RAPID Risk-Register Assistant

Plain-English prototype summary · 23 June 2026 · Bangladesh test build

> **In one paragraph.** WFP country offices keep a formal risk register — a structured list of the things that could go wrong in their operation (funding shortfalls, cyclones, fraud, staff security and so on), each scored for severity and paired with planned actions to reduce it. Today these registers are written and reviewed by hand, so they drift out of date and can contain risks that are vaguely worded or actions that don't meet WFP's own quality standard. We have built a working prototype of an assistant that helps keep a register healthy: it checks the register against WFP's rules, spots risks the register is missing using our live country intelligence, and proposes improvements — but it never changes anything by itself. Every suggestion is queued for a person to approve or reject, with the reasons and evidence recorded.

Why it is useful

A risk register is only as good as it is current and well-written. Three problems recur in practice, and the assistant targets each one:

- **Quality drift** — risks worded loosely, or mitigating actions that aren't specific, measurable or time-bound the way the methodology requires.
- **Blind spots** — a real and growing risk in the country that simply isn't on the register yet.
- **Staleness** — the register reflects a moment months ago, while the situation has moved on.

How it works

Once a day the assistant looks at the current register and does three jobs:

1. **Checks the rules (working now).** It tests every risk and action against WFP's R2 methodology — is the severity score arithmetic correct, does a risk the office has flagged as “beyond our tolerance” actually have an action or a recorded decision to accept it, are deadlines realistic and not all dumped on one date, does each action have an owner. It writes a plain report of everything that falls short.
2. **Finds missing risks (working now).** It reads our live intelligence on the country and proposes risks the register doesn't yet cover — written up in the proper format, always citing the evidence, and always choosing from WFP's official list of risk types (it can't invent one).
3. **Judges relevance (not built yet).** Deciding whether an existing risk is still relevant, duplicated, or out of date — this is the next piece, deliberately left for after you've tried the first two.

The safety principle. The assistant never edits the register itself. Every change — whether suggested by the assistant or typed in by a risk officer — becomes a proposal with its justification and evidence attached, and waits for a human to approve it. Only approved entries reach the register that other systems read. Nothing can be deleted automatically. There is a complete audit trail of who decided what, and why.

How to test it

Everything below runs on the project's development server — the machine you reach with the usual `gcloud` SSH command. Once you are connected, type the lines one at a time. The register has already been loaded with the mock Bangladesh data, so you can run these straight away.

Step 1 — set up access to the AI service (do this once per session):

```
export GOOGLE_APPLICATION_CREDENTIALS=/home/james_king/vertex-sa-key.json
cd /home/james_king/wfp-intelligence
```

Step 2 — do a dry run (looks at everything, writes nothing):

```
.venv/bin/python -m pipelines.risk_register.run_custodian --dry-run
```

You'll see it load 12 risks and 22 actions, run the checks, and report what it found — without changing anything.

Step 3 — do a real run (writes the report and any proposals):

```
.venv/bin/python -m pipelines.risk_register.run_custodian
```

This produces three files in `data/shared/risk-register/proposals/`:

- `BGD-<date>-conformance.md` — the plain report of rule-breaches.
- `BGD-<date>-proposals.md` and `.json` — the missing-risk suggestions, ready for approval.

Open the *.md* files in any text viewer to read them. A run costs about four US cents.

Step 4 — look at the register itself (optional). In the database, this shows the current approved register:

```
SELECT risk_title, seriousness, appetite_status FROM risk_register_current;
```

Step 5 — start over any time, to reset to the clean mock register:

```
.venv/bin/python scripts/seed-bgd-risk-register.py --reseed
```

What it found on the Bangladesh test

Run against your mock register, the rule-checker correctly flagged real problems — proof it isn't just going through the motions:

What it flagged	Why it matters
Two high-severity risks (camp violence; inflated partner reporting) marked “beyond tolerance” but with **no action and no decision to accept them**	The methodology requires either a mitigating action or an explicit management decision — here there was neither.
Almost all actions given the **same end-of-year deadline**; two with **no deadline**; one with **no owner**	The guidance explicitly warns against bunching deadlines, and every action needs a date and an owner.
The **entire “Financial” risk category was absent**	One of WFP's four official categories wasn't represented at all.
A **missing risk** proposed from live intelligence, with evidence cited	Shows the blind-spot detection working on real country data.

What is not done yet

- **Relevance judgement** (job 3 above) — the next build.
- **Other countries** — today it is Bangladesh only; it was built to generalise.
- **Daily automatic schedule and a screen for the risk officer** — for now it is run by hand.

These are scoped and waiting on your direction once you've had a chance to try it.