

CanadaBuysTenderScope

Application overview & workflow guide

CanadaBuysTenderScope is a local tool for browsing government tender notices, screening them against an editable capability profile, analysing individual notices with a local AI model, and saving favourite notice copies with notes. This guide explains what the dashboard shows, how notices are screened and analysed, and where information is stored. It is written for hiring managers, developers and engineers without requiring a full codebase review.

The application separates three activities: browsing saved tender notices, deterministic term screening, and model-assisted analysis. A shortlist match is a reason to inspect a notice—not a suitability or eligibility decision.

1. CSV rows saved

Shows: The number of nonblank rows in the currently saved CSV. Duplicate opportunities can count more than once.

Populated by: Clicking “Fetch latest from CanadaBuys” downloads and validates the CanadaBuys CSV, then saves it locally. The card reports the saved dataset, rather than a count of unique opportunities.

Clicking the card: Shows all saved CSV notices in the notices section, with search and pagination.

Workflow: Read the CSV → build notice records → apply search → display the requested page.

Database use: The notices come from the CSV, not SQLite.

AI use: None. Fetching, counting and browsing do not call the model.

CanadaBuys data source

CanadaBuys open tender notice CSV

<https://canadabuys.canada.ca/opendata/pub/openTenderNotice-ouvertAvisAppelOffres.csv>

2. Shortlisted

Shows: The number of current CSV notices matching at least one screening term in the capability profile.

Populated by: Code checks titles and descriptions against the profile’s terms and records matching reasons.

Clicking the card: Shows only rule-matched notices, with search and pagination.

Workflow: Read CSV and profile → match terms → attach reasons → filter → display.

Database and AI use: No database collection determines shortlist membership, and no model is called. The shortlist is recalculated from the saved dataset and profile.

Analysed & favourites

3. Analysed

Shows: The number of current CSV notices with an accepted analysis saved in SQLite. Accepted unrelated results count; failed attempts alone do not.

Populated by: Clicking “Analyse notice”, receiving a valid model response, passing the evidence check and successfully saving the result.

Clicking the card: Shows current CSV notices with saved accepted analyses. Opening a notice restores its result without rerunning AI.

Workflow: Read description and profile → call model → validate JSON and exact quote → save → display and update count.

Database use: Stores the accepted result, model, timestamp, source hash and latest attempt/error. Reanalysis replaces the accepted result for that notice rather than increasing the count.

AI use: The model interprets the buyer’s request and relevance, supplies a quote, and suggests an offering hypothesis and next investigation. Clicking the card itself does not call AI.

Current limitation: changed CSV files produce new notice IDs. Older analyses do not automatically carry over into the new dataset’s Analysed view.

4. Favourites

Shows: The number of notice copies deliberately bookmarked, including notices no longer in the current CSV.

Populated by: Clicking the bookmark button, optionally entering a note, then clicking “Save”.

Clicking the card: Shows archived copies, with search and pagination. Opening one displays its saved content, date and note.

Workflow: Select notice → open modal → save full notice copy and optional note → update count. Notes can later be edited, and favourites can be removed.

Database use: A separate SQLite table stores the full notice JSON, favourite key, first saved date and note. CSV updates do not overwrite these copies.

AI use: None for bookmarking or browsing. New analysis is disabled in the archived-copy view; an existing stored analysis may be displayed if available.

The important distinction

Analysed tracks accepted results associated with notices in the current dataset. Favourites preserves deliberately saved notice copies, even when they disappear from that dataset.

The capability profile

The editable `capability_profile.json` file tells the application what types of work to look for. Each area has an ID, a readable name and a list of screening terms.

Four screening areas

1. Local-first AI and document search
2. Web applications and backend development
3. Business systems and workflow automation
4. Reporting and data integration

Terms include “chatbot”, “software development”, “Power Automate” and “Power BI”.

One profile, two uses

Shortlisting: Code searches notice titles and descriptions for explicit terms from the profile.

Analysis: The model uses the profile to assess whether the work requested in the description fits the screening areas.

The profile describes screening interests. It is not proof that an organization can deliver the work, meets the buyer’s requirements, or is ready to bid.

Profile excerpt

Partial excerpt from `capability_profile.json`.

```
"description": "Initial rule-based screening profile ...",
"areas": [
  {
    "id": "local_ai",
    "name": "CanadaBuysTenderScope AI and document search",
    "terms": [
      "artificial intelligence",
      "generative AI",
      "large language model",
      "large language models",
      "retrieval augmented generation",
      "semantic search",
      "document search",
      "chatbot",
      "chatbots",
      "conversational AI",
      "local inference",
      "on premises AI"
    ]
  }
],
```

The description is abbreviated for layout; this is an excerpt, not a complete standalone JSON document.

How the shortlist works

The shortlist identifies notices containing capability-profile terms. It does not use AI, assign a score or establish eligibility.

1. Check the title first, then the description.
2. Look for terms from each capability area.
3. Match case-insensitively, allowing spaces, line breaks or common hyphens between phrase words.
4. Shortlist the notice if at least one term matches.
5. Record the matching text and its capability area, with at most one reason per area.

Example: a notice mentioning “Power Automate” matches the business systems and workflow automation area. Nonmatching notices remain available to browse and analyse.

Code excerpt

Inside `shortlist_notice()` in `shortlist.py`; this excerpt relies on the surrounding function and `_find_term()` helper.

```
reasons = []
matched_areas = []

for area in profile["areas"]:
    area_match = None

    for field_name, text in (
        ("Title", title),
        ("Description", description),
    ):
        for term in area["terms"]:
            match = _find_term(term, text)

            if match is not None:
                area_match = (
                    f'{field_name} contains "{match.group(0)}" '
                    f'— screening term for {area["name"]}.'
                )
                break

        if area_match is not None:
            break

    if area_match is not None:
        matched_areas.append(area["id"])
        reasons.append(area_match)

return {
    "shortlisted": bool(reasons),
    "reasons": reasons,
    "matched_areas": matched_areas,
}
```

At least one recorded reason sets `shortlisted` to `True`. Each matching area contributes its ID and a human-readable explanation.

How analysis works

Central question: “Does the work explicitly requested in this description fit the supplied capability profile?”

From click to accepted result

1. Select a notice and click “Analyse notice”.
2. The server retrieves its description from the saved CSV.
3. The app checks that the description is usable and within its input limits.
4. It loads the capability profile and analysis instructions.
5. It sends the description and profile to the local model.
6. The model returns a seven-field assessment.
7. The app validates the JSON, required fields and supporting quote.
8. An accepted result is saved and displayed. A failure is recorded and shown as an error.
9. The Analysed count updates after a successful save.

What the model contributes

The model interprets the buyer’s request and its relevance to the profile, provides a supporting quote, and proposes an offering hypothesis and next investigation. This is a separate step from term matching.

What the application checks

Before accepting the response, the application checks its JSON structure, required fields and exact supporting quote. The quote check verifies that the quoted text appears in the supplied description; it does not independently establish that the interpretation is correct.

What acceptance means

An accepted result passed the application’s validation and was saved successfully. It does not necessarily mean the opportunity is relevant: an accepted unrelated result still counts as Analysed.

Saved analyses can be reopened without calling the model again. Reanalysis replaces the accepted result for the same notice rather than creating an additional counted notice.

Putting the pieces together

Where information lives

Information	Storage / behaviour
Current tender notices	Saved CSV; browsed with search and pagination
Capability areas and terms	Editable capability_profile.json
Shortlist flags and reasons	Recalculated from current notices and profile
Accepted analyses and attempts	SQLite; tied to notice identity
Favourite notice copies and notes	Separate SQLite table; retained across CSV updates

Typical workflow

1. Fetch the latest CSV and review the CSV rows saved count.
2. Open saved notices and use search and pagination.
3. Open Shortlisted and inspect a term-match reason.
4. Analyse a notice and review the accepted result or reported error.
5. Open Analysed to view a saved result without rerunning the model.
6. Bookmark a notice, add a note, and open its archived copy in Favourites.

Boundaries to keep in mind

- Counts refer to different things: CSV rows, term-matched rows, notices with accepted analyses, and saved favourite copies.
- The shortlist count covers the entire saved dataset, regardless of the current search. Filtering happens before pagination.
- Changed CSV files produce new notice IDs, so older analyses do not automatically carry into the new dataset's Analysed view.
- Archived favourites preserve notice content; new analysis is disabled in that view.
- Neither a term match nor a saved analysis establishes business suitability, eligibility, profitability or bid readiness.

About this guide

This guide describes the application's intended behaviour and includes an excerpt from its shortlist code. It is not an independent audit of the application.