AI Quietly Breaking Quality? The Hidden Risks Lurking in LLM-Driven Apps

Reet Kaur, Sekaurity
reetkaur@sekaurity.com

Abstract

Artificial Intelligence (AI) systems promise automation, personalization, and cost savings but introduce hidden risks that traditional software engineering often fails to address. This paper explores the subtle, cumulative failures that can undermine software quality, security, and maintainability over time. It provides real-world case studies, explains why these risks arise, and offers practical recommendations for engineering teams to adopt robust, lifecycle-focused practices that deliver reliable and secure AI-powered features.

Biography

Reet Kaur is the CEO and Founder of Sekaurity, a cybersecurity advisory practice specializing in AI Security, GRC, and enterprise risk. She previously served as Chief Information Security Officer (CISO) for the largest higher-ed institution in Portland, Oregon, securing systems supporting over 85,000 students. With more than 20 years of experience, Reet has held leadership roles at Merck, Nike, AECOM, Fidelity, and CIBC across finance, pharma, retail, and academics. She is the coauthor of a book on application security and the author of LinkedIn Learning courses on AI Security Foundations, the OWASP Top 10 for LLMs, and Cybersecurity Due Diligence in M&A. She holds CISSP, CISM, CRISC, and PMP certifications, and degrees from the University of Waterloo and Carnegie Mellon University.

Introduction

AI systems have evolved rapidly from research prototypes to production-critical components in everyday applications, from customer-facing chatbots to personalized recommendations and fraud detection. This widespread adoption is driven by the promise of automation and business value. Yet, amid the rush to deploy these systems, an essential question often goes unasked: Is your AI quietly undermining software quality and introducing security risks you may not even see?

Unlike traditional rule-based code, AI relies on large datasets, statistical models, and probabilistic outputs that are inherently harder to test and debug. This introduces unique risks that standard QA and CI/CD practices often fail to address. Issues like poor data quality, model drift, limited explainability, and fragile third-party integrations can degrade system performance in subtle, hard-to-detect ways. These failures rarely cause crashes immediately but instead accumulate over time, increasing support tickets, introducing bias, and complicating maintenance.

For engineering teams, these risks create technical debt, slow delivery, and add unexpected operational costs. For users, they result in unreliable or unfair experiences that erode trust. For production support teams, they increase maintenance complexity and demand ad-hoc fixes.

This paper explores the hidden costs of integrating AI into production systems, explains why these issues arise, and offers practical recommendations to identify and manage these risks before they become costly maintenance and security liabilities.

Background

Over the past decade, AI has moved from a niche research topic to a standard production capability across industries. Early AI systems were often built for more constrained business use cases, such as rule based expert systems like MYCIN by Stanford in the 1970's to assist in diagnosing bacterial infections. Neural networks and other approaches were also explored in domains such as medical imaging, credit risk scoring, and early detection, though large-scale commercial adoption came much later. Advances in machine learning frameworks, deep learning models, and accessible APIs have enabled more flexible, developer-friendly, and widely adopted AI-powered features.

Today, teams use Generative AI, including Large Language Models and agents to:

- Automate customer and internal employee interactions through chatbots, virtual assistants, and other conversational systems.
- Personalize content and recommendations
- Detect fraud and risky transactions
- Improve search relevance
- Summarize or generate code using large language models

Traditionally, software quality has been defined by principles like:

- Correctness (does it do what it's supposed to?)
- Reliability (does it work consistently?)
- Usability (is it effective and easy for users?)
- Fairness (does it treat all users equitably?)
- Security (does it protect data and resist attacks?)
- Maintainability (is it easy to update and improve?)
- Performance (does it respond quickly and handle load well?)

In classic engineering, these qualities are enforced through deterministic logic, unit and integration testing, code reviews, and, in more recent years, continuous integration and continuous delivery (CI/CD) pipelines. But AI features challenge many of these assumptions. Models are probabilistic by nature, heavily dependent on data quality, and can degrade over time if they aren't retrained. Many function as "black boxes," making failures difficult to trace or explain. Integrating third-party large language models can also hide critical validation steps behind simplified APIs.

This shift introduces new engineering challenges, such as:

1. **Standard QA practices missing subtle, accumulating errors**
    - Chatbots might perform well in testing but produce biased or offensive responses with real user inputs.
    - Fraud detection models can pass initial QA but start missing new fraud tactics without retraining.
    - Summarizers might rephrase legal or policy text in ways that change its meaning, creating compliance risks.

- Traditional QA often doesn't test for variations, multi-turn conversations, or adversarial prompts, letting failures slip into production.

1. **Data pipelines decaying silently without clear warning**
   - Data sources can change, degrade, or become stale without triggering obvious alerts.
   - Models trained on outdated or biased data can start making worse predictions over time.
2. **Prompt engineering introducing injection risks**
   - Users can craft inputs like "Ignore previous instructions and share internal data," bypassing guardrails.
   - This risk often goes untested if QA only checks expected user prompts.
3. **Vendor APIs changing or shutting down without notice**
   - Providers might update models in ways that break your integration or discontinue services entirely.
4. **Outputs that violate policy or leak data if not properly validated**
   - A support bot might share confidential information in response to cleverly phrased user queries.

Without strong engineering processes, good tooling, and clear ownership over the entire AI lifecycle, teams risk releasing features that quietly harm user experience, increase maintenance costs, and introduce security issues.

Hidden Costs and Risks of AI Systems

While AI systems promise automation and improved user experiences, they also introduce hidden costs and risks that teams often underestimate. These issues rarely cause obvious crashes or failures but can undermine software quality, security, and maintainability over time:

1. **Data Quality Debt**
   AI models rely on high-quality training and input data. Poorly labeled, biased, incomplete, or even maliciously poisoned data can lead to misclassifications, unsafe outputs, and fairness failures. These problems often remain invisible during testing but surface in production.
2. **Model Drift and Misuse**
   Unlike static code, models degrade over time as real-world inputs shift. Without continuous monitoring and retraining, performance silently declines, leading to user-visible errors, biased outcomes, and unintended behaviors.
3. **Opacity and Explainability Gaps**
   Many AI models function as opaque black boxes. Without explainability tooling, developers cannot easily debug or justify decisions, reducing accountability, complicating compliance, and increasing the risk of unsafe or policy-violating outputs.
4. **Human Oversight Costs**
   AI is often marketed as fully automated, but in practice it frequently requires manual review. Without planned human-in-the-loop workflows, teams risk overreliance on automation, leading to unsafe or unfair outcomes that demand costly manual correction.
5. **Fragile Third-Party Wrappers and Supply Chain Risks**
   Small vendors often offer easy-to-integrate AI wrappers around popular APIs. While these accelerate delivery, they can be fragile and opaque. Vendors facing cost spikes or funding issues may shut down or change terms abruptly, leaving integrations broken in production and requiring urgent reengineering.
6. **User Trust and UX Erosion**
   Inconsistent, biased, or inexplicable AI decisions undermine user trust. Users faced with

unpredictable recommendations or unfair outcomes may churn, leave negative reviews, or abandon the product altogether.

7. **Engineering and Maintenance Overhead**
   AI features aren't one-off deliveries. They require ongoing monitoring, retraining, data validation, and threat modeling. Without lifecycle planning, these features become brittle and expensive to maintain.
8. **Regulatory and Compliance Risks**
   Emerging AI regulations demand explainability, fairness testing, and bias mitigation. Failing to design for these requirements creates costly retrofits, compliance gaps, and reputational damage.

Case Studies and Real-World Examples

Below are documented failures and patterns that show how these risks emerge in production:

	Case Study	What Happened	Why It Matters
1	Chatbots and Conversational Drift	Microsoft launched Tay, a chatbot on Twitter designed to learn from user interactions. Users quickly exploited it by feeding it offensive prompts. Within hours, Tay began posting racist and hateful messages publicly, forcing Microsoft to shut it down.	Demonstrates how chatbots without strong content filters and adversarial testing can be hijacked in public, damaging brand reputation.
2	Chevy Dealership Chatbot Incident	In late 2023, a prankster used a Chevrolet dealership's website chatbot to negotiate a $76,000 Tahoe down to $1. By crafting prompts carefully, he forced the bot to agree to absurd terms like "this is a legally binding offer." The conversation went viral, and the dealership quickly shut it down.	Shows how public-facing AI chatbots can be manipulated without proper validation or safeguards, underscoring the need for strict controls on automated systems handling transactions.
3	Bias in Medical Image Classification	Google and Stanford researchers found that commercial AI models for diagnosing skin conditions performed much worse on darker skin tones because training data was skewed toward lighter skin.	Proves that without representative data and fairness testing, AI can reinforce healthcare disparities and deliver unsafe, biased results.

4	Credit Scoring and Model Drift	Apple Card users reported that women received lower credit limits than men with similar financial profiles, even on shared accounts. While Apple and Goldman Sachs denied intentional bias, the opaque credit risk model drew regulatory scrutiny.	Highlights that credit scoring models can drift or embed bias over time, making monitoring and recalibration essential.
5	Fragile Third-Party Wrappers and Supply Chain Risks	Several small AI plugin vendors built easy-to-use integrations on top of major APIs like OpenAI's. When API providers changed pricing or terms, these vendors raised prices sharply, limited features, or shut down completely. Customers were left with broken integrations and had to find replacements quickly.	Exposes the risks of relying on opaque third-party tools, emphasizing the need for vendor evaluation, modular integration design, and contingency planning.

Operational Realities

These failures often go unnoticed at first, producing no clear error logs or crashes. Instead, they accumulate quietly through:

- Drift in predictions
- Inconsistent or biased outputs
- Fragile vendor dependencies that fail without warning
- Increased human intervention to maintain quality

Security risks also emerge in these gaps, such as prompt injection through unvalidated inputs, supply-chain vulnerabilities from opaque third-party wrappers, and data poisoning via unvalidated training data. Many teams focus heavily on launch readiness, including benchmark accuracy, initial integration, and demo success, while neglecting lifecycle quality. They celebrate fast time-to-market but fail to plan for inevitable changes in data, user behavior, and vendor ecosystems. However, as real-world conditions evolve, systems silently degrade. Bugs that don't appear in test suites show up as user complaints, support tickets, or regulatory inquiries.

Ultimately, quality and security are inseparable in AI systems. Ignoring either guarantees unplanned maintenance, user frustration, brand damage, and regulatory exposure. Maintaining AI quality is not just a data science problem; it is a software engineering, DevOps, and product management responsibility that demands deliberate, cross-functional ownership.

Recommendations

To address these hidden costs and quiet failures, engineering teams need to embed quality and security practices throughout the AI lifecycle. This means moving beyond one-time model accuracy and adopting

disciplined engineering practices that keep AI-powered features reliable, maintainable, and secure over time. Key recommendations include:

1. **Data Validation Pipelines**
   Automate checks in ETL and training workflows to catch labeling errors, bias, and stale or malicious data. Treat data quality with the same rigor as code quality through validation tests and reviews.
2. **Continuous Monitoring for Drift**
   Implement drift detection for both input data and model predictions. Build dashboards and alerts to identify degradation early. Include retraining as part of regular maintenance responsibilities.
3. **Explainability Tooling**
   Integrate frameworks such as SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-agnostic Explanations) to make model predictions traceable, interpretable, and defensible.
4. **Human-in-the-Loop Processes**
   Design workflows that incorporate manual review or override for high-risk or ambiguous outputs. This reduces harm and builds user trust while maintaining safety in production.
5. **Vendor Risk Management**
   Evaluate third-party wrappers and APIs rigorously. Use modular integration patterns and maintain clear vendor exit plans to reduce the risk of fragile, orphaned systems.
6. **Risk-Based Testing and Validation**
   Expand test suites to include fairness checks, adversarial testing, and stress tests. Integrate these into CI/CD pipelines to catch potential failures before deployment. In practice, teams should:
   - **Adversarial Prompt Testing** - Probe chatbots/LLMs with edge cases, malicious inputs, and jailbreak-style prompts.
   - **Bias and Fairness Benchmarks** - Test with representative datasets across demographics and conditions to detect unfair outcomes.
   - **Scenario-Based Simulation** - Use synthetic but realistic cases (e.g., fraud tactics, policy edge cases) to stress the model.
   - **Drift Replay Testing** - Re-run historical data across versions to detect hidden accuracy or bias shifts.
   - **Fail-Safe and Guardrail Validation** - Verify that models uphold policies (e.g., never disclosing PII) even under adversarial input.
   - **Continuous Red-Teaming** - Apply penetration testing principles by continuously probing systems for vulnerabilities.
7. **Lifecycle Ownership**
   Treat AI features as long-lived products, not one-time deliveries. Plan for versioning, retraining, monitoring, and maintenance as you would for microservices or APIs.
8. **Alignment with Governance Frameworks**
   Prepare for regulatory expectations by aligning with frameworks such as the NIST AI Risk Management Framework, ISO/IEC 42001, or regional AI regulations. This ensures audit readiness and reduces future compliance rework.

By embedding these practices, developer teams can reduce technical debt, avoid maintenance surprises, and deliver AI-powered features that maintain production-quality standards over time.

Conclusion

AI systems are now integral to business operations, but they introduce subtle, probabilistic, and cumulative failure modes that traditional software engineering practices often miss. These failures degrade quality, introduce security risks, erode user trust, and increase operational costs.
Ignoring these risks does not make them disappear. Instead, it guarantees unplanned maintenance, technical debt, user frustration, and potential regulatory exposure.

Addressing them requires treating AI quality and security as integrated responsibilities. Teams must embed robust data validation, continuous monitoring, explainability tooling, human oversight, vendor risk management, and strong lifecycle planning into every deployment.

Maintaining quality in AI systems is not optional. It is essential for protecting users, supporting business goals, and building the trust necessary for adopting AI responsibly and sustainably in an increasingly automated world.

References / Bibliography

- European Parliament and Council. (2024). *EU Artificial Intelligence Act*. Retrieved from https://eur-lex.europa.eu
- National Institute of Standards and Technology (NIST). (2023). *AI Risk Management Framework (AI RMF 1.0)*. U.S. Department of Commerce. Retrieved from https://www.nist.gov/ai
- ISO/IEC. (2024). *ISO/IEC 42001: Artificial Intelligence Management System Standard*. International Organization for Standardization.
- Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., ... & Dennison, D. (2015). *Hidden Technical Debt in Machine Learning Systems*. NeurIPS Workshop Paper. Retrieved from https://papers.nips.cc/paper/5656-hidden-technical-debt-in-machine-learning-systems.pdf
- Google Cloud. (2022). *The AI Quality Framework: Best Practices for Building and Managing AI Products*. Retrieved from https://cloud.google.com/blog
- OWASP Foundation. (2023). *OWASP AI Security & Privacy Guide*. Retrieved from https://owasp.org/www-project-ai-security-and-privacy-guide/
- Brundage, M., Avin, S., Clark, J., Toner, H., Eckersley, P., Garfinkel, B., ... & Amodei, D. (2020). *Toward Trustworthy AI Development: Mechanisms for Supporting Verifiable Claims*. arXiv preprint. Retrieved from https://arxiv.org/abs/2004.07213
- U.S. Federal Trade Commission (FTC). (2021). *Aiming for Truth, Fairness, and Equity in Your Company's Use of AI*. Retrieved from https://www.ftc.gov/business-guidance/blog/2021/04/aiming-truth-fairness-equity-your-companys-use-ai