

App registry template and app record

<https://prompt-to-prototype.com/worksheets/app-record-template/> · as of 2026-10-21

Use this app registry template to record every worker-built app. A builder types 12 core fields; the intake form, the registry's formulas and the reviewers fill the rest, and each field says who fills it and which risk tiers need it. Download the builder core or the full registry sheet as CSV, plus two extension sheets and a data dictionary, then copy the one-page app record and the runbook. [Jump to the downloads.](#)

Who this is for

Registry administrators setting up the [app registry](#), and builders and app owners filling in a record. You get the fields grouped by who fills them, the tier each is required at, five CSV files and a filled-in example. Reading time: about 8 minutes.

Download the CSV files

Version 1, 2026-10-21.

- [Builder core sheet: the 12 fields a builder types \(CSV, 12 columns\).](#)
- [Full registry sheet: every main column, grouped by who fills it \(CSV, 66 columns\).](#)
- [AI services, agent identities and grants extension sheet \(CSV, 15 columns\).](#)
- [Audit, records and legal extension sheet \(CSV, 17 columns\).](#)
- [Data dictionary: what to enter in every column, who fills it and when it's required \(CSV, 9 columns\).](#)

Delete the first row of each file before you import; it carries this cite line:

Prompt to Prototype, App registry template and app record, <https://prompt-to-prototype.com/worksheets/app-record-template/>, as of 2026-10-21. Licensed CC BY-SA 4.0.

Using or adapting these templates inside your business isn't distribution under the licence. The main sheet has one row per app, with the builder's 12 columns first; the extension sheets are keyed by `app_id` and used only when they apply, so you can restrict the grant inventory more tightly than the rest of the registry.

How do I use this template?

1. Create a spreadsheet, or an IT service management or service-catalog table, from [the full registry sheet](#), and add the extension sheets keyed by `app_id`. Delete the `EXAMPLE` rows. See [the app registry](#) for how to run the inventory around these columns, including who may edit which columns.
2. Point the intake form at the "intake form" columns, and set formulas or workflow rules for the "computed" columns. Builders never type a computed field.
3. Fill in what your tier needs and leave the rest blank. A Sandbox app needs only the one-time sandbox workspace record; the first answer above Sandbox starts an app-level record ([gate matrix, intake](#)).

4. Copy the one-page app record and the runbook into your documents tool, and link them from `decision_records` and `runbook_link`.
5. Record tiers by name only (Sandbox, Team, Important, Critical). `risk_tier_rank` is a hidden sort key (1 for Sandbox to 4 for Critical).

Registration takes about 10 minutes once the intake form has skip logic (A-031); expect 15–20 minutes the first time ([get a worker-built app approved](#)).

Which columns does the app registry template have?

Key: **Yes** means required at that tier; a condition ("If used") means required only when it applies; *Good practice* means recommended, not a gate; **No** means not needed. The Sandbox column is the one-time sandbox workspace record. **Filled by** is the builder, the intake form (from the builder's answers), computed (a formula or workflow; nobody types it), or a named role. The [data dictionary](#) says what to enter in every column; the tiers are defined on [risk tiers](#).

People are recorded twice. Each `_person` field holds a directory account (the stable sign-in ID or work email), which is the key for leaver checks, the orphan check and help-desk routing. The role fields are the durable part: when the person changes, the role stays.

1. Builder core (typed by the builder)

Field	What to enter	Sandbox	Team	Important	Critical
<code>app_name</code>	A plain name (for a Sandbox workspace, the workspace name)	Yes	Yes	Yes	Yes
<code>purpose</code>	One sentence: what it does and for whom	No	Yes	Yes	Yes
<code>builder_person</code>	The builder's directory account	Yes	Yes	Yes	Yes
<code>app_owner_person</code>	The app owner's directory account	No	Yes	Yes	Yes
<code>app_owner_role</code>	The business role accountable once anyone besides the builder relies on it	No	Yes	Yes	Yes
<code>app_owner_successor</code>	The directory account of whoever takes over if the owner leaves	No	Yes	Yes	Yes
<code>team</code>	The team or function that uses it	No	Yes	Yes	Yes
<code>user_group</code>	Who uses it, and roughly how many	No	Yes	Yes	Yes
<code>platform</code>	The tool or platform it's built on	Yes	Yes	Yes	Yes
<code>location</code>	Where it runs: URL or workspace	No	Yes	Yes	Yes
<code>source_location</code>	Where the exported source and configuration are kept	No	Yes	Yes	Yes
<code>kill_criteria</code>	The result that would make you stop	No	Yes	Yes	Yes

2. Filled by the intake form

Field	What to enter	Sandbox	Team	Important	Critical
hosting_pattern	The approved pattern, from a list (a private workspace for Sandbox)	Yes	Yes	Yes	Yes
secrets_location	Where each secret is held: the secrets-manager entry or the platform's server-only secret-store name, never the value	No	If used	If used	If used
q1_data to q6_actions	The answer letter for each tier question (for example c or b2), one column each	No	Yes	Yes	Yes
ai_feature, customer_facing_ai, agent_or_automation	Yes or no for M1, M2 (asked only if M1 is yes) and M3; built-in SaaS AI features count as M1	No	Yes	Yes	Yes
jurisdiction_flag, consent_gate	Yes or no (for the jurisdiction flag, yes if you don't know)	No	Yes	Yes	Yes
audit_scope_flag, audit_scope_reason	Yes or no, and one sentence saying why	No	Yes	Yes	Yes
sandbox_attestation_on	The date the builder attested to the counsel gates (workspace registration form)	Yes	No	No	No

The [audit-scope flag](#) never changes the tier, and an audit-scope app gets no minimum tier: the compliance owner's change-management and evidence requirements apply at whatever tier the app is, Team included.

3. Computed (nobody types these)

Field	Sandbox	Team	Important	Critical
app_id, builder_role, risk_tier, support_level, registered_on, next_review_on	Yes	Yes	Yes	Yes
tier_driver, retire_by	No	Yes	Yes	Yes
risk_tier_rank	Optional	Optional	Optional	Optional

The enablement lead confirms `risk_tier` at triage, and only a signed downgrade changes it.

4. People filled by the owner and reviewers

Field	Filled by	Sandbox	Team	Important	Critical
named_backup_person	app owner	No	No	Yes	Yes
maintainers	app owner	No	If used	If used	If used
business_reviewer_person	business reviewer	No	No	Yes	Yes

Field	Filled by	Sandbox	Team	Important	Critical
technical_reviewer_person	technical reviewer	No	No	Yes	Yes
security_reviewer_person	security reviewer	No	No	No	Yes
data_owner_person, data_owner_role	enablement lead	No	No	If Q1 is (c) or (d)	If Q1 is (c) or (d)
data_owner_signoff_by, data_owner_signoff_on	data owner	No	No	If Q1 is (c) or (d)	If Q1 is (c) or (d)

A data owner never signs off their own app's data use ([separation of duties](#)): if they own the app, their manager or deputy signs. If `data_owner_signoff_by` equals `app_owner_person`, flag the record in the registry's health checks.

5. Lifecycle, checks, tier changes and exceptions

Field	Filled by	Sandbox	Team	Important	Critical
status (Sandbox, active, quarantined, retiring, retired, orphaned)	enablement lead	Yes	Yes	Yes	Yes
last_review_on	enablement lead	No	Yes	Yes	Yes
last_access_review_on	IT/platform owner	No	No	No	Yes
business_review_on	business reviewer	No	No	Yes	Yes
technical_check_on	technical reviewer	No	No	Yes	Yes
last_exposure_scan_on, last_exposure_scan_result	IT (or the technical reviewer it delegates to)	If a floor breach was fixed	If scanned	Yes	Yes
last_restore_test_on	named backup	No	If an agent writes to live systems	<i>Good practice</i>	<i>Good practice</i>
counsel_routed_on	enablement lead	No	If routed	If routed	If routed
floor_breach, floor_breach_as_found, floor_breach_fix_by	enablement lead	If open	If open	If open	If open
downgrade, downgrade_record, downgrade_review_on	enablement lead	No	If used	If used	If used
exceptions_open, exceptions_next_expiry	enablement lead	No	If used	If used	If used
runbook_link	app owner	No	No	Yes	Yes
graduation_status	IT/platform owner	No	If a trigger fired	If a trigger fired	Yes
decision_records	enablement lead	No	Yes	Yes	Yes

An app's status becomes active at [go-live sign-off](#).

Extension sheet: AI services, agent identities and grants

One row per AI service, identity, OAuth grant, token, MCP server or scheduled job, keyed by `app_id`; required from Team when `ai_feature` or `agent_or_automation` is yes. The app owner fills it, and the IT/platform owner checks it against the identity provider's grant reports. Columns: `app_id`, `item_type`, `item_name`, `version` (pinned model or server version), `runs_as_person` (blank for a service identity, which is preferred; this is what leaver matching checks), `scope`, `data_class`, `trifecta_check` (which leg was removed, and when), `eval_threshold`, `eval_last_pass_on`, `expiry`, `owner_role`, `secret_entry_name` (where the secret is held, never the value), `log_location` and `recorded_on`.

Extension sheet: audit, records and legal

One row per app, keyed by `app_id`. If `audit_scope_flag` is yes, the enablement lead routes the app to the [compliance owner](#), who fills `audit_scope`, `control_evidence`, `controls_evidenced`, `compliance_requirements_set_on` (go-live waits for it), `vendor_assurance` and `cuecs_relied_on`; the IT/platform owner fills `hosting_vendor`; the app owner fills `change_history_location`. Whatever the audit flag, the app owner fills `confidential_data`, `retention_period` and `disposal_method` for every Important and Critical app, `accessibility_regime` and `accommodation_route` when people outside the business use it or receive its messages or staff must use it, and `ads_employment_use` and `ads_corroboration_and_notice` when Q5 isn't (a).

Don't: store secret values in the record

The registry holds the *name and location* of a secret, never the value (the five never-dos, number 2; AP11). A record that holds a token is a new leak. Record where each secret is held, its scope and its expiry only.

What goes on the one-page app record?

The app record is required from Team up. Copy this into your documents tool.

```

# App record: <app name>          (registry ID: <app_id>)

**Purpose.** One or two sentences: what it does, for whom, and what it replaced.
**Tier and why.** <risk tier>. Answers: Q1 _, Q2 _, Q3 _, Q4 _, Q5 _, Q6 _. Modifiers: M1 _, M2 _, M3
_.
Flags: jurisdiction _, consent gate _, audit scope _.

**People.** Builder: <name, directory account>. App owner: <name, role>, successor: <name>.
Named backup (has run it): <name>. Business reviewer: <name>. Technical reviewer: <name>.

**Data.** What it reads, stores and sends; the highest data class; where each is stored;
the data owner, and who signed off and when (never the app owner for their own app).

**How it works.** A short description, and a link to a diagram if one exists.

**Connections.** Systems it reads or writes. For each: the credential's NAME and where it
is held (never the value), its scope, and its expiry.

**Tests and checks.** Where the test cases are and when they last passed; the last
technical check, exposure scan and restore test.

**Decisions.** Links to the intake, downgrade and exception records, and any good-practice
line skipped, with the reason.

**Change log.** Where version history lives (platform history, version control or a log).

```

For each gate running on a [bounded exception](#), add one row. An exception changes when or how a gate is met, never whether it applies.

Gate	Approver	Reason, and what covers the risk meanwhile	Expires (90 days or less)
(for example, the runbook)	(the approving role, never the builder)	(why the gate can't be met as written, and the interim cover)	YYYY-MM-DD

What goes in the runbook?

The runbook is required from Important up. Write it so that someone who has never touched the app can use it at 2 a.m.

```

# Runbook: <app name>

**Who to call.** Owner, backup, escalation (role and contact route, not personal numbers).
**What "working" looks like.** One check anyone can run, and what the result should be.
**If it stops.** The first three things to check, in order.
**How to restart or restore.** Steps, and where the last tested restore is recorded.
**How to turn it off safely.** What stops, who is told, and what the users do meanwhile.
**Access.** Who can change it, and how access is granted and removed.
**Known limits.** What it can't do, and what users must not rely on it for.

```

What does a filled-in record look like?

Illustrative composite

App: invoice exception sorter (invented). **Tier:** Important, from Q1(c) (supplier bank details) and Q4(c) (the ERP re-performs the match). **Answers:** Q1 c, Q2 b2, Q3 a, Q4 c, Q5 a, Q6 a; M1 and M3 no. **Flags:** jurisdiction and consent gate no; **audit scope yes**, because it touches financial reporting (invoice data flows to the ledger). The compliance owner set its requirements on 2026-09-10 and recorded `control_evidence` = no, because the ERP match is the control the auditors test; the tier stays Important. **People:** builder u-1042; owner, the AP manager (u-0311); backup u-0726. **Data owner:** the finance master-data lead (u-0207) signed off; the AP manager couldn't, as the app's owner. **Checks:** technical check and business review 2026-09-14; last exposure scan 2026-10-01, clean. The matching rows are in [the full registry sheet](#) and [the audit, records and legal sheet](#); [the AI and agents sheet](#) shows separate invented rows.

Assumptions and evidence

- **A-001** (Low): we expect builders to register their apps if registration is quick and unlocks something they want; test this by timing five builders on the 12 core fields, and by comparing registrations with apps found by discovery.
- **A-031** (Low): we expect the intake form, which fills the second group of columns, to take under 10 minutes once it uses skip logic; test this by timing it in the pilot.
- **A-044** (Low): we expect agent and automation identities, grants and tokens can be inventoried and tied to an owner role at a reasonable cost; test this by comparing the AI and agents sheet with your identity provider's grant reports.

Evidence: the field list is derived from the risk-tier model, the audit-scope decision and the legal research behind it, with OWASP guidance on orphaned apps¹ and non-human identities². No controlled study of registry design for worker-built apps exists. Facts on this page are as of October 2026.

Related

- [App registry](#): why each field exists, how to find apps to register and how to keep the registry current.
- [Risk tiers for worker-built apps](#): the six questions, the flags and the gate matrix behind the columns.
- [Get a worker-built app approved](#): the intake form that fills the second group of columns.
- [Document an AI-built app](#): how builders write the app record and runbook.
- [Roles and decision rights](#): who signs downgrades and exceptions, and the separation-of-duties rules.
- [When the app's builder leaves](#): how the person fields drive leaver matching.

1. OWASP Citizen Development Top 10, CD-SEC-09 "Asset Management Failures" (2022; the orphan risk persists). <https://owasp.github.io/www-project-citizen-development-top10-security-risks/content/2022/en/CD-SEC-09-Asset-Management-Failures> [primary], accessed 2026-10-21.

2. OWASP, "Non-Human Identities Top 10" (2025). <https://owasp.org/www-project-non-human-identities-top-10/> [primary], accessed 2026-10-21.

