

Risk tier workshop for existing apps

<https://prompt-to-prototype.com/worksheets/risk-classification-workshop/> · as of 2026-10-22

A 90-minute session that takes a list of discovered apps and produces a tier, a provisional owner and a next action (register, fix or retire) for each, ready to load into the registry. It applies the six tier questions to apps that already exist, answer by answer, with no scoring. Expect about 3–4 minutes per app, so 12–15 apps per session. Downgrade requests and disputes are recorded and routed, not settled in the room.

Who this is for

IT managers and enablement leads running the triage of apps found by discovery or an amnesty. You get inputs, attendees, an agenda, a recording sheet and the outputs to load. Reading time: about 10 minutes; the session itself is 90 minutes.

The tier model itself is on [risk tiers for worker-built apps](#). This worksheet is how to apply it to a batch.

Inputs

Circulate these two working days before the session:

- ☐ The list of discovered apps, from [finding existing apps](#): name, platform, builder, sharing or user count, data sources, whether it's reachable from the internet, whether it calls an AI model (including a tool's built-in AI features, such as an AI formula in a spreadsheet or an AI step in a workflow tool), and any OAuth grants, tokens or service identities held by agents and automations (with whether each token can be scoped).
- ☐ The recording sheet below, pre-filled with one row per app.
- ☐ The six tier questions and two flags, printed in full from [the six tier questions](#). The [intake form's plain-text version](#) prints well.
- ☐ The [approved-tools list](#), by data class, including each tool's built-in AI features, so the room can check the AI approval gate.
- ☐ The list of downgrade signers and exception approvers (enablement lead, security reviewer, IT/platform owner, technical reviewers for standard downgrade 3) and the counsel and compliance contacts (see [the counsel route](#)).
- ☐ One worked case from the [worked test cases](#) to calibrate on.

If the list is longer than 15 apps, split it by department or platform rather than rushing answers.

Who to invite

The first three roles are the minimum; without them, don't run the session. One person can hold more than one role (see [when one person holds several roles](#)).

Role	Needed?	Why they're there
Enablement lead (facilitator)	minimum	Runs the session; owns the registry load
Technical reviewer	minimum	Answers Q3 (reach) and Q6 (actions) from platform settings and grants; checks the trifecta legs
A business representative for each area in the batch	minimum	Answers Q2 (users), Q4 (dependence) and Q5 (decisions about people); proposes owners

Role	Needed?	Why they're there
Security reviewer	recommended	Checks the floors; flags Critical candidates. The session can run without one: the technical reviewer checks the floors, and Critical candidates are held at their current scope until a named security reviewer has seen them (see minimum staffing)
Data owners or the privacy officer for the data in the batch	if the batch has Q1(c) or (d) data	Answers Q1 (data) where the class isn't obvious
IT/platform owner	for the Critical candidates segment	Hears graduation needs

Builders don't need to attend, but keep them reachable for questions.

Agenda (90 minutes)

Time	Segment	Output
0–10	Calibrate: walk one worked case through all six questions	Shared reading of the wording
10–60	The six-question pass, one app at a time (about 3–4 minutes each)	Six answers and both flags per app
60–70	Floors check	List of floor breaches, each with an owner and a fix-by date
70–80	Owners and next actions	Provisional owner and register, fix or retire per app
80–90	Downgrade requests, disputes and gates	Routing list; follow-up dates

The six-question pass

For each app, answer every question for what the app **can** do with the data, reach and permissions it holds, not for what it's meant to do. For an agent or automation, answer for every permission it has been granted. If a token can't be scoped, answer for everything it can do, then note it for standard downgrade 3. Record the letter. Then take the **highest** answer or modifier as the tier. Never add, average or score the answers.

The question wording below is exact; the answer labels are short handles for the room. The full answer wording, which is what counts, is on [the six tier questions](#): print it and settle any doubt from it, not from the labels.

Question (exact wording)	Answer labels → tier
Q1. Data. What's the most sensitive data the app reads, stores or sends?	(a) Public, or your own work files → Sandbox · (b) Internal → Team · (c) Confidential → Important · (d) Restricted → Critical . Note the consent gate (route to counsel before collecting)
Q2. Users. Who uses it or relies on its output? People who only <i>receive messages</i> from it (reminders, notifications, emails, replies) are answered under Q6, not here.	(a) Only you → Sandbox · (b1) One team → Team · (b2) Several teams or the whole company, all inside the business → Team · (c) Anyone outside the business → Important
Q3. Reach. Can someone reach it from the internet without company sign-in?	(a) No → Sandbox · (b) Yes → Important , on an approved public-facing hosting pattern only
Q4. Dependence. What happens if it stops, or gives wrong results, for a day?	(a) Only your own work → Sandbox · (b) Users work around it → Team · (c) A process stops, or output is re-performed or checked before use → Important · (d1) Posted, or kept as control evidence, without independent recomputation → Critical · (d2) Filings, safety actions or customer transactions with no per-output check → Critical

Question (exact wording)	Answer labels → tier
Q5. Decisions about people. Does its output decide, rank, score, recommend or flag anything, or produce a score, ranking, flag, summary or recommendation that a person relies on to decide, about a person's job (hiring, firing, promotion, performance rating), pay, discipline, credit, insurance, housing, education, health care, or access to public benefits?	(a) No → Sandbox · (b) Every outcome reviewed first by a person who meets the meaningful-human-review test → Important · (c) An outcome can take effect before that review → Critical
Q6. Actions. What can it do without a person approving each action?	(a) Nothing beyond itself → Sandbox · (b) Its own data store, or messages to colleagues → Team · (c) A shared system of record, or automated messages to outsiders → Important · (d) Money, deletions, bulk changes over the change cap, or anything it can't undo → Critical

Jurisdiction flag. Does the app handle information about people in Quebec, the EU or the UK, or about staff who work in the EU or UK, or is an AI feature's output used in the EU? If you don't know, answer yes. If yes, **route to counsel**. The flag never changes the tier.

Audit-scope flag. Ask the four questions under [the audit-scope flag](#). Any yes: **route to the compliance owner**. The flag never changes the tier.

AI modifiers. Check [M1](#), [M2](#) and [M3](#) for every app. Count a tool's built-in AI features (an AI formula in a spreadsheet, an AI step in a workflow tool) as M1 when they process the app's data; the approval gate is met if that feature is on the approved-tools list for the data class in Q1, and otherwise the app is blocked until it is. M2 makes the app **Critical**. For M3, list the identity and grants for the registry.

Don't: score and sum

Don't give each question points and add them up. A severe answer gets averaged away: an app that deletes records in bulk but has few users "scores" low. One Critical answer makes the app Critical. This is the high-water-mark principle in NIST's categorization standard¹.

Don't: tier by how the app was built

"It's just a macro" and "it was built with AI" aren't answers. Tier on what the app does with its data, users, reach, dependence, decisions and actions; the UK National Cyber Security Centre's guidance calibrates scrutiny the same way².

Recording sheet

Record the as-found answer, and if it's higher only because of a floor breach, the answer as fixed and the fix-by date.

App	Q1	Q2	Q3	Q4	Q5	Q6	Juris.	Audit	M1–M3	Floors OK?	Fix-by date	Tier	Provisional owner (role)	Next action	Routing
Example: desk-booking reminders	b	b1	b as found; a as fixed	b	a	b	no	no	none	No: public bucket (Floor 2)	5 working days after the session	Team as fixed (Important if not fixed by the fix-by date)	Facilities manager	Fix: make the storage private, confirm with a clean exposure scan	—

Illustrative composite

The example row is invented. The app holds staff names and desk bookings (Q1(b)) and serves the facilities team (Q2(b1); staff who get reminders are Q6(b), not Q2). Its files sit in publicly readable storage, so Q3 is (b) as found and Floor 2 is broken: a Team app may not be reachable without sign-in. Under the [floor-breach rule](#), the room closes public access the same day, records the as-found answer, the breach and a fix-by date (5 working days at most), and tiers the app as fixed: Team. If no clean exposure scan confirms the fix by then, it's Important, with every Important gate, until one does. The fix is to make the storage private, not to move the app to a public-facing pattern it doesn't need.

Floors check

For every app, check both floors before anything else is loaded:

1. **The trifecta rule.** No agent or AI feature may combine private data, untrusted content and a way to send data out. For each AI feature or agent, ask: does it read private data, does it read outside content (email, web pages, uploaded files, tickets), and can it send anything out (email, web requests, rendered links or images, public storage)? If all three, it stops until one leg is removed.
2. **No internet reach without company sign-in below Important.** If a Sandbox or Team app is reachable from the internet without sign-in, make it private now, or move it to an approved public-facing pattern and register it at Important.

A floor breach is contained first, with a named owner and a fix-by date no more than 5 working days away. It doesn't wait for the registry load. The floor-breach rule applies only to mistakes: if the owner wants to keep a public link, the as-found answer is the answer.

Owners and next actions

For each app, record a provisional owner (a role, usually the process owner or the builder's manager) and one next action:

- **Register:** the app has an owner, its floors hold and it's worth keeping. Open an app record at its tier and send it through [intake](#) to confirm.
- **Fix:** something must change first (a floor breach, an unapproved AI service, a personal account, a credential in the code, an over-broad token that could be scoped). Record the fix and re-answer the questions afterwards.
- **Retire:** nobody will own it, it's unused, or it's been replaced. Start [retirement](#).

Downgrade requests and disputes

- Downgrades: if an owner wants a lower tier because compensating controls remove the reason for an answer, record the request and route it to the signers named in the [downgrade path](#). For a token the tool can't scope, note whether the app could qualify for standard downgrade 3 (no AI feature, not an agent, code calls no write or delete endpoint, token in a secrets manager) and route it to a technical reviewer. Don't decide it in the room. No downgrade can go below a (legal) answer or touch a floor.
- Exceptions: a request to change *when or how* a gate is met (for example, a runbook due later) is a [bounded exception](#): it needs a named approver, a written reason, an expiry of 90 days or less and an entry in the app record. Record the request and route it to the approver. There are no exceptions to the floors or the legal triggers.
- Disputes: if the room can't agree on an answer, record the higher answer, note the wording that caused the disagreement, and route it per [resolving disputes](#).
- Gates: list every app with the jurisdiction flag, the consent gate or the audit-scope flag, and send each to counsel (through [the counsel route](#)) or the compliance owner.

Outputs

- ☐ A tier for every app, with its six answers and both flags.
- ☐ A provisional owner (role) and next action for every app.
- ☐ A registry load file built from the recording sheet (see the [app registry](#)).
- ☐ A floor-breach list with owners, as-found answers and fix-by dates.
- ☐ A data-owner sign-off list: every app with a Q1(c) or (d) answer, and the data owner (role) who must sign off before it registers at Important or Critical.
- ☐ Routing lists: downgrade requests (including standard downgrade 3 candidates), exception requests, disputes, counsel gates and compliance gates.
- ☐ A list of Critical apps for the IT/platform owner's graduation planning.

Follow-up

Within five working days:

- ☐ Confirm every provisional owner with the owner's manager.
- ☐ Load the registry and send each "register" app through intake.
- ☐ Confirm every floor-breach fix by its fix-by date (for Floor 2, with a clean exposure scan); apply the as-found answer to any that aren't fixed.
- ☐ Send each data-owner sign-off request.
- ☐ Start retirement for every "retire" app.
- ☐ Schedule the next session if the batch wasn't finished.

Assumptions and evidence

- **A-018** (Medium): we expect four tiers set by the highest of six answers to be enough to route support, review and graduation for discovered apps; test this by counting apps the room couldn't place, and disputes per session.
- **A-045** (Low): we expect 35% or less of registered apps to land in Important or above; test this by reporting the tier mix of each workshop batch.

Evidence: published government policy with any-of tier triggers³, security-agency guidance and a durable federal standard; no controlled study of triage workshops for worker-built apps. The 3–4 minutes per app is a planning estimate, not a measurement; time your first session. Facts on this page are as of October 2026.

Related

- [Risk tiers for worker-built apps](#): the full model and the gate matrix.
- [App registry](#): where the workshop's outputs go.
- [Find the apps your staff already built](#): how to build the input list.
- [Get a worker-built app approved](#): the intake each registered app goes through next.

-
1. NIST, "FIPS 199: Standards for Security Categorization of Federal Information and Information Systems", Feb 2004, section 3 (high-water mark). <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.199.pdf> [primary], accessed 2026-10-21.
 2. UK National Cyber Security Centre, blog post on the "vibe coding spectrum" approach to AI-assisted software development, 18 Jun 2026. <https://www.ncsc.gov.uk/blogs/the-vibe-coding-spectrum-approach-to-ai-assisted-software-development> [regulator], accessed 2026-10-21.
 3. Government of Saskatchewan, "Low-Code/No-Code Development and Maintenance Policy", May 2026. <https://taskroom.saskatchewan.ca/-/media/project/taskroom/documents/policy/lowcodenocodedevelopmentandmaintenancepolicy.pdf> [primary], accessed 2026-10-21.

Prompt to Prototype, Risk tier workshop for existing apps, <https://prompt-to-prototype.com/worksheets/risk-classification-workshop/>, as of 2026-10-22. Licensed [CC BY-SA 4.0](#).