

General OpSec Rules for AI

For everyone · ~8 min read · the cheat-sheet lesson

This lesson is the master cheat-sheet for everyday AI use. The earlier lessons explained *why* the rules exist; this one lists them in a form you can pin to the side of your monitor. Every rule here has already been justified elsewhere in the course — this is the distilled version you come back to when you are about to hit enter on a prompt and something in the back of your mind is whispering "wait".

Read it once now. Then treat the two numbered lists as the parts you actually memorise.

The 10 rules

1. **Classify every prompt before you send it.** Spend two seconds deciding whether the text in your box is public, internal, confidential, or secret — and never send anything above your tool's approved classification level.
2. **Never mix personal and corporate AI accounts or identities.** Keep them on different tools, different browser profiles, and different logins; cross-contamination is how private chats end up attached to work identities.
3. **Approved tools only for corporate work. Personal tools only for personal work.** A consumer ChatGPT tab is not the same product as your employer's enterprise contract, even when the interface looks identical.
4. **Redact values, keep shapes.** Replace real names, keys, URLs and IDs with placeholders that preserve the *structure* of the problem — the model can still help with the shape, but the secrets never leave your machine.
5. **Assume prompts are retained, reviewed, and breachable.** Treat every message as if it will sit in a log for years, be read by a human someday, and eventually appear in a breach disclosure.
6. **Treat screenshots, file uploads, and IDE autocomplete as prompts.** A PDF you dropped in or a suggestion you accepted in your editor counts the same as a pasted message — the bytes still leave your machine.

7. **One browser profile per jurisdiction. No cross-login.** Keep EU work in an EU profile and US work in a US profile; never sign into a second jurisdiction's tools from the first.
8. **Stop-drop-tell on a mistake. Do not delete the chat.** If you realise you sent something you shouldn't have, stop using the tool, capture the evidence, and tell your security team — do not hit the trash icon.
9. **Rotate any secret you suspect touched an AI — even "maybe".** Rotation is cheap and deterministic; hoping a token wasn't captured is neither.
10. **When in doubt, ask the security team before pasting.** A thirty-second question is always cheaper than a post-incident review.

Approved tools

"Approved" is a legal word, not a taste word. It does not mean "a tool my colleagues also use" or "a tool with a nice interface". It means your employer has done the paperwork that makes it safe to send corporate data into that tool.

In practice, an approved tool has all four of these:

- **A contract** with the provider — an enterprise agreement or DPA, not the click-through consumer terms of service.
- **Zero-retention** (or a bounded, documented retention window) written into that contract, so your prompts are not kept beyond what's necessary.
- **Admin audit** — your security team can see who used the tool, when, and (if needed) what for.
- **An acceptable-use policy** — the rules you agreed to, in writing, during onboarding.

Typical approved categories at a modern company: enterprise ChatGPT, Claude for Work or Claude Enterprise, Gemini for Workspace, approved IDE assistants (Copilot Business, Cursor Business, Claude Code under an enterprise plan), and approved MCP servers that connect to internal systems through a vetted gateway. Anything else — a personal account, a newly announced tool, a browser extension that "adds AI" — is unapproved until the paperwork says otherwise.

Stop-drop-tell

Mistakes happen. The point of a response protocol is to make the recovery fast and boring. When you realise you pasted something you shouldn't have, do exactly three things, in this order:

1. **Stop** using the tool. Don't send a "wait, ignore that" follow-up. Don't keep chatting. Close the tab if that helps you resist the urge.
2. **Capture** the evidence. Take a screenshot of the prompt *and* the response. Note the timestamp, the account you were signed in as, and the tool you were using.
3. **Tell** security. Use whichever channel your company has set up for this — a dedicated Slack channel, a hotline, a ticket queue. Share the screenshots and a one-line description.

Do not delete the chat

Deletion does not remove the data from the provider and it destroys the evidence your security team needs. The bytes are already on the provider's disks — a UI delete only hides them from *you*. Meanwhile your responders now have nothing to triage, no timestamp to cross-reference, and no prompt text to scan for other exposed secrets. Leave it there. Let the professionals handle it.

Near-miss vs. incident

Two words that sound similar and mean very different things on paper. Getting them right makes reporting much less scary.

A **near-miss** is when something *could* have gone wrong but didn't, because you caught it in time. You realised in seconds, you stopped, you reported, and no credential or data was used downstream. Near-misses are logged, maybe discussed in a retro, and then they go away. Nobody is in trouble.

Example: A support engineer pastes a customer's email into an AI summariser out of habit, then realises within a minute. They stop, screenshot the chat, and ping the security channel. The token, the account, and the customer are all fine. Near-miss.

An **incident** is when data actually left your control in a way that matters: a real credential, real PII, real customer data, or a real production secret made it into a tool —

and either wasn't caught, was caught too late, or was discovered later by someone else. Incidents trigger rotation, investigation, and in some regulated industries, external notifications.

Example: A developer pastes an AWS access key into a consumer chatbot to "decode" it, and forgets about it. A week later the key shows up in the cloud audit logs being used from an unfamiliar IP. Incident.

When to escalate

If any of the following touched an AI tool — even briefly, even in a tool you closed immediately — escalate to the security team as an incident, not a near-miss:

- **Any credential exposure** — API keys, tokens, passwords, JWTs, SSH keys, OAuth secrets.
- **Any PII exposure** — names combined with identifiers, email addresses, phone numbers, government IDs, addresses, dates of birth.
- **Any customer data** — anything a customer entrusted to you, regardless of how sensitive it "seems".
- **Any production secret** — database URLs, signing keys, internal endpoints with live tokens, infrastructure topology.

The test is simple: if a downstream attacker could use what you pasted, escalate. The security team would always rather spend thirty minutes closing a false alarm than three days cleaning up a silent one.

Rules to remember

1. Classify every prompt before you send it.
2. Never mix personal and corporate AI accounts or identities.
3. Approved tools only for corporate work. Personal tools only for personal work.
4. Redact values, keep shapes.
5. Assume prompts are retained, reviewed, and breachable.
6. Treat screenshots, file uploads, and IDE autocomplete as prompts.
7. One browser profile per jurisdiction. No cross-login.

8. Stop-drop-tell on a mistake. Do not delete the chat.
9. Rotate any secret you suspect touched an AI — even "maybe".
10. When in doubt, ask the security team before pasting.