Design Guidelines to Prevent Harmful AI Experiences (version 1)

Overarching Problem: Harmful AI Interactions

AI-infused products can produce outputs that mislead, misinform, violate rights, or cause unsafe real-world consequences. These harms often arise not from explicit failures but from design decisions that do not account for the specific risks of AI-mediated experiences and interactions. These guidelines address that gap. Based on 155 real-world AI harmful cases across diverse categories of taxonomy (AI capabilities, harm types, harm initiator, AI's involvement, harm intent, and harm severity), we developed 19 design patterns/strategies across 6 principles, each targeting a distinct failure mode: unbounded AI authority, blind trust, misattributed content, impulsive harm, hidden influence, and safety drift. The principles cover the full lifecycle of a harmful AI interaction, from the moment the AI is permitted to act through to whether safety mechanisms remain effective over time. The guidelines are intended to be applicable across AI product types and interaction modalities.

Contributing authors

Pitch Sinlapanuntakul, Auli Badoni*, Avleen Kaur*, Chang Li*, Hsiang-Yi Lung*, Otto Roesler*, Te-Hsin Angela Shen*, Tirata Vittayaareekul*, Mark Zachry

* *Authors contributed equally to this version of work.*

PRINCIPLE 1. Design Against Unbounded AI Authority

Establish what the AI is permitted to do, and build in human control for when those permissions are exceeded.

Overall Problem

AI operating in high-risk contexts, including automated decision-making, real-time recommendations, and autonomous task execution, can act or advise beyond its safe scope without appropriate oversight. When no boundaries exist, users may act on outputs that cause safety, legal, or ethical harm, often without realising the AI was never authorised to act in the first place. When AI completes several steps before a human reviews what happened, the problem compounds: some actions may already be irreversible.

Core Idea / Definition

AI must operate within defined boundaries. When risk increases, the AI must narrow what it can do and transfer control to a human. When AI acts across multiple steps or on behalf of users over time, this boundary-setting becomes more complex and more critical.

Solution

Define what the AI is and is not permitted to do before a product ships. Build explicit authority transitions into the interaction so that when risk increases, control visibly moves to a human. For products where AI acts across multiple steps, define authority at the task level, not only at the product level. Escalation should be a designed mode with its own interaction logic, not a hidden backend rule.

PATTERN A. Escalate When Risk Is Detected

Exit normal operation when the potential for harm is identified, before any harmful output or action occurs.

What it prevents: The AI continuing normal operation when consequences could be serious.

Concept level

Treat escalation as a distinct interaction mode, not a backend decision. The product should have a clearly defined high-risk state that behaves differently from normal operation. Risk categories (e.g., medical, legal, financial, safety, crisis, and automated decision-making contexts) should be defined during concept development, not after launch. When AI acts across multiple steps, escalation triggers should also account for risk that emerges mid-task: a task that seemed low-risk at the start

may become high-risk several steps in. When a trigger fires, the AI exits normal operation and does not resume until the risk condition is resolved or a human has intervened.

Practice level

- Signal the mode change through a clear transition cue delivered through the appropriate channel: a visual state change on screen, a verbal acknowledgment in voice products, a haptic pattern in wearables, or a status flag in autonomous pipelines.
- Communicate the reason for escalation plainly. Avoid internal terminology.
- Deliver an explicit next-step prompt so users know what will happen. For example: “I need to pause here. A human will review this before I continue.” Users should never have to guess whether the AI paused, escalated, or ignored a request.
- When AI has been acting across multiple steps, surface a task-state summary at the point of escalation so users understand where the AI stopped and why.
- Design the escalation transition early in the process. Do not treat it as edge-case error handling.

———————————————————————————————————

PATTERN B. Require Human Review for High-Impact Outputs

Hold consequential outputs in a pending state until a qualified person has reviewed and approved them.

What it prevents: Serious outputs being finalised or acted on without human validation.

Concept level

Human review is a designed part of the workflow, not a fallback. Identify which output types require review during product definition. High-impact outputs should enter a pending state until a human has approved them. For products where AI acts across multiple steps, review gates should be placed at defined checkpoints within a task chain, not only at the end, so humans can intervene before a sequence of actions has already produced downstream effects.

Practice level

- Communicate the pending state through the appropriate channel: a visual badge or hold screen on screen, a verbal confirmation in voice products, or a workflow hold state in autonomous products.
- Identify clearly which parts of an output are awaiting validation, rather than flagging the whole output vaguely.
- Give reviewers a dedicated review interaction separate from the main product experience so they can act efficiently.

- When AI has been acting across multiple steps, show a summary of completed steps alongside the output awaiting review so reviewers have full context.

PATTERN C. Separate Suggestion from Execution

Never allow the AI to progress from recommendation to action without an explicit human decision between the two.

What it prevents: The AI carrying out actions without explicit user consent.

Concept level

A suggestion and its execution are two separate steps. The AI recommends; the human decides whether to act. This boundary should be built into the product architecture, not left to convention. For products where AI acts across multiple steps, this principle extends across the task chain: the AI should not automatically progress from one consequential action to the next without confirmation. Passive agreement such as a casual acknowledgment or uninterrupted silence is not sufficient consent for irreversible actions.

Practice level

- Place a confirmation gate between suggestion and execution through the appropriate channel: a review screen with explicit approve and cancel controls on screen, a verbal confirmation request in voice products, a haptic prompt in wearable products, or a structured hold state in autonomous pipelines.
- Communicate what will happen, what cannot be undone, and who is affected, in plain language rather than formal disclaimers.
- For irreversible actions, require a deliberate confirmation gesture appropriate to the channel: a checkbox, typed acknowledgment, or explicit verbal or physical confirmation.
- For products where AI acts across multiple steps, present a summary of the full intended task chain before execution begins, not only the next single step.
- Provide a clearly accessible cancel or undo path at every confirmation point.

PATTERN D. Block Actions Outside Approved Boundaries

Prevent the AI from taking any action that falls outside its defined safe scope, regardless of user instruction.

What it prevents: The AI operating beyond its defined safe scope.

Concept level

Some actions should never be allowed regardless of user instruction. These product-level limits are set during design and are not adjustable by users. When a limit is reached, the AI should communicate this in terms of the harm it is preventing rather than policy language, and offer a safer path forward. For products where AI acts across multiple steps or with delegated authority, boundaries should be defined per task type and per delegation level, because the scope of acceptable AI action varies depending on what a user has authorized the AI to do on their behalf.

Practice level

- Communicate the block through the appropriate channel: a disabled state with a clear explanation on screen, a verbal refusal with a reason in voice products, or a task termination flag with explanation in autonomous pipelines.
- Pair every blocked action with a short explanation naming the specific reason and a suggested alternative path.
- Avoid vague refusals. Naming the specific limit builds more trust than a generic block.
- For products with delegated authority, surface the boundary that was reached and explain which delegated scope it falls outside of.

PRINCIPLE 2. Design Against Blind Trust

Make the AI's capability boundaries and output reliability visible so users can judge when and how much to rely on what they receive.

Overall Problem

Users frequently overtrust AI or misread outputs because the product's limits and confidence levels are hidden. When nothing signals unreliability, users treat outputs as authoritative and may make harmful decisions as a result. Product messaging that implies expertise the AI does not have makes this worse.

Core Idea / Definition

Users must be able to accurately judge what the AI can do and how reliable each output is. Competence and uncertainty must be visible at the moment users interpret results.

Solution

Make capability, scope, and uncertainty visible throughout the product, not only in onboarding or fine print. Reliability must be communicated through interaction design. Users should never need to infer whether an answer can be trusted.

———————————————————————————————————————

PATTERN E. Declare Capabilities and Limits Upfront

Set accurate expectations about what the AI can and cannot do before users begin relying on it.

What it prevents: Users arriving with inflated expectations and relying on outputs the product cannot reliably produce.

Concept level

Limitations are part of responsible product design, not a marketing risk. Setting accurate expectations from the start reduces the likelihood of overtrust later. The product should also recognize when a request falls outside its reliable scope and say so directly rather than producing a weak or uncertain answer.

Practice level

- Deliver a capability summary at the start of use through the appropriate channel: an introductory explanation on screen, an opening verbal briefing in voice products, or a scoped task definition in products that act across multiple steps.

- Surface a scope boundary signal when a request approaches the edge of reliable capability through the appropriate channel: an inline alert on screen, a verbal note in voice products, or a task scope flag in autonomous products.
- Avoid product messaging that implies authority the AI does not have.
- Test whether new users can correctly describe the product's limits after first use. If they cannot, the framing needs work.

PATTERN F. Reinforce Limits in Sensitive Contexts

Remind users that the AI is a support tool, not a professional authority, whenever sensitive or high-stakes topics arise.

What it prevents: Users treating AI output as a substitute for professional judgment in high-stakes situations.

Concept level

The AI's role in sensitive domains is to support, not replace, professional judgment. When the product detects a sensitive topic area such as mental health, medical, legal, financial, or crisis contexts, it should position itself as a starting point and direct users toward qualified human help. This reminder should be delivered before the AI responds, not after.

Practice level

- Deliver the limit reminder through the appropriate channel before responding: a banner or callout on screen, a verbal acknowledgment in voice products, or a contextual flag in automated workflows.
- Direct users toward trusted resources clearly and prominently, not buried after the response.
- Keep reminders brief. Long disclaimers are skipped regardless of modality.

PATTERN G. Show Confidence and Credibility

Attach reliability signals directly to every output so users can judge trustworthiness without having to look for it.

What it prevents: Users treating all outputs as equally reliable when reliability varies across responses.

Concept level

Uncertainty is part of the product experience. Every substantive output carries a degree of confidence, and that confidence should be visible without requiring the user to seek it out. Sources should be shown where claims are made. When the product does not have enough information to answer reliably, it should say so and suggest how the user can verify independently.

Practice level

- Attach a confidence signal directly to outputs through the appropriate channel: a visual confidence indicator on screen, a verbal qualifier in voice products, or a reliability flag in autonomous outputs.
- Cite sources at the point where claims appear, not collected separately at the end.
- Use differentiation signals to distinguish verified information, inference, and speculation: visually on screen, verbally in voice products, or structurally in text outputs.
- Provide an on-demand detail layer for users who want more context, without making it part of the default experience.
- When confidence is low, name it clearly and suggest a verification path.

PRINCIPLE 3. Design Against Misattributed Content

Ensure AI-produced outputs remain clearly identifiable as machine-generated, so it cannot be mistaken for human work.

Overall Problem

AI-generated outputs can appear indistinguishable from human-created material. When labels are absent, stripped during editing, or lost during export, users and third parties may treat machine-generated content as authentic. This can spread misinformation, enable impersonation, or create legal and reputational harm. The same risk applies beyond generative AI: a credit score, hiring recommendation, or medical triage result passed between teams without attribution can be mistaken for human judgment.

Core Idea / Definition

AI-produced outputs must remain identifiable as machine-generated throughout their lifecycle. Users should never confuse AI outputs, whether text, decisions, recommendations, scores, or media, with human judgment or real-world evidence.

Solution

Treat attribution as part of the output itself, not as surrounding decoration. Labels must persist through editing, export, and sharing. Design for the scenarios where users will screenshot, crop, repost, or forward outputs outside their original context.

PATTERN H. Label AI-Generated Content Clearly and Persistently

Mark every AI-produced output as machine-generated in a way that survives distribution beyond the original product.

What it prevents: AI outputs being mistaken for authentic human or real-world material.

Concept level

AI labeling is a core transparency requirement, not a legal disclaimer. It should be perceivable immediately and designed to survive cropping, screenshotting, forwarding, and reposting. For non-generative AI outputs such as risk scores, recommendations, or automated decisions, the same logic applies. Users should know when a result was machine-generated regardless of whether it looks like media.

Practice level

- Apply an attribution marker to all outputs through the appropriate channel: a persistent visual badge on screen, a verbal attribution in voice outputs, a metadata tag in files and documents, or a provenance flag in downstream data.
- Place the attribution marker on the content itself, not only in the surrounding interface.
- Repeat the attribution signal at export and share moments.
- Design the label to survive the most likely distribution paths, including screenshots, crops, and forwarded messages.

———————————————————————————————————————

PATTERN I. Preserve Attribution Across Sharing and Editing

Keep AI origin markers attached to outputs as they are modified, exported, or passed to other people and products.

What it prevents: AI labels being stripped away when content is exported, modified, or passed to another context.

Concept level

Attribution is part of the output, not decoration applied to it. It should be embedded so it persists outside the originating product. This applies equally to non-generative AI outputs that move between products or teams: an AI-generated assessment passed into a human report should carry its origin with it. The default should be attribution included, not attribution as an opt-in.

Practice level

- Embed attribution in content metadata so it travels with the output outside the product.
- Deliver a warning when editing, cropping, or exporting would remove attribution through the appropriate channel: a visual prompt on screen, a verbal alert in voice products, or a workflow flag in automated pipelines.
- Set the default share or export state to attribution-included rather than requiring users to activate it.
- For outputs passed between teams or integrated into other documents, ensure the AI origin is visible in the receiving context.

———————————————————————————————————————

PATTERN J. Record Human and AI Contributions

Maintain a traceable history of what the AI produced and what humans changed so accountability is clear throughout.

What it prevents: Unclear or contested accountability for decisions and edits in AI-assisted workflows.

Concept level

Traceability is part of trust. In any workflow where AI and humans both contribute, there should be a clear record of who did what and when. This record should be accessible without requiring a separate compliance process, and should distinguish AI contributions from human ones in a way that is meaningful to the people using the product, not only to technical teams.

Practice level

- Provide a contribution log accessible through the appropriate channel: a version timeline on screen, a summary on request in voice products, or an audit export in high-stakes automated workflows.
- Use differentiation signals to distinguish AI contributions from human ones: visual distinction in version history, verbal identification in voice outputs, or structured metadata in documents.
- Allow audit log export for workflows where accountability is required.
- Make contribution summaries available without overwhelming users who do not need them by placing detailed logs behind an on-demand detail layer.

PATTERN K. Warn When Outputs Resemble Protected Material

Alert users before sharing or exporting any output that closely resembles a real person, brand, or protected work.

What it prevents: Users unknowingly sharing or acting on content that could infringe copyright, enable impersonation, or create legal and reputational harm.

Concept level

When an output closely resembles a real person, brand, or protected work, authorship ambiguity becomes a risk to the user and to others. The product should detect this before the output leaves its context, not after. Users need enough context to understand the specific concern, not just a vague caution.

Practice level

- Detect resemblance to real individuals, brands, or protected works before the output is shared or exported.
- Deliver a specific warning through the appropriate channel: a confirmation gate on screen, a verbal alert before completion in voice products, or a release hold in automated outputs.

- Explain specifically what the concern is: resemblance to a real person, a brand, or a protected work.
- Link to practical guidance rather than terms of service.

PRINCIPLE 4. Design Against Impulsive Harm

Introduce deliberate pauses at moments where fast decisions or fast automated action could produce harm that is difficult to reverse.

Overall Problem

Convenient, fast interactions are a design goal, but in AI products that same convenience can allow users to take harmful actions before they have considered the consequences. Rapid content sharing, single-click execution, and the absence of natural stopping points all increase the likelihood of mistakes and misuse. When AI completes sequences of actions on a user's behalf, the pace and volume of those actions can outrun any meaningful oversight, and by the time a user reviews what happened, some actions may already be irreversible.

Core Idea / Definition

Interactions must deliberately slow users when actions could cause significant harm. Speed and convenience should not be the default when consequences are serious. This applies equally to automated AI actions that can outpace meaningful human oversight.

Solution

Identify the moments in your product where speed amplifies harm and insert deliberate pauses. Friction at these points should be informative, not just blocking. Users should leave a review moment understanding what they were about to do, not just that something stopped them. Automation thresholds and mid-task checkpoints should be defined during design, not discovered after something goes wrong.

PATTERN L. Insert Frictions Before High-Risk Actions

Require at least one deliberate friction or review moment before any action whose consequences are serious or irreversible.

What it prevents: Users or automated processes executing high-impact actions without understanding what they are doing.

Concept level

Friction is a safety tool, not a design flaw. The right amount of friction at the right moment prevents harm without creating unnecessary barriers. Review steps should communicate consequences in plain language, not generate generic warnings. Every product team should identify where speed amplifies harm during design, not during incident review. For products that act across multiple

steps, review steps should be placed at defined points within a task chain, not only before the first action and after the last.

Practice level

- Place a review checkpoint before high-impact actions through the appropriate channel: a dedicated review screen on screen, an explicit verbal confirmation request in voice products, a confirmation prompt in spatial or wearable products, or a structured hold state in autonomous pipelines.
- Communicate the specific consequence in plain terms
- Require at least one deliberate review step for high-impact actions. A passive confirmation is not enough when consequences are serious.
- Provide a clearly accessible cancel path through every modality.
- State explicitly which actions can be undone and which cannot.
- For products that act across multiple steps, show where in the task chain the review checkpoint falls and what has already been completed.

PATTERN M. Confirm Before Sharing AI-Generated Content

Interrupt the sharing action with a brief confirmation step so users make a conscious decision rather than an automatic one.

What it prevents: Rapid spread of unverified or misattributed AI content.

Concept level

Sharing is a decision, not a reflex. When AI-generated content moves outside the product, the window for correction closes. A brief confirmation moment at the point of sharing is enough to interrupt habitual behaviour and give users a chance to reconsider. The reminder should appear at the moment of sharing, not earlier in the session where it will be forgotten.

Practice level

- Place a sharing confirmation gate at the point of distribution through the appropriate channel: a lightweight confirmation screen on screen, a verbal confirmation step in voice products, or a release gate in automated distribution pipelines.
- Keep the confirmation brief. Two or three lines is enough. Long confirmations are dismissed without reading regardless of modality.
- Remind users the content is AI-generated at that exact moment.
- Make source inspection available with a single action rather than through settings.

PATTERN N. Prevent Automation From Bypassing Human Judgement

Set explicit thresholds for automated action and require human reauthorisation when those thresholds are reached or the original scope changes.

What it prevents: Automated AI action executing at a speed or scale that removes meaningful human involvement.

Concept level

Automation designed for efficiency can remove the human judgment that should govern it. When AI executes sequences of actions on a user's behalf, the original authorisation does not mean the user has consented to every downstream consequence. Thresholds for automation, including action count, time elapsed, financial impact, and scope change, should be defined during design. When automation approaches a threshold or encounters an unexpected condition, it should pause and request human input rather than continue or fail silently. When a task requires actions beyond what the user originally authorised, this should always trigger a reauthorisation step rather than proceeding on assumption.

Practice level

- Display a running awareness signal showing what automated actions have been taken through the appropriate channel: a live count or summary on screen, a periodic verbal update in voice products, or a task log in autonomous pipelines.
- Provide a pause or stop control that is always reachable during automated runs, regardless of modality.
- When automation approaches a defined threshold, deliver a checkpoint prompt before proceeding: a visual gate on screen, a verbal pause in voice products, or a hold state in pipelines.
- When completing a task would require actions outside the original scope, surface a scope summary and do not proceed without explicit reauthorisation.
- Treat scope expansion as a new consent event, not a continuation of the original approval.

PRINCIPLE 5. Design Against Hidden Influence

Make visible both what data the product collects or draws on and how it shapes the output, so users can see and adjust the forces acting on what they receive.

Overall Problem

AI outputs are shaped by forces users cannot see. Personal data may be collected and used without users knowing it is being drawn on at all. A recommendation may be personalised using data a user never knowingly provided. A classification may reflect bias in the training data. A summary may reflect a particular framing or optimisation goal. In each case, users interpret results as neutral and objective and make decisions on that assumption. In addition to influence alone, users often have no visibility into what data is being collected, when it is being used, or how it is driving what they receive. When consent is buried in sign-up flows and controls are hidden in settings, users have no practical way to see, question, or adjust any of it.

Core Idea / Definition

Users must be able to see what data the product is drawing on and how that data is shaping what they receive. Data collection is not a backend detail. It is a design decision that users have a right to see at the moment it happens, not discover after the fact.

Solution

Make data collection and its influence on outputs visible and adjustable at the point where users encounter the result. Show what data was used. Show how it shaped the output. Place controls next to the result, not in account settings. Seek consent when data is actually being collected or used, not only at account creation when the context is too abstract to be meaningful.

———————————————————————————————————————

PATTERN O. Seek Consent at the Point of Data Collection

Ask for permission when data is actually being collected or used, not only at account sign-up where the context is abstract.

What it prevents: Users being unaware that data is being collected mid-interaction and used to shape current or future outputs.

Concept level

Consent at sign-up is not the same as informed consent at the moment data is used. Users who agreed to terms during account creation are unlikely to remember what they consented to when they encounter a personalised result later. Seeking permission at the moment of collection is more

honest and more useful. Users should be able to decline data collection for specific interactions without losing access to core features.

Practice level

- Deliver a contextual permission prompt at the moment data is first accessed or collected during an interaction through the appropriate channel: an inline prompt on screen, a verbal request in voice products, or a task-scoped consent step in products that act across multiple steps.
- Explain in plain language what the data will be used for at that moment.
- Distinguish between data used for the current interaction only and data retained for future personalisation.
- Provide a live signal when the product is actively collecting or using data.
- Allow users to decline without losing access to core features.

PATTERN P. Reveal How Personal Data Shapes Results

Show users which personal data influenced an output and let them adjust those signals immediately from the same place.

What it prevents: Users receiving personalised outputs without knowing which data drove them.

Concept level

Data influence is a visible part of the product experience, not a backend detail. When personal signals shape an output, users should be able to see which signals were used and adjust them immediately. Controls placed far from the result are not a meaningful substitute for in-context transparency.

Practice level

- Deliver an influence disclosure near the output through the appropriate channel: an expandable explanation on screen, a verbal summary in voice products, or a metadata annotation in document or data outputs.
- Place adjustment controls near the result, not in account settings.
- Allow users to remove or adjust signals immediately and show how results change when they do.
- Confirm when a signal has been removed so users know the adjustment took effect.

PATTERN Q. Signal the AI's Framing or Stance

Communicate clearly when an output reflects a particular viewpoint, optimisation goal, or framing rather than a neutral position.

What it prevents: Users interpreting outputs that reflect a particular goal or viewpoint as if they were neutral.

Concept level

AI products are often optimised for a specific goal, whether that is engagement, persuasion, efficiency, or balance. That optimisation shapes outputs in ways users may not recognise. The operating mode should be communicated clearly so users can interpret results with the right frame. When bias risk increases, this should be delivered as an active signal rather than left to users to notice.

Practice level

- Communicate the active mode through the appropriate channel: a persistent mode indicator on screen, a verbal label at the start of a response in voice products, or a mode tag in structured outputs.
- Provide an on-demand explanation of what the current mode means in practice, in plain language.
- Use consistent signals across modes so users learn to read them over time, whether those signals are visual, verbal, or structural.
- Deliver a bias risk signal when relevant rather than letting the output speak without context.

PRINCIPLE 6. Design Against Safety Drift

Keep safety signals and correction mechanisms active and accessible across the full duration of an interaction, not only at its start.

Overall Problem

Safety interventions are often designed as single moments: a warning appears once and disappears, an escalation fires and fades, a reporting option exists but requires navigating away. Over time and across long sessions, these one-off interventions stop doing meaningful work. Risk persists, but the signals have gone quiet. AI products can also degrade silently, continuing to produce outputs after their reliable operating conditions have changed, with no indication to the user that anything is different.

Core Idea / Definition

Safety mechanisms must remain visible and actionable across the full length of an interaction. Protections that exist at the start of a session must still be working at the end.

Solution

Treat safety as a continuous state of the product experience, not a series of one-off events. Warnings should remain present for as long as the risk condition exists. Feedback mechanisms should be embedded directly in outputs so reporting never requires an interruption. The product should also communicate its own failures proactively rather than waiting for users to notice something is wrong.

PATTERN R. Keep Safety States Visible While Risk Persists

Maintain active risk signals throughout an interaction for as long as the underlying risk condition remains unresolved.

What it prevents: High-risk conditions continuing in the background while the experience looks normal.

Concept level

Safety is a continuous state, not a single notification. A warning that appears once and disappears has done almost no work. If the risk condition that triggered a warning has not been resolved, the signal should remain present. Repeated high-risk behaviour within a session should escalate the response rather than repeat the same alert. Transitions into and out of a high-risk state should be obvious to the user regardless of the interaction channel.

Practice level

- Maintain a persistent risk signal through the appropriate channel for as long as the risk condition is active: a persistent visual indicator on screen, a recurring verbal acknowledgment in voice products, or a continuous status flag in autonomous products.
- Do not allow risk signals to fade after a single message if the underlying risk has not changed.
- Escalate the response when high-risk behaviour recurs within the same session rather than repeating the same low-level alert.
- Deliver a clear state transition signal both when entering and when exiting a high-risk mode.
- Include a brief explanation of why the persistent signal is present, not just the signal itself.

———

PATTERN S. Enable In-Context Reporting and Correction

Give users a simple way to flag problems directly from any output without having to leave the interaction to do so.

What it prevents: Unsafe outputs persisting because providing feedback is inconvenient or invisible.

Concept level

Feedback is a meaningful input to product improvement, not a complaint form. When reporting is easy and embedded in normal interaction, users are more likely to flag problems. When it requires leaving the interaction to do so, they rarely do. The product should also confirm that reports are received and explain how they are used. Feedback that feels ignored will not be provided a second time.

Practice level

- Embed a reporting mechanism directly alongside every output through the appropriate channel: an inline report option on screen, a voice command in voice products, or a correction trigger in products that act across multiple steps.
- Keep the reporting interaction minimal: a category selection and an optional note is enough.
- Offer a small set of clear categories rather than a blank input.
- Deliver a brief confirmation after submission that explains how the report is used: "Your report helps improve safety in this product."
- Optionally surface a history of flagged items for users who want transparency over their own contributions.