



Pramaan

प्रमाण · proof

Citizen voice → funded, verified public works

A citizen reports in their own language. Pramaan understands it, ranks it in the open, finds the national scheme that can pay for it, and calls it fixed only when the citizens who asked say so.

Ramkumar K R · Mahakisore M · Jaswanth S github.com/Monolithic-Dev/Pramaan



11

languages by voice, text or photo



5

intake channels incl. WhatsApp and SMS



36

states and UTs, plus Brazil



₹0

to run a pilot on free tiers



+60%

more people reached (demo data)



69/69

end-to-end checks on real Gemini

One broken culvert. Ten reports. No answers.



Fragmented

The same broken culvert is reported ten times, in five languages, across a helpline, WhatsApp groups, a portal and a ward meeting. Nobody can say how many people are really affected.



Misaligned

Budgets follow habit or whoever shouts loudest. Nobody checks whether the most vulnerable areas get served, or which national scheme could have paid for the work.



Unmeasured

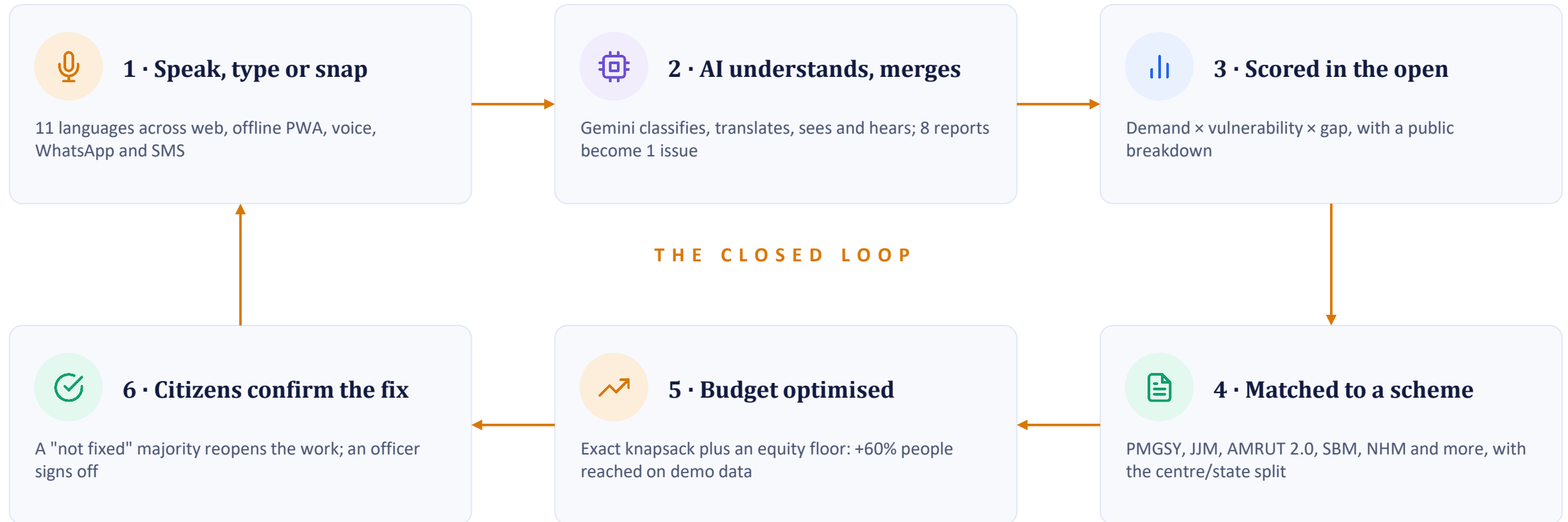
A project is "completed" on paper. Nobody asks the people who reported it whether anything actually changed.



The Track 01 challenge

Build a multilingual Digital Public Good that gathers citizen requests by voice, text and messaging apps, fuses them with demographic, infrastructure and investment data, and recommends priority projects to policymakers.

Pramaan closes the loop, from voice to verified fix



Maps one to one to the Track 01 brief: aggregate multilingual requests, fuse them with demographic, infrastructure and investment data, surface hotspots, recommend priority projects.

Three audiences, one platform



Citizens

- Report in 11 languages by voice, text or photo
- Snap a photo: Gemini drafts the report and flags hazards
- No smartphone? Text STATUS <code> to hear the outcome
- Track by code with no account; confirm the fix
- Get updates in your own language



Officers

- Ranked, explainable priorities and a live map
- SLA clocks with automatic escalation
- Budget optimiser and national scheme matcher
- Need-vs-spend map and scheme impact ledger
- Grounded AI co-pilot and printable weekly briefing



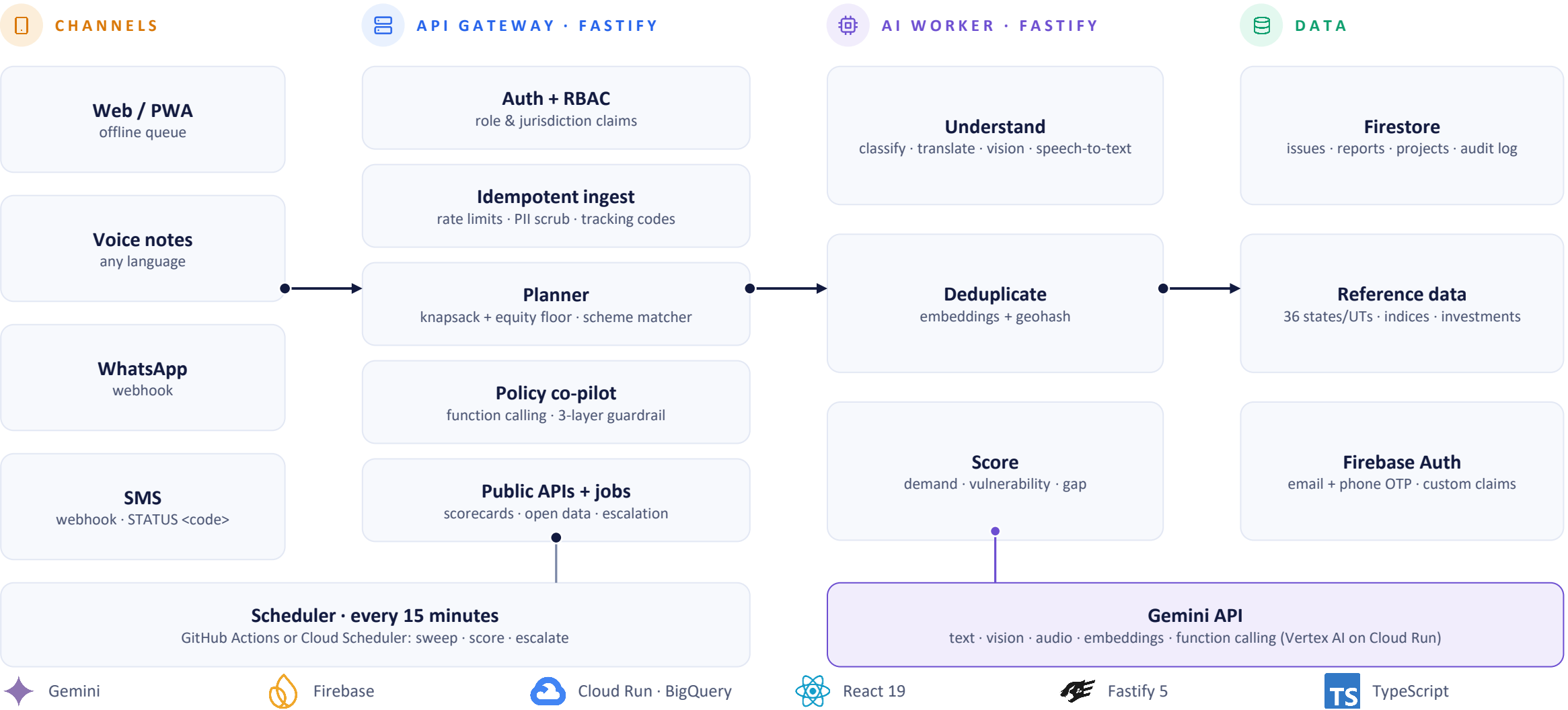
The public

- Follow the money: does spending reach need?
- District scorecards graded A to E by a published formula
- Open data: CSV plus a try-it API
- Aggregates only, k-anonymous, no login

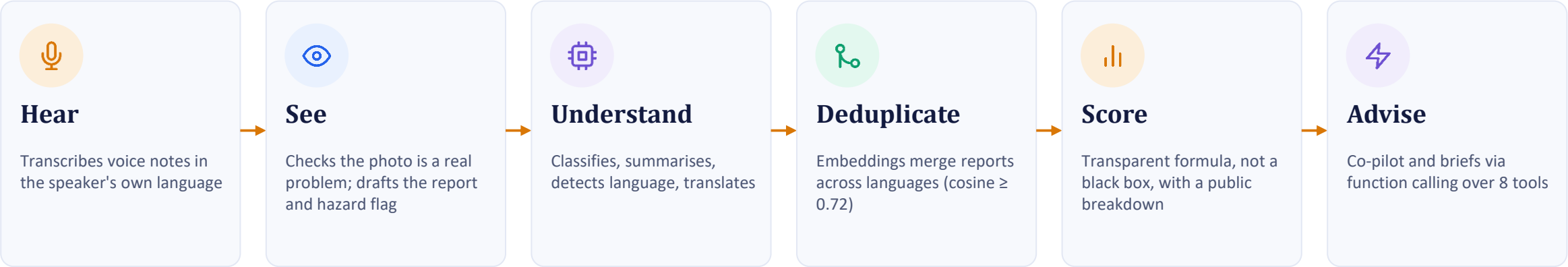


Access is enforced on the server, on every request. Citizen, field officer, district collector, state or national admin. Jurisdiction is a region tree (country, state, district): a Pune collector gets a 403 on a Delhi issue.

Architecture: stateless and Gemini-native



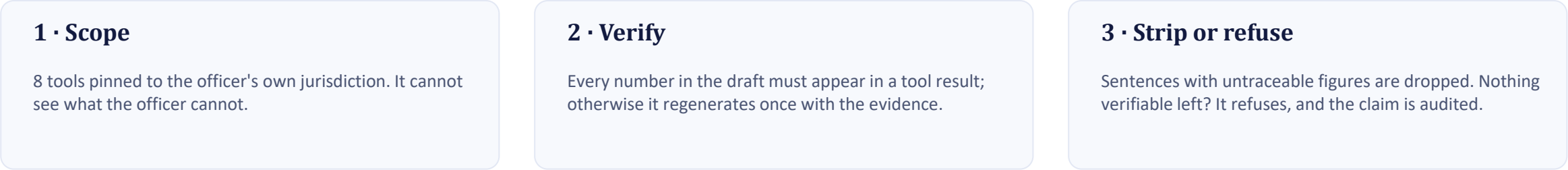
Gemini at every stage, a fallback at every step



$$priority = (0.40 \times demand + 0.30 \times vulnerability + 0.30 \times gap) \times duplication \times efficacy$$

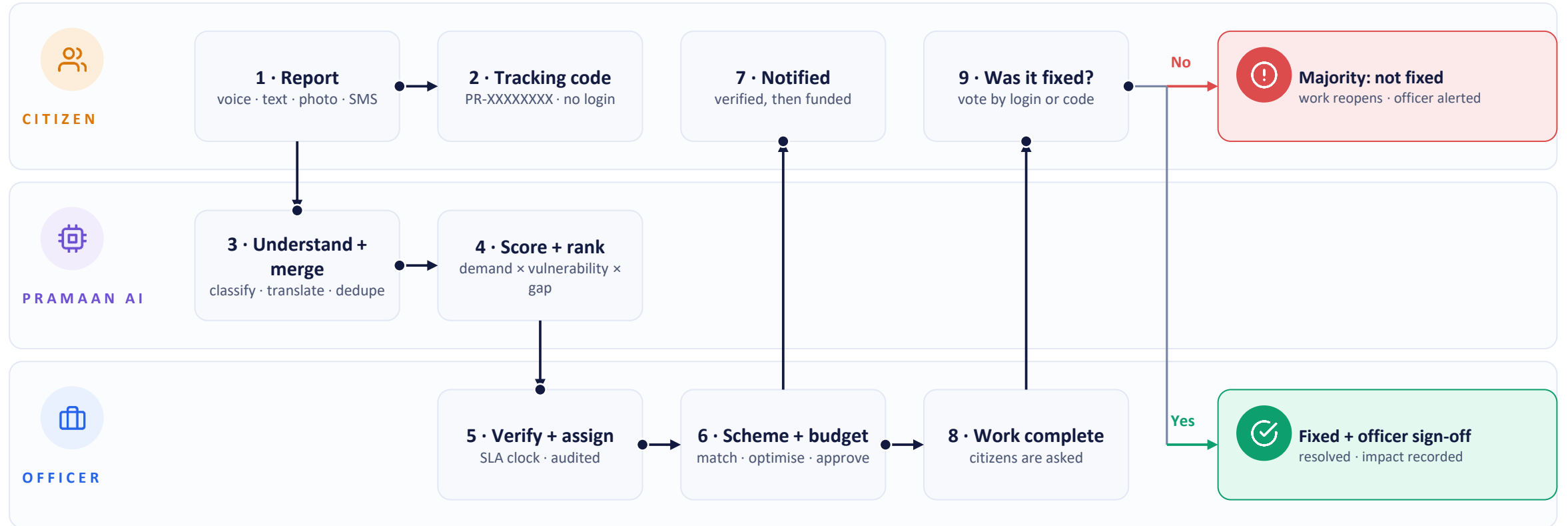
Every issue shows its full breakdown. Weights are configurable, and the weights lab shows policymakers how the ranking would move.

THE CO-PILOT'S THREE-LAYER GUARDRAIL



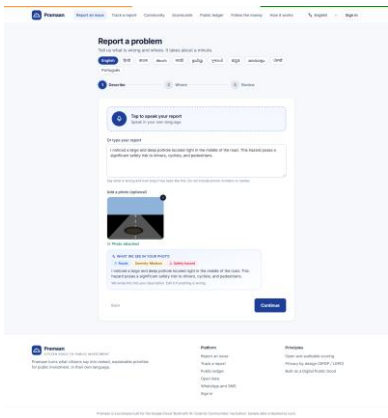
Resilience: a model pool skips hung or overloaded Gemini models, and malformed output falls back to raw text, so a report is never dropped.

User flow: from complaint to confirmed fix

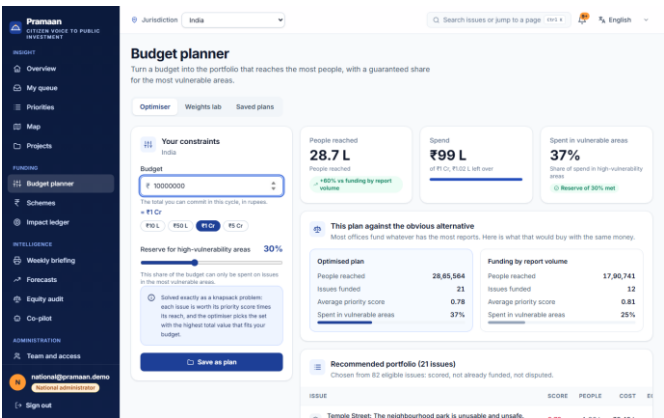


Only citizens plus an officer's sign-off can close an issue. Every status change is made on the server, audited, and sent to the reporter in their own language. No smartphone? Text STATUS <code> and get the answer by SMS.

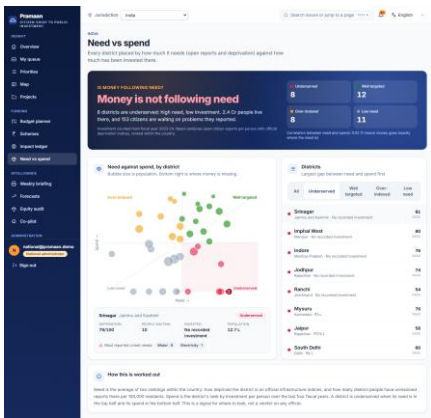
A working prototype, not a concept



Citizen: snap a photo, Gemini writes the report



Officer: budget optimiser with an equity reserve



Policymaker: is money following need?

+60%

more people reached than funding by report volume*

$r = 0.02$

need vs investment; 8 districts, 2.4 crore people underserved*

328

unit and route tests passing

69/69

end-to-end checks on real Gemini

*Measured on illustrative sample data (about 430 issues and 1,850 reports), labelled as such in the app and API.

Built against the rubric, criterion by criterion

25%**AI / Technical Execution**

Gemini across text, vision, audio, embeddings and function calling. Guardrailed co-pilot, model-fallback pool, exact knapsack. 328 tests plus 69/69 end-to-end checks on real Gemini.

20%**Problem-Solution Fit**

Each gap in the brief has a feature. Fragmented: AI dedup (8 reports, 1 issue). Misaligned: optimiser, scheme matcher, need-vs-spend. Unmeasured: citizen-confirmed fixes.

20%**Depth & Reach Across India**

11 languages, 36 states and UTs, 5 intake channels including offline PWA and SMS status for feature phones. Brazil runs on the same codebase.

20%**Deployability & Scalability**

₹0 pilot on free tiers from one Render blueprint; Cloud Run, Pub/Sub, BigQuery and Vertex AI to scale. Stateless, idempotent, scheduled jobs.

15%**Impact Potential**

Ties demand to real central schemes (PMGSY, JJM, AMRUT 2.0, SBM, NHM) and measures people benefited, days to fix, cost per person and citizen confirmation.

Deployable this week. National by design.



Free pilot path · ₹0

Render static site, API and worker, Firebase Spark (Firestore + Auth), a Gemini API key and a 15-minute GitHub Actions cron. Runs locally on the Firebase emulators too.



Scale path · same code

Cloud Run gateway and worker, Pub/Sub, BigQuery reference data, Vertex AI, Cloud Scheduler. Switching is configuration only: WORKER_URL vs Pub/Sub, Firestore vs BigQuery.

HOW IT SCALES ACROSS INDIA



Adding a state is a form; adding a country is a config document



11 languages loaded on demand, 36 states and UTs in the reference geography



Offline PWA queue for rural connectivity: reports, photos and voice sync later



Feature phones get the whole loop: report by SMS, text STATUS <code>



MIT-licensed Digital Public Good that any state or country can run and adapt

NEXT IVR phone line · Learned ranking · Live LGD and PFMS data · DigiLocker (India Stack) · National aggregation

Pramaan (प्रमाण): proof a citizen was heard, proof of where the money went, proof that it worked.

Ramkumar K R · Mahakisore M · Jaswanth S
github.com/Monolithic-Dev/Pramaan