Demo: Better Prompts → Better Eval Scores

Same task · same 4 test cases · same LLM grader · **5 prompts**, each one small step better

The Setup

Task: turn a raw user bug report into an engineering triage ticket
(Title, Severity P0–P3, Component, Steps, Expected, Actual, Missing Info)

Test cases — each probes a different failure mode:

- Clear repro, workaround exists (expected P2)
- Data loss, version unknown (expected P0)
- Vague report (should ask, not invent)
- Angry tone, cosmetic issue (expected P3)

Grader: `PromptEvaluator` (LLM-as-judge, 1–10, criteria per case) · 2 runs per prompt, averaged

The Prompt Ladder — Average Score

	What changed	Avg score	
v1	Bare request — "Summarize this bug report"	**2.1** ±0.1	
v2	+ Role & clear goal — QA-engineer system prompt; the goal is an actionable triage ticket	**3.4** ±0.4	
v3	+ Output format — Exact fields and layout spelled out	**4.4** ±2.1	
v4	+ XML tags, policy & rules — Severity policy, ignore tone, don't invent, specific questions	**8.9** ±0.1	
v5	+ Example & think first — One worked example (few-shot); decide severity before writing	**9.0** ±0.0	

± = spread between runs. Same grader and criteria every time — only the prompt changed.

Score per Test Case

	Clear repro, workaround exists	Data loss, version unknown	Vague report	Angry tone, cosmetic issue
v1 Bare request	2 / 2	2 / 2	2 / 2	2 / 3
v2 + Role & clear goal	2 / 4	5 / 4	2 / 5	3 / 2
v3 + Output format	9 / 2	9 / 2	6 / 3	2 / 2
v4 + XML tags, policy & rules	8 / 9	9 / 9	9 / 9	9 / 9
v5 + Example & think first	9 / 9	9 / 9	9 / 9	9 / 9

Each cell: score from run 1 / run 2 … (of 2).

v1: Bare request — avg 2.1

Change: "Summarize this bug report"

Grader's reasoning:

> **[2/10] Data loss, version unknown (expected P0)** — The solution fails to meet multiple mandatory requirements. It uses 'High' instead of the required 'P0' severity designation for data loss.

> **[3/10] Angry tone, cosmetic issue (expected P3)** — The solution correctly identifies severity as P3 (cosmetic) and maintains a neutral tone without inflating based on the reporter's urgent language. However, it fails to meet mandatory requirements: the 'Missing Info' field is completely absent, and the field labels don't match the required format (e.g., 'Issue' instead of 'Title', 'Environment' instead of 'Component').

v2: + Role & clear goal — avg 3.4

Change: QA-engineer system prompt; the goal is an actionable triage ticket

Grader's reasoning:

> **[2/10] Clear repro, workaround exists (expected P2)** — The solution violates two mandatory requirements: (1) Severity is marked as P1 when the criteria explicitly state it should be P2 due to the existence of a workaround (credit card payment works), and (2) the required 'Component' field is completely missing. Additionally, the Expected and Actual behaviors are incorrectly placed within the Steps to Reproduce section rather than as separate fields.

> **[5/10] Data loss, version unknown (expected P0)** — The solution correctly identifies P0 severity due to data loss and properly avoids inventing details not in the report. The Missing Info section appropriately asks for app version, platform, and sync information.

v3: + Output format — avg 4.4

Change: Exact fields and layout spelled out

Grader's reasoning:

> **[2/10] Data loss, version unknown (expected P0)** — The solution fails a mandatory requirement. The criteria explicitly states 'Severity is P0, justified as user data loss' but the solution marks it as P1.

> **[9/10] Clear repro, workaround exists (expected P2)** — The solution meets all mandatory requirements: it correctly assigns P2 severity with explicit workaround justification, provides a neutral title under 10 words, includes all required fields, uses a numbered list of steps derived directly from the report, and separates expected/actual behaviors. Missing Info contains specific, relevant questions without inventing details.

v4: + XML tags, policy & rules — avg 8.9

Change: Severity policy, ignore tone, don't invent, specific questions

Grader's reasoning:

> **[8/10] Clear repro, workaround exists (expected P2)** — The solution meets all mandatory requirements: it has a neutral title under 10 words, correctly assigns P2 severity with proper justification, identifies the component, provides a numbered list of steps from the report, separates expected and actual behaviors, and includes specific questions in Missing Info. No mandatory requirements are violated.

> **[9/10] Clear repro, workaround exists (expected P2)** — The solution meets all mandatory requirements: it uses all required fields, correctly assigns P2 severity with proper justification referencing the credit card workaround, provides a numbered list of steps accurately derived from the report, and separately states expected (order completes) and actual (infinite spinner) behaviors. Missing Info consists entirely of specific questions about browser/device scope and console errors, with no invented details.

v5: + Example & think first — avg 9.0

Change: One worked example (few-shot); decide severity before writing

Grader's reasoning:

> **[9/10] Angry tone, cosmetic issue (expected P3)** — The solution meets all mandatory requirements: correct P3 severity assignment, neutral title under 10 words, proper component identification, all required fields present, and no invented details. The missing info questions are appropriately specific.

Before / After — "Data loss" Case

v1 — score 2/10

```
## Bug Report Summary

**Issue**: Mass data loss – all user drafts deleted after app update

**Severity**: High

**Details**:
- ~40 saved draft notes disappeared following an overnight app update
- User did not manually delete anything
- Lost content includes unrecoverable work-related notes

**Expected behavior**: Drafts should persist across app updates

**Priority**: High – user reports critical work data loss
```

v5 — score 9/10

```
Title: All saved drafts missing after app update

Severity: P0 – data loss with no workaround (all 40 drafts perman

Component: Drafts / Notes feature

Steps to Reproduce:
1. Unknown – see Missing Info

Expected: Saved drafts persist after app update
Actual: All drafts are gone; the Drafts folder is empty

Missing Info:
- What platform is the app on (iOS, Android, web)?
- What was the previous app version and the version installed in
- Was there any error message or warning during the update proces
```

Takeaways

- **The model can't read your mind.** v1–v2 lost points on fields and a severity scale they were never told about
- **Format alone isn't enough.** v3 got the fields right but *guessed* severity — scores swung between runs
- **Biggest jump = policy + rules** (v4): explicit severity policy, "ignore tone", "don't invent"
- **Examples buy consistency** (v5): same score every run, steps marked *Unknown* instead of made up
- **Judges are fallible too:** a v3 ticket invented "Mobile App" and plausible steps, and still got 9/10 — spot-check outputs, don't just trust the number