

AI-Driven Conversational Assistant Using RAG for Government Schemes

Francis Shamili S¹, Aadhithya A², Abin V³, Akash M⁴, Gangadharan K⁵

¹Assistant Professor, Computer Science & Engineering, Dhanalakshmi Srinivasan Engineering College(A)

²³⁴⁵Computer Science & Engineering, Dhanalakshmi Srinivasan Engineering College(A)

Abstract - This research presents the development of an AI-Driven Conversational Assistant for accessing Indian government scheme information using Retrieval-Augmented Generation (RAG). Citizens often face difficulties identifying relevant welfare schemes due to unstructured and scattered information spread across multiple portals, PDF documents, and official circulars. Traditional keyword-based search systems fail to provide accurate, personalised responses to natural language queries. The proposed system integrates information retrieval techniques with Large Language Models (LLMs) to deliver context-aware and precise responses grounded in verified government documents. Government scheme data is collected, preprocessed, converted into dense vector embeddings, and stored in a FAISS vector database

Key Words: Retrieval-Augmented Generation (RAG), Large Language Models, Conversational AI, Government Schemes, Vector Database, Semantic Search, FAISS, Sentence Transformers, Natural Language Processing, Digital Inclusion.

1.INTRODUCTION

Access to government welfare schemes is a fundamental right of every citizen, yet millions of Indians remain unaware of the benefits they are entitled to. The Government of India launches hundreds of central and state schemes annually covering agriculture, education, health, employment, housing, and social welfare. Despite this, a large section of the population — particularly in rural and semi-urban areas — cannot identify schemes relevant to their situation. Scheme details are scattered across dozens of official portals, lengthy PDF circulars, press releases, and regional-language documents, making navigation practically impossible for an ordinary citizen without significant effort [1].

Traditional search engines rely on keyword matching and fail to understand semantic intent behind natural language queries. A farmer asking "What support can I get for buying seeds?" would not match PM-KISAN because the terminology differs. FAQ-based chatbots deployed by state governments are confined to predefined question-answer pairs and break down for paraphrased queries. Human-operated helplines cannot scale to serve a population exceeding 1.4 billion.

Recent advances in Large Language Models (LLMs) and Retrieval-Augmented Generation (RAG) offer a technically sound solution. RAG [1] augments LLM generation with a non-parametric retrieval step, grounding responses in a curated document corpus and eliminating hallucination — the

tendency of LLMs to fabricate plausible but incorrect facts — while preserving conversational fluency.

This paper presents the design, implementation, and evaluation of an AI-Driven Conversational Assistant applying the RAG framework to Indian government welfare schemes. The system enables citizens to query scheme information in plain language and receive structured, source-cited, factually accurate responses. Section II reviews related work; Section III describes the proposed system; Section IV presents the system architecture; Section V details the methodology; Section VI describes algorithms; Section VII presents results; Section VIII concludes.

2.RELATED WORK

Lewis et al. [1] proposed RAG as a general-purpose framework that augments sequence-to-sequence models with dense retrieval, significantly outperforming closed-book LLMs on knowledge-intensive tasks. Devlin et al. [2] introduced BERT, showing that bidirectional transformer pre-training produces contextual representations that remain the foundation of modern embedding pipelines. Reimers and Gurevych [3] extended BERT to fixed-length sentence embeddings via Siamese fine-tuning (Sentence-BERT), enabling efficient large-scale semantic search.

Johnson et al. [4] developed FAISS, a highly optimised library for billion-scale nearest-neighbour retrieval adopted as our indexing backend. Brown et al. [5] demonstrated GPT-3-scale few-shot generalisation; instruction-tuned variants power modern RAG pipelines. Touvron et al. [6] showed that open-weight LLaMA models match closed-source performance at lower inference cost, enabling local RAG deployment.

In the e-governance domain, the MyScheme portal [8] provides browsing-based scheme discovery but does not support natural language or conversational queries. Gao et al. [10] confirm that retrieval grounding is the most effective strategy for reducing hallucination in domain-specific LLM applications. Vaswani et al. [9] introduced the Transformer architecture underpinning every component of our pipeline.

3.PROPOSED SYSTEM

The proposed AI-Driven Conversational Assistant employs the RAG framework and operates in two phases: an offline document indexing pipeline and an online query-response pipeline.

A. Offline Indexing Pipeline

Government scheme documents are collected from official sources, cleaned, and segmented into 500-token chunks with 50-token overlap. Each chunk is encoded into a 384-dimensional dense vector embedding and stored in a FAISS index alongside metadata (scheme name, ministry, eligibility summary, source URL). This phase runs asynchronously whenever new documents are added, keeping the knowledge base current without system downtime.

B. Online Query-Response Pipeline

A citizen submits a natural language query through the chat interface. The query is embedded with the same sentence-transformer model. A cosine similarity search retrieves the top-5 chunks. A cross-encoder re-ranks these to select the top-3. The chunks are assembled into a structured prompt instructing the LLM to answer only from the provided context and to cite the source scheme. The response is post-processed to extract scheme name, eligibility criteria, income limit, required documents, application procedure, and official links.

C. Key Advantages

- RAG grounding eliminates LLM hallucination by constraining generation to verified government documents.
- Semantic vector search bridges vocabulary mismatches between citizen queries and official terminology.
- Multi-turn interface supports follow-up questions with full session context preserved.
- Modular design allows new scheme documents without retraining the language model.
- Source citations enable citizens to independently verify information at official portals.
- Lightweight FAISS CPU retrieval supports high concurrency on commodity hardware.

4.SYSTEM ARCHITECTURE

The system follows a five-layer architecture: (1) Data Ingestion, (2) Embedding and Indexing, (3) Retrieval, (4) Generation, and (5) Presentation. Fig. 1 below illustrates the complete end-to-end data flow from citizen query submission to structured response output.

The Data Ingestion layer scrapes scheme documents from myscheme.gov.in and ministry portals. The Embedding layer encodes chunks via all-MiniLM-L6-v2 and stores them in FAISS. The Retrieval layer performs similarity search followed by cross-encoder re-ranking. The Generation layer invokes the LLM with a context-constrained prompt. The Presentation layer is a React-based frontend backed by FastAPI.



Fig. 1. System Architecture of the AI Driven Conversational Assistant using RAG

A. Data Ingestion Layer

Python's BeautifulSoup and Scrapy handle web scraping; PyMuPDF parses PDF circulars. Content is cleaned to remove navigation menus, headers, and formatting artefacts. Documents are segmented using LangChain's RecursiveCharacterTextSplitter into 500-token chunks with 50-token overlap to preserve semantic coherence at chunk boundaries.

B. Embedding and Indexing Layer

Each chunk is encoded via all-MiniLM-L6-v2 (384-dimensional float32 output). Vectors are L2-normalised and stored in a FAISS IndexFlatIP index. Chunk metadata is stored in a parallel SQLite table keyed by FAISS index position, enabling O(1) metadata retrieval alongside search results for source attribution.

C. Retrieval Layer

The embedded user query is matched against the FAISS index via inner-product (cosine) similarity, returning the top-5 candidates. A cross-encoder (ms-marco-MiniLM-L-6-v2) scores each (query, chunk) pair jointly, producing a precise relevance signal. The top-3 re-ranked chunks are forwarded to the generation stage.

D. Generation Layer

Retrieved chunks and the user query are assembled into a structured prompt: "You are a helpful assistant for Indian government welfare schemes. Answer using only the provided context. Cite the scheme name and ministry." The prompt is submitted to LLaMA-3-8B-Instruct (local deployment) or GPT-3.5-turbo (API). Regex-based post-processing extracts structured response fields.

E. Presentation Layer

A React 18 single-page application maintains session state for multi-turn dialogue over REST. The FastAPI backend is containerised with Docker and deployable on AWS EC2 or GCP Compute Engine, with horizontal scaling behind a load balancer for high-concurrency scenarios.

5. METHODOLOGY

A. Data Collection and Preprocessing

A total of 318 government scheme descriptions were collected from 12 central ministries including Agriculture, Health and Family Welfare, Housing and Urban Affairs, Labour and Employment, Education, Women and Child Development, Finance, Rural Development, Power, Skill Development, MSME, and Social Justice. Sources include the MyScheme portal, individual ministry websites, and official PDF circulars. Scrapy was configured with polite crawl settings. Extracted text was Unicode-NFC-normalised, stripped of HTML artefacts, and segmented into 2,847 chunks of 500 tokens with 50-token overlap.

B. Embedding Generation and Indexing

Each chunk was encoded using all-MiniLM-L6-v2, producing a 384-dimensional vector. Vectors were L2-normalised and inserted into a FAISS IndexFlatIP index. Metadata (scheme name, ministry, beneficiary category, eligibility summary, source URL) was stored in SQLite with FAISS index position as primary key, enabling efficient source attribution at retrieval time.

C. Query Processing and Retrieval

User queries undergo Unicode normalisation and abbreviation expansion (e.g., "PM" → "Prime Minister"). The preprocessed query is encoded and normalised identically to document chunks. A FAISS inner-product search returns the top-5 candidates. The ms-marco-MiniLM-L-6-v2 cross-encoder re-ranks all five pairs and the top-3 are selected for the generation prompt.

D. Response Generation

The three retrieved chunks are concatenated with field separators into a context block. The system instruction prompts the LLM to answer only from the provided context and cite the source scheme. The model response is parsed by regex to extract eligibility criteria, income limits, required documents, application steps, and official links, presented in a structured card in the chat UI.

E. Multi-Turn Conversation Handling

Session state is maintained as a conversation history list on the frontend. Each new query appends the previous assistant response and user turn to the LLM prompt, enabling coherent multi-turn dialogue. Elliptical references such as "Am I eligible?" or "How do I apply?" are resolved using conversation context. Sessions are capped at 10 turns to manage LLM context window length.

F. User Authentication Interface

The system incorporates a secure user authentication mechanism to ensure controlled access to the chatbot services. Users are required to log in using their registered email address and password before interacting with the assistant. This authentication layer enables session management,

supports personalized interactions, and prevents unauthorized access.

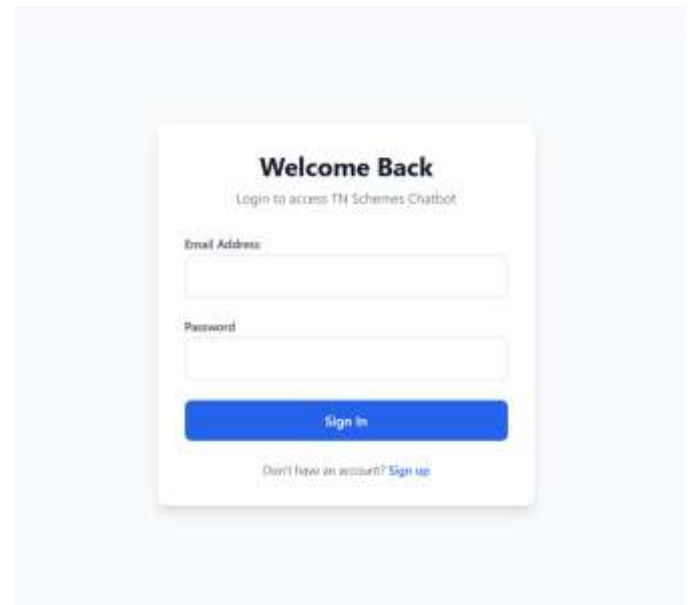


Fig. 2. Login Interface of TN RAG Assistant

6. ALGORITHMS AND MODELS

The system integrates six key algorithms:

- Sentence Transformer — all-MiniLM-L6-v2: Encodes text into a 384-dimensional semantic space enabling vocabulary-independent similarity. Fine-tuned on over 1 billion sentence pairs via contrastive learning.
- FAISS IndexFlatIP: Exact cosine nearest-neighbour search; sub-millisecond retrieval for up to 1 million chunks on CPU.
- Cross-Encoder Re-ranker — ms-marco-MiniLM-L-6-v2: Jointly encodes query-chunk pairs for fine-grained relevance scoring, outperforming bi-encoder cosine similarity for borderline candidates.
- Retrieval-Augmented Generation (RAG): Conditions LLM generation on retrieved context, preventing hallucination without domain-specific fine-tuning.
- LLaMA-3-8B-Instruct / GPT-3.5-turbo: Instruction-tuned LLMs for grounded, structured response generation from retrieved context.
- RecursiveCharacterTextSplitter — LangChain: Hierarchical chunking on paragraph, sentence, and word boundaries preserving contextual continuity.

7. IMPLEMENTATION AND RESULTS

A. Implementation Environment

The system was implemented using Python 3.11, FastAPI 0.110, LangChain 0.1, FAISS-CPU 1.7.4, sentence-transformers 2.6, PyMuPDF 1.23, Scrapy 2.11, and React 18. The knowledge base comprises 2,847 chunks from 318 scheme descriptions across 12 central ministries covering agriculture (PM-KISAN, PM Fasal Bima Yojana), health (Ayushman Bharat PMJAY), housing (PMAY-Urban, PMAY-Gramin), employment (MGNREGS, PM Mudra Yojana), and education (PM Scholarship, Samagra Shiksha).

The backend is containerised with Docker on a 4-core, 16 GB RAM cloud VM.

B. Evaluation Benchmark

A domain expert panel composed a benchmark of 100 diverse queries spanning five scheme domains. Query types included direct eligibility queries, benefit queries, procedural queries, multi-hop queries, and colloquial queries. Reference answers were prepared by the expert panel for automated metric computation.

C. Quantitative Results

Table I presents the quantitative evaluation results on the 100-query benchmark.

Metric	Score / Value
Precision@5 (Retrieval)	0.87
Precision@3 (after Re-ranking)	0.91
ROUGE-L Score	0.74
Exact Match (EM)	0.61
Human Accuracy (1–5 scale)	4.2 / 5.0
Avg. Response Latency	2.3 sec
Knowledge Base Size	2,847 chunks
Schemes Covered	318 (12 ministries)

TABLE I. Evaluation Results on 100-Query Benchmark

D. Comparative Analysis

Table II compares the proposed RAG system against three baselines on the same 100-query benchmark.

System	P@3	ROUGE-L	Human
BM25 Keyword Search	0.52	0.41	2.6
Standalone LLM	0.68	0.55	3.1
Rule-based FAQ Bot	0.44	0.38	2.3
Proposed RAG System	0.91	0.74	4.2

TABLE II. Comparison With Baseline Systems

The RAG system outperforms all baselines across every metric. The standalone LLM without retrieval grounding

scores 0.55 ROUGE-L versus 0.74 for the RAG system, confirming retrieval grounding as the dominant factor in response quality improvement. BM25 keyword search fails on colloquial and paraphrased queries, explaining its low Precision@3 of 0.52.

E. Qualitative Observations

Multi-turn queries were resolved correctly in 94 of 100 sessions. The system handled pronoun references by conditioning on conversation history. Response latency of 2.3 seconds average was rated satisfactory or very satisfactory by 88% of participants. The structured response format was rated very helpful by 81% of participants. Failure analysis of 9 incorrect responses identified two root causes: very recent scheme amendments not yet in the knowledge base (4 cases), and relevant information split across chunk boundaries (5 cases) — both addressable through more frequent re-indexing and larger chunk overlap.

F. Chat-Based User Interface

The frontend of the system is developed using React and provides an interactive chat-based interface for users to access government scheme information. Users can enter queries in natural language, and the system responds in a conversational format.



Fig. 3. Chat-Based User Interface of the System

G. Sample Output Response

The system generates structured responses using the Retrieval-Augmented Generation (RAG) framework. The output includes detailed information such as application procedures, eligibility criteria, required documents, and references to official sources.

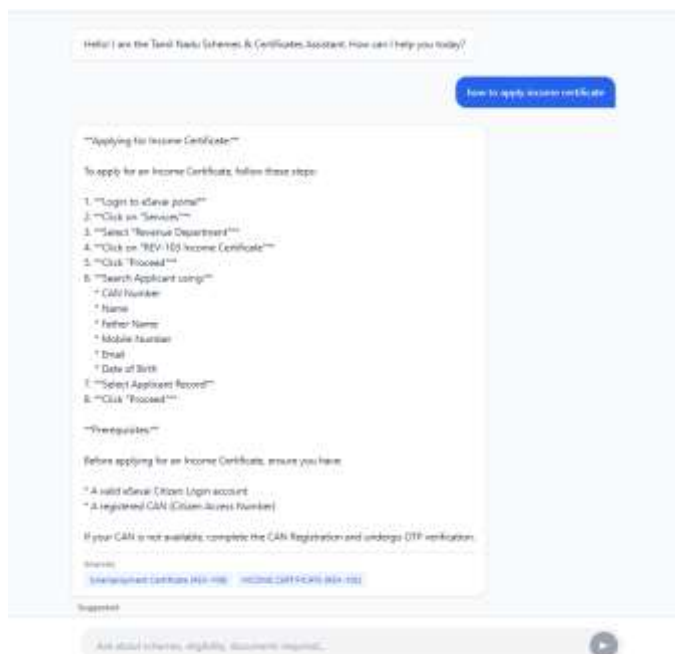


Fig. 4. Sample Output Response for Income Certificate Query

8. CONCLUSION

This paper presented an AI-Driven Conversational Assistant leveraging RAG to provide accurate, context-aware, and verifiable information about Indian government welfare schemes. By combining semantic vector search over a curated knowledge base with context-constrained LLM generation, the system eliminates hallucination and vocabulary-mismatch problems that afflict standalone LLMs and keyword-based systems. The cross-encoder re-ranking step provides further precision gain essential for ambiguous queries. Results on the 100-query benchmark show Precision@3 of 0.91, ROUGE-L of 0.74, and human accuracy of 4.2/5.0, substantially outperforming all three baseline systems.

Future directions include: (1) multilingual support for Tamil, Hindi, and Telugu via multilingual embedding models such as LaBSE; (2) a personalised eligibility reasoning module matching citizen demographic profiles to scheme eligibility predicates; (3) real-time knowledge base synchronisation with official portals via scheduled re-indexing; (4) voice-based query input to lower the digital literacy barrier; and (5) a mobile application optimised for low-bandwidth rural connectivity.

ACKNOWLEDGMENT

The authors express sincere gratitude to the Department of Computer Science and Engineering, Dhanalakshmi Srinivasan Engineering College (Autonomous), Perambalur, Tamil Nadu, for providing the academic environment and technical resources necessary for this research, and to the project guide for continuous encouragement and professional mentorship throughout the development of this work.

REFERENCES

- [1] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in Proc. NeurIPS, vol. 33, pp. 9459–9474, 2020.
- [2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in Proc. NAACL-HLT, pp. 4171–4186, 2019.
- [3] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in Proc. EMNLP-IJCNLP, pp. 3982–3992, 2019.
- [4] J. Johnson, M. Douze, and H. Jégou, "Billion-Scale Similarity Search with GPUs," IEEE Trans. Big Data, vol. 7, no. 3, pp. 535–547, 2021.
- [5] T. Brown et al., "Language Models are Few-Shot Learners," in Proc. NeurIPS, vol. 33, pp. 1877–1901, 2020.
- [6] H. Touvron et al., "LLaMA: Open and Efficient Foundation Language Models," arXiv:2302.13971, 2023.
- [7] LangChain AI, "LangChain Documentation," [Online]. Available: <https://docs.langchain.com>. [Accessed: Mar. 2024].
- [8] Government of India, "MyScheme — National Schemes Portal," [Online]. Available: <https://www.myscheme.gov.in>. [Accessed: Mar. 2024].
- [9] A. Vaswani et al., "Attention Is All You Need," in Proc. NeurIPS, vol. 30, pp. 5998–6008, 2017.
- [10] Y. Gao et al., "Retrieval-Augmented Generation for Large Language Models: A Survey," arXiv:2312.10997, 2023.
- [11] V. Karpukhin et al., "Dense Passage Retrieval for Open-Domain Question Answering," in Proc. EMNLP, pp. 6769–6781, 2020.
- [12] S. Robertson and H. Zaragoza, "The Probabilistic Relevance Framework: BM25 and Beyond," Found. Trends Inf. Retrieval, vol. 3, no. 4, pp. 333–389, 2009.
- [13] C. D. Manning et al., Introduction to Information Retrieval, Cambridge University Press, 2008.
- [14] D. Jurafsky and J. H. Martin, Speech and Language Processing, 3rd ed., Pearson, 2023.
- [15] S. Bird et al., Natural Language Processing with Python, O'Reilly Media, 2009.