9]. The results of these assessments are not, of course, conclusive evidence of the effectiveness of any specific assessment framework, but do highlight the need to make assessment criteria explicit and traceable. Usability and security analysis of interfaces agree on compatible conclusions [5] and research that takes the usable-security principles into developers and suggests that “people who build or configure security mechanisms are also important users of these mechanisms” [27]. An eye tracking study of 49 developers revealed that users typically skimmed through documentation and mostly absorbed the code examples. Secure Content Security Policy implementations were achieved in only 33% and 30% of cases across two tasks, and 15 of the 49 participants copied and pasted the supplied header [3]. These results drive a design process that involves an explicit consideration of security-relevant interactions and implementation-relevant requirements, rather than implicit based on visual completeness.

1.2. Generative AI Has Changed Where Prototypes Come From

Generative artificial intelligence (GenAI) is transforming the way product teams develop and improve interface prototypes. Design2Code provides a benchmark of 484 real-world webpages whose code averages 31,216 tokens per file, with a minimum of 784 and a maximum of 98,637 tokens; the webpages have an average document object model (DOM) tree depth of 13 and contain 158 HTML tags per example [1]. Interaction2Code introduces the evaluation of interface generation to 127 real-world webpages with 374 different interactions, and reports that over half of the cases studied include interactive elements [12]. Sketch2Code contains 731 hand-drawn sketches from 484 webpages, showing the feasibility of a preliminary visual concept to be translated into interface prototypes [19]. Previous studies, such as pix2code, proved that it is possible to create interface code from graphical user-interface screenshots with reported accuracy of more than 77% on three platforms [6]. The more recent UICoder boosted the compilation rate of a 15.5-billion-parameter model from 0.03 to 0.82, with a synthetic corpus of almost one million SwiftUI programs being filtered using a compiler and a vision-language model, instead of expensive human feedback [2]. A systematic review also indicates that in the studies reviewed, token-based input representations were seen in about 97.75% of cases [20]. Together, these advancements highlight the increasing capacity of AI systems to transform visual and textual information into initial interface designs. This means that product managers and other stakeholders can make reasonable prototypes without relying on the involvement of formal designers or developers, which can help to achieve earlier agreement and design exploration [7].

However, the ability to generate a plausible interface does not establish that the resulting prototype is correct, consistent, complete, or appropriate for a security-sensitive context. Manual detection of design-system violations is identified in research on hybrid design-system linting as time-consuming, and the authors determine that AI alone didn't suffice for adoption. The study suggests a mixture of heuristic and model-based assessment, along with a minimum of 90% of successful attainment and 95% of confidence on 90% of cases [18]. These figures are to be interpreted as target figures only, until it is expressly stated that they were obtained experimentally. The use of iterative agent-based workflows can enhance the outputs but it does not remove the need for review. For Sketch2Code, simulated user-agent feedback was accurate and informative in 83.3% of cases and accurate in answering the questions in 86.7% of cases, and iterative refinement increased visual similarity by up to 7.1% [19]. These findings illustrate the benefits of feedback-informed generation and show the need to examine the reliability and limitations of automated support.

For cybersecurity products, these limitations translate into three governance concerns. Firstly, the AI-generated interfaces might vary from adhering to approved design tokens, components, accessibility standards, and interaction norms. Second, exception handling, recovery scenarios, and those rejected for permissions as well as other security aware paths are not included in generated screens, especially those involving exception handling. Thirdly, unrestricted generation can create new interaction patterns for which there may not be enough organizational review and validation. These concerns drive a structured governance process that ties together design systems and compliance, user-flow validation, and pattern-based prototyping. The

proposed framework tackles these three concerns by recognizing the interfaces created by AI as preliminary artifacts to be explicitly checked and reviewed by humans prior to being accepted for downstream design or implementation.

1.3. Governance Is the Missing Layer

Addressing these risks requires more than prohibiting generative AI use. Governance mechanisms are most effective when they are designed into practitioners' practices and backed by design artefacts that advocate for action. However, a co-design exercise with 48 practitioners from 12 technology firms identified that abstract ethics and fairness checklists can be hard to operationalize, and that we need governance artefacts that are grounded in established practices, that do not shift responsibility onto individual advocates [4]. UX practitioners are also invited to participate in the whole responsible-AI lifecycle, from linking user needs, design decisions, and responsible development practices [8].

Guidance already exists that will provide a basis for this integration. Eighteen human–AI interaction guidelines were generated through four phases of development, including a user study with 49 practitioners, and cover such topics as capability disclosure (G1), explanation of system behaviour (G11), communication of consequences (G16), global controls (G17) and notification of changes (G18) [25]. On the organizational level, the National Institute of Standards and Technology (NIST) AI Risk Management Framework outlines four primary factors: GOVERN, MAP, MEASURE and MANAGE, as well as characteristics of trustworthy AI systems [17]. It is also described by its Generative AI Profile in 12 categories of risk, such as confabulation, data privacy, human–AI configuration, information integrity [16].

While these resources provide sound principles relevant to responsible AI and to interaction with humans, they do not provide enough operational linkage between the requirements of governance and the activities of design, particularly around the design of AI-assisted cyber security products. Product teams, in other words, should have a mechanism for mapping generated interfaces to approved systems, in order to ensure full and security-related user journeys and to establish a safeguard on the addition of unapproved interaction patterns. This paper addresses this need by recommending a conceptual three-capability governance framework that can be included as a workflow of iterative prototyping and review. The framework conveys existing guidance into proposed design activities and criteria, which needs to be assessed with future empirical validation.

1.4. Contributions

This paper has five contributions. First, it suggests a conceptual design-governance framework for AI-aided UX engineering of cybersecurity products, including design-system compliance, user-flow validation and pattern-based prototyping. Second, it creates a traceable evidence base out of 28 documents ranging from design-to-code benchmarks, design-system tooling, usable security, security operations, and AI risk management; this evidence base is presented in Table I. Third, it aggregates reported quantitative performance measures for the generation and refinement of AI-driven interfaces into Table IV, offering context on the potential of and problems with existing approaches. Fourth, it outlines, in Table III, an eight-metric testing procedure, which includes the published quantitative target and proposed targets which must be empirically calibrated. Fifth, it provides a prototype of the security-operations console and the accompanying security checks in Fig. 3 and Table V, and suggests directions for future empirical assessment. The framework is not in fact an empirically validated method or an improvement on the existing approaches, but rather a conceptual proposal presented in a structured manner based on previous research.

1.5. Organization

The remainder of this paper is organized as follows. In Section II, related work is reviewed in five strands. Section III outlines the proposed governance framework and three governance capabilities that are outlined, as well as its iterative validation loop. The methodology for the evidence synthesis and evaluation protocol is described in Section IV. The synthesized evidence, resulting requirements, and an illustrative representation of the framework demonstration are introduced in Section V, which discusses their implications. The study's limitations and threats to validity are discussed in Section VI. Finally, Section VII concludes the paper and gives directions for future research.

2. Related Work

2.1. Design-to-Code and AI-Assisted Prototyping

AI-assisted interface generation has evolved from demonstrating that converting a screenshot into code is possible, to testing more complex interface structures and interactions. Early research for converting screen shots to code for three platforms claimed over 77% accuracy [6]. Design2Code built on this work by using a benchmark of 484 curated webpages, and added evaluation measures for matching blocks, text, position, colour and CLIP similarity. The benchmark also recorded impressive implementation complexity, with an average number of 31,216 tokens per page and an average depth of 13 pages of the document object model (DOM) [1]. Interaction2Code widened its evaluation focus from static appearance to interactive behaviour and looked at 127 webpages and 374 interactions, finding ten common failure categories [12]. Sketch2Code moved the shift from the input further upstream: 731 sketches were drawn by the user, off 484 webpages [19].

Other studies focus on how to improve the code produced by the automated feedback. UICoder investigated compiler-based and vision-language-model feedback as alternatives to human feedback during interface-model fine-tuning [2]. The systematic review of LLMs in SE highlights the growing importance of LLMs in programming tasks and also shows an 87% success rate for a model in complex Python programming problems [20]. The studies are examples of progress made in interface generation, reproduction of interaction and code refinements. Their main

concern, though, is mostly capability generation or benchmarking, not organisational governance of generated prototypes. Benchmarking by itself is not indicative of an interface being compliant to the approved design system of a product, or if being used, of all security-sensitive paths, or of the use of only approved interaction patterns within the product. The scheme outlined aims to tackle the concerns, with governance being added in the process of generation and refinement.

2.2. Design Systems, Linting, and Design Governance

Design systems offer durable elements, design tokens, modes for interactions, and rules for implementation, helping organizations be consistent across their products. However it can be challenging to enforce those standards if the requirements are broad, manual inspection involves a lot of time, or if there isn't enough information about the requirements to be adequately represented in a model's context window. Research into design system linting finds limitations with both manual and artificial-intelligence-based linting, suggesting a hybrid method should be used that involves both a heuristic and a model-based evaluation. The study specifies targets of at least 90% successful completion and 95% confidence on at least 90% of cases [18]. These values are different from the ones which have been attained in an experimental test that has been conducted and reported explicitly as measured values by the original research.

Governance research also provides an understanding of why rules of any kind need to be tied to practice. In a co-design study conducted with practitioners from 12 technology companies, abstract ethics and fairness checklists proved to be hard to operationalize; and it was argued that the requirement for governance artefacts to fit practitioners' workflows was the pressing need [4]. Guidelines for human–AI interaction present another source of actionable requirements on interface behaviour [25], and while human use and experience in AI human-centred research puts emphasis on the need for user expertise throughout different stages of the AI lifecycle [8].

These studies collectively offer complementary foundations for design-system enforcement and responsible-AI governance. Automated linting, interaction guidelines and governance artefacts, however, utilise other aspects of the design process. In this paper, a framework is proposed that links these concerns in an AI-supported prototyping process. The first of its abilities tests it against design-system rules that have been approved; the second tests user-flow completeness; and the third capability encourages re-use of user-flow reviewed interaction-patterns rather than unfettered invention. The aim of this integration is to allow for governance action to be taken at an early stage of design thinking and only conformance checking in later stages.

2.3. Usable Security and Security-Sensitive Interaction

Usable-security research shows that security mechanisms may lose effectiveness when their interaction requirements are in conflict with users' needs to complete their everyday work and their cognitive ability [28]. This field of systematic reviews groups it by Authentication Usability, Developer-Faced Usability, Behaviour-Influencing Design Strategies, and Evaluation Models. An opportunity that was identified in one study that looked at 55 studies was that there was only a handful of studies that had formal evaluation models [9] to try to make formal evaluation criteria more explicit in security-product design.

Studies of browser warnings illustrate how interaction design can influence security-related behaviour. Research has reported adherence rates of 79% for an active warning and 13% for a passive warning [23]. Another analyzed reported that an overhaul of the Chrome HTTPS warning had adherence of around 70%, versus about 30% for an early version [15]. High scale field tests revealed bypass rates of 23.2%, 18.0%, and 33.0% of Chrome's malware warnings, Chrome's phishing warnings, and Firefox's SSL warning, respectively [26]. Security indicators were gradually removed to a low of one and all 25 of the bankers who used their bank accounts provided their account details without difficulty were asked to do so [14]. Despite the variation in setting and outcome measures across these studies, they collectively highlight the importance of studying security-sensitive interfaces from a standpoint of visibility, comprehension, decision sequencing, and the outcomes or impact of user action.

Security indicators are also impacted by the representation of information. Research evaluating 40 icons and seven strings involved 1,329 respondents before proposing three candidate designs. The study's inter-coder Fleiss' kappa was 0.89, coder agreement on 91% of answers and prevalence of estimated colour vision deficiency was about 8% of males [22]. More dimensions come with developer-facing research, as security guidance might not be understood or applied as intended, and developer implementation practices can affect security mechanisms [3], [27].

The significance of these findings for AI-assisted UX engineering is that a generated screen could have a complete feel and look to it while not including any essential decisions, recovery states, or explanations about any security implications. Security-sensitive user flows are thus named targets for validation explicitly as opposed to relying on visual similarity or component-level conformances. The user-flow capability is designed to aid in user reviewer finding missing paths in the prototype, when exception handling and permission related states are involved, before a prototype is accepted for downstream design or implementation.

2.4. Security-Operations Context and Analyst Workflow

These are challenging scenarios to explore the real-world impacts of AI-supported UX governance. Research shows that 51% of security teams are overloaded with alerts, 25% or more of security analysts' time is devoted to dealing with false positives and 63% of practitioners say some form of burnout has occurred. According to the same source, 93% of analysts are willing to use AI assistance and about 49% of

the alerts are successfully completed by teams [13]. An accompanying study shows that around 40% of alerts were ignored, 49% were assumed to be false positive alerts and analysts depended on their colleagues in 50% of the cases: It also projects cybercrime costs of 13.82 trillion U.S. dollars by 2028 [24]. These are the operational pressures reported in the studies mentioned and should not be seen as pervasive across all SOCs nor as a proof that interface design would solve all these challenges.

Research on visualization focuses on the human-centered approach and structured threat representations, such as MITRE ATT&CK, in developing security analysts' tools [21]. Research carried out by multi-criteria decision making methods also indicates that usable-security criteria vary in expert weight from 7.37% to 20.99%, per the six competing models [10]. This difference indicates that stakeholders' priorities may vary, and that this must be made transparent in mapping out requirements and assessing them, rather than merging them into an unexamined average.

A context is a framework that communicates to the analysts the represented application domain for the proposed framework, during which analysts might need to annotate alerts, review evidence, assess impact, and enact controlled responses. The design-governance workflow is intended to help teams check whether a generated prototype follows approved interface conventions and represents the required investigative and response paths. It is not assumed these checks themselves can solve the problem of alert fatigue, elevate analyst efficiency or increase incident resolution rates, which would need independent empirical verification.

2.5. AI Risk Management and Trustworthy-AI Guidance

The NIST AI Risk Management Framework (AI RMF 1.0) offers a framework that identifies AI-related risks and addresses them within four components: GOVERN, MAP, MEASURE, MANAGE [17]. It also addresses attributes of trustworthy AI such as valid and reliable, safe and resilient, accountable and transparent, explainable, privacy enhancement and fairness with negative bias managed. The NIST Generative AI Profile builds on this by enumerating 12 risk categories: including confabulation, privacy of input or output of data, integration into the value chain or components, configuration of humans and AI, and data or information integrity [16].

These resources provide a basis for identifying and managing risks associated with the use of generative AI. Human–AI interaction guidelines follow and cover aspects such as communicating capabilities, explaining behaviour, communicating consequences, supporting user control and notifying users of changes in AI-enabled systems [25] the research on UX practitioners' role in responsible AI also highlights the importance of user-centred design in the process of translating the global levels of principles to product-development activities [8].

However, interaction guidelines or risk-management methods do not always describe the method for evaluating a generated prototype against a specific organization’s components, interaction ways and its accepted interaction patterns. Their application involves product-specific requirements, application stages and evidence for conformance. The framework proposed in this paper uses these sources as conceptual foundations for its governance requirements. Does not state that the framework meets all the recommendations from NIST nor make independent assessments of the safety, security or trustworthiness of a prototype. Rather, it draws on three bounded capabilities that could be used to significantly support a more systematic review in early-stage cybersecurity-product prototyping.

2.6. Positioning and the Research Gap

Supplementary capabilities and constraints are seen in the literature reviewed. There are prototyping and reproduction of interfaces and interactions methods that are built from a design-to-code and AI-prototyping research [1][2][6][12][19]. Design-system linting and governance research is related to conformance checking and operationalization of organizational requirements [4] [8] [18] [25]. Evaluating the security sensitive interactions and the uncertainty in users consistently recognizing or following security advice is important [3], [9]–[14] [22]–[28]. Based on the aforementioned, SOC research also focuses on the operational context in which analysts have to investigate and respond to alerts under workload (or time-pressure) constraints [10], [13], [21], [24]. Finally, NIST guidance offers a broader context when trying to identify and manage AI related risks [16], [17].

The research challenge in this context is to integrate these threads into an integrated and operationally formulated research process for AI-supported cybersecurity-product prototyping. Specifically, the proposed framework incorporates three capabilities: (1) Design-system compliance to review adherence of the components to the design system and design rules; (2) User-flow validation to review the completeness of required user flows, alternative user flows, exception user flows, and security-sensitive user flows; and (3) Pattern-based prototyping to encourage re-use of reviewed user interaction patterns, and to explicitly review unfamiliar patterns. These capabilities are then interconnected via human oversight and iterative validation for the entire prototyping process.

Proposed integration of generation, design-system conformance, workflow validation, and controlled generation reuse as part of a cybersecurity oriented governance process is then the framework's contribution. The literature reviewed justifies the integration but is not conclusive that it would be more effective than other approaches. Section III provides a more detailed look into the framework and its work-flow, while Section V shows the supporting evidence as well as an illustration of the instantiation. Empirical testing is needed and evaluation against other criteria is required to determine the reliability, efficiency, and usefulness of its application.

3. The Proposed Framework

3.1. Design Principles

The framework has five principles. P1, constraint over prohibition: generation of prototypes is allowed, but they are executed with explicit organizational constraints not in an unconstrained mode [18] [25]. P2, groundedness: Governance artifacts should be formulated as checks that fit practitioners' workflows so that abstract guidance is not operationalized in practice [4]. P3, traceability: every prototype contains a provenance record, which consists of the model, the requirement set, the design-system version and the review verdicts, as in the GOVERN and MAP functions [17]. The security-flow primacy of authentication, consent, warning, data-sharing and privileged-action flows are validated as first-class artifacts in P4 [9], [15], [22], [23], [26]. P5 human authority: Automated verdicts are advisory; a designated human reviewer retains responsibility for assessing conformance in the assessment of conformity, as AI-only conformance has been proven to be insufficient [18].

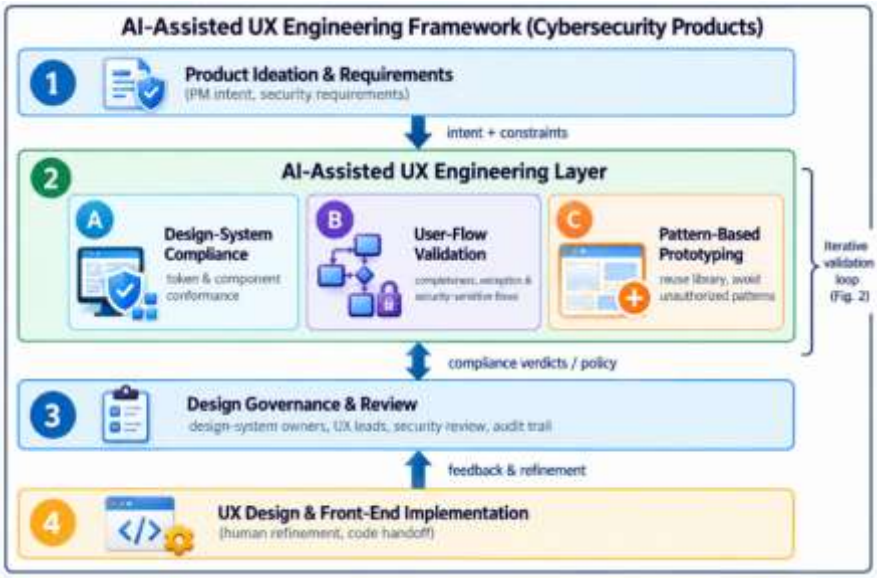


Figure 1. AI-Assisted UX Engineering Framework for Cybersecurity Products

3.2. Capability 1: Design-System Compliance

The first capability evaluates a generated prototype against the organization's design system and returns actionable violations. It is specified as a hybrid of deterministic and model-based checks, because hybrid linting is necessary for practical adoption: heuristics reliably detect syntactic violations while models handle contextual judgement that resists reduction to rules [18]. Eight rule families are defined: colour-token conformance, including semantic roles such as severity, success, warning, and destructive; typography scale and weight; spacing and layout-grid conformance; component identity, detecting substituted or re-implemented non-approved components; state completeness across default, hover, focus, disabled, loading, and error; accessibility constraints including contrast, target size, and non-colour-only encoding [22]; content and tone rules, which design systems centralize [18]; and density and information-hierarchy conformance for data-dense analyst screens.

Outputs are structured: every violation has a unique identifier for the affected element or token, a level, and a suggested replacement from the design system. The capacity reports express confidence with probability measures, similar to the public quality goals of minimum 90% successful completion with 95% confidence on minimum 90% of reports [18]. If the design system does not provide the required pattern, the pattern-returning capability should not create a pattern, but should return "no approved pattern exists" leading to Capability 3 and to a human owner.

3.3. Capability 2: User-Flow Validation

A second capability, however, looks at a prototype as a set of flows and not a screen. There are four types required for all user interactions such as the primary flows (the intended happy path), alternative flows (valid paths like alternate roles, tenants, entry points, etc), exception flows (recovery, failure, timeout, etc, or valid partial success and failure), security-sensitive flows (authentication, step up authentication, etc, consent, warning acknowledgment, data sharing and privileged/destructive actions). Taxonomy follows from usable-security literature, with authentication usability as the largest cluster [9], warning adherence based on not passive but active presentation [23] being improved after redesign of the warning copy and interaction [15] and the indicator design needing to resist removal and degradation and remain comprehensible [14] [26]. In the case of security products, the flow-validity questions are whether a user is capable of performing the security task, whether the impact of each response is communicated [25], and whether the flow remains under operational conditions within the domain where analysts are already overloaded, with about half of all alerts that they receive not being acted upon [13], [24].

Capability 2 generates a flow-coverage matrix that indicates whether each class has been covered, partially covered, or unassigned states; is missing exception paths; has missing security confirmations; or has ambiguous or passive security messages. Interaction completeness is also a unique failure axis, as interactive elements are dominant on real pages and models fail differently on interactive elements, with complete visual similarity consistently surpassing the rate of interactive elements on feedback throughout the various benchmark measurements [12] and specific feedback measurably improving outputs [19]; the capability, thus, can transform review comments into structured prompts, as is the operation of the loop in Section III-E.

3.4. Capability 3: Pattern-Based Prototyping

The third comes with the restriction that generation is limited to be within patterns of interactions validated by the organization. It keeps a pattern registry, which includes the pattern name, goal it will help a user achieve, the flow classes it will satisfy, design system components, accessibility notes, and validation status. Generation is pattern-first: only those requirements that do not match an item in the registry are eligible for novel generation. Novel patterns are considered unvalidated and are clearly marked as such requiring explicit human approval and will not accumulate in the registry without being validated, thus preventing having the registry filled with duplicate solutions that are not in line with the existing product experience.

There are two literature signals that inspire this capability. First, interaction patterns are where generated interfaces fail most frequently, and explicit prompting to make painstaking selections of components, style, and text produces measurable improvements to output – up to 7.1% more similar to the visual output, over five rounds of feedback; and, for correctly formatted HTML, the range was 82.4% to 98.0% across model selections [19]. Second, the existing interaction vocabulary already exists and is considerable for analyst-facing security tooling, such as a manner of interacting with multi-sourced data in threat-model driven visual analytics workflows [21] and human-AI guidelines that would outline who gets informed, how it should be explained, the resulting consequences, and the required controls [25]. Capability 3 also restricts against passively issued security notices which can be safely ignored without any acknowledgement [23]; security indications which rely on fine decoration (including variation by size and colour blind individuals) [22]; and destructive and privileged actions with no consequence text or safe default [25].

3.5. The Iterative Validation Loop

Figure 2 specifies the six-stage loop that operationalizes the framework: (1) product-manager intent and security requirements are captured as a structured requirement set; (2) a prototype is generated, pattern-first and constraint-bound; (3) design-system compliance is evaluated, returning token- and component-level verdicts; (4) user-flow validation returns a coverage matrix and defect list; (5) pattern reuse is checked and unvalidated patterns are flagged; and (6) a named UX or design reviewer accepts, revises, or rejects, with defects returned as structured prompts to stage 2. The loop terminates in a design freeze that hands artifacts to human UX design and front-end implementation; a generated prototype is never treated as a deliverable specification.

There are two design decisions that are driven empirically. First, the iteration is limited by a review gate rather than by self-assessment of the model; iterative agent's refinement has the ability of improving the outcomes only when the feedback signal is meaningful, and it was so when it was used in the context we studied: 83.3% of it were meaningful; 86.7% of the answers we got were accurate [19]; This residual uncertainty reinforces the need for human review and final acceptance authority. Second, the prototype does not need to be high-fidelity in order to be validated: controlled comparison of low fidelity and high fidelity prototypes across both paper and computer media lead to the conclusion that neither was more effective in uncovering problems of use; both received 28 participants (mean 1,270 comments, SD 18), and the most common problem categories were match between system and real world (28%), visibility of system status (17%), aesthetic and minimalist design (10%) [11]. Thus, early and inexpensive validation using low fidelity prototypes is warranted, hence the economic value of using AI for prototyping in the first place.

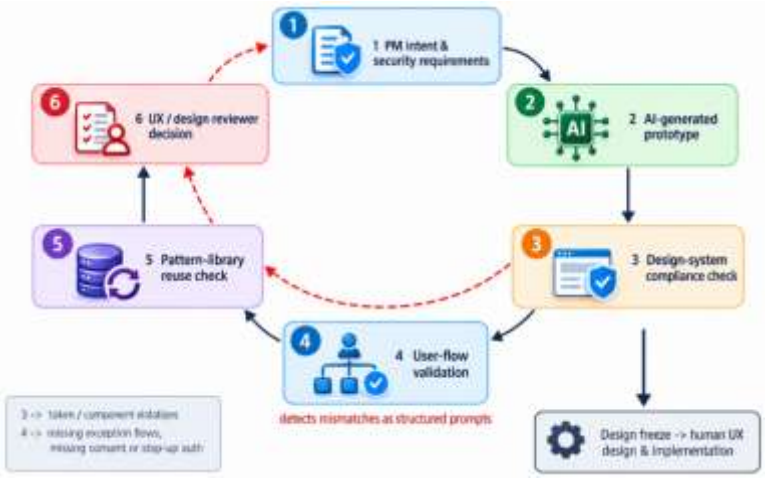


Figure 2. Iterative Validation and Review Workflow for AI-Assisted UX Engineering in Cybersecurity Products

3.6. Roles, Artifacts, and Governance Integration

There are four roles: Acceptance criteria for security apply as well, and this set of acceptance criteria is owned by the product manager. The UX designer or UX lead is accountable for the flow validation, pattern approval and the ultimate verdict on the review. The design-system owner owns the rule set and the token and component catalogues used by Capability 1. Security review is a participant in the security-sensitive flow catalogue, as well as signing off for security-sensitive flows that involve authentication, consent, warnings, and privileged actions. Engineering works on artifacts that have been frozen, and doesn't need to decode purpose out of code.

Five artifacts are created during each iteration: a requirements set, a provenance log including model, prompt, design-system version and dates, a design-system violation report, flow-coverage matrix and a pattern-reuse report including flags for unvalidated patterns. This artifact set supports the GOVERN and MAP functions of the AI RMF [17] and maintains an audit trail of decisions—important, because poorly grounded governance documents are ineffective in practice [4]. Section V-K describes generative-AI risk categories with regard to confabulation, data privacy, human-AI configuration and information integrity mapped to framework controls [16].

4. Methodology

4.1. Research Design

The methodology is split into two parts: evidence-based synthesis, in which the components of the framework are extracted from results in the literature, as well as specification of the empirical evaluation protocol needed for the framework. This paper reports the evidence synthesis and specifies a proposed evaluation protocol; it does not report a new user study, and every quantity in Section V is a value reported by the cited source under the condition of the cited source.

4.2. Corpus Assembly and Extraction

The corpus consists of the 28 documents collected as the reference corpus for this work. For inclusion, a document needed to include empirical results on the use of AI for interface generation or design-system tooling, or empirical or standards-based results on usable security, warning design, security operations or AI risk management. It spans ACM venues (CHI 2008 to CHI 2024, SOUPS 2016, and ACM Transactions journals), USENIX and IEEE security venues (USENIX Security 2013, IEEE Symposium on Security and Privacy 2007, IEEE SecDev 2016), IEEE journals and conferences (Transactions on Software Engineering, Access, VizSec 2017), HFES 2002, MDPI journals (Information and Sustainability), NAACL 2025, arXiv preprints, and two NIST publications (AI 100-1 and AI 600-1).

The evidence-extraction process involved four steps: (1) layout-preserving conversion of each PDF to text; (2) reading each document for statements containing a quantity, population, threshold, or validated guideline set; (3) mapping each statement to one or more framework components; and (4) the tabulation and source attribution in Table I. All tabulated values were programmed in accordance with the language used in their original source and any value that does not exactly match that language was dropped. Only the tabulated values are used in Sections V and VI, and no value appears in this paper that is not traceable to one of the 28 documents.

4.3. Derivation of Requirements, Metrics, and Targets

Framework requirements were derived by asking, for each mapped evidence item, what a design process would have to do differently to avoid the failure the evidence documents. The metrics and target thresholds were based on the same evidence, using a published threshold where available or a proposed threshold requiring empirical calibration; the distinction between these two cases is explicit in Table III.

4.4. Proposed Evaluation Protocol

There are eight metrics defined in the protocol. M1: Design-Sys Violation Rate (infractions/screen), broken down by rules. M2, compliance verdict reliability: agreement between automated verdicts and expert adjudication, reported as precision and recall per rule family, with a target of at least 90% completion and 95% confidence on at least 90% of cases. [18] M3, Flow coverage completeness: percentage of features having all four flow classes present and accessible. M4, determining security flow defects (missing confirmations, passive notices or missing consequence text per feature), and relevant to security, this is a high-level category. Pattern reuse ratio for M5, counter-metric is number of unvalidated patterns. M6, time from requirement set to first prototype passing the review gate. M7, iterations to design freeze, with defects resolved per iteration. M8 assessed reviewer-reported usefulness, trust calibration, and effort, in line with tools developed in human-AI interaction guideline evaluation [25] with explicit controls for the role-playing effect, where participants who were assigned a role seemed to behave significantly less securely than the other participants who were assigned to act on their personal risk [14].

The proposed evaluation will recruit 8–12 practitioners (depending on the case) including at least two cybersecurity product teams who each propose a comparable feature brief, but under different types of prototyping (ungoverned versus framework-governed) with balanced order. Prototype fidelity does not lead to confusion in the discovery of usability problems [11] so the low fidelity condition is methodologically acceptable and it reduces the cost of the comparison. Analysis is envisaged as within subject or matched case comparison of M1-M8 using non-parametric statistics and qualitative analysis of rationales provided by the reviewers. Like in many security-critical systems, where similar criteria were found to have weights ranging between 7.37% and 20.99% by six expert models, disagreement in the weights of usability and security criteria must be modelled rather than averaged [10].

4.5. Scope Note

Table I-VI are literature synthesis and design derivation and not experimental results. Table V presents an illustrative instantiation where the verdict is the output of specified checks applied to a specified prototype generated via inspection of a rule set instead of by a deployed system. This distinction continues in Section V.

5. Results and Discussion

5.1. RQ1 — What Does Published Evidence Quantify About AI-Assisted Interface Generation?

RQ1 asks how mature AI-assisted interface generation is and where it falls short. Table I is a summary of the quantitative findings from the literature reviewed and how they relate to the proposed framework. Design2Code shows that generation is possible at the scale of real-world product interfaces: 484 curated webpages with a mean code size of 31,216 tokens per file, a mean DOM depth of 13 and a mean number of HTML tags per example of 158 [1]. Interaction2Code highlights the challenges of reproducing interactive behaviour: its benchmark covers 127 webpages and 374 interactions and identifies ten categories of failure. Across the evaluated models, full-page CLIP similarity is lower than Interaction IR, including GPT-4o (0.7114 versus 0.7754), Claude 3.5 Sonnet (0.7172 versus 0.7914), Qwen2.5-VL-72B (0.6430 versus 0.6524), and Gemini-1.5-Flash (0.5967 versus 0.6738) [12]. Iterative feedback has been found to yield better generated outputs: After 5 rounds, sketch2code achieved visual-similarity gains of up to 7.1% and IoU-based layout gains of 2.7%. However, simulated-user feedback was meaningful in 83.3% of cases, and answers to expert questions were accurate in 86.7% of cases [19]. The results of this study provide evidence that suggests that one should not assume the visual plausibility of an interface as a sign of functional completeness but rather rely on explicit interactions validation.

5.2. RQ2 — What Do Usable-Security and SOC Findings Imply for Governance Requirements?

RQ2 asks what governance must enforce. Table II translates the evidence into ten verifiable requirements. The usable-security strand sets the consequence. The literature of usable security shows that the design of warnings can shape user behaviour; in the redesign of the Chrome HTTPS warning, the use of a new warning design was accompanied by a doubling of reported adherence, from 30% to 70% [15]; similarly, a laboratory study of a phishing-warning raised adherence to 79% for active warnings, compared with 13% for passive warnings [23]. In a separate study involving 67 bank customers, 23 of the 25 participants using their own accounts entered their passwords after security indicators were removed [14]. Field measurements also showed that the click-through rates were from 18.0% for Chrome phishing warnings to 81.8% for the worst-performing Chrome SSL variant with an untrusted issuer [26]. The following findings refer to various study contexts and to various outcome measures and are not to be understood as one comparable experiment. The security-operations literature also offers more background: Security teams report handling about 49% of incidents (nearly half), 51% of teams report being overwhelmed with alerts, and more than 25% of analyst time is spent on false-positives, according to Tariq et al. [13]. Baruwal Chhetri et al. found that around 40% of alerts were ignored and 49% were assumed to be false alarms, with 50% of analysts depending on colleagues for assistance [24]. These are contextual and not universal estimates of SOC performance, but rather figures representing different populations and instruments. The combined results lead to the following findings that motivate the requirements R3 (explicit flow coverage) and R4 (active security notices, acknowledgement, consequence communication, and safe defaults). For requirement R5, the separate requirement that colour shall not be the only indicator of a security-relevant meaning, 40 icons and seven strings were evaluated, resulting in 1329 responses to be analyzed, with the reported estimate of approximately 8% of men who are colour vision deficient [22] to take into account.

5.3. RQ3 — What Evaluation Protocol Can Convert the Framework Requirements Into Measured Claims?

RQ3 asks how the framework would convert requirements into measured claims. Table III specifies eight metrics with operational definitions, instruments, and targets, separating published from proposed targets. Only M2, conformance-verdict reliability, inherits a published quantitative threshold (at least 90% completion with 95% confidence on at least 90% of cases, from the hybrid UI-linting study [18]); the remaining seven are proposed and must be calibrated empirically. This is itself a finding: the systematic usable-security review located only 4 formal evaluation-model papers out of 55, and cited a preceding review in which only 6% of articles reached the final analytical stage [9]. The evidence base therefore does not yet support fixed targets for flow coverage, security-flow defects, pattern reuse, time-to-prototype, iterations-to-freeze, or reviewer experience, which is consistent with the small-sample multi-case design offered in Section IV-D and with the multi-criteria precedent in usable cybersecurity where Fuzzy-AHP expert weights span 7.37% to 20.99% across six expert models [10], demonstrating that stakeholder weighting itself must be modelled.

5.4. Synthesis of AI-Prototyping Performance

Figure 5 summarises the reported performance indicators for AI-assisted UI generation and iterative refinement. The accuracy of screenshot-to-code exceeds 77% [6] for the signal-rich configurations, UICoder raises SwiftUI compilability from 3% to 79-82% and Feedback-driven refinement increases CLIP from 0.334 to 0.404 and preference Elo from 773 to 1099 [2] for direct Sketch2Code generation, while open-source VLMs achieve correct formatted HTML accuracy of 82.4% (Llava-1.6) to 98.0% (InternVL2) with 7.1% visual similarity and 86.7% question-answer accuracy added by feedback refinement [19]. The human assessment of Design2Code outputs shows that 49% of the self-revision pages generated by GPT-4V were rated as being exchangeable with the reference, while 64% were rated as preferred [1]. The numbers support all three claims above: that AI prototyping is helpful in the early stages, that there is no such thing as a single-pass generation, and that

explicit feedback channels increase a measurable proportion of the attempts that are considered acceptable. The structural mechanisms that reverse this fraction from acceptance by default to acceptance labour are Capability 1 and the loop of Section III-E.

5.5. Visual Synthesis of SOC Operating Conditions

Figure 4 visualises the SOC strand values reported by Tariq et al. [13] (SOC survey synthesis) and Baruwal Chhetri et al. [24] (SOC field/survey study). The headline numbers are saturation: 94% of large corporations have a SOC, and 93% of analysts were open to an AI teammate; leakage of attention: 51% of analysts were overwhelmed, 27% of time went into false positive, 49% assumed false positive, only 49% of alerts were resolved, 40% ignored them altogether; colleague reliance: 50% of the time. The figure is not a single-population measurement, as the two studies were based on different populations and instruments. The framework translates this operating envelope into Requirement R4 (which represents the state of active security notices and safe defaults) and into the prototype call to validate against exception flows as the partial-attention state of the analyst is the norm.

5.6. Illustrative Instantiation: A Security-Operations Console

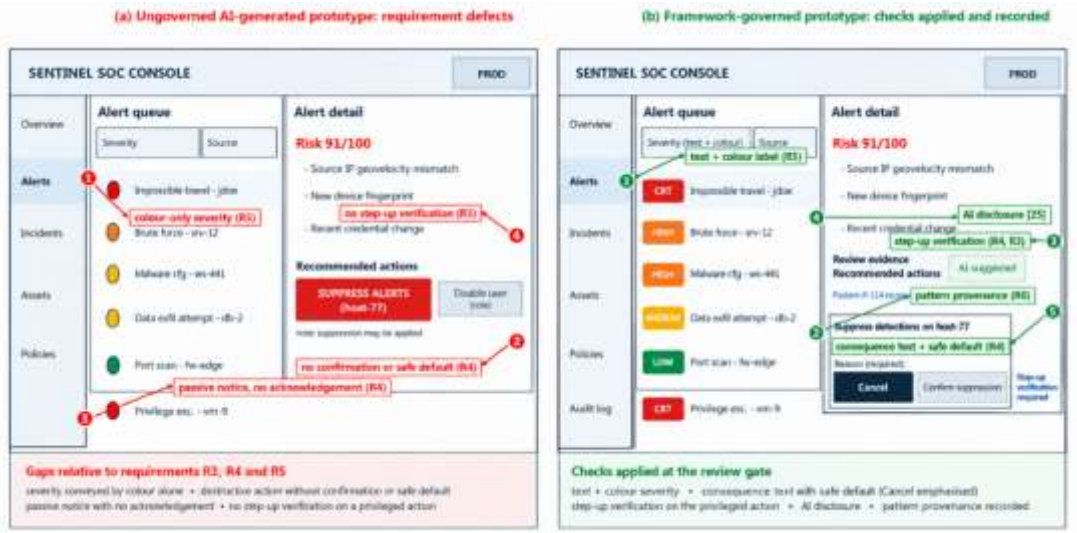


Figure 3. (a) Ungoverned Prototype; (b) Framework-Governed Prototype

Figure 3 shows an annotated mockup of a security-operations console that is in two states. The ungoverned prototype (Panel A) shows five numbered defect callouts covering three principal failure categories: colour-only severity, unsafe or insufficiently confirmed actions, and passive security notices without acknowledgement. First, severity is expressed by colour alone; Requirement R5 says that the security-related meaning should be understandable without colour [22]. Secondly, a destructive action is introduced without sufficient confirmation or safe default, which is contrary to Requirement R4 and evidence that active warnings are more adhered to than passive warnings (79% to 13%) [23]. thirdly, a passive, dismissible security note is not in line with Requirement R4 (active notice, acknowledgement, consequences and safe defaults). This reported increase is another indication that warning design can influence user behaviour, from about 30% to 70% after the Chrome HTTPS warning redesign [15]. The examples listed here do not constitute a complete list of possible prototype defects.

Panel B shows five governance checks. First, the severity pills include color with other information (such as text) or non-colour information, which supports Requirement R5, Capability 1. Secondly, a confirmation dialog provides feedback on consequences and highlights cancel as the safe choice, fulfilling R4, Capability 2. Third, a step-up verification indicator is displayed on the privileged-action path, to aid in the security-sensitive flow validation in Requirement R3 and the associated safe-action expectations in Requirement R4. Fourth, an AI-activity disclosure strip maps the human–AI interaction guidelines of capability disclosure, explanation, consequence communication, global controls, and change notification [25]. Fifth, a pattern-provenance citation identifies an approved registry entry, supporting the pattern-first governance approach under Requirement R6 and iteration traceability under Requirement R8. The figure is an illustrative instantiation of the proposed framework, not an empirical evaluation of its effectiveness.

Table V shows twelve representative checks applied to the prototype with the given rule set. The four checks (C4, C5, C6, and C11) result in a Revise verdict, a structured set of revision actions, and not an automatic acceptance. The verdicts in Table V are illustrative expected outputs from applying the specified rule set to the prototype; they are not measurements from a deployed system.

Table VI is a mapping table that illustrates four categories of potential risks in generative AI that are relevant to interface prototyping and their corresponding proposed framework controls. Structured verdicts in Capability 1 and the provenance log containing model, prompt and design-system version are designed to aid in the management of

confabulation risk [16, 17]. The no-production-data policy and review gate are intended to reduce data-privacy risk [3], [16]. Capabilities 2 and 3, along with the AI-activity disclosure strip, are designed to mitigate the risk of human–AI configuration, to include covering flows, tracing patterns back to their source, and appropriate disclosure [16], [25]. Finally, security-flow validation and security-restriction of unverified free-form copy in security-relevant surfaces are designed to reduce information-integrity risk [15, 16, 23]. These are proposed governance controls, which will need to be empirically evaluated.

5.7. Consolidated Findings

Three consolidated findings follow from Tables I, II, and III and Figures 4 and 5. First, AI-assisted prototyping is a measurable capability: from 0.03 to 0.82 compile rate [2], 77% screenshot-to-code accuracy [6], 49% of generated pages judged exchangeable with reference by humans [1], and up to 7.1% iterative improvement on visual similarity [19]. Second, AI-assisted prototyping is not a sufficient authority: usable-security measures show soft adherence (30-79%) and hard failure modes (92% credential entry after indicator removal [14], 81.8% bypass of the worst Chrome SSL variant [26]); SOC evidence shows the analyst environment the prototype must respect (51% overwhelmed [13]; 49% presumed false positives, 40% disregarded, 50% colleague reliance [24]). Third, M2 is the only metric in the proposed evaluation protocol that inherits a published quantitative target from the hybrid UI-linting study [18]; the remaining seven metrics are proposed and require empirical calibration.

5.8. Discussion of Implications for Product Teams

For product organisations, the question the evidence most directly answers is which parts of the AI-prototyping pipeline warrant governance investment. Three are evident. The first is the conformance bottleneck: design-system guidance rarely fits a single model context window and AI alone was found insufficient for adoption [18]; Capability 1 (hybrid heuristics plus model checks) is therefore the architecture supported by the literature, not a discretionary choice. The second is the security-sensitive flow surface: If teams find that a prototype is polished but incomplete, they can accept it as a specification, which is not found in the empirical literature where security goals of high visual similarity to generated artifacts are met [1] [19]; Capability 2 (R3-R5) addresses this by providing verifiable flow coverage. The third is creative latitude: low-fidelity prototypes can be as effective as high-fidelity prototypes at discovering usability problems in a comparison of 28 teams with 1,270 comments [11] and therefore a team's loss of fidelity-driven insight need not be offset with front-loading the validation process.

5.9. Discussion of Implications for Security Review and the SOC

The framework maps three categories of security-product organizations' SOC operational pain into concrete artefact requirements for security review. If the level of analysts overloading exceeds 50%, and the time on false-positive alerts exceeds 25% [13] then the information density of prototypes and the amount of alerts for users must also be controlled by artefacts (R3-R4). In the case of the 49%-presumed-false-positive rate and the 40%-disregarded rate [24], exception paths and free-form copy that are not verified by the AI pass through unchecked (R3-R4, R5, and the bounded AI-text policy associated with [25]). The 92% own-account credential entry after indicators were removed [14] and the 23.2% to 81.8% click-through rates by browser and variant [15], [26] mean security-relevant indicators must survive degradation with consequence text and a safe default (R4, R5), which Capability 2 enumerates as a check and the human reviewer accepts.

5.10. Discussion of Implications for AI Risk Management

The four categories of generative-AI risk identified in this paper are aligned to proposed framework controls in the NIST Generative AI Profile [16]. Structured verdicts in Capability 1 and a provenance log of the version of the design-system, prompt and model are intended to provide support for confabulation risk management [16, 17]. To minimize the data-privacy risk, a no-production-data policy and a review gate are proposed [3, 16]. To deal with human–AI configuration risk, we have Capability 2's flow-coverage matrix, Capability 3's pattern-provenance mechanism, and the AI-activity disclosure strip. Last but not least, security-flow validation and limitation of unverified free-form copy on security-relevant surfaces are supposed to reduce the information-integrity risk [15], [16] and [23]. These controls are suggested governance actions, not an action that proves the risks are gone. Their effectiveness, implementation feasibility, and impact on cybersecurity-product design outcomes require empirical evaluation.

5.11. Summary of Results and Discussion

The sequence of evidence synthesis, requirement derivation, metric specification, and illustrative instantiation demonstrates the framework's load-bearing chain. The evidence addresses RQ1 by characterizing AI-assisted interface generation as capable but still vulnerable to interaction-related failures, RQ2 by deriving governance requirements that prioritize security-sensitive flows and safe defaults, and RQ3 by specifying an eight-metric evaluation protocol in which M2 inherits a published quantitative target while the remaining seven metrics require empirical calibration. This chain is made visible by Fig. 3 and Table V which maps the chain to one prototype in two states, and is mapped to the AI RMF risk categories by Table VI. The value of the framework is thus not just the checks themselves, but the embedding of the language of generation, governance, review and risk-management into a single auditable process for prototyping cybersecurity products.

6. Threats to Validity

6.1. Construct Validity

The framework relies on a hybrid of heuristic and model-based checks (Capability 1) for which the only published quantitative target (90% completion with 95% confidence on 90% of cases) is specific to a design-system linting study using heuristic-enhanced UI checks [18]; the

target has not been reproduced in cybersecurity-product design systems or under registry-mediated pattern validation. Metrics M1, M3, M4, and M5 lack published quantitative targets and are drawn from a usable-security literature that has only 4 of 55 surveyed studies that provide formal evaluation models [9].

6.2. Internal Validity

The second threat is internal validity. The mapping from evidence to framework requirements is the authors' interpretive judgement; other mappings are possible for R4 (active notices) and R6 (pattern-first generation). There are laboratory and field studies of the browser phishing and HTTPS contexts that support the evidence chain from warning research to R4 and there is no direct evidence of the transfer process between the two that arises from warning research and R4 (alert suppression, step-up verification, privileged actions). The sketch-to-code evidence (Sketch2Code, 731 sketches; agent faithfulness 83.3%, accuracy 86.7%) [19] is for HTML sketches which are not security-product workflows, and the Interaction2Code measurements (127 webpages, 374 interactions) [12] are for webpage interaction, not SOC workflows.

6.3. External Validity

The third threat is external validity. The corpus consists of papers from the ACM, IEEE, and USENIX venues and English-language web and desktop contexts, and there is a single study on energy-management systems [10] and none on mobile-first or industrial control contexts. The alert-fatigue figures derive from a few survey-based sources with differing populations and instruments (51% / 54% in different sub-sections of [13]; 49% presumed false positives and 40% disregarded in [24]). Several useful-security results are based on lab experiments that involve small samples (60 participants in [23] and 67 bank customers with 25 using their own accounts in [14]). The warning literature covers over 10 years of interface changes, and older adherence and bypass data refer to interfaces that no longer exist (Chrome HTTPS warning cases date back to 2013 on [15] and Chrome malware / phishing cases date back to 2013 on [26]). Benchmark complexity statistics are for curated webpage collections and not for internal enterprise applications (Design2Code 484 webpages, mean 31,216 tokens [1] and Interaction2Code 127 webpages, 374 interactions [12]), and design-to-code performance indicators have different metric definitions, making cross-study comparisons indicative only.

Second, all of the 4 models evaluated, and the 6% figure reported earlier in the usable-security systematic review, apply a stricter screen at the synthesis stage that excludes relevant work on source coding [9]; also, the Albarrak fuzzy-AHP weights (7.37-20.99% across 6 models) [10] indicate that expert weighting is an interpretive process as well; and the Gorski eye-tracking study with N=49 measured attention and CSP-implementation outcomes under controlled documentation, but not under production documentation [3]. These are all transferable limits.

6.4. Reliability

The fourth threat is reliability. The framework assumes that the organization has a codified design system and a maintained pattern registry; otherwise, it would be impossible to have machine-checkable checks; token and component catalogues would be needed before being able to use Capability 1; and the invalidated-pattern count in M5 would be high always if the pattern registry were not maintained. The proposed evaluation design of 8-12 practitioners per case means that small effect sizes are possible but unlikely to be detected, and is designed as a first validation with case-level triangulation, and not as a confirmatory trial. The proposed protocol introduces explicit control conditions that are required for role playing effects as documented in [14] but are not measured in this paper.

7. Conclusion and Future Work

7.1. Conclusion

This paper introduced a conceptual model for AI-aided UX engineering for cybersecurity products. The framework views generative AI as a limited drafting tool with three governance mechanisms that involve design-system compliance by returning verdicts at token, component, and state levels for a versioned rule set; user-flow validation by treating main, alternative, exception and security-sensitive flows as mandatory artifacts that are validated against an enumerated catalogue; and pattern-based prototyping by preferring validated patterns over new invention, and highlighting unvalidated patterns for human review before inclusion in the registry. These capabilities are executed within an iterative sequence where results must be submitted for a named human review verdict and generate a set of artefacts that align with the GOVERN and the MAP functions of the NIST AI Risk Management Framework.

Three durable conclusions are supported by Tables I-III and Figures 4-5. Deterministic rules and model-based assessment are complementary, and can be combined to facilitate design-system conformance checking, as per previous research on UI-linting [18]. The published quantitative targets, however, have not yet been validated for cybersecurity-product design systems in the framework proposed in this paper. Interface refinement can be facilitated by iterative agent feedback, as suggested in the Sketch2Code study, which received meaningful simulated-user feedback in 83.3% of the cases and accuracy in answering expert questions of 86.7% [19]. These results are used to provide feedback and are helpful for refinement, but also demonstrate the importance of human review. Benchmark interaction is the recurring failure axis (Implement Rate lags Full-Page CLIP for every model in [12]). Security interaction quality is decided by flow design rather than visual polish: active vs passive warning adherence is 79% vs 13% [23], adherence rises from 30% to 70% after warning redesign [15], 23 of 25 own-account participants entered passwords after indicators were removed [14], and click-through rates by browser and variant range from 18.0% to 81.8% [26]. Governance artifacts survive only when grounded in practitioner workflows [4], which is why the framework

specifies checks, registries, and audit records rather than principles alone.

7.2. Future Work

Five research directions follow. First, the evaluation protocol in Section IV-D should be performed on a multi-case study basis across at least two cybersecurity product teams, but with the distinction between published-versus-proposed rule families that was provided in Table III preserved; and second, the conformance rule families should be applied and optimized for real cybersecurity-product design systems to achieve attainable precision and recall, beyond web-style component libraries as in [18]. Third, a catalogue of security-flow patterns must be built and tested, based on [3], [15], [22], [23], [26] and on threat-model-driven analyst workflows [21]. Fourth, the impact of the framework on team collaboration and role boundaries (specifically the change of the UX-designer from early drafts to constraint specification and pattern curation) needs to be explored longitudinally, because the framework reallocates UX early-stage labour [8]. Fifth, transfer should be tested before the framework can be generalized, in areas that have similar usability-security tensions, such as critical-infrastructure energy management [10]. In the meantime, organizations can implement this framework as a structured approach to working with generative AI as a design assistance tool, rather than supplanting existing UX engineering and design governance practices.

References

[1] C. Si, Y. Zhang, R. Li, Z. Yang, R. Liu, and D. Yang, "Design2Code: Benchmarking multimodal code generation for automated front-end engineering," arXiv:2403.03163, 2024.

[2] J. Wu, E. Schoop, A. Leung, T. Barik, J. P. Bigham, and J. Nichols, "UICoder: Finetuning large language models to generate user interface code through automated feedback," arXiv:2406.07739, 2024.

[3] P. L. Gorski, S. Moller, S. Wiefling, and L. Lo Iacono, "'I just looked for the solution!' On integrating security-relevant information in non-security API documentation to support secure coding practices," IEEE Trans. Softw. Eng., vol. 47, no. 6, pp. 1255-1269, 2021.

[4] M. A. Madaio, L. Stark, J. Wortman Vaughan, and H. Wallach, "Co-designing checklists to understand organizational challenges and opportunities around fairness in AI," in Proc. CHI 2020, doi: 10.1145/3313831.3376445.

[5] U. O. Nwokedi, B. A. Onyimbo, and B. B. Rad, "Usability and security in user interface design: A systematic literature review," Int. J. Inf. Technol. Comput. Sci., vol. 8, no. 5, pp. 72-80, 2016, doi: 10.5815/ijitcs.2016.05.08.

[6] T. Beltramelli, "pix2code: Generating code from a graphical user interface screenshot," arXiv:1705.07962, 2017.

[7] T.-S. Kim, M. J. Ignacio, S. Yu, H. Jin, and Y.-G. Kim, "UI/UX for generative AI: Taxonomy, trend, and challenge," IEEE Access, vol. 12, 2024, doi: 10.1109/ACCESS.2024.3502628.

[8] Q. V. Liao, M. Vorvoreanu, H. Subramonyam, and L. Wilcox, "UX matters: The critical role of UX in responsible AI," Interactions, vol. 31, no. 4, pp. 28-35, Jul.-Aug. 2024.

[9] F. Di Nocera, G. Tempestini, and M. Orsini, "Usable security: A systematic literature review," Information, vol. 14, no. 12, Art. 641, 2023, doi: 10.3390/info14120641.

[10] A. M. Albarrak, "Integration of cybersecurity, usability, and human-computer interaction for securing energy management systems," Sustainability, vol. 16, no. 18, Art. 8144, 2024, doi: 10.3390/su16188144.

[11] M. Walker, L. Takayama, and J. A. Landay, "High-fidelity or low-fidelity, paper or computer? Choosing attributes when testing web prototypes," in Proc. HFES 46th Annu. Meet., 2002, pp. 661-665.

[12] J. Xiao, Y. Wan, Y. Huo, Z. Wang, X. Xu, W. Wang, Z. Xu, Y. Wang, and M. R. Lyu, "Interaction2Code: Benchmarking MLLM-based interactive webpage code generation from interactive prototyping," arXiv:2411.03292, 2024.

[13] S. Tariq, M. Baruwal Chhetri, S. Nepal, and C. Paris, "Alert fatigue in security operations centres: Research challenges and opportunities," ACM Comput. Surv., vol. 57, no. 10, Art. 240, Oct. 2025, doi: 10.1145/3723158.

[14] S. E. Schechter, R. Dhamija, A. Ozment, and I. Fischer, "The emperor's new security indicators: An evaluation of website authentication and the effect of role playing on usability studies," in Proc. IEEE S&P, 2007, doi: 10.1109/SP.2007.35.

[15] R. W. Reeder, A. Porter Felt, S. Consolvo, N. Malkin, C. Thompson, and S. Egelman, "An experience sampling study of user reactions to browser warnings in the field," in Proc. CHI 2018, doi: 10.1145/3173574.3174086.

[16] National Institute of Standards and Technology, "Artificial intelligence risk management framework: Generative artificial intelligence profile," NIST AI 600-1, Jul. 2024, doi: 10.6028/NIST.AI.600-1.

[17] National Institute of Standards and Technology, "Artificial intelligence risk management framework (AI RMF 1.0)," NIST AI 100-1, Jan. 2023, doi: 10.6028/NIST.AI.100-1.

[18] T. Knearem, Y. Lu, S. Dutta, J. Blass, C. Kliman-Silver, and F. Bentley, "AI is not enough: A hybrid technical approach to AI adoption in UI linting with heuristics," in Proc. CHI 2024, doi: 10.1145/3613905.3637135.

[19] R. Li, Y. Zhang, and D. Yang, "Sketch2Code: Evaluating vision-language models for interactive web design prototyping," in Proc. NAACL 2025 (Long Papers), pp. 3921-3955, Apr. 2025.

[20] X. Hou, Y. Zhao, Y. Liu, Z. Yang, K. Wang, L. Li, X. Luo, D. Lo, J. Grundy, and H. Wang, "Large language models for software engineering: A systematic literature review," ACM Trans. Softw. Eng. Methodol., vol. 33, no. 8, Art. 220, 2024, doi: 10.1145/3695988.

[21] L. Franklin, M. Pirrung, L. Blaha, M. Dowling, and M. Feng, "Toward a visualization-supported workflow for cyber alert management using threat models and human-

centered design," in Proc. IEEE VizSec, 2017, doi: 10.1109/VIZSEC.2017.8062200.
[22] A. Porter Felt, R. W. Reeder, A. Ainslie, H. Harris, M. Walker, C. Thompson, M. E. Acer, E. Morant, and S. Consolvo, "Rethinking connection security indicators," in Proc. SOUPS 2016, doi: 10.1145/3670009.
[23] S. Egelman, L. F. Cranor, and J. Hong, "You've been warned: An empirical study of the effectiveness of web browser phishing warnings," in Proc. CHI 2008, doi: 10.1145/1357054.1357219.
[24] M. Baruwal Chhetri, S. Tariq, R. Singh, F. Jalalvand, C. Paris, and S. Nepal, "Towards human-AI teaming to mitigate alert fatigue in security operations centres," ACM Trans. Interact. Intell. Syst., vol. 24, no. 3, Art. 12, Jul. 2024, doi: 10.1145/3670009.
[25] S. Amershi, D. Weld, M. Vorvoreanu, A. Fourney, B. Nushi, P. Collisson, J. Suh, S. Iqbal, P. N. Bennett, K. Inkpen, J. Teevan, R. Kikin-Gil, and E. Horvitz, "Guidelines for human-AI interaction," in Proc. CHI 2019, doi: 10.1145/3290605.3300233.
[26] D. Akhawe and A. Porter Felt, "Alice in warningland: A large-scale field study of browser security warning effectiveness," in Proc. 22nd USENIX Security Symp., 2013.
[27] Y. Acar and S. Fahl, "You are not your developer, either: A research agenda for usable security and privacy research beyond end users," in Proc. IEEE SecDev, 2016, doi: 10.1109/SecDev.2016.013.
[28] A. Adams and M. A. Sasse, "Users are not the enemy," Commun. ACM, vol. 42, no. 12, pp. 40-46, Dec. 1999, doi: 10.1145/322796.322806.

TABLES and FIGURES

TABLE I
Evidence log: quantitative findings extracted from the corpus and mapped to framework components.

Ref.	Source / Venue	Reported finding (verbatim or directly quoted)	Framework component
[1] Si et al., 2024	Design2Code	484 webpages; mean 31,216 tokens/file (min 784, max 98,637, SD 23,903); mean DOM depth 13 (min 4, max 32); mean 158 HTML tags (min 12, max 528). Human eval: 49% of GPT-4V self-revision pages exchangeable; 64% preferred over reference.	Scale, Capability 3
[2] Wu et al., 2024	UICoder	5 iterations; ~1M generated SwiftUI programs; initial mined 0.4%. Compile: StarChat-Beta 0.03 -> UICoder-Filtered 0.79 / Top 0.82 / DPO 0.75. CLIP 0.334 -> 0.404. Elo 773 -> 1099.	Capability 1 (automated verdicts)
[3] Gorski et al., 2021	IEEE TSE	N=49 eye-tracking between-group study; secure CSP: 33% (task 3), 30% (task 4), control 0%; 15/49 copy-pasted supplied header; 9/13 code-comment group aware via code comment.	Capability 2; governance artifacts
[4] Madaio et al., 2020	CHI 2020	48 practitioners, 12 companies, 37 products; NLP 10, CV 10, predictive analytics 9, conversational AI 6, IR/search 6.	Governance grounded in workflow
[5] Nwokedi et al., 2016	IJITCS	Systematic review of usability and security in UI design; 90% successful access-code entry reported as a design-dependent outcome.	Capability 2 (authentication flows)
[6] Beltramelli, 2017	pix2code	>77% accuracy on iOS, Android, web; greedy error: iOS 22.73%, Android 22.34%, web 12.14%; 1,500 training instances per platform.	Prototyping baseline
[7] Kim et al., 2024	IEEE Access	Taxonomy, trends, and challenges for UI/UX of generative AI.	Framework positioning
[8] Liao et al., 2024	Interactions	UX role in responsible AI across the development lifecycle.	Governance ownership
[9] Di Nocera et al., 2023	Information	55 studies: authentication 34 (56%), developer support 7, design strategies 10, formal models 4; preceding review retained only 6% at final analytical stage.	Capability 2; evaluation gap
[10] Albarrak, 2024	Sustainability	EMS CAGR 15.65% (2024-2029) and 14.70% (to 2032); Fuzzy-AHP weights of six existing models 7.37% to 20.99% (EM#2-EM#6).	Metric prioritization
[11] Walker et al., 2002	HFES	28 participants, 1,270 comments (M=45, SD=18); low=high fidelity; issues: match 28%, visibility 17%, aesthetics 10%.	Prototype fidelity; early validation
[12] Xiao et al., 2024	Interaction2Code	127 webpages, 374 interactions, 15 topics, 31 categories. Full-page CLIP / Interaction IR: GPT-4o 0.7114/0.7754;	Capability 2; interaction risk

		Claude-3.5-Sonnet 0.7172/0.7914; Qwen2.5-VL-72B 0.6430/0.6524; Gemini-1.5-Flash 0.5967/0.6738. Ten failure types.	
[13] Tariq et al., 2025	ACM Comput. Surv.	94% of large corporations operate a SOC; 51% (54% in sub-section) teams overwhelmed; >25% (27%) time on false positives; 63% burnout; >80% workload increase; 93% open to AI; teams resolve ~49% of alerts.	Domain context; flow requirements
[14] Schechter et al., 2007	IEEE S&P	67 bank customers; 23/25 (92%) own-account participants entered passwords after site-auth image removal; no participant withheld passwords; role-playing significantly reduced security behaviour.	Evaluation controls; flow validation
[15] Reeder et al., 2018	CHI 2018	Earlier Chrome HTTPS warning 30% adherence -> redesigned up to 70%. Weekly: SSL 3.4%, malware 0.4%, phishing 0.2%. 79% active vs 13% passive adherence; 68% early Firefox HTTPS click-through cited.	Capability 2 (warning flows)
[16] NIST AI 600-1, 2024	NIST	12 generative-AI risk categories: confabulation, dangerous-or-violent recommendations, data privacy, value chain and component integration, harmful bias and homogenization, CBRN information, human-AI configuration, obscene-or-abusive content, information integrity, information security, intellectual property.	Governance; risk register
[17] NIST AI 100-1, 2023	NIST	GOVERN, MAP, MEASURE, MANAGE; trustworthy-AI characteristics including validity, safety, security and resilience, accountability, explainability, privacy, and fairness.	Governance structure
[18] Knearem et al., 2024	CHI 2024	Manual UI linting time-consuming; design-system guidance rarely fits one context window; AI alone insufficient; targets >=90% completion with 95% confidence on >=90% of cases.	Capability 1
[19] Li et al., 2025	NAACL 2025	731 sketches from 484 webpages (18.0% by two designers, 16.5% by 3+); feedback meaningful 83.3%, answers accurate 86.7%; +7.1% visual similarity, +2.7% IoU layout over 5 rounds; correctly formatted HTML 82.4% (Llava-1.6)/98.0% (InternVL2); best-model text-augmented layout: Claude 3.5 Sonnet 22.26, GPT-4o 21.33, Gemini 1.5 Pro 18.72.	Capabilities 2-3; iterative loop
[20] Hou et al., 2024	ACM TOSEM	395 primary studies; token-based inputs ~97.75% of studies; reported Codex (12B) solved 72.31% of complex Python problems.	Prototyping and implementation
[21] Franklin et al., 2017	IEEE VizSec	Threat-model-driven, human-centred analyst workflow; ATT&CK-style frame recommended.	Capability 3 (pattern library)
[22] Porter Felt et al., 2016	SOUPS	1,329 respondents; 40 icons + 7 strings; inter-coder Fleiss kappa 0.89 (0.81 earlier round); 91% coders agreed; ~8% of men colour-blind.	Capability 1 (accessibility); Capability 3
[23] Egelman et al., 2008	CHI 2008	60 participants; 97% fell for >=1 phishing message; 79% active vs 13% passive adherence; 6/10 passive-condition participants never noticed warning.	Capability 2 (warning flows)
[24] Baruwal Chhetri et al., 2024	ACM TIIS	Cybercrime $7.08T (2022) -> $13.82T by 2028; 600% increase since COVID-19; ~80% of surveyed orgs operate SOC 24/7 (SANS 2023); ~40% alerts disregarded; 49% presumed false positives; 50% colleague reliance.	Domain context; human-AI teaming
[25] Amershi et al., 2019	CHI 2019	18 guidelines; 4-phase process; user study 49 practitioners x 20 AI products; G1 capabilities, G11 explanations, G16 consequences, G17 global controls, G18 change notifications.	Capability 3; governance
[26] Akhawe & Felt, 2013	USENIX Sec	Click-through: Chrome malware 23.2%, Chrome phishing 18.0%, Firefox SSL 33.0% (beta), Chrome SSL 70.2% (stable); worst Chrome SSL variant (untrusted	Capability 2 (warning patterns)

		issuer) 81.8%.	
[27] Acar & Fahl, 2016	IEEE SecDev	Developers and their tooling as the relevant user population.	Governance audience
[28] Adams & Sasse, 1999	Commun. ACM	Users compromise mechanisms due to cognitive and procedural burden; remediation through design and education rather than user blame.	Foundational motivation

TABLE II
Derived framework requirements R1-R10 with source evidence

ID	Requirement	Source evidence
R1	Generated prototypes must be evaluated against a versioned design-system rule set covering colour tokens, typography, spacing, components, state completeness, accessibility, and content tone.	[18], [22]
R2	Conformance verdicts must report confidence; they must return "no approved pattern exists" rather than substituting an invented pattern.	[18], [25]
R3	Every feature must enumerate main, alternative, exception, and security-sensitive flows; security-sensitive flows must be reviewed against a catalogue.	[9], [14], [15], [23], [26]
R4	Security notices must be active, require acknowledgement, and state consequences; destructive and privileged actions must present a safe default.	[15], [23], [25], [26]
R5	Colour must never be the sole carrier of security-relevant meaning.	[22]
R6	Generation must be pattern-first against a maintained registry; unvalidated patterns must be flagged and human-approved before reuse.	[12], [19], [21], [25]
R7	A prototype is never a specification: design freeze requires a named human review verdict after the validation loop.	[11], [18], [19]
R8	Each iteration must append a provenance log recording model, prompt, design-system version, and verdicts.	[16], [17]
R9	Low-fidelity artifacts are acceptable inputs to validation, since fidelity does not determine usability-problem discovery.	[11]
R10	Evaluation must control for role-playing and self-report bias when assessing security behaviour.	[14]

TABLE III
Proposed evaluation metrics, instruments, and targets. Reference indicates a threshold inherited from a published source; Proposed indicates a target that requires empirical calibration

ID	Metric	Instrument	Target	Status
M1	Design-system violation rate per screen, by rule family	Capability 1 rule engine output	Baseline; reduction vs ungoverned	Proposed
M2	Conformance verdict reliability (precision/recall vs expert adjudication)	Automated verdicts vs two-reviewer adjudication	>=90% completion; 95% confidence on >=90% of cases	Reference [18]
M3	Flow-coverage completeness (features with all four flow classes reachable)	Flow-coverage matrix	100% coverage at design freeze	Proposed
M4	Security-sensitive flow defects per feature	Catalogue audit (confirmations, consequence text, acknowledgement, safe default)	Zero critical defects at design freeze	Proposed
M5	Pattern reuse ratio; unvalidated-pattern count	Pattern-reuse report	Increasing reuse; zero unapproved patterns at freeze	Proposed
M6	Time to first reviewable prototype	Provenance-log timestamps	Reduction vs baseline	Proposed
M7	Iterations to design freeze; defects resolved per iteration	Loop records	Decreasing iterations with non-decreasing defect resolution	Proposed
M8	Reviewer-reported usefulness, trust calibration, and effort	Adapted human-AI evaluation instrument [25]; role-play controls [14]	Acceptance at pre-registered threshold	Proposed

V-E. Visual synthesis of SOC operating conditions

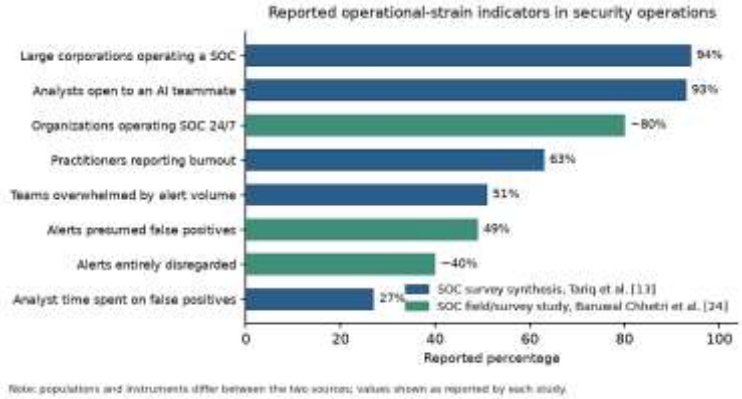


Fig. 4. Reported operational-strain indicators in security operations. Values are presented as reported by Tariq et al. [13] (SOC survey synthesis) and Baruwal Chhetri et al. [24] (SOC field/survey study); populations and instruments differ between the two sources

V-D. Synthesis of AI-prototyping performance

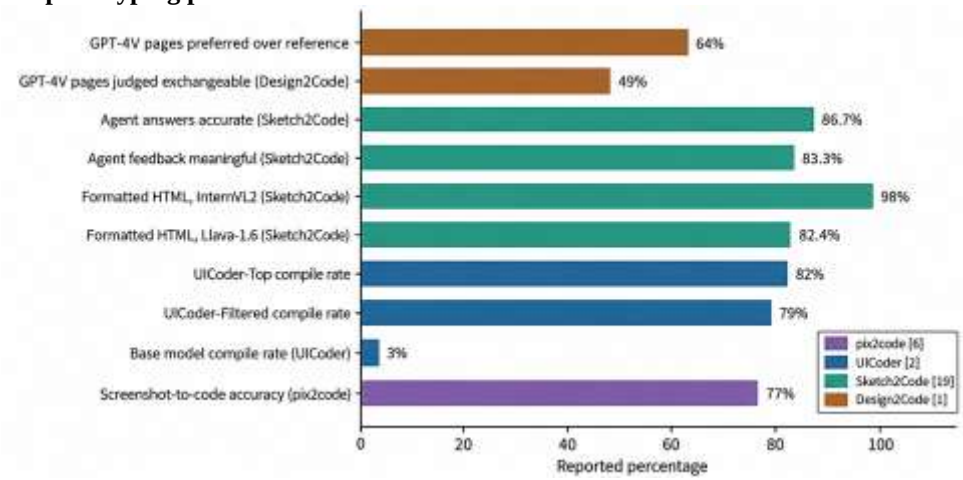


Fig. 5. Reported performance indicators of AI UI-generation and refinement studies [1], [2], [6], [19]. Metrics, tasks, and populations differ across studies; values are shown for magnitude reference only, not for direct comparison.

TABLE IV

Reported quantitative performance indicators across the AI UI-generation corpus.

Indicator	Benchmark / source	Reported value	Ref.
Screenshot-to-code accuracy (single-image)	pix2code	> 77% across iOS, Android, web	[6]
Compile rate (StarChat-Beta baseline)	UICoder	0.03	[2]
Compile rate top configuration	UICoder-Top	0.82	[2]
Improvement in CLIP after filtering	UICoder	0.334 -> 0.404 (Elo 773 -> 1099)	[2]
Direct-prompting correct HTML rate (open)	Sketch2Code	Llava-1.6 82.4%; InternVL2 98.0%	[19]
Best-model text-augmented layout similarity	Sketch2Code	Claude 3.5 Sonnet 22.26; GPT-4o 21.33	[19]
Iterative agent feedback (mean rounds)	Sketch2Code	Meaningful 83.3%; Acc@questions 86.7%	[19]
Iterative visual-similarity gain (5 rounds)	Sketch2Code	+7.1% visual; +2.7% IoU layout	[19]
Human preference (Design2Code, GPT-4V self-revision)	Design2Code	Exchangeable 49%; Preferred over reference 64%	[1]
LLM4SE: token-based input share	TOSEM SLR	~97.75% of 395 studies	[20]

TABLE V

Illustrative instantiation: representative checks applied to the security-operations console prototype. Verdicts are expected outputs of the specified rule set applied to the specified prototype, not measurements from a deployed system.

Check	Rule family	Prototype element	Verdict
C1	Colour-token conformance	Severity pills in the alert	Pass: pills mapped to approved semantic severity

		queue	tokens (Fig. 3 B, callout 1)
C2	Colour independence	Severity pills and status indicators	Pass: each pill carries a text label in addition to colour [22]
C3	Typography and spacing tokens	Header, column headers, table cells	Pass: token scale and grid applied
C4	Component identity	Filter row and pagination controls	Revise: a non-approved control was substituted for the design-system filter control
C5	Interaction state completeness	Alert rows and action buttons	Revise: loading and disabled states missing on two controls
C6	Exception flow coverage	Alert detail retrieval	Revise: timeout and partial-data paths not represented
C7	Security-consequence communication	Privileged acknowledgement dialog	Pass: consequence text present with a required reason field [25]
C8	Safe-default emphasis	Acknowledge or suppress alert	Pass: Cancel is the emphasised default [23], [26]
C9	Step-up verification	Privileged action gate	Pass: step-up chip present on the privileged path [9], [14]
C10	AI-activity disclosure	Suggested-response area	Pass: activity notice present for AI-suggested content [25]
C11	Pattern provenance	Whole composable block	Revise: two composable blocks outside the registry marked unvalidated
C12	Accessibility baseline	Contrast and target size	Pass with one advisory on a secondary label

TABLE VI
Mapping of NIST AI 600-1 generative-AI risk categories [16] to framework controls

GenAI risk category [16]	**Framework control**	**Sources**
Confabulation	Capability 1 structured verdicts; provenance log records model, prompt, design-system version.	[16], [17]
Data privacy	No-production-data policy in requirement set, enforced at provenance and review gate.	[3], [16]
Human-AI configuration	Capability 2 flow-coverage matrix; Capability 3 pattern provenance; AI-activity disclosure strip.	[16], [25]
Information integrity	Capability 2 prohibits free-form copy in security-relevant surfaces; rule R4 requires active notices with consequence text.	[15], [16], [23]