International Journal of
LAW, ARTS AND HUMANITIES
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0523 www.zesterapublications.com Original Research Paper

Smart AI Caller: A Survey and Implementation Report on Real-Time Multilingual Voice Agents

Student:
Ms. Priyanka Singh and Mr. Satyajit Nayak
Student, Dept. Of CSE, GIFT Autonomous, Bhubaneswar

Faculty:
Dr. Saurav Ghosh
Professor/ Asst. Professor, Dept. Of CSE, / CSEIT GIFT Autonomous, Bhubaneswar

Abstract

Smart AI Caller is an advanced open-source AI Voice Agent platform designed to transform the way organizations interact with customers through telephone and web-based voice channels. The platform leverages modern artificial intelligence technologies to create natural, real-time conversational experiences that closely resemble human interaction. Built using a scalable microservices architecture, the system integrates powerful AI components including Large Language Models (LLMs), Speech-to-Text (STT), and Text-to-Speech (TTS) technologies into a unified real-time voice processing pipeline. This architecture allows the system to accurately interpret spoken language, generate intelligent responses based on context, and deliver them through natural-sounding synthesized speech. By combining these technologies with a modular infrastructure, Smart AI Caller enables flexible deployment, high scalability, and efficient performance for businesses seeking automated voice-based customer support and engagement solutions.

A key capability of the platform is its multilingual support, designed to address the linguistic diversity of users. Smart AI Caller natively supports more than ten Indian languages, including Hindi, Tamil, Telugu, and Bengali, enabling organizations to communicate with users in their preferred language and improve accessibility and user satisfaction. The system is optimized to deliver sub-second response latency, ensuring smooth and natural conversations without noticeable delays. In addition, the platform integrates seamlessly with enterprise telephony providers such as Twilio, Vonage, and SIP-based systems, allowing organizations to incorporate AI voice agents directly into existing communication infrastructures. The system also supports web-based voice interactions, enabling businesses to deploy AI-powered assistants on websites and digital platforms.

To simplify the design and management of conversational agents, the platform provides a visual drag-and-drop workflow builder that allows users to create complex conversation flows without requiring extensive programming knowledge. Furthermore, Smart AI Caller incorporates a Retrieval-Augmented Generation (RAG) knowledge base that enables document-based question answering by retrieving relevant information from organizational documents and knowledge repositories. This ensures that responses remain accurate, contextual, and informative during conversations. The project report presents the overall system architecture, component design, data flow mechanisms, database schema, infrastructure technologies, and performance benchmarks, demonstrating that production-grade AI voice capabilities can be achieved within an open, extensible, and scalable platform suitable for modern enterprise applications.

Keywords—Voice AI, Large Language Models, Speech Recognition, Text-to-Speech, Multilingual Systems, Telephony, RAG, FastAPI, Pipecat, pgvector, Microservices.

1 Introduction

Modern organizations increasingly depend on automated communication systems to handle customer support, appointment scheduling, service inquiries, information delivery, and campaign-based calling. Conventional interactive voice response systems rely on rigid menu trees and predefined options, which often frustrate users because they cannot interpret natural speech or adapt to changing conversation contexts. Recent advances in large language models (LLMs), neural speech recognition, streaming audio pipelines, and speech synthesis have made it possible to replace static IVR flows with dynamic voice agents that listen, reason, retrieve knowledge, and respond naturally.

Smart AI Caller is developed as a practical platform for such real-time conversational automation. It receives caller audio through telephony providers such as Twilio, Vonage, or SIP-compatible infrastructure; streams audio through WebSocket channels; converts speech into text; generates contextual responses using LLMs; optionally retrieves organization-specific knowledge using a RAG knowledge base; and converts the generated response back into natural speech. The complete pipeline is designed for low latency so that the user experiences a continuous conversation rather than a delayed question-answer exchange.

This paper-style report consolidates the major project report into a concise research presentation. Similar to a survey paper, it first discusses background and challenges, then reviews enabling technologies, presents the proposed architecture, evaluates the system, and concludes with open research and development directions.

2 Background and Motivation

Automated voice systems are widely used in banking, healthcare, education, e-commerce, telecommunications,

International Journal of
LAW, ARTS AND HUMANITIES
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0523 www.zesterapublications.com Original Research Paper

and public-service environments. However, many existing systems are rule-based, expensive, proprietary, or difficult to customize for small and medium-sized organizations. Their limitations become more visible in linguistically diverse regions such as India, where users may prefer Hindi, Bengali, Tamil, Telugu, or other regional languages.

The motivation of Smart AI Caller is to provide an accessible and scalable alternative. Instead of requiring users to follow a fixed menu, the system allows natural spoken interaction. Instead of relying only on hand-coded scripts, it can use LLM reasoning and document retrieval to answer questions. Instead of limiting deployment to one communication channel, it supports both telephony and web-based voice interaction. The project therefore connects recent AI progress with the engineering requirements of production voice automation.

3 Challenges in AI Voice Calling Systems

Real-time AI calling platforms face challenges comparable to those discussed in survey literature: data flow, computational complexity, scalability, visualization, and security. In Smart AI Caller, these challenges appear in the following forms.

3.1 Real-Time Audio Processing

Voice interaction requires continuous audio capture, streaming, transcription, response generation, synthesis, and playback. Even small delays can make the conversation feel unnatural. The system must therefore minimize end-to-end latency across STT, LLM, RAG, and TTS components.

3.2 Multilingual Understanding

Indian users frequently communicate in multiple languages and accents. A useful voice agent must recognize regional speech accurately and respond with clear synthesized speech. This requires careful integration of language-aware STT and TTS services.

3.3 Knowledge Grounding

LLMs can generate fluent responses, but enterprise callers often require accurate answers from organization-specific documents. Smart AI Caller uses a RAG pipeline with vector search to ground responses in uploaded knowledge sources.

3.4 Scalability and Deployment

A production call center may handle many simultaneous calls. Each call requires an active pipeline instance, asynchronous task processing, database access, and media streaming. The architecture must support horizontal scaling without tightly coupling services.

3.5 Workflow Flexibility

Business teams may need to create conversation flows without writing code. A visual workflow builder helps define call logic, decision nodes, and scripted pathways while still allowing AI-generated responses.

4 Related Technologies

Smart AI Caller integrates several modern technologies into one platform.

Large Language Models: LLMs provide intent understanding, response generation, context maintenance, and reasoning. The report identifies Groq-hosted LLM inference as a key component for fast response generation.

Speech-to-Text: Streaming STT converts caller audio into text. The project discusses Deepgram and Sarvam AI as speech-processing services, with emphasis on multilingual performance.

Text-to-Speech: TTS services such as ElevenLabs, Cartesia, and Sarvam AI produce natural spoken output. Voice quality is important because robotic speech reduces user trust and engagement.

Retrieval-Augmented Generation: RAG retrieves relevant document chunks from a vector database before the LLM generates a response. This improves factual accuracy for organization-specific questions.

Telephony and Web Streams: Twilio, Vonage, SIP systems, and web audio interfaces connect external callers to the real-time voice pipeline.

5 Proposed System Architecture

Smart AI Caller follows a modular microservices architecture. The major components are the frontend dashboard, backend API, real-time voice pipeline, AI service layer, database, cache and queue layer, object storage, and deployment infrastructure.

The backend API is implemented using FastAPI and coordinates users, agents, workflows, calls, transcripts, recordings, and knowledge-base operations. The voice pipeline is based on Pipecat-style real-time processing, where audio frames move asynchronously through VAD, STT, LLM, RAG, and TTS stages. PostgreSQL 17 stores structured application data, while the pgvector extension supports vector similarity search for RAG. Redis and ARQ provide asynchronous task execution, and MinIO stores files such as uploaded documents, recordings, and generated assets. Docker Compose supports local and production-style deployment.

[Architecture Diagram: Caller/Web User → Telephony or Web Audio → Real-Time Voice Pipeline (STT, LLM+Workflow, TTS) ↔ FastAPI Backend ↔ Next.js Dashboard; RAG Knowledge Base / PostgreSQL+pgvector; Redis/ARQ Async Tasks; MinIO Storage]

Figure 1: High-level architecture of Smart AI Caller showing caller input, streaming voice processing, AI reasoning, speech synthesis, dashboard control, knowledge retrieval, asynchronous tasks, and storage.

6 Call Flow

A typical call proceeds through the following sequence:

1. A user initiates a phone or web-based voice session.
2. The telephony provider streams audio to the backend over WebSocket.
3. Voice activity detection identifies active speech segments.

International Journal of
LAW, ARTS AND HUMANITIES
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0523 www.zesterapublications.com Original Research Paper

4. The STT component converts speech into text.
5. The workflow engine checks whether the conversation should follow a predefined node path.
6. The RAG module retrieves relevant document context when knowledge grounding is required.
7. The LLM generates a response using conversation history and retrieved context.
8. The TTS component synthesizes the response into speech.
9. The generated audio is streamed back to the caller.
10. Transcripts, usage data, and recordings are stored for later review.

7 Implementation Features

The project implements a service-factory pattern so that AI providers can be configured or replaced without rewriting the pipeline. The visual workflow engine allows users to create conversation flows through nodes and edges, making the system useful for both technical and non-technical users. The RAG knowledge base enables document upload, chunking, embedding generation, vector storage, and semantic retrieval. The dashboard supports agent configuration, call monitoring, transcript access, and recording downloads.

A major implementation concern is maintaining asynchronous execution. Blocking operations in STT, LLM, or TTS would increase latency and reduce concurrent call capacity. Therefore, the architecture separates real-time media handling from background tasks and storage operations.

8 Results and Evaluation

The project report evaluates Smart AI Caller across latency, speech recognition accuracy, speech synthesis quality, resource utilization, scalability, and testing coverage. Table 1 summarizes the reported results.

Table 1: Performance summary of Smart AI Caller

Metric	Reported Result
End-to-end response latency	Below 1 second under test conditions
Speech recognition WER	4.2% for English; 7.8% for Hindi
Voice synthesis MOS	4.3 for English; 4.1 for Hindi
CPU use per active pipeline	3.2% for English; 3.5% for Hindi
Concurrent call simulation	50 WebSocket call sessions with stable latency
Target benchmark	Less than 10% WER and above 4.0 MOS

These results suggest that the architecture is suitable for real-time interaction. Low word error rates help the LLM receive accurate user intent, while high MOS values indicate natural and understandable synthesized speech. Stable concurrent WebSocket testing demonstrates that the asynchronous pipeline, Redis queue, and modular service design can support practical multi-call workloads.

9 Open Research and Development Issues

Although the platform demonstrates strong initial results, several open issues remain. First, multilingual robustness must be improved for noisy environments, code-mixed speech, and regional accents. Second, RAG responses require stronger evaluation methods to measure factual grounding and hallucination reduction. Third, enterprise deployments require privacy controls, role-based access, encryption, audit logs, and compliance workflows. Fourth, the system can benefit from adaptive interruption handling so callers can naturally speak over the AI response. Finally, large-scale production deployment should include autoscaling, observability dashboards, failure recovery, and cost optimization.

10 Future Scope

Short-term enhancements include improved dashboard analytics, better workflow templates, enhanced transcript search, and richer agent configuration. Medium-term enhancements include CRM integration, advanced call campaign management, sentiment analysis, speaker diarization, and better multilingual coverage. Long-term work may include self-hosted speech models, autonomous agent testing, enterprise-grade orchestration, and deployment across high-volume call centers.

11 Conclusion

Smart AI Caller presents a complete approach to building real-time multilingual AI voice agents using open and modular technologies. By combining telephony integration, streaming audio processing, LLM reasoning, RAG-based knowledge retrieval, multilingual STT and TTS, visual workflow design, and containerized infrastructure, the platform addresses many limitations of traditional IVR systems. The reported evaluation demonstrates sub-second latency, strong speech recognition accuracy, natural voice synthesis, efficient resource usage, and stable concurrent call handling. Overall, Smart AI Caller provides a practical foundation for intelligent customer communication and a useful direction for future research in scalable voice AI systems.

References

1. P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," NeurIPS, 2020.
   A. Radford et al., "Robust Speech Recognition via Large-Scale Weak Supervision," OpenAI Technical Report, 2022.
2. Meta AI Research, "Llama 3: Open Foundation and Fine-Tuned Chat Models," 2024.
3. Sarvam AI, "Saaras: Indian Language Speech Foundation Models," 2024.
4. Groq Inc., "LPU Inference Engine: Architecture and Performance," 2024.
5. Pipecat Framework, "Real-Time AI Voice and Multimodal Agent Framework," 2024.

International Journal of
LAW, ARTS AND HUMANITIES
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0523 www.zesterapublications.com Original Research Paper

6. FastAPI Documentation, "FastAPI—Modern, Fast Web Framework for Building APIs with Python," 2024.
7. pgvector, "Open-Source Vector Similarity Search for PostgreSQL," 2024.
8. Twilio, "Twilio Media Streams—Real-Time Audio Streaming over WebSocket," 2024.
9. Redis, "Redis 7.0—In-Memory Data Structure Store," 2024.
10. Docker Inc., "Docker Compose—Define and Run Multi-Container Applications," 2024.
11. MinIO, "MinIO High Performance Object Storage," 2024.
12. ElevenLabs, "Voice AI—Text-to-Speech and Voice Cloning," 2024.
13. Deepgram, "Nova-2 Streaming Speech Recognition API," 2024.
14. LangChain, "Framework for LLM Applications," 2024.
15. OpenAI, "Text Embeddings and Semantic Search," 2024.
16. Babburi, S. (2023). Hybrid blockchain architecture for verifiable data provenance in cloud pipelines. International Journal of Intelligent Systems and Applications in Engineering, 11(4s), 711–719.
17. Gaddam, S. From Fixed Specifications to Self-Adapting Systems: A Machine Learning Perspective on Software Engineering.
18. Reddy, S. K. R. Developing a Modular AI Framework to Enhance Scalability and Personalization in Next-Generation Reward Platforms.
19. Poojari, R. Enhancing Healthcare Decision-Making through Machine Learning and the Analysis of Large-Scale Medical Data.
20. Mahimalur, R. K., Vasgam, M., & Manoharan, D. (2025). From Assessment to Automation: DevOps Lifecycle Management for Secure Cloud Migration and CICD Implementation. Power System Technology, 49(3).
21. Purmani, S. S. R. (2025). Enhancing IT strategic planning and decision making through data visualization. International Journal of Enhanced Research in Management & Computer Applications, 14(4), 75–81
22. Purmani, S. S. R., & Kotte, G. Intelligent Project Orchestration: How Generative AI is Reshaping Go-to-Market Strategy Planning and Cross-Functional Delivery. environments, 4, 5.
23. Kotte, G. (2025). Revolutionizing Stock Market Trading with Artificial Intelligence. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.5283647
24. Cyril, H. P., & Kumara, S. Identification of Anomalies via Deep Learning-Based Models for High-Dimensional Telecom Traffic Data.
25. Kotte, G. (2025). Enhancing Zero Trust Security Frameworks in Electronic Health Record (EHR) Systems. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.5283668
26. Viswanathan, V. (2024). Embedding Ethical Principles into Generative AI Workflows for Project Teams.
27. Viswanathan, V., Polagani, S. S., Agarwal, R., Akula, S., Dey, S., & Kashyap, R. (2025, September). AI-Augmented Threat Intelligence for Proactive Intrusion Detection in Multi-Cloud Ecosystem. In 2025 IEEE International Conference on Advanced Computing Technologies (ICACT) (pp. 567-572). IEEE.
28. Mudusu, S. K. (2026, March 26). A data trust scoring framework for reliable and responsible AI systems. InfoWorld (Foundry Expert Contributor Network).
29. Mudusu, S. K. (2025, June 3). Transforming legacy IT systems with AI-driven data engineering for improved efficiency and insights. Hampton Global Business Review (HGBR).
30. Gajula, S. (2025). Next-Gen Secure Cloud-Native Platforms For Financial Institutions: A Microservices And Zero Trust-Based Resilience Model. Journal of International Crisis & Risk Communication Research (JICRCR), 8.
31. Maturi, S. Y. (2025). Blockbond Hardening: Securing Pooled-Hash Protocols Against Traffic Tampering, MITM Hash-Rate Hijacking, and Template Coercion. https://doi.org/10.20944/preprints202512.2064.v1
32. 29. [25] Ranjbareslamloo, S., Dzukey, G. A., Islam Muhit, M. M., & Qattawi, A. (2025). Numerical and experimental study of residual stress in additively manufactured IN718. Manufacturing Letters, 44, 915–927. https://doi.org/10.1016/j.mfglet.2025.06.108
33. 30. [26] Manoharan, D. (2026). Synthetic EDI Test Data Generation For Secure, Scalable, And PHI-Free Healthcare Claims Quality Engineering. Journal of International Crisis and Risk Communication Research, 9(1).
34. P. Venkata Ramana. (2024). AI-driven predictive analytics in ERP systems for proactive supply chain optimization. International Journal of Innovative Engineering and Management Research (IJIEMR).
35. Kavuri, S. (2025). Critical Review of Software Testing Problems in the Current Decade. International Journal on Science and Technology, 16(2). https://doi.org/10.71097/ijsat.v16.i2.9469
36. Venkata Pavan Kumar Gummadi. (2024). API Design and Implementation: RAML and OpenAPI Specification. Journal of Electrical Systems, 16(4), 76–85. https://doi.org/10.52783/jes.9329
37. Gajula, S. (2025). Cloud transformation in financial services: A strategic framework for hybrid adoption and business continuity. International Journal of Scientific Research in Computer Science, Engineering and Information technology.
38. Gajula, S. (2025). Intelligent Customer Churn Analytics in Digital Banking Using Advanced Machine Learning Models. 2025 1st International Conference on Emerging Trends in Information Systems and Informatics (ICETISI), 1–6. https://doi.org/10.1109/icetisi67983.2025.11406030