# INDIC-22 — MULTILINGUAL RAG RELIABILITY & EVALUATION PLATFORM

**An Enterprise AI Quality Engineering System for Grounded, Culturally Aware and Trustworthy Indian-Language AI**

**MISSION: GET INTO RELIANCE INTELLIGENCE — An Independent AI Strategy & Innovation Challenge**

| Document field | Details |
|---|---|
| Proposal ID | RI-03 |
| Proposed ecosystem alignment | Reliance Intelligence — Multilingual AI and Enterprise AI |
| Solution category | AI Evaluation, RAG, LLMOps, Multilingual AI, Responsible AI |
| Primary users | Enterprise AI teams, AI platform engineers, model evaluators and AI governance teams |
| Initial language scope | Hindi, English, Hinglish, Tamil and Telugu |
| Long-term language ambition | Evaluation coverage across 22 Indian languages |
| Research focus | Multilingual groundedness, retrieval reliability, evaluator calibration and continuous AI quality monitoring |
| Document status | Independent R&D and product proposal |
| Date | 14 November 2026 |

---

## 1. Executive Summary

Retrieval-Augmented Generation (RAG) has become an important architecture for enterprise AI applications.

It allows language models to retrieve information from organisational documents, knowledge bases and other approved sources before generating responses.

However, deploying a RAG system does not automatically make it reliable.

A system can retrieve irrelevant documents, misunderstand a question expressed in a regional language, combine information incorrectly, produce unsupported claims or cite evidence that does not actually support its answer.

These failures become especially important in multilingual environments.

Varkala Sai Vignesh

A system may perform well in English while producing inconsistent answers in Hindi, Hinglish, Tamil or Telugu. It may retrieve English documents for a regional-language query but fail to reason correctly across the language boundary.

INDIC-22 proposes an enterprise AI evaluation platform that measures these failures before and after deployment.

The platform would connect to RAG applications, execute controlled multilingual test suites, evaluate retrieval and generation independently, inspect evidence supporting individual claims, compare automatic evaluators against human assessments and monitor quality regressions across model or knowledge-base changes.

### Core product capabilities

1. Multilingual RAG benchmarking.
2. Cross-lingual retrieval and reasoning evaluation.
3. Retrieval relevance and ranking diagnostics.
4. Claim-level groundedness verification.
5. Hallucination and unsupported-answer detection.
6. Citation accuracy and evidence traceability.
7. Cultural and contextual evaluation.
8. LLM-as-a-judge calibration.
9. Automated model and prompt comparison.
10. Continuous RAG regression monitoring.
11. Enterprise evaluation dashboards.
12. Explainable failure analysis.

### Central product principle

An enterprise AI system should not be considered reliable merely because its answers sound convincing.

It should demonstrate that it retrieves relevant evidence, reasons correctly from that evidence, responds appropriately in the requested language and acknowledges when the available information is insufficient.

---

# 2. Verified Strategic Context

## 2.1 Reliance Intelligence

Reliance Industries' FY2025–26 public corporate reporting describes Reliance Intelligence as building trusted, affordable and multilingual AI services designed for accessibility across 22 Indian languages.

The same reporting describes a broader ecosystem of consumer, enterprise and government AI applications.

Varkala Sai Vignesh

Reliance Enterprise Intelligence Limited (REIL), established as a joint venture between Reliance Intelligence and Meta, is positioned to develop enterprise AI solutions for Indian businesses.

These public priorities create a credible strategic context for an independent multilingual AI quality and reliability proposal.

The proposal does not assume that Reliance has an internal evaluation platform with the architecture described here.

It also does not claim that INDIC-22 is an existing Reliance product, internal requirement or approved development initiative.

### 2.2 Why an evaluation layer matters

As organisations deploy AI applications across different languages and business functions, they need to answer questions such as:

- Does the system retrieve the correct information?
- Does it understand a question written in a regional language?
- Can it reason over documents written in a different language?
- Are generated statements supported by retrieved evidence?
- Are citations accurate?
- Does the system refuse to answer when evidence is insufficient?
- Does a model update reduce performance in a previously supported language?

INDIC-22 proposes a reusable evaluation layer for answering these questions.

## 3. Existing Research and Competitive Landscape

A major requirement of this proposal is to recognise existing research and avoid overstating novelty.

### 3.1 IndicRAGSuite

The 2025 research paper *IndicRAGSuite: Large-Scale Datasets and a Benchmark for Indian Language RAG Systems* introduces IndicMSMarco for multilingual retrieval and response-generation evaluation across 13 Indian languages.

The associated work also describes training resources based on 19 Indian-language Wikipedias.

**Implication:** Building another basic Indian-language RAG benchmark would not, by itself, be a sufficient research contribution.

INDIC-22 should instead investigate enterprise evaluation workflows, claim-level diagnostics, evaluator reliability and continuous monitoring.

### 3.2 MEMERAG

Vaikala Sai Vignesh

The 2025 ACL paper *MEMERAG: A Multilingual End-to-End Meta-Evaluation Benchmark for Retrieval Augmented Generation* investigates whether automatic evaluators can reliably assess multilingual RAG outputs against human judgements.

Its released research resources include English, German, Spanish, French and Hindi evaluation data.

**Implication:** INDIC-22 must explicitly evaluate the reliability of its evaluators rather than assuming that an LLM judge is correct.

## 3.3 XRAG

The 2025 EMNLP Findings paper *XRAG: Cross-lingual Retrieval-Augmented Generation* studies situations where user queries and retrieved documents use different languages.

It identifies challenges involving response-language correctness and reasoning across retrieved languages.

**Implication:** Cross-lingual reasoning must be treated as a separate test category, not merely another translation test.

## 3.4 BordIRLines

The 2025 ACL Findings paper introduces a multilingual benchmark for RAG in culturally sensitive settings across 49 languages.

It examines how retrieved documents and language selection affect the consistency and robustness of generated responses.

**Implication:** Cultural-context evaluation should be grounded in carefully selected use cases and expert-reviewed criteria, not subjective assumptions about entire language communities.

## 3.5 RAGAS and established evaluation frameworks

RAGAS already provides metrics covering faithfulness, context precision, context recall, response relevance and other evaluation dimensions.

These capabilities should be integrated or used as baselines where appropriate.

INDIC-22 should not claim to have invented established RAG metrics.

## 3.6 Differentiation summary

| Existing research or tool | Existing contribution | INDIC-22 research direction |
|---|---|---|
| IndicRAGSuite | Indian-language RAG datasets and benchmark | Enterprise evaluation and failure diagnostics |
| MEMERAG | Multilingual evaluator meta-evaluation | Calibrated evaluator workflow and operational reporting |

Varkala Sai Vignesh

| | | |
|---|---|---|
| XRAG | Cross-lingual RAG benchmark | Enterprise cross-language regression testing |
| BordIRLines | Culturally sensitive cross-lingual RAG evaluation | Domain-specific contextual risk assessment |
| RAGAS | General RAG evaluation metrics | Language-stratified evaluation and evidence-level diagnostics |

The proposed differentiation is the integration of these capabilities into a repeatable enterprise evaluation and monitoring product.

---

# 4. Problem Definition

## 4.1 English-centric evaluation

Many RAG implementations are initially tested using English questions and English reference documents.

This leaves important questions unanswered:

- Will the same system understand Hindi questions?
- Can it process Hinglish written in Latin script?
- Can it retrieve Telugu documents for a Telugu question?
- Can it use English technical documentation to answer a Hindi question?
- Does its answer remain accurate after translating the retrieved information?

## 4.2 Retrieval failures

A system may generate a plausible response using incomplete or irrelevant context.

For example, a question about an enterprise reimbursement policy may retrieve an outdated policy document instead of the latest version.

A generation-only evaluation may not reveal the underlying retrieval failure.

## 4.3 Grounding failures

A response may contain several statements, of which only some are supported by the retrieved evidence.

A single overall faithfulness score may hide these individual failures.

## 4.4 Citation failures

An answer may include citations that:

- Point to irrelevant documents.

Vankala Sai Vignesh

- Reference the wrong passage.
- Support only part of a claim.
- Refer to outdated content.
- Do not exist in the retrieved context.

### 4.5 Evaluator reliability

LLM-as-a-judge systems can themselves be inconsistent.

A judge may prefer fluent answers, fail to identify unsupported claims or produce different scores for semantically equivalent responses in different languages.

An enterprise evaluation product must test the evaluator as well as the RAG system.

# 5. Product Vision and Objectives

Build a model-agnostic enterprise AI quality platform that continuously evaluates retrieval, generation, grounding, multilingual reasoning and response safety across supported Indian-language workflows.

### Product objectives

1. Measure RAG performance independently across supported languages.
2. Identify failures at retrieval, reasoning and response levels.
3. Evaluate claim-level evidence support.
4. Detect unsupported answers and citation errors.
5. Assess cross-lingual knowledge retrieval.
6. Calibrate automated evaluators against expert human judgement.
7. Detect quality regressions after system updates.
8. Generate actionable engineering recommendations.
9. Provide reproducible evaluation reports.
10. Support enterprise governance and auditability.

# 6. Proposed Product Features

### Feature 1 — Multilingual Benchmark Manager

<text color="secondary" size="xs">CORE FEATURE</text>

A central workspace for creating, importing and managing RAG test suites.

Capabilities:

- Language-specific test collections.
- Domain-specific test collections.
- Native-language questions.
- Code-mixed questions.

Varkala Sai Vignesh

- Cross-lingual questions.
- Reference answers.
- Relevant-document annotations.
- Expected citation references.
- Negative and unanswerable queries.
- Dataset versioning.

Initial language scope:

- Hindi.
- English.
- Hinglish.

The next evaluation phase should introduce Tamil and Telugu.

Expansion towards 22 languages should depend on available native-language reviewers, dataset quality and validated evaluation procedures.

## Feature 2 — Retrieval Quality Analyzer

Measures whether the RAG system retrieves the correct evidence.

Capabilities:

- Recall@K.
- Precision@K.
- Mean Reciprocal Rank.
- Normalized Discounted Cumulative Gain.
- Context relevance.
- Duplicate retrieval detection.
- Retrieval diversity.
- Language mismatch detection.
- Outdated-document retrieval detection.

The dashboard should distinguish retrieval failures from generation failures.

## Feature 3 — Cross-Lingual RAG Evaluator

Tests whether systems can answer questions when the user language differs from the retrieved document language.

Example:

Question: "कर्मचारी की छुट्टी की नीति क्या है?"

Retrieved evidence: English-language HR policy.

Expected response: Hindi explanation grounded in the English policy.

Evaluation dimensions:

Varkala Sai Vignesh

- Query understanding.
- Cross-lingual retrieval relevance.
- Correct interpretation of retrieved evidence.
- Response language correctness.
- Preservation of important technical terms.
- Citation accuracy.
- Contradiction handling.

## Feature 4 — Claim-Level Grounding Engine

The system decomposes generated responses into atomic factual claims.

Each claim is compared against retrieved evidence.

Illustrative output:

| Generated claim | Evidence status |
|---|---|
| Employees receive 20 annual leave days | Supported |
| Leave approval is mandatory | Supported |
| Unused leave always expires | Not supported |

The engine should identify the exact supporting or contradicting passage.

A claim that is not supported by the supplied context should not automatically be labelled false. It should be classified as unsupported unless additional authoritative evidence establishes otherwise.

## Feature 5 — Hallucination and Unsupported-Answer Detection

Evaluate:

- Unsupported claims.
- Contradictions with source documents.
- Fabricated numbers.
- Incorrect dates.
- Unsupported policy statements.
- Missing qualifications.
- Overconfident responses.
- Answers generated despite insufficient evidence.

The platform should distinguish factual incorrectness from lack of evidence.

## Feature 6 — Citation and Provenance Verification

Capabilities:

- Citation existence checks.
- Passage-level support verification.
- Source-document identification.
- Page and section references.

Vaikala Sai Vignesh

- Evidence coverage.
- Source version validation.
- Broken citation detection.

Every evaluation result should be traceable to the documents used in that test run.

## Feature 7 — Cultural and Contextual Grounding Evaluation

This feature assesses whether a response preserves relevant meaning and context.

Potential test categories:

- Local terminology.
- Region-specific institutional references.
- Formal versus informal address.
- Ambiguous expressions.
- Code-mixed language.
- Culturally sensitive subject matter.
- Translation-induced meaning changes.

Cultural evaluation must be based on defined rubrics and qualified reviewers.

The system should not use stereotypes or assume that every speaker of a language shares the same cultural preferences.

## Feature 8 — LLM-as-a-Judge Calibration

The platform compares automated evaluator decisions against human annotations.

Capabilities:

- Multiple judge-model comparison.
- Evaluator agreement analysis.
- Human-review workflows.
- Prompt-version comparison.
- Language-specific calibration.
- False-positive and false-negative analysis.
- Confidence estimation.
- Judge drift monitoring.

The platform must never treat an LLM judge's score as unquestionable ground truth.

## Feature 9 — Model and Pipeline Comparison

Compare different RAG configurations:

- Embedding models.
- Retrieval algorithms.
- Rerankers.

Varkala Sai Vignesh

- Chunking strategies.
- Language models.
- System prompts.
- Context windows.
- Query rewriting methods.

The system identifies which configuration performs better for each language and evaluation dimension.

## Feature 10 — Continuous Regression Monitoring

Whenever a model, prompt, embedding model, document corpus or retrieval configuration changes, the system can execute relevant evaluation suites again.

It identifies:

- Performance regressions.
- Language-specific degradation.
- New citation failures.
- Increased hallucination rates.
- Retrieval instability.
- Increased latency.
- Increased inference cost.

## Feature 11 — Enterprise Evaluation Dashboard

A unified interface containing:

- Overall evaluation results.
- Language-wise performance.
- Retrieval metrics.
- Grounding metrics.
- Hallucination indicators.
- Citation accuracy.
- Evaluator agreement.
- Cost and latency.
- Regression history.
- Failed test examples.

## Feature 12 — Explainable Failure Reports

Instead of reporting only a score, INDIC-22 should explain why a test failed.

Example:

"Telugu query failed because the retriever returned English documents with low semantic relevance. The generator then produced an answer containing two claims not supported by the retrieved passages."

Venkata Sai Vignesh

This converts evaluation into actionable engineering feedback.

---

# 7. End-to-End Product Workflow

## 7.1 Evaluation setup

1. Enterprise connects a RAG application through a supported API adapter.
2. The user defines the application configuration.
3. An evaluation dataset is selected.
4. Language and domain coverage are configured.
5. Baseline models and metrics are selected.
6. The platform validates dataset and configuration compatibility.

## 7.2 Test execution

1. Test questions are submitted to the RAG application.
2. Query, retrieved passages and generated answers are captured.
3. The system stores trace metadata.
4. Retrieval evaluation is executed.
5. Generation evaluation is executed.
6. Claim-level grounding is evaluated.
7. Citation checks are performed.
8. Language and contextual quality are evaluated.
9. Results are aggregated by language and test category.

## 7.3 Human review

1. Low-confidence evaluations are flagged.
2. Human reviewers inspect the question, answer and evidence.
3. Reviewers assign labels using a defined rubric.
4. Disagreements are recorded.
5. Adjudication is performed when required.
6. Human decisions become part of the evaluator calibration dataset, subject to appropriate data permissions.

## 7.4 Final reporting

The platform produces:

- Overall quality report.
- Language-wise scorecard.
- Failure classification.
- Model comparison.
- Citation audit.
- Human-review summary.

Vamsala Sai Vignesh

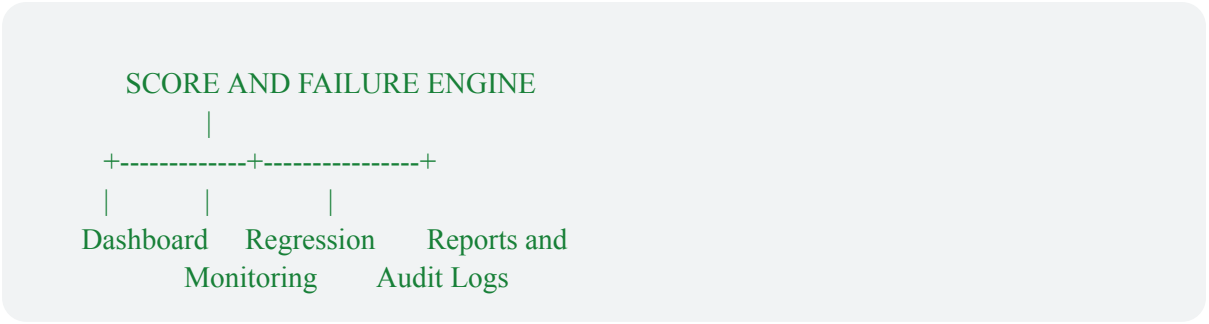
- Regression report.
- Recommended engineering actions.

---

# 8. Detailed Technical Architecture

## 8.1 High-level architecture






## 8.2 Layer 1 — Application and API gateway

Responsibilities:

- Authentication.
- Tenant identification.
- API request validation.
- Application registration.
- Rate limiting.
- Evaluation-job management.
- Adapter selection.
- Secure trace ingestion.

The platform should support REST-based integrations first.

Additional adapters can support different RAG frameworks as the product evolves.

## 8.3 Layer 2 — Dataset management

Responsibilities:

- Dataset ingestion.
- Schema validation.
- Language metadata.
- Document references.
- Annotation management.
- Dataset versioning.
- Test-split generation.
- Licensing metadata.

Every dataset should include provenance and permitted-use information.

## 8.4 Layer 3 — RAG application adapter

The adapter captures:

- Original query.
- Normalized query, if available.
- Retrieved document identifiers.

Varkala Sai Vignesh

- Retrieved text passages.
- Retrieval ranks.
- Similarity scores, if available.
- Generated response.
- Citations.
- Model identifier.
- Prompt version.
- Latency.
- Token usage.

The adapter should preserve the original trace instead of silently transforming it.

## 8.5 Layer 4 — Retrieval evaluation engine

This module evaluates retrieval independently of response generation.

Candidate metrics:

- Recall@K.
- Precision@K.
- MRR.
- nDCG@K.
- Context precision.
- Context recall.
- Retrieval diversity.
- Language coverage.

Ground-truth relevance labels are required for supervised retrieval metrics.

Where such labels are unavailable, the system must clearly identify the result as an automated estimate.

## 8.6 Layer 5 — Generation and grounding engine

The response is decomposed into atomic claims.

Each claim is evaluated against the retrieved evidence.

Candidate classifications:

- Supported.
- Partially supported.
- Contradicted.
- Unsupported.
- Unverifiable.

The system retains the evidence span, evaluator output and evaluation configuration for every claim.

## 8.7 Layer 6 — Multilingual evaluation engine

Varkala Sai Vignesh

This layer handles:

- Language identification.
- Script detection.
- Code-mixing identification.
- Transliteration-aware comparison.
- Response-language correctness.
- Semantic similarity.
- Cross-lingual reasoning tests.
- Language-specific rubric execution.

Translation can be used as a diagnostic aid, but translated text must not replace native-language human evaluation.

### 8.8 Layer 7 — Evaluator calibration engine

This layer assesses the evaluator itself.

It compares:

- LLM judge labels.
- Human reviewer labels.
- Multiple judge models.
- Different judge prompts.
- Language-specific performance.

It calculates agreement and error patterns.

### 8.9 Layer 8 — Analytics and reporting

Stores evaluation runs and produces:

- Quality trends.
- Language-wise breakdowns.
- Failure distributions.
- Model comparisons.
- Regression alerts.
- Cost and latency analysis.

All results should remain linked to the exact application and dataset versions used.

---

## 9. Core Data Architecture

### 9.1 Main entities

| Entity | Purpose |
|---|---|

Vaikala Sai Vignesh

| | |
|---|---|
| Organisation | Enterprise tenant |
| User | Platform user and reviewer |
| RAG Application | Registered application |
| Application Version | Model, prompt and pipeline configuration |
| Dataset | Evaluation collection |
| Dataset Version | Immutable test-data version |
| Test Case | Individual evaluation example |
| Document Reference | Relevant evidence source |
| Evaluation Run | Executed test batch |
| Query Trace | Application query and retrieval trace |
| Retrieved Chunk | Retrieved passage and rank |
| Generated Response | Captured model answer |
| Claim | Atomic response statement |
| Evaluation Result | Metric and judgement output |
| Human Review | Expert assessment |
| Regression Event | Detected performance change |
| Audit Event | User and system activity |

Vignesh

### 9.2 Illustrative test-case schema

```
None
{
 "test_id": "indic_0001",
 "language": "hi",
 "query_type": "cross_lingual",
 "query": "कर्मचारी की छुट्टी की नीति क्या है?",
 "expected_response_language": "hi",
 "relevant_document_ids": [
  "hr_policy_2026"
 ],
 "expected_evidence": [
  {
   "document_id": "hr_policy_2026",
   "section": "Annual Leave"
  }
 ],
 "answerability": "answerable",
 "domain": "human_resources",
 "annotation_status": "human_reviewed"
}
```

### 9.3 Illustrative evaluation result

```
None
{
 "run_id": "run_001",
 "test_id": "indic_0001",
 "retrieval": {
  "recall_at_5": 1.0,
  "relevant_document_found": true
 },
 "generation": {
  "response_language": "hi",
  "claim_count": 3,
  "supported_claims": 2,
  "unsupported_claims": 1
 },
 "citation": {
  "valid_citations": 1,
  "invalid_citations": 0
 },
 "human_review_required": true
}
```

These examples illustrate proposed schemas, not actual evaluation results.

---

# 10. Evaluation Methodology

INDIC-22 should evaluate RAG quality at multiple levels rather than compressing all quality dimensions into a single score.

## 10.1 Retrieval metrics

| Metric | What it measures |
|---|---|
| Recall@K | Whether relevant evidence appears in the top K results |
| Precision@K | Proportion of retrieved results that are relevant |
| MRR | Ranking of the first relevant result |
| nDCG@K | Quality of ranked retrieval |
| Context precision | Relevance of retrieved context |
| Context recall | Coverage of relevant evidence |

## 10.2 Generation metrics

| Metric | What it measures |
|---|---|
| Faithfulness | Whether answer claims are supported by context |
| Answer relevance | Whether the response addresses the query |
| Factual correctness | Agreement with reference information |

Varkala Sai

| | |
|---|---|
| Completeness | Whether important requested information is included |
| Abstention quality | Whether the system appropriately declines unsupported questions |

## 10.3 Evidence metrics

Proposed metrics:

- Claim support precision.
- Claim support recall.
- Unsupported claim rate.
- Contradiction rate.
- Citation precision.
- Citation coverage.
- Evidence-span accuracy.

For claim-level evaluation:

\frac{\text{Supported Claims}}
{\text{All Claims Evaluated}}
]

This should be reported alongside coverage and uncertainty, not as a standalone proof of factual correctness.

## 10.4 Multilingual metrics

- Response-language accuracy.
- Native-language semantic relevance.
- Cross-lingual retrieval recall.
- Code-mixed query robustness.
- Transliteration robustness.
- Language-specific answer correctness.
- Human-rated contextual appropriateness.

## 10.5 Evaluator metrics

- Agreement with expert reviewers.
- Cohen's kappa for appropriate two-rater categorical settings.
- Fleiss' kappa for suitable multi-rater settings.
- Krippendorff's alpha where annotation structure permits.
- Macro F1.
- False-positive rate.
- False-negative rate.
- Calibration error.
- Performance by language and task.

The selected agreement metric must match the annotation design and label distribution.

Varkala Sai Vignesh

# 11. Benchmark and Dataset Strategy

## 11.1 Existing public resources

INDIC-22 should build upon existing resources where licensing and intended use permit.

Potential research baselines include:

1. IndicRAGSuite / IndicMSMarco.
2. MEMERAG.
3. XRAG.
4. BordIRLines.
5. MIRACL.
6. IndicGenBench.

These resources should be used for comparison, not indiscriminately merged into a single dataset.

## 11.2 Proposed additional dataset

The independent R&D contribution should be an enterprise-focused evaluation collection covering scenarios not adequately represented in the selected baselines.

Proposed categories:

- Internal HR policy questions.
- Enterprise IT support.
- Customer service knowledge.
- Product documentation.
- Process and operations manuals.
- Multilingual policy retrieval.
- Cross-lingual evidence reasoning.
- Outdated document detection.
- Contradictory document handling.
- Unanswerable queries.
- Citation verification.
- Prompt-injection attempts in retrieved content.

The first version should use synthetic and public documents to avoid unnecessary exposure of confidential information.

## 11.3 Proposed initial dataset scale

| Component | Initial research target |
|---|---|
| Evaluation queries | 2,000–3,000 |
| Supported languages | 4 initial evaluation tracks |
| Cross-lingual scenarios | 300–500 |

Varkala Sai Vignesh

| | |
|---|---|
| Unanswerable queries | 300–500 |
| Adversarial cases | 200–300 |
| Human-reviewed claim annotations | 1,000–2,000 |
| Enterprise document domains | 3–5 |

These are proposed collection targets, not existing dataset statistics.

The project should prioritize annotation quality over simply increasing dataset size.

### 11.4 Annotation process

Each test case should be reviewed using a documented rubric.

Reviewers should assess:

- Query meaning.
- Relevant evidence.
- Expected answer.
- Acceptable response language.
- Supported and unsupported claims.
- Citation correctness.
- Whether abstention is appropriate.

A subset should receive independent double annotation.

Disagreements should be analysed and adjudicated rather than hidden by automatic majority voting.

### 11.5 Dataset rights

Before reusing external resources:

- Verify the licence of each dataset.
- Review commercial-use restrictions.
- Retain required attribution.
- Avoid using evaluation-only data for training.
- Record source and version metadata.
- Respect dataset redistribution terms.

For example, the MEMERAG repository describes its data as intended for evaluator model and prompt selection and states that it should not be used to train models.

This restriction must be respected.

---

## 12. Research Questions and Proposed Novelty

Varkala Sai Vignesh

## RQ1 — How reliably do RAG applications perform across Indian-language query conditions?

Measure whether performance varies by language, script, code-mixing and retrieval-language configuration.

## RQ2 — Can claim-level evidence evaluation identify failures hidden by aggregate faithfulness scores?

Compare overall response scoring with atomic claim decomposition and evidence-span verification.

## RQ3 — How well do LLM judges correlate with native-language expert reviewers?

Evaluate agreement by language, task type and claim category.

## RQ4 — Does cross-lingual retrieval improve or degrade enterprise response reliability?

Compare:

- Same-language query and documents.
- Regional-language query with English-only retrieval.
- Regional-language query with multilingual retrieval.
- Hybrid retrieval with language-aware reranking.

## RQ5 — Can continuous evaluation identify quality regressions after pipeline changes?

Measure whether versioned test suites identify changes in retrieval, grounding, response language and citation accuracy.

## RQ6 — Can a risk-based evaluation strategy reduce enterprise evaluation cost?

Study whether high-risk and low-confidence examples can receive more expensive human or model evaluation while simpler examples use cheaper deterministic checks.

## Proposed contribution

The research contribution is a reproducible enterprise evaluation methodology combining:

1. Language-stratified RAG testing.
2. Cross-lingual retrieval and reasoning analysis.
3. Claim-level evidence diagnostics.
4. Human-calibrated automated evaluators.
5. Version-aware regression testing.
6. Risk-based evaluation orchestration.

Varkala Sai Vignesh

This is a proposed integration and research direction. Its novelty must be established through further comparison with existing academic and commercial systems before publication claims are made.

---

# 13. Experimental Design

## 13.1 Baseline systems

Compare the proposed platform against:

- Standard English-centric RAG evaluation.
- Existing RAGAS metrics.
- IndicRAGSuite evaluation resources.
- MEMERAG evaluator comparison.
- XRAG cross-lingual test scenarios.
- Human expert assessment.

## 13.2 Experimental configurations

Test different combinations of:

| Component | Configurations |
|---|---|
| Embedding model | General multilingual and Indic-focused |
| Retrieval | Dense, sparse and hybrid |
| Reranking | None, multilingual reranker |
| Query processing | Original and normalized query |
| Retrieval corpus | Same-language, English-only and multilingual |
| Generation model | Multiple supported model families |
| Evaluation | Rule-based, LLM judge and human |
| Evidence analysis | Response-level and claim-level |

## 13.3 Controlled experiments

**Experiment A — Language parity**

Compare performance across the supported language tracks.

**Experiment B — Cross-lingual reasoning**

Test whether models can use evidence retrieved in a different language from the user query.

**Experiment C — Grounding reliability**

Compare aggregate faithfulness scores with claim-level evidence annotations.

**Experiment D — Judge calibration**

Varkala Sai Vignesh

Measure how well automated evaluators reproduce expert human judgements.

**Experiment E — Regression detection**

Introduce controlled changes to models, prompts and retrieval configurations and measure whether the platform detects quality changes.

**Experiment F — Cost and latency**

Measure evaluation cost and time under different evaluation strategies.

### 13.4 Proposed acceptance targets

These are initial research targets, not achieved results or guaranteed performance.

| Metric | Proposed target |
|---|---|
| Evaluation test reproducibility | ≥ 99% on deterministic test configurations |
| Retrieval trace capture completeness | ≥ 98% on supported adapters |
| Citation existence validation | 100% on deterministic citation checks |
| Claim-level human-review agreement | Target to be established through pilot annotation |
| Evaluator agreement with human labels | Target macro F1 ≥ 0.80 on a defined held-out set |
| Regression detection recall | ≥ 90% on deliberately introduced material regressions |
| Language-wise reporting coverage | 100% for configured test tracks |
| Evaluation run auditability | 100% of completed runs linked to configuration and dataset versions |

Important qualifications:

- A high evaluator agreement score does not establish that the evaluator is unbiased.
- Citation existence checks do not prove that a citation supports a claim.
- Regression detection targets must be measured against a pre-defined set of material regressions.
- Targets should be revised after pilot data and confidence intervals are available.

---

## 14. Evaluator Calibration and Human Review

This is one of the most important technical components.

### 14.1 Why LLM judges need validation

An automated evaluator may:

Varkala Sai Vignesh

- Prefer fluent but incorrect answers.
- Penalize valid regional-language expressions.
- Misinterpret code-mixed sentences.
- Overlook unsupported numbers.
- Miss contradictions.
- Treat semantic similarity as factual correctness.

## 14.2 Proposed evaluator architecture

```
None
Generated RAG Answer
        |
Claim Decomposition
        |
Evidence Retrieval
        |
    +---+---+
    |       |
 LLM Judge  Rule Checks
    |       |
    +---+---+
        |
 Confidence Assessment
        |
 Human Review Routing
        |
 Expert Annotation
        |
 Evaluator Calibration
        |
 Updated Evaluation Policy
```

## 14.3 Review routing

Send a result for human review when:

- Judge confidence is low.
- Multiple judges disagree.
- A critical claim is unsupported.
- A citation contradicts the answer.
- The language is outside validated coverage.
- The case involves a high-impact enterprise decision.

## 14.4 Human review safeguards

- Reviewers receive the original query and evidence.
- Labels follow documented definitions.

- Reviewers can mark a case unassessable.
- Disagreement is preserved.
- Adjudication decisions are recorded.
- Reviewer access is controlled.

Human review data should not automatically be used for model training.

---

# 15. Security, Privacy and Governance

INDIC-22 may process sensitive enterprise documents and AI application traces.

The platform must be designed to avoid turning an evaluation tool into another data exposure risk.

## 15.1 Security controls

- Tenant isolation.
- Role-based access control.
- Encryption in transit and at rest.
- API authentication.
- Secrets management.
- Audit logging.
- Retention controls.
- Secure dataset storage.
- Evaluation-job isolation.
- Rate limiting.

## 15.2 Sensitive data handling

- Redact personal information where possible.
- Avoid collecting unnecessary source documents.
- Support document-reference-based evaluation where feasible.
- Restrict access to raw prompts and retrieved passages.
- Define deletion and retention procedures.
- Separate customer data from shared evaluation datasets.

## 15.3 Prompt injection and hostile documents

Retrieved documents may contain malicious instructions.

Evaluation test suites should include cases where retrieved content attempts to:

- Override system instructions.
- Request confidential information.
- Manipulate citations.
- Cause unsafe tool execution.
- Change the evaluation criteria.

Vadala Sai Vignesh

The evaluator must treat retrieved content as untrusted evidence, not executable instructions.

### 15.4 Responsible AI

The platform should report:

- Language-specific weaknesses.
- Uncertainty.
- Unsupported answers.
- Evaluation limitations.
- Human-review requirements.
- Known blind spots.

A single overall score should never conceal serious weaknesses in a particular language or domain.

---

## 16. Proposed Technology Stack

| Layer | Candidate technologies |
|---|---|
| Frontend | React / Next.js |
| Backend | Python, FastAPI |
| Job orchestration | Celery or Temporal |
| Database | PostgreSQL |
| Vector storage | pgvector or a compatible vector database |
| Object storage | S3-compatible storage |
| Dataset management | Versioned Parquet / JSONL |
| Evaluation orchestration | Custom evaluation service |
| Existing evaluation baselines | RAGAS and compatible libraries |
| Language processing | Indic NLP and multilingual model toolkits |
| Model integration | Model-agnostic REST adapters |
| Analytics | SQL and analytical dashboards |
| Observability | OpenTelemetry and centralized logging |
| Deployment | Docker and Kubernetes where scale requires |

These are candidate implementation technologies, not confirmed Reliance technology selections.

The architecture should allow evaluation modules to be replaced independently.

---

## 17. Enterprise Integration Architecture

INDIC-22 should operate as an evaluation layer rather than requiring enterprises to rebuild their RAG applications.

Varkala Sai Vignesh

### Integration methods

1. REST API.
2. SDK integration.
3. Batch evaluation.
4. Scheduled regression jobs.
5. CI/CD evaluation hooks.
6. Supported RAG framework adapters.

### Example API workflow

```
None
POST /api/v1/applications
POST /api/v1/datasets
POST /api/v1/evaluation-runs
GET  /api/v1/evaluation-runs/{run_id}
GET  /api/v1/reports/{report_id}
POST /api/v1/reviews/{review_id}
```

These are proposed API routes for the prototype.

### CI/CD integration

A new model or prompt version can trigger a test run.

The deployment pipeline can then apply configured quality gates.

Example:

- Block deployment if citation integrity falls below a defined threshold.
- Flag deployment if Hindi groundedness regresses materially.
- Require human review if unsupported claims increase.
- Allow deployment only after critical test cases pass.

Quality gates must be configurable by business risk and should not rely on one aggregate score.

---

# 18. Business Model and Strategic Value

## 18.1 Potential customers

- Large enterprises deploying RAG applications.
- Financial services organisations.
- Telecommunications providers.
- Retail and commerce companies.
- Government-facing AI service providers.

Varkala Sai Vig

- Enterprise software vendors.
- AI platform engineering teams.

### 18.2 Potential commercial models

| Model | Description |
|---|---|
| Usage-based | Pricing by evaluation volume |
| Enterprise subscription | Platform access and reporting |
| Private deployment | Customer-controlled infrastructure |
| Evaluation services | Custom benchmarks and human review |
| Continuous monitoring | Recurring quality and regression monitoring |

These are commercial hypotheses requiring market validation.

### 18.3 Business value

- Lower risk of deploying unreliable RAG applications.
- Faster model and pipeline comparison.
- Reduced manual evaluation effort.
- Better multilingual application quality.
- More transparent AI governance.
- Reproducible enterprise AI testing.

### 18.4 Strategic value to Reliance Intelligence

The product could provide reusable quality infrastructure across multilingual AI applications and enterprise deployments.

It could support the broader objective of making AI systems trusted and usable across Indian languages.

However, integration into Reliance platforms would require internal product review, technical compatibility assessment and security approval.

---

# 19. Implementation Roadmap

### Phase 1 — Research and benchmark design

#### Weeks 1–2

- Review existing RAG benchmarks.
- Define evaluation taxonomy.
- Select initial language tracks.
- Design annotation rubrics.
- Identify dataset licences.
- Build synthetic enterprise document collections.

Varkala Sai Vignesh

Deliverable: benchmark specification and dataset design.

## Phase 2 — Evaluation engine

**Weeks 3–4**

- Build application adapters.
- Implement evaluation trace capture.
- Integrate retrieval metrics.
- Implement generation metrics.
- Add citation checks.
- Create evaluation result storage.

Deliverable: working RAG evaluation backend.

## Phase 3 — Multilingual and cross-lingual evaluation

**Weeks 5–6**

- Implement language metadata.
- Add Hindi and Hinglish evaluation.
- Introduce cross-lingual test cases.
- Add language-wise reporting.
- Implement code-mixed query testing.

Deliverable: multilingual evaluation prototype.

## Phase 4 — Claim-level grounding and evaluator calibration

**Weeks 7–8**

- Implement claim decomposition.
- Add evidence-span verification.
- Integrate LLM judges.
- Build human review interface.
- Measure evaluator agreement.

Deliverable: evidence-level evaluation module.

## Phase 5 — Dashboard and regression monitoring

**Weeks 9–10**

- Build evaluation dashboard.
- Add model comparison.
- Implement version tracking.
- Introduce regression tests.
- Add report generation.
- Run controlled experiments.

Vantala Sai Vignesh

Deliverable: complete prototype and evaluation report.

### Phase 6 — Research analysis

**Weeks 11–12**

- Perform error analysis.
- Compare baselines.
- Evaluate language disparities.
- Analyse evaluator failures.
- Document reproducibility.
- Prepare research findings.

Deliverable: final R&D report and demonstration.

This is an independent prototype schedule, not a Reliance delivery commitment.

---

# 20. Minimum Viable Product

The first version should prove three hypotheses:

1. A multilingual RAG system can be evaluated beyond English-centric metrics.
2. Claim-level evidence diagnostics reveal failures hidden by aggregate scores.
3. Automated evaluators can be assessed against human judgement rather than assumed reliable.

### MVP features

- Application registration.
- Dataset upload.
- Hindi, English and Hinglish test suites.
- RAG trace capture.
- Retrieval evaluation.
- Faithfulness and relevance baselines.
- Citation verification.
- Claim-level grounding.
- Human review.
- Model comparison dashboard.
- Versioned evaluation reports.

### Deferred features

- Full 22-language coverage.
- Production-scale continuous monitoring.
- Advanced cultural evaluation across all domains.
- Automated remediation.
- Complex multi-agent evaluation.

Vaikala Sai Vignesh

- Large-scale enterprise integrations.

This keeps the initial research achievable and measurable.

---

# 21. Risk Register

| Risk | Potential impact | Mitigation |
|---|---|---|
| Existing benchmark overlap | Weak novelty | Position as enterprise evaluation and monitoring |
| Inadequate native-language annotations | Invalid results | Qualified reviewers and double annotation |
| LLM judge bias | Incorrect quality scores | Human calibration and multiple judges |
| Translation-based evaluation | Cultural and semantic loss | Native-language test cases |
| Dataset licensing restrictions | Legal and research limitations | Licence review and provenance tracking |
| Unstable evaluation metrics | Inconsistent comparisons | Versioned metrics and reproducible runs |
| Aggregate scores hiding failures | Misleading enterprise decisions | Language and dimension-specific reporting |
| High evaluation cost | Poor scalability | Risk-based evaluation and caching |
| Confidential enterprise data | Privacy exposure | Data minimization and private deployment |
| Metric gaming | False sense of quality | Hidden test sets and adversarial evaluation |

---

# 22. Expected R&D Deliverables

The independent project should produce:

1. Complete product requirements document.
2. Enterprise evaluation architecture.
3. Dataset and annotation specification.
4. Multilingual RAG evaluation prototype.
5. Cross-lingual evaluation test suite.
6. Retrieval and grounding metrics implementation.
7. Claim-level evidence verification module.
8. LLM-judge calibration module.
9. Human-review interface.
10. Regression monitoring dashboard.
11. Benchmark comparison report.
12. Error-analysis and reproducibility report.
13. Security and privacy design.

Varkala Sai Vignesh

14. Technical demonstration.

---

# 23. Strategic Relevance and Product Positioning

INDIC-22 is designed as a reusable AI quality engineering layer for multilingual RAG applications.

Its strongest strategic connection is the need for reliable multilingual AI across consumer, enterprise and government use cases.

It is particularly relevant to enterprise AI because the product evaluates systems against measurable evidence rather than relying on subjective impressions of answer quality.

The proposed architecture could support evaluation of:

- Enterprise knowledge assistants.
- Customer support RAG applications.
- Internal policy assistants.
- Multilingual document search.
- Government information assistants.
- Domain-specific knowledge applications.

The platform should remain independent of any single language model or RAG framework.

Its value comes from providing consistent evaluation across different systems.

---

# 24. Conclusion

Multilingual AI quality cannot be established by testing a few English prompts and assuming the results generalize to every language.

A system that performs well in English may fail to retrieve relevant regional-language information, reason across languages or provide evidence for its claims.

INDIC-22 proposes an enterprise platform that makes these weaknesses measurable.

By combining multilingual benchmarks, retrieval diagnostics, claim-level evidence verification, calibrated automated evaluators and continuous regression testing, the platform aims to turn AI reliability from a subjective expectation into an engineering discipline.

The central idea:

**Do not just measure whether an AI system produces an answer. Measure whether it retrieves the right evidence, understands it correctly and can justify what it says.**

---

Varkala Sai Vignesh

# 25. Verification Notes and Research References

## A. Reliance corporate context


1. Reliance Industries Limited, Chairman's Statement, FY2025–26. Publicly describes Reliance Intelligence's ambition to build trusted, affordable multilingual AI services accessible across 22 Indian languages.
2. Reliance Industries Limited, Annual Report 2025–26, Digital Services Business Overview. Describes Reliance Intelligence's enterprise AI ecosystem and REIL's establishment as a joint venture with Meta.
3. Reliance Industries Limited, August 29, 2025. *Reliance and Meta Partner for Enterprise AI Solutions for India.* Official announcement of the enterprise AI joint venture.


## B. Core RAG evaluation research


4. Prasanjith, P., More, P. B., Kunchukuttan, A., and Dabre, R. (2025). *IndicRAGSuite: Large-Scale Datasets and a Benchmark for Indian Language RAG Systems.* arXiv:2506.01615.
5. Cruz Blandón, M. A., Talur, J., Charron, B., Liu, D., Mansour, S., and Federico, M. (2025). *MEMERAG: A Multilingual End-to-End Meta-Evaluation Benchmark for Retrieval Augmented Generation.* Proceedings of ACL 2025.
6. Liu, W., Trenous, S., Ribeiro, L. F. R., Byrne, B., and Hieber, F. (2025). *XRAG: Cross-lingual Retrieval-Augmented Generation.* Findings of EMNLP 2025.
7. Li, B. et al. (2025). *Multilingual Retrieval Augmented Generation for Culturally-Sensitive Tasks: A Benchmark for Cross-lingual Robustness.* Findings of ACL 2025. Introduces BordIRLines.
8. Singh, H. et al. (2024). *IndicGenBench: A Multilingual Benchmark to Evaluate Generation Capabilities of LLMs on Indic Languages.* Proceedings of ACL 2024.


## C. Evaluation frameworks and documentation


9. RAGAS documentation — Available metrics for RAG and agentic application evaluation.
10. Microsoft Learn — Evaluation metrics for generative AI and RAG applications.
11. NVIDIA NeMo documentation — Retrieval and generation evaluation metrics.


## D. Evidence classification

| Claim | Status |
|---|---|
| Reliance Intelligence publicly describes a 22-language AI ambition | Verified corporate statement |
| REIL is a Reliance Intelligence–Meta enterprise AI joint venture | Verified corporate statement |
| IndicRAGSuite already provides Indian-language RAG evaluation resources | Published research |
| MEMERAG evaluates multilingual RAG evaluators | Published research |
| XRAG studies cross-lingual RAG | Published research |
| INDIC-22 product features and architecture | Independent proposal |

Vaikala Sai Vignesh

| | |
|---|---|
| Proposed enterprise benchmark extensions | Research hypotheses |
| Proposed performance targets | Unvalidated engineering targets |
| Potential Reliance integration | Strategic hypothesis requiring internal review |

### E. Research limitations

- The proposal does not claim that multilingual RAG evaluation is an unexplored field.
- The proposed platform must be compared against existing tools and research before novelty or superiority claims are made.
- Four initial language tracks are proposed; coverage across 22 languages is a long-term ambition, not an MVP capability.
- Cultural and contextual quality cannot be reliably reduced to a single automated score.
- LLM judges require human calibration.
- External datasets must be used in accordance with their licences and intended-use restrictions.
- No internal Reliance systems, confidential documents or unpublished roadmap have been assumed.

---

## 26. Independence and Disclaimer

This document is an independently developed AI product and R&D proposal prepared as part of:

**MISSION: GET INTO RELIANCE INTELLIGENCE — An Independent AI Strategy & Innovation Challenge.**

It is intended to demonstrate independent research, enterprise AI product strategy, technical architecture and applied AI evaluation capabilities.

It is not an official Reliance Industries Limited, Reliance Intelligence Limited or Reliance Enterprise Intelligence Limited product or document.

No employment, internship, partnership, endorsement, internal access, project approval or commissioning is implied.

All proposed features, architecture, research methods, performance targets and commercial models require independent implementation and validation.

Varkala Sai Vignesh