
Model selection for Hindi form-filling voice agents under noisy speech, cost and latency constraints

Aman Dalmia¹ Sanskriti Midha¹ Jigar Doshi²

¹ARTPARK, I-Hub, Indian Institute of Science ²Indian Institute of Science
{aman.dalmia,sanskriti,jigar}@artpark.in

Abstract

We describe how we evaluated a voice agent, now being piloted in India with low-income, Hindi-speaking mothers, that fills a structured maternal-care enrollment form over a phone call. A wrong value captured or a call ended early means a mother is not enrolled. Those stakes necessitate pre-deployment testing and rule out building an evaluation dataset from real callers’ data, which is personally identifiable. We therefore built our own: native Hindi-speaking women re-recorded failure modes observed across dozens of pilot calls under the noisy conditions callers speak from, giving 380 recordings assembled into 960 simulated calls and 3,760 turn-level tests drawn from those multi-turn calls. We score each component in isolation and in an end-to-end pipeline to separate the errors a component makes from those it inherits. The best model extracts almost perfectly from reference transcripts (99.79%), but median form completion falls to 85–92% on real transcripts. Per-turn evaluation also ranks models incorrectly: under the deployed transcription model, the model ranked 10th per turn completes more form fields than the model ranked 3rd because some of its errors are recoverable by rules. This lets smaller, cheaper models match larger ones on form completion. Finally, no model wins on accuracy, cost and latency at once. We show how we make that trade-off when choosing the models we deploy.

1 Introduction

Many social benefits in India start with a form, yet more than half the women in the lowest wealth quintile are not literate [7]. Someone must talk to them and fill the form on their behalf. Today that work falls to frontline health workers who enroll beneficiaries one at a time, a poor use of their stretched capacity. We built FormBharo (“fill the form” in Hindi), a voice agent that fills a structured form over a phone call under tight latency and cost budgets. It is being piloted with ARMMAN [1], a non-profit organization working on maternal and child health, to enroll low-income, Hindi-speaking mothers in maternal care programs in rural Maharashtra.

The form is not a flat list of questions (Figure 1, Appendix A): “name” is open-ended, district is restricted to a fixed set of values, answers to conditional questions (like, whether the user is pregnant) dictate which questions come next, and incorrect values could end the call prematurely. FormBharo (Figure 2, Appendix B) fills it with a hybrid design: a speech-to-text (STT) model transcribes the caller’s speech and an LLM (called EXTRACT) extracts the relevant form values. Rules validate the extracted value (*is the phone number 10 digits? is the child’s date of birth a valid past date?*); valid values are stored, and a second set of rules picks the next question to ask, given the current state of the form. A second LLM, REPLY, phrases that question naturally, which a text-to-speech (TTS) model speaks back to the caller, or it ends the call once every field has been answered.

Plenty goes wrong on a real call. Users call from areas with background noise, and the hardest case is multiple background speakers. They speak in colloquial terms and alternate between Hindi and

Hinglish. Named entities remain difficult for end-to-end ASR [9]. Callers often say a number in groups of two or three digits, pausing between them. The agent has to distinguish a pause that means the caller has finished from one that means more digits are coming. All of this has to feel real-time to the caller, so each LLM gets a per-turn latency budget (Section 4). Cost matters as much: our target is below Rs 2 per call at scale, which the current deployment does not yet meet.

Existing voice-agent benchmarks do not measure completion of a constrained, branching form [3, 4, 8], and Indic speech benchmarks score transcription accuracy, not the task the transcription feeds into [2, 6], so we built an evaluation for that gap and used it to choose the models we deploy.

2 Evaluation design

Each error above belongs to one component: the STT model, the extraction LLM or the reply LLM. Several TTS models sound good enough, so we exclude TTS from this first evaluation. We test the agent at two levels. *Unit tests* score each component in isolation; *end-to-end tests* pass outputs from one component to the next. Together they give us 380 audio recordings and 3,760 turn-level tests drawn from 960 multi-turn simulated calls.¹

2.1 Dataset creation

We define 5 *simulated user profiles* (“personas”), each with fixed personal details corresponding to the fields we intend to capture, and generate several simulated calls per persona. The expected flow for a call is the agent asking one question at a time and the user answering it. The form’s conditional questions create multiple branches, and the user’s past answers decide which one a call follows. In total, the four conditional points in the form produce $2 \times 3 \times 4 \times 2 = 48$ distinct paths through it, one call per path (Appendix C), so across the 5 personas we get $5 \times 48 = 240$ calls.

We also account for 3 *acoustic variations* drawn from the conditions callers speak from: background noise (ambient chatter and nearby speakers), microphone distance and speaking pace. Each call uses a single condition throughout, since a caller’s environment does not change mid-call, and each of the 240 calls is recorded under each variation and in an ideal environment, yielding $240 \times 4 = 960$ calls. The *value* is generated once per persona whereas the *transcript* is generated per call with that value as a reference, under field-specific constraints defined with ARMMAN from the failure modes seen in the pilot: names and districts with phonetically hard spellings, or a user unable to recall her phone number. Both use GPT-5.5 [11].

Deduplicating the transcripts of all 960 calls leaves 380 unique ones to be recorded. We recruited five native Hindi speakers close to the target demographic: all female, aged 18–35, from Uttar Pradesh and Maharashtra, to cover differences in accent and colloquialisms. They recorded each answer under a directive such as “*keep the mic at least 25 cm away*”, and supervisors checked every clip against it. No real caller data enters the benchmark at any stage: every recording is a synthetic transcript read by a speaker recruited and compensated for this study, who consented to its release.

2.2 Unit tests

Speech-to-text. Word Error Rate (WER) is commonly used for comparing STT models, but it is a poor metric for Indic languages [13] and for agents [12]: it counts every surface difference as an error (“nine” vs “9”), even when the meaning is unchanged. So we use LLM-WER [13] instead: an LLM judge reviews every mismatch flagged by WER and classifies whether it is semantically equivalent or phonetically similar, and WER is recomputed over the genuine errors.

Data extraction accuracy. We prepare unit tests at the *turn level* from every call, since that is where every mistake happens. Each test case contains the latest user response and the preceding conversation history as *input*, and the expected form values as the *ground truth*. Numbers, dates and booleans are compared exactly, but open-ended fields like names can have many phonetic forms (“*Lakshmi*” versus “*Laxmi*”), so we use a binary LLM judge [14] for them, calibrated against human labels (Appendix D). To ensure every error is attributable to the agent’s mistake in that turn, the agent

¹Every evaluation run, with the benchmark dataset and each model’s outputs, is public on Calibrate [5] (Appendix F).

is assumed to have responded perfectly to all previous turns, and only its last response is scored. Deduplication yields 1,880 unit tests, over which we report the mean *extraction accuracy*.

Response quality. We prepare these unit tests the same way, with two differences. Each test case also receives a tool call carrying the rule-based layer’s decision about which question to ask next. REPLY then generates that question or ends the call; we evaluate the end-call tool call by exact match. Every other reply is graded on 6 independent dimensions, each with its own binary LLM judge: *correctness* (asks the question the rule-based layer selected), *Hindi adherence* (the reply is in Hindi), *conciseness* (one line), *no acknowledgment* (does not acknowledge the caller’s previous answer), *no value echo* (does not repeat the caller’s answer back) and *script fidelity* (uses the wording the form prescribes for that question). A response is accurate if it passes all six judgments or ends the call at the right time. Deduplication yields 1,880 unit tests: 920 require a reply, and 960 check whether the call ends correctly.

2.3 End-to-end testing

Unit tests assume perfect inputs, but the deployed agent passes one component’s output to the next. We therefore replace the reference transcripts with STT transcripts and compute *form-completion accuracy*. For each call, we start with an empty form, accumulate the values extracted at each turn and compare the result with the expected values. Form-completion accuracy is the fraction of fields that are correct at the end of the call, averaged over calls. Conversation history is held fixed, so an incorrect extraction does not change which question appears next.

3 Findings

We benchmark five STT models and 11 LLMs on accuracy, p95 latency and cost; full results and implementation details are in Appendix I.

Transcription quality. WER and LLM-WER disagree: Nova-3 has the best WER yet the second-worst LLM-WER. Our WER is inflated too: the references are Romanized Hindi, the predictions Devanagari. So we rank by LLM-WER. Chirp 3 has the best LLM-WER but costs at least twice as much as any other model, Scribe v2 performs close to it at less than half of the cost, and GPT-4o-transcribe is the least accurate. Eliminating the first and last leaves Scribe v2, Saaras v3 and Nova-3 for the end-to-end tests.

Turn-level extraction accuracy. With perfect transcription, frontier models are almost perfectly accurate (Appendix I). With real transcripts the median drop is modest for Scribe v2 (0.42 percentage points [pp]) but larger for Saaras v3 and Nova-3 (3.14 points). Individual drops are not uniform across transcription models: GLM-5.1 falls 35 points on Scribe v2 but holds 94.52% on Saaras v3 and 95.43% on Nova-3. The leaderboard changes too: the best accuracy falls to 98.94% (Gemini 3.5 Flash and Claude Sonnet 4.6, Scribe v2) from GPT-5.5’s 99.79% with perfect transcription.

End-to-end form completion. With the deployed Scribe v2 transcripts, GPT-4.1 ranks 10th per turn (78.94%) yet completes more form fields (91.50%) than GPT-5.5, ranked 3rd (98.62%) at 89.37% (Table 1); Mistral Medium 3.5 inverts the same way, 8th per turn (92.34%) yet 91.84%. Per-turn accuracy scores the raw LLM output; form completion scores what is stored after the rule-based layer runs. The errors we inspected are numeric fields returned as strings, which that layer normalizes before storing. On real transcripts, form completion degrades far more than extraction: a median of 7.67, 7.38 and 13.56 points against per-turn drops of 0.42–3.14 points (Table 2). The rule layer cannot repair them all. These drops are a lower bound for a live call.

Response quality. REPLY behaves the same way, losing a median of 2.23 points on real transcripts (Table 3).

4 Selecting which models to deploy

We choose the three models in the order the call flows through them, since each consumes the output of the one before it.

Table 1: Form-completion accuracy (%) under perfect transcription and under the deployed STT model. The leaderboard differs from per-turn extraction: under Scribe v2, GPT-5.5 is 3rd on extraction but 10th here, while GPT-4.1 is 10th on extraction but 7th here. Saaras v3 and Nova-3 columns and uncertainty values are in Appendix I.

Model	Reference	Scribe v2
Gemini 3 Flash	100.00	89.48
Gemini 3.5 Flash	100.00	92.50
GPT-5.5	99.95	89.37
Claude Sonnet 4.6	99.66	93.01
GLM-5.1	99.56	58.66
Claude Opus 4.8	99.41	92.03
Gemini 2.5 Flash	99.06	92.39
Gemini Pro	98.90	93.47
GPT-5.4-mini	98.35	90.10
GPT-4.1	95.99	91.50
Mistral Medium 3.5	92.83	91.84

Table 2: Drop in accuracy when real transcripts are used. Med. is the median across 11 LLMs; Δ is the median of each model’s drop from its reference score.

Model	Extraction		Form completion	
	Med. (%)	Δ (pp)	Med. (%)	Δ (pp)
Saaras v3	94.52	−3.14	90.74	−7.67
Scribe v2	97.29	−0.42	91.84	−7.38
Nova-3	95.16	−3.14	85.50	−13.56

Table 3: REPLY accuracy (%), p95 latency and cost, with Gemini 3.5 Flash as EXTRACT.

Model	Ref.	Scribe	Lat. (s)	\$/turn
Claude Sonnet 4.6	100.00	97.77	5.15	0.0146
GPT-4.1	100.00	96.60	3.65	0.0025
GPT-5.4-mini	98.09	96.70	3.36	0.0008
Gemini 3 Flash	97.23	96.17	2.66	0.0017
Gemini 3.5 Flash	97.02	94.73	3.50	0.0052

STT. Scribe v2 gives the highest median extraction (97.29%) and form-completion (91.84%) accuracy, and the smallest drops (Table 2), so we deploy it.

EXTRACT. We first require form-completion accuracy of at least 90% on Scribe v2 transcripts and p95 latency below 5 seconds. This removes GLM-5.1, GPT-5.5 and Gemini 3 Flash on accuracy, and Gemini 2.5 Flash and Gemini Pro on latency. Six candidates remain. Claude Sonnet 4.6 is the most accurate (93.01%), Mistral Medium 3.5 the fastest (1.75 s p95) and GPT-5.4-mini the cheapest (\$0.0011 per extraction turn; Appendix G).

All six candidates lie on the Pareto frontier: none leads on accuracy, latency and cost at once (Figure 3, Appendix G). We therefore min–max normalize the three axes and rank models by weighted-sum scalarization [10]. With latency weight fixed at 0.1, Gemini 3.5 Flash leads when the accuracy weight is 0.50–0.61, while Claude Sonnet 4.6 leads above that range. Our aim is to bring each call below Rs 2, which the current stack does not yet meet. We therefore operate at the cost-conscious end of the weight range and deploy Gemini 3.5 Flash. Its form completion overlaps Sonnet’s within the reported uncertainty (92.50 ± 0.53 vs. 93.01 ± 0.50), while it responds faster and costs less than half as much per turn.

REPLY. Applying the same procedure with p95 latency below 4 seconds and accuracy above 95% selects GPT-5.4-mini (Table 3).

5 Conclusion

We could not test our agent on real callers, so we reproduced their failure modes synthetically. That was enough to choose the three component models we deployed. End-to-end testing revealed that per-turn accuracy ranked our models incorrectly, and rules recovered enough errors for a cheaper model to match a larger one. Public benchmarks can inspire, but every agent requires bespoke evaluation.

Limitations. The benchmark uses five simulated personas and scripted recordings, with one acoustic condition per call. Its end-to-end tests hold conversation history fixed, so they measure error propagation but not how an incorrect extraction changes the next question or triggers a retry. GPT-5.5 generated the reference values and transcripts, and tops the reference extraction leaderboard. So, the benchmark’s hard cases may reflect that model’s own biases.

Acknowledgments and Disclosure of Funding

We thank Amrita Mahale, Parina Anand and Hetvi Lodaya at ARMMAN for designing the enrollment form, testing the agent through successive iterations, and sharing the insights from the field that guided its design. We thank the data ops team at ARTPARK for their help with data collection and annotation. Finally, we are grateful to the frontline health workers and mothers who tested FormBhara and shared their feedback. This work was funded by the Government of Karnataka.

References

- [1] ARMMAN. ARMMAN: Advancing reduction in mortality and morbidity of mothers, children and neonates. <https://armman.org>, 2008. Accessed: 2026-09-29.
- [2] A. Bhanushali, G. Bridgman, D. G. P. K. Ghosh, P. Kumar, S. Kumar, A. R. Kolladath, N. Ravi, A. Seth, A. Singh, V. N. Sukhadia, S. Umesh, S. Udupa, and L. V. S. V. Durga Prasad. Gram vaani ASR challenge on spontaneous telephone speech recordings in regional variations of Hindi. In *Proceedings of Interspeech 2022*, pages 3548–3552. ISCA, 2022. doi: 10.21437/Interspeech.2022-11371. URL https://www.isca-archive.org/interspeech_2022/bhanushali22_interspeech.pdf.
- [3] T. Bogavelli, G. Gauthier Melançon, K. Stankiewicz, O. Bangbose, F. Riols, H. H. Nguyen, R. Mehndiratta, L. D. Brin, J. Marinier, H. Subramani, A. Madamala, S. K. Nemala, and S. Sunkara. EVA-Bench: A new end-to-end framework for evaluating voice agents, 2026. URL <https://arxiv.org/abs/2605.13841>.
- [4] Y. Chen, X. Yue, C. Zhang, X. Gao, R. T. Tan, and H. Li. VoiceBench: Benchmarking LLM-based voice assistants. *Transactions of the Association for Computational Linguistics*, 14: 378–398, 2026. doi: 10.1162/tacl.a.628. URL <https://doi.org/10.1162/tacl.a.628>.
- [5] A. Dalmia and J. Doshi. Calibrate: An open-source framework for evaluating AI agents. <https://calibrate.artpark.ai>, 2026. Source code: <https://github.com/ARTPARK-SAHAI-ORG/calibrate>.
- [6] A. Diwan, R. Vaideeswaran, S. Shah, A. Singh, K. M. Srinivasa Raghavan, S. Khare, V. Unni, S. Vyas, A. Rajpuria, C. Yarra, A. R. Mittal, P. K. Ghosh, P. Jyothi, K. Bali, V. Seshadri, S. Sitaram, S. Bharadwaj, J. Nanavati, R. Nanavati, and K. Sankaranarayanan. MUCS 2021: Multilingual and code-switching ASR challenges for low resource Indian languages. In *Proceedings of Interspeech 2021*, pages 2446–2450. ISCA, 2021. doi: 10.21437/Interspeech.2021-1339. URL https://www.isca-archive.org/interspeech_2021/diwan21_interspeech.pdf.
- [7] IIPS and ICF. National family health survey (NFHS-5), 2019–21: India report. Technical Report FR375, International Institute for Population Sciences, Mumbai, 2021. URL <https://dhsprogram.com/pubs/pdf/FR375/FR375.pdf>.
- [8] D. Jain, H. Shukla, G. Rajeev, A. Kulkarni, C. Khatri, and S. Agarwal. VoiceAgentBench: Are voice assistants ready for agentic tasks?, 2025. URL <https://arxiv.org/abs/2510.07978>.
- [9] Y. Li, M. Zhang, C. Su, Y. Li, X. Qiao, M. Ren, M. Ma, D. Wei, S. Tao, and H. Yang. A multitask training approach to enhance Whisper with contextual biasing and open-vocabulary keyword spotting. In *Interspeech*, 2024. URL <https://arxiv.org/abs/2309.09552>.
- [10] R. T. Marler and J. S. Arora. The weighted sum method for multi-objective optimization: new insights. *Structural and Multidisciplinary Optimization*, 41(6):853–862, 2010. doi: 10.1007/s00158-009-0460-7.
- [11] OpenAI. GPT-5.5 system card. <https://openai.com/index/gpt-5-5-system-card/>, Apr. 2026. Accessed: 2026-07-23.
- [12] Pipecat AI. stt-benchmark: Benchmarking speech-to-text with semantic WER and TTFS latency. <https://github.com/pipecat-ai/stt-benchmark>, 2026. Accessed: 2026-07-23.
- [13] Sarvam AI. Evaluating Indian language ASR. <https://www.sarvam.ai/blogs/evaluating-indian-language-asr>, Apr. 2026. Accessed: 2026-06-29.
- [14] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Appendix

A Call flow of the enrollment form

Figure 1 shows the flow the agent follows on a call.

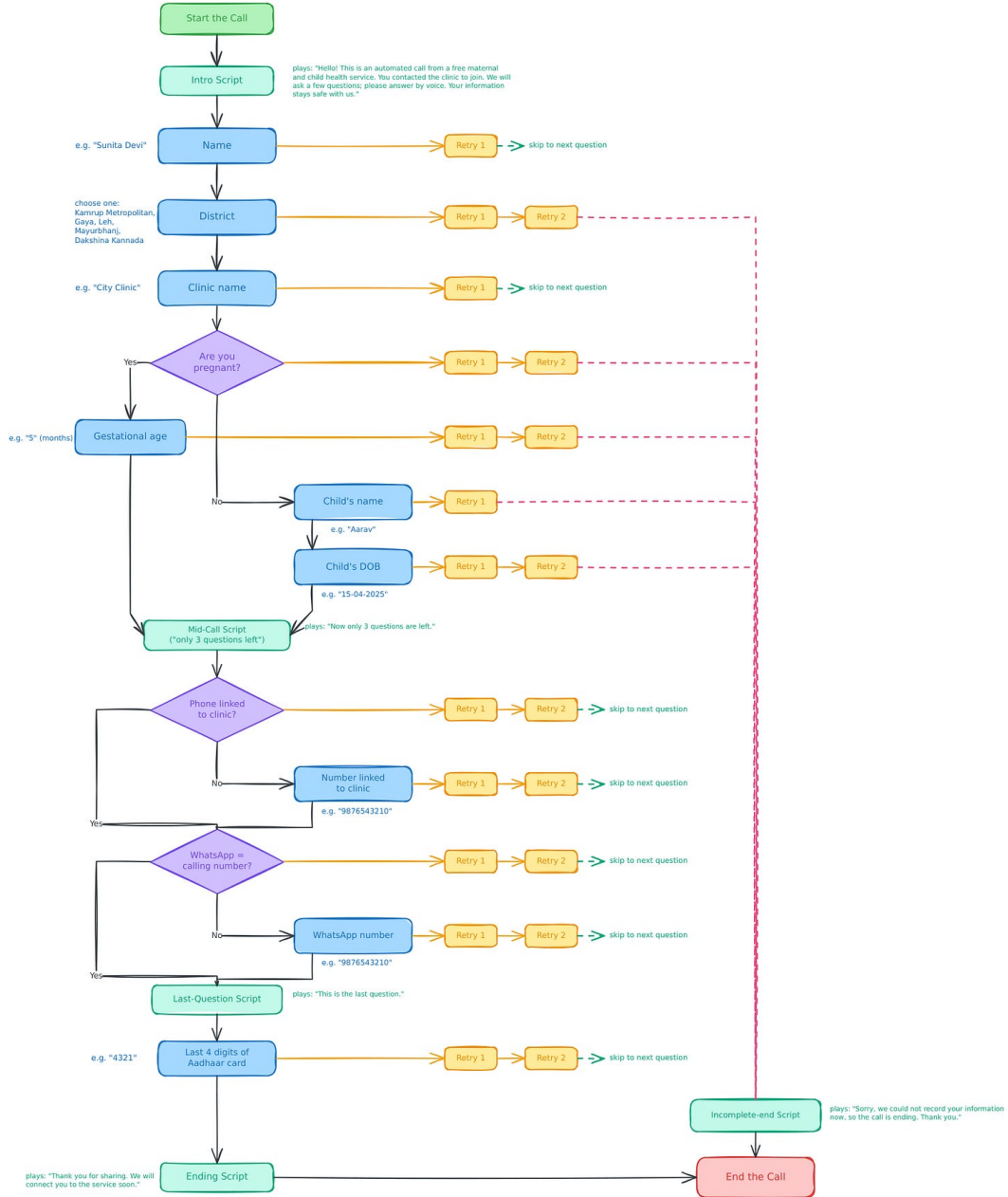


Figure 1: Call flow of the enrollment form being tested. The agent speaks first and asks one question at a time, in the order shown. The answers to the conditional questions (“are you pregnant”) decide which questions are asked next. Invalid answers either cause the flow to skip to the next question or end the call prematurely. Aadhaar is India’s national ID. The district options shown are evaluation values chosen for phonetically hard spellings and spread across states; they are not the pilot’s district set.

B How FormBharo works

Figure 2 shows how a single turn moves through the agent.

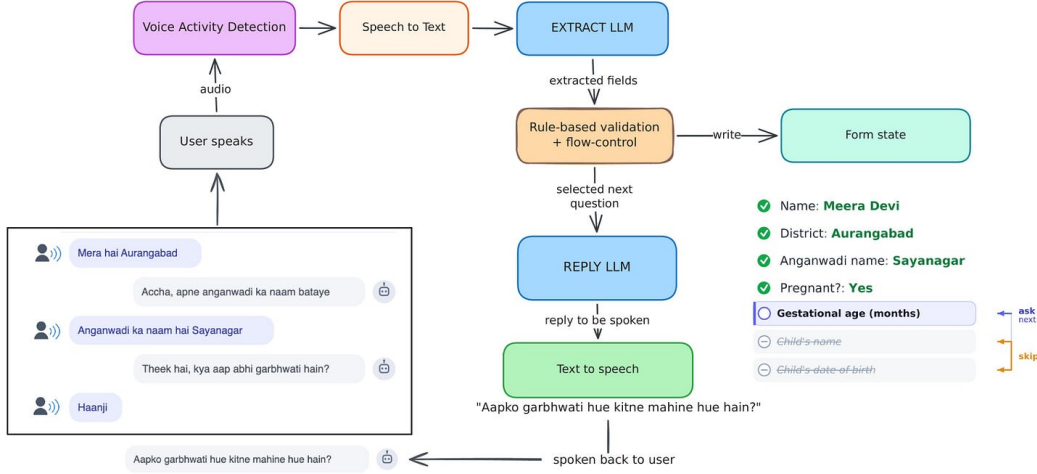


Figure 2: How FormBharo works. EXTRACT LLM extracts the form values from the transcribed text. A rule-based layer validates the extracted values, updates the form, and selects which question to ask next. REPLY LLM phrases that question to be spoken back to the user through a TTS model. The dialogue is schematic rather than a scored benchmark item. The “Anganwadi name” (shown as “Clinic name” in Figure 1) is the public clinic’s name.

C Branches used to generate the calls

The 48 paths per persona come from four fields whose outcome varies, including retry exhaustion:

- two pregnancy branches (*pregnant or not*);
- three branches for whether the calling number is linked to the clinic (*linked; not linked and correct value provided; or not linked but not able to remember the linked number*);
- four choices for whether the WhatsApp number is the same as the calling number (*same; not same and answered; not same but unable to recall the WhatsApp number; cannot recall if they are the same*);
- two branches for the Aadhaar digits (*answered or not known*).

D LLM judge calibration

We compared every LLM judge’s output with our own labels on a small subset of the unit tests and based on the disagreements, we tuned each judge’s prompt until it agreed with us on every test case in that subset. We then ran each updated judge on a larger set of unit tests it had not been tuned on, and found that it agreed with our labels on every test case. Although full agreement with LLM judges is rare, we believe we reached it because we created a separate judge for each narrow, objective dimension of the response, rather than one judge for its overall quality, which is subjective.

E Speech-to-text results

Table 4 compares the five STT models on transcription quality and cost. Since the models have different billing units, the total cost of transcribing the entire benchmark is reported.

The absolute WER values in Table 4 are separately inflated by a script mismatch: our reference transcripts are Romanized Hindi whereas most models return Devanagari, so a correctly transcribed word still counts as a substitution. That is why WER exceeds 1.0 here.

Table 4: Comparison of STT models. Lower is better throughout. Ranking by LLM-WER rather than WER changes which model we would pick.

Model	WER ↓	LLM-WER ↓	Cost (\$) ↓
Scribe v2	0.835 ± 0.015	0.062 ± 0.024	0.172
Chirp 3	0.981 ± 0.007	0.055 ± 0.025	0.424
Saaras v3	1.007 ± 0.004	0.074 ± 0.028	0.137
GPT-4o-transcr.	1.012 ± 0.004	0.127 ± 0.033	0.159
Nova-3	0.826 ± 0.015	0.104 ± 0.031	0.127

We omit worked examples because this document cannot render Devanagari alongside Romanized Hindi. Transliterating the model output would make the displayed text disagree with the WER computed on the original Devanagari.

F Public evaluation results

Every evaluation described in this paper was run using Calibrate [5], an open-source framework for evaluating AI agents. The results of each evaluation, with every test case, all the audio samples, the output of every model, and every judge’s verdict along with its reasoning chain, are publicly available at the links in Table 5.

Table 5: Links to the public evaluation results described in the paper.

Benchmark	Page
Speech-to-text	calibrate.artpark.ai/public/stt/9d25da9f-2727-42ee-931d-422858017383
Extraction, reference transcripts	calibrate.artpark.ai/public/benchmark/d77d790f-9c09-4a18-b4bf-2abadd84a9e8
Extraction, Scribe v2 transcripts	calibrate.artpark.ai/public/benchmark/4431ba9d-0ca2-457b-b924-e194d5414b1c
Response, reference transcripts	calibrate.artpark.ai/public/benchmark/5e974652-6568-49c3-875f-61b7ed253589
Response, Scribe v2 transcripts	calibrate.artpark.ai/public/benchmark/a508373f-1ee9-486c-90e3-214abd567d8f

G Model selection

Among the models that satisfy the deployment constraints, we min-max normalize accuracy a_i , latency ℓ_i and cost c_i within the Pareto frontier, reversing the scales of latency and cost so that higher is always better:

$$\tilde{a}_i = \frac{a_i - a_{\min}}{a_{\max} - a_{\min}}, \quad \tilde{\ell}_i = \frac{\ell_{\max} - \ell_i}{\ell_{\max} - \ell_{\min}}, \quad \tilde{c}_i = \frac{c_{\max} - c_i}{c_{\max} - c_{\min}}.$$

We then rank the models by weighted-sum scalarization [10], $U_i = w_a \tilde{a}_i + w_\ell \tilde{\ell}_i + w_c \tilde{c}_i$ with $w_a + w_\ell + w_c = 1$, and select $i^* = \arg \max_i U_i$. After applying the minimum accuracy and latency requirements, we set $w_\ell = 0.1$ and sweep w_a from 0.5 to 0.9, with $w_c = 1 - w_a - w_\ell$, to make the remaining accuracy-cost trade-off explicit. For reference, under equal weights Mistral Medium 3.5 ranks highest for EXTRACT ($U = 0.81$), ahead of GPT-4.1 ($U = 0.78$), because it pairs the lowest latency with 91.84% form-completion accuracy, while Claude Sonnet 4.6, the most accurate candidate, scores $U = 0.51$ for being slower and more expensive. Only two models lead across that range: Gemini 3.5 Flash for $0.50 \leq w_a \leq 0.61$ and Claude Sonnet 4.6 for $0.62 \leq w_a \leq 0.90$. For REPLY, Pareto filtering leaves GPT-5.4-mini and Gemini 3 Flash, and GPT-5.4-mini is optimal throughout the same sweep with a consistent $U = 0.9$.

H Ethics statement

No real user data was used to build the dataset. Paid native Hindi-speaking annotators recorded the scripts and explicitly consented to public release of their recordings. The scripts contain no real caller data or caller PII; the recordings contain the consenting annotators’ voices. Formal IRB review was not required under the applicable institutional policy and was not obtained. The agent is intended to widen access to care, but incorrect data capture could instead delay or deny it. Deployment at scale therefore requires safeguards that let callers confirm or correct captured information.

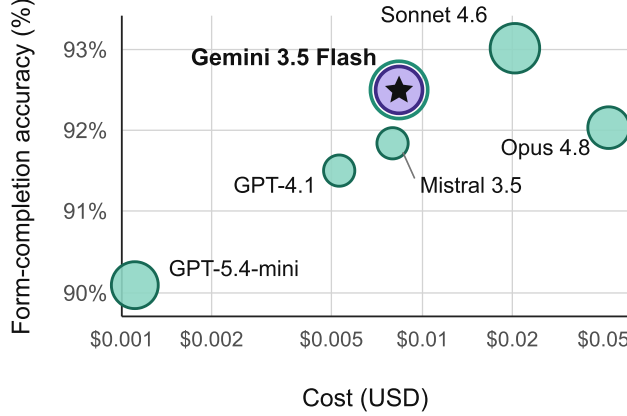


Figure 3: Cost–accuracy–latency trade-off across the six EXTRACT candidates that satisfy the deployment constraints. The logarithmic x-axis is cost per turn, the y-axis is form-completion accuracy on Scribe v2 transcripts, and marker area encodes p95 latency per turn (larger is slower). All six are Pareto-optimal across the three objectives, so no candidate can be discarded without a weighting. The model selected for deployment is starred. The per-turn cost and latency of every model are in Table 7.

I Full benchmark results

The temperature is set to 0 for non-reasoning models. The reasoning models use “medium” reasoning effort for EXTRACT and “low” for REPLY. Table 6 gives the turn-level extraction accuracy and Table 7 the end-to-end form-completion accuracy, each under perfect transcription (Reference) and under the transcripts produced by each STT model. Table 8 gives the response accuracy. Two judges are not pinned to a fixed model version: the binary judge for open-ended extraction fields runs on GPT-5.4-mini, and the published LLM-WER implementation [13] does not fix its judge model.

Table 6: Comparison of LLMs for turn-level extraction accuracy (%).

Model	Reference	Saaras v3	Scribe v2	Nova-3
GPT-5.5	99.79 ± 0.99	95.53 ± 1.03	98.62 ± 0.64	95.16 ± 1.07
Gemini 3.5 Flash	99.36 ± 1.22	95.69 ± 1.01	98.94 ± 0.58	96.38 ± 0.94
Claude Opus 4.8	99.15 ± 1.32	96.22 ± 0.96	98.40 ± 0.67	96.01 ± 0.98
Claude Sonnet 4.6	99.15 ± 1.32	96.01 ± 0.98	98.94 ± 0.58	95.96 ± 0.99
GLM-5.1	98.72 ± 1.48	94.52 ± 1.12	63.40 ± 2.20	95.43 ± 1.05
Gemini Pro	97.23 ± 1.90	94.26 ± 1.15	97.29 ± 0.84	94.63 ± 1.12
Gemini 2.5 Flash	96.17 ± 2.14	95.27 ± 1.06	98.03 ± 0.73	95.43 ± 1.05
Gemini 3 Flash	95.96 ± 2.19	84.79 ± 1.70	88.83 ± 1.50	86.28 ± 1.63
GPT-5.4-mini	92.34 ± 2.76	90.05 ± 1.43	96.76 ± 0.91	91.54 ± 1.34
Mistral Medium 3.5	89.36 ± 3.11	87.82 ± 1.56	92.34 ± 1.29	87.77 ± 1.56
GPT-4.1	85.11 ± 3.51	75.53 ± 1.99	78.94 ± 1.91	79.84 ± 1.87

Table 7: Comparison of LLMs for form-completion accuracy (%), with p95 latency and cost per extraction turn. The cost and latency are measured on the Reference run.

Model	Reference	Saaras v3	Scribe v2	Nova-3	p95 latency (s)	Cost (\$/turn)
Gemini 3 Flash	100.00 $\pm$ 0.00	85.24 $\pm$ 0.77	89.48 $\pm$ 0.60	83.92 $\pm$ 1.01	4.77	0.0028
Gemini 3.5 Flash	100.00 $\pm$ 0.00	90.74 $\pm$ 0.61	92.50 $\pm$ 0.53	87.09 $\pm$ 0.76	4.09	0.0084
GPT-5.5	99.95 $\pm$ 0.09	89.97 $\pm$ 0.73	89.37 $\pm$ 0.64	84.73 $\pm$ 0.80	2.97	0.0076
Claude Sonnet 4.6	99.66 $\pm$ 0.23	91.55 $\pm$ 0.71	93.01 $\pm$ 0.50	84.54 $\pm$ 0.77	4.78	0.0204
GLM-5.1	99.56 $\pm$ 0.27	90.92 $\pm$ 0.75	58.66 $\pm$ 1.46	86.09 $\pm$ 0.75	7.34	0.0022
Claude Opus 4.8	99.41 $\pm$ 0.31	92.22 $\pm$ 0.61	92.03 $\pm$ 0.56	85.83 $\pm$ 0.88	3.36	0.0417
Gemini 2.5 Flash	99.06 $\pm$ 0.38	91.76 $\pm$ 0.70	92.39 $\pm$ 0.52	85.50 $\pm$ 0.73	13.95	0.0012
Gemini Pro	98.90 $\pm$ 0.42	92.55 $\pm$ 0.52	93.47 $\pm$ 0.45	86.53 $\pm$ 0.70	8.52	0.0127
GPT-5.4-mini	98.35 $\pm$ 0.48	90.68 $\pm$ 0.75	90.10 $\pm$ 0.55	84.40 $\pm$ 0.79	4.01	0.0011
GPT-4.1	95.99 $\pm$ 0.70	88.82 $\pm$ 0.74	91.50 $\pm$ 0.53	86.33 $\pm$ 0.74	1.85	0.0053
Mistral Medium 3.5	92.83 $\pm$ 0.68	90.39 $\pm$ 0.68	91.84 $\pm$ 0.49	83.37 $\pm$ 0.76	1.75	0.0080

Table 8: Comparison of LLMs for response quality (%), p95 latency and cost per turn, with Gemini 3.5 Flash as the EXTRACT LLM. The cost and latency are measured on the Reference run.

Model	Reference	Scribe v2	p95 latency (s)	Cost (\$/turn)
Claude Sonnet 4.6	100.00 $\pm$ 0.00	97.77 $\pm$ 0.75	5.15 $\pm$ 0.39	0.0146
GPT-4.1	100.00 $\pm$ 0.00	96.60 $\pm$ 0.91	3.65 $\pm$ 0.66	0.0025
GPT-5.4-mini	98.09 $\pm$ 1.28	96.70 $\pm$ 0.85	3.36 $\pm$ 0.52	0.0008
Gemini 3 Flash	97.23 $\pm$ 1.49	96.17 $\pm$ 0.96	2.66 $\pm$ 0.24	0.0017
Gemini 3.5 Flash	97.02 $\pm$ 1.28	94.73 $\pm$ 1.11	3.50 $\pm$ 1.32	0.0052