Generative AI-Based Chatbot for Customer IT Support Automation

Abstract—**Enterprises increasingly face challenges in providing efficient, adaptive, and secure IT support as traditional ticketing systems struggle with latency, rigidity, and limited scalability. Existing chatbot solutions built on commercial large language models (LLMs) often introduce high costs, token inefficiency, and data privacy risks, while current retrieval and multi-agent systems lack personalization, multimodal intelligence, and dynamic adaptability. This research presents the design and implementation of a generative AI–based IT support chatbot that integrates four core innovations: a multi-agentic architecture enabling distributed, resilient, and scalable task execution; a retrieval-augmented knowledge base enhanced with knowledge graphs and episodic memory for accurate, personalized response generation; conversational form automation that leverages Rasa and transformer models to extract structured information from natural queries, minimizing user effort in ticket creation; and an interactive 3D conversational avatar with real-time speech-to-text, text-to-speech, and multimodal emotion detection to foster empathetic, human-like interactions. The system was evaluated using RAGAS metrics to benchmark response accuracy, context recall, and faithfulness. Results show strong domain-specific performance, with high accuracy for IT-related queries and effective filtering of irrelevant requests, thereby reducing reliance on manual ticketing processes. Moreover, caching, adaptive slot filling, and feedback-driven updates significantly improved efficiency and reduced computational overhead. This study contributes a novel, privacy-preserving IT support framework that combines multi-agent coordination, contextual retrieval, conversational automation, and multimodal interaction. Beyond reducing operational costs and latency, the proposed approach sets the foundation for scalable enterprise AI support systems, with future extensions planned for multilingual capabilities and ITSM platform integration.**

Keywords—generative Ai, agentic ai, knowledge base, knowledge graphs, 3d avatar, conversational form filling automation, conversational ai, large language model, natural language processing, retrieval augmented generation, prompt engineering, episodic memory, computer vision, convolutional neural networks.

I. INTRODUCTION

The increasing demand for efficient and context-aware internal IT support in organizations has revealed critical limitations in traditional ticketing systems. These systems often suffer from latency, rigidity, and poor scalability, leading to reduced employee efficiency and higher operational costs. Moreover, existing solutions typically lack adaptability, real-time learning, and cost-effective integration with enterprise infrastructures. Commercial large language model (LLM) integrations pose significant financial and data privacy challenges, while the absence of effective knowledge base management and retrieval mechanisms further restricts their ability to handle repetitive queries and emerging technical issues [1].

To address these challenges, this paper presents the design and implementation of a secure, intelligent IT Support Chatbot system tailored for enterprise environments. The proposed system integrates four major components: a Multi-Agentic architecture for distributed task execution and modular scalability, a dynamic Knowledge Base powered by Retrieval-Augmented Generation (RAG) and Knowledge Graphs for precise and context-aware response generation, a 3D Conversational Avatar that provides real-time, interactive engagement with lip synchronization and facial expressions, and Conversational Form Automation mechanism that automatically generates customized support forms when user queries cannot be resolved by the knowledge base.

A distinctive feature of the system is its ability to maintain efficient memory and personalization through episodic memory integration. By leveraging user history and previous interactions, the system generates contextually relevant and adaptive responses. In addition, advanced caching and indexing mechanisms reduce redundant token usage in retrieval, thereby lowering computational costs. The pipeline-based training and feedback loop further enhance the reliability of the knowledge base by continuously incorporating user interactions while employing guardrail mechanisms to prevent irrelevant or unsafe responses.

To ensure privacy and regulatory compliance—particularly in enterprise environments—the chatbot operates on a locally deployed LLM, minimizing the risk of exposing sensitive data to external services while supporting lightweight, cost-effective models that enterprises can manage internally. This design reduces reliance on manual ticketing processes by delivering real-time, intelligent, and secure IT support.

II. LITERATURE REVIEW

Despite advances in conversational AI and multi-agent systems, existing IT support solutions face high costs, limited memory efficiency, and insufficient automation. Many lack integrated multimodal intelligence, adaptive feedback, and real-time personalization, reducing effectiveness and scalability. LLM-based systems struggle with conversation memory, RAG knowledge bases often require manual updates, and multi-agent frameworks face coordination and knowledge-

sharing challenges. These gaps highlight the need for a unified architecture combining dynamic memory management, emotional intelligence, adaptive learning form automation, and scalable multi-agent collaboration to enhance enterprise IT support performance and user experience.

A. *Multi-Agentic Systems*

Modern improvements in AI reasoning capabilities have sparked interest in agentic systems, particularly multi-agent frameworks. These frameworks enable agents to collaborate toward shared goals by incorporating dynamic reasoning, contextual awareness, and scalability. The AutoGPT framework [2] exemplifies such systems, employing an Agentic Collaboration Process wherein multiple AI agents operate under human supervision via a Conversable Agent. This setup supports collaborative problem-solving and flexible task allocation.

AutoGPT, LangChain, and CrewAI attempted to address orchestration and collaboration. AutoGPT demonstrated autonomous task chaining but lacked reliable error handling. LangChain offered modular integration, though it depended on manual orchestration. CrewAI explored collaborative reasoning but struggled with scalability and execution speed. These studies underline the need for more robust orchestration strategies.

Distributed systems research offers parallels, particularly in retry mechanisms, exponential backoff, and fault-tolerant microservices. Similarly, LLM integration introduces issues of hallucination and unpredictability, where techniques like temperature tuning and human-in-the-loop feedback improve reliability.

B. *Limitations of current emotional identificatoin of 3d avartar*

TTS and STT technologies have significantly contributed to the development of voice-based conversational interfaces. Systems such as Google WaveNet [3] and Amazon Polly [4] have achieved high levels of speech naturalness using deep learning techniques. Similarly, OpenAI Whisper and IBM Watson STT optimize transcription in noisy environments [5]. Despite these advancements, TTS and STT systems remain limited in emotional expressiveness, constraining their effectiveness in emotionally sensitive interactions [6] [7].

Recent deep learning models, including BERT, GPT, and CNN-RNN architectures, have demonstrated the capacity to perform sentiment analysis and detect emotions from both vocal and facial inputs [8] [9]. Nevertheless, real-time multimodal emotion sensing and seamless integration into conversational systems remain active research challenges [10].

The incorporation of 3D avatars enhances user experience by providing visually expressive interfaces. Platforms such as Unity 3D, Blender, and NVIDIA Omniverse support the creation of avatars with expressive capabilities [11]. However, current avatar implementations lack dynamic emotive behavior. Integrating real-time sentiment and emotion analysis with avatars could improve user engagement and establish deeper human–computer interactions [12].

C. *Conversational Form-Filling Automation*

Conversational AI has increasingly been applied to simplify digital form filling, improving accessibility and reducing user effort. Meshram et al. [13] demonstrated how chatbots can act as interactive interfaces for structured data entry. More advanced approaches have combined LLMs with contextual augmentation, such as GPT-3.5 with RAG for browser-based form filling [14]. While effective for generic workflows, these systems lack domain-specific adaptability, conversational flexibility, and multilingual accessibility, limiting their applicability to IT support scenarios. Similarly, Dhvani, a Kannada voice-based chatbot designed for government forms, addressed accessibility for aged and monolingual users [15] but was restricted to a single regional language and scripted interactions.

Rule-based and traditional NLP approaches, such as chatbot implementations using NLTK, spaCy, and Selenium, enable automated data entry but fail to incorporate advanced contextual reasoning or LLM-driven understanding [16]. Context-aware frameworks leveraging clustering algorithms and semantic grouping improved autofill accuracy [17], while enterprise-focused AI agents applied NLP and adaptive learning to provide real-time guidance [18]. However, these solutions remain confined to structured data, lack conversational adaptability, and do not integrate semantic classification or predictive autofill mechanisms.

Overall, existing work highlights progress in automated form-filling but also reveals significant gaps. Current systems are largely general-purpose, accessibility-focused, or enterprise-oriented, with limited support for free-form conversational input, contextual adaptation, or IT-specific workflows. None address semantic ticket classification, LLM-assisted contextual suggestions or adaptive slot filling—capabilities critical to efficient IT support automation.

D. *Limitaions in Personalize Answer Geneation in RAG Systems*

Transformer-based LLMs combined with RAG have significantly impacted knowledge-intensive tasks, including IT support and automated literature review. RAG integration with domain-documentation improves factual accuracy and reduces hallucinations [19]. The use of knowledge graphs further enhances response relevance and retrieval efficiency in customer support applications.

Despite these advances, current systems do not prioritize memory efficiency [20] or scalable feedback mechanisms.

ProxyLLM [19] illustrates how LLMs can transform emotionally charged queries into neutral formats, supporting professional agent responses. Improvements in vector search and domain-specific model tuning have broadened chatbot applicability in areas such as HR and internal IT systems.

Nevertheless, these systems often suffer from token inefficiency, static dataset reliance, and limited adaptability to dynamic enterprise environments [21][22]. Consequently, even with powerful models, personalized, contextual, and cost-efficient support remains an ongoing challenge.

III. METHODOLOGY

The IT support system, primarily used by end-users, integrates multi-agent collaboration, 3D avatar interaction with emotional intelligence, conversational form-filling automation, and a knowledgebase component. The methodology combines local LLMs, vector-based retrieval, and modular microservices to deliver scalable, accurate, and human-like support while ensuring personalization and efficient solutions.

A. Multi-Agentic System

This proposed system implements a custom Multi-Agent System (MAS) designed from scratch without reliance on existing agentic frameworks. The architecture is structured in a layered approach, including the API Gateway Layer, Orchestrator Layer, and Core System Layer, ensuring modularity, maintainability, and extensibility.

The workflow starts with client input, where a user submits a goal and contextual information representing the high-level objective and relevant background. The Orchestrator receives this input and decomposes the goal into smaller, actionable tasks. To enhance reliability, the orchestrator uses robust error handling, including try/catch blocks, exponential backoff for repeated failures, and dynamic adjustment of LLM parameters such as temperature based on retry count. Each generated task is stored in the Task Ledger via the API layer, ensuring traceability and persistence.

The Agent System then generates a planning prompt using the client's goal and context. This prompt is forwarded to the Model System, which integrates multiple LLMs, including OpenAI, Ollama, and Groq, to interpret the prompt. Model outputs, typically structured in JSON, are parsed into actionable tasks. All tasks, session states, and agent progress are persisted in a MongoDB-backed database, which employs abstract interfaces to enable extensibility and support task, session, and progress tracking.

For task assignment, the orchestrator allocates tasks to agents via an assignment queue, updating each agent's Progress Ledger. Agents execute their assigned tasks, leveraging the Tool System when external operations are required, such as calculations, API calls, or web searches. Tools follow a factory design pattern, providing base interfaces and specific implementations. After processing, outcomes are validated, and workflow states are updated. Snapshots of progress are recorded to ensure reproducibility.

The system iterates through pending tasks, continuously updating progress and states in both the Task Ledger and Progress Ledger, maintaining a synchronized, reliable, and adaptive workflow for multi-agent collaboration.

B. AI 3D Avatar and Emotion Detection

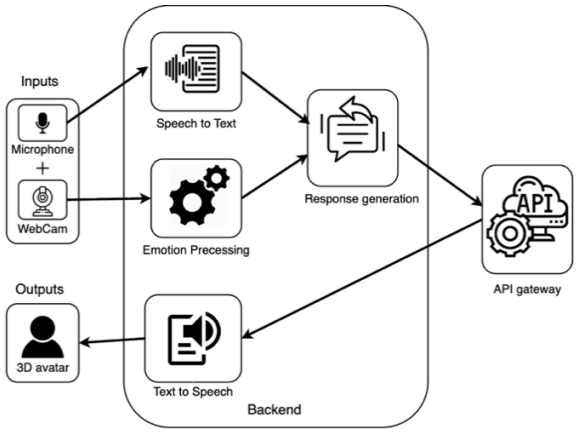

Fig 1: 3D avatar with Voice Integration and Emotion Intelligence

The development of the 3D avatar module with voice integration and emotion intelligence follows a user-centered design tailored for enterprise IT support. The system processes spoken inputs, detects user emotions in real time, and delivers empathetic responses through a dynamic 3D avatar, transforming traditional technical support into a more human-like, supportive interaction.

The module is implemented as a microservice within a scalable chatbot framework, consisting of five key components: User Input, Emotion Detection, Response Generation, 3D Avatar, and API Gateway. Voice is captured via microphone and facial cues through webcam (with consent). Emotion detection applies multimodal analysis, response generation creates empathetic outputs, and the avatar synchronizes with text-to-speech (TTS) responses. Inter-component communication is managed through REST APIs and WebSockets.

For avatar creation, the Ready Player Me SDK was used to generate customizable 3D models exported as GLB files and integrated into Unity. Blendshapes were designed for facial expressions such as frustration, satisfaction, or neutrality, while Unity's Animator Controller handled gesture transitions like nods or head tilts. Lip synchronization was achieved with Rhubarb, aligning viseme data from TTS audio with mouth movements to ensure natural speech delivery. The backend, developed in Flask, provided APIs for real-time transmission of emotion and TTS data to Unity, with WebSocket integration ensuring smooth synchronization. Testing focused on optimizing load times and ensuring performance across hardware tiers.

Voice integration combined Whisper for speech-to-text (STT) and ESPNet for TTS. The STT pipeline, developed with PyAudio, accurately transcribed queries, which were routed to the emotion detection module. TTS generated natural speech with adjustments to reflect emotions—for example, adopting slower, softer tones for empathetic responses. The outputs were seamlessly synchronized with avatar lip movements, enabling hands-free troubleshooting with a natural conversational flow.

The emotion intelligence subsystem fused three modalities to ensure reliable sentiment recognition: facial expression analysis with DeepFace, voice tone analysis via SpeechBrain, and text sentiment analysis using an IMDb-trained model. Results from each were combined through majority voting, yielding a holistic view of user emotions. Additional optimizations such as threading, noise filtering, and lighting adjustments supported real-time, reliable performance in IT support contexts.

C. Conversational Form filling automation

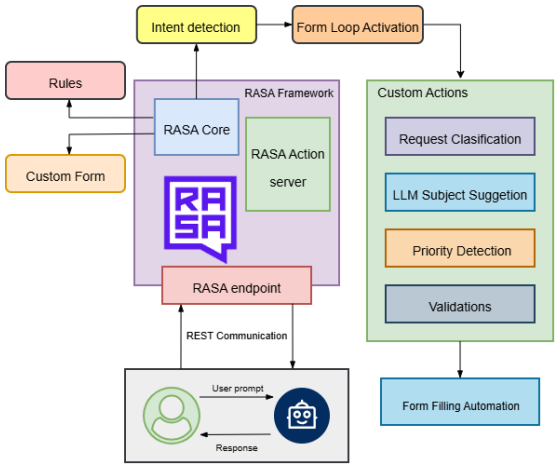

Fig. 2 2: Conversational form filling automation

As depicted in Fig. 3 component was developed to automate IT support ticket creation by extracting structured information from natural user inputs. Support request intent and slot detection were implemented using Rasa Open Source, it allows direct access to the underlying code, enabling modification of dialogue policies, NLU pipelines, and form behaviors to meet the specific requirements of IT support ticket automation. Rasa also allows integration of external models such as sentence transformers and LLMs, A conversational form loop was configured to activate automatically upon detection of a support request intent and remains active until all required fields—request type, category, subject, description, and priority—are collected. Interruption handling and fallback strategies were applied to manage out-of-scope or ambiguous inputs.

Request type and category prediction was implemented using a sentence transformer model (all-MiniLM-v6). Each user request is compared against a structured metadata repository derived from curated IT support problems. The original dataset, stored in YAML format, was converted into JSON to facilitate analysis and efficient metadata extraction. The model then selects the most semantically similar issue from this repository to classify the request type and category.

For ticket subject and description generation, a LLM (Phi-3 via Ollama) is employed. The model receives contextual information from the initial request and predicted labels to generate suggested text. A hybrid approach allows users to either accept or override these suggestions.

Ticket priority is determined through a hybrid method combining keyword weighting (60%) and semantic similarity (40%), supported by domain-specific business rules. Critical terms and operationally sensitive issues, such as hardware failures or outages, are automatically assigned higher priority scores. Predicted priorities are presented to users for confirmation, allowing manual adjustment when necessary. Priority detection scoring mechanism is mathematically formulated as:

$$Priority\ score = (Keyword\ Matching\ \times 60\%) + (Semantic\ Similarity\ \times 40\%) + Business\ Rules$$

D. Adaptive Knowledgebase with RAG

The system is primarily a Retrieval-Augmented Generation (RAG)-based system that integrates vector-based retrieval, LLMs, and episodic memory to deliver accurate and personalized responses. IT support documents are first mined from PDF manuals and transformed into JSON format with rich metadata, capturing issue types, categories, and relevant keywords. To ensure granular and precise retrieval, the content is segmented using recursive text splitting, and each segment is encoded using the Sentence Transformer model (all-MiniLM-L6-v2) before stored as a vector collection in Chroma DB.

When a user query is received, it is semantically searched against the Chroma DB collection to retrieve the top five most relevant document chunks. These chunks are then re-ranked using a cross-encoder model (ms-marco-MiniLM-L-6-v2) to enhance retrieval accuracy based on semantic similarity, custom prompt engineering, and AI-generated outputs. To improve personalization and context awareness, an episodic memory module processes historical user queries and responses stored in MongoDB through semantic embeddings. The system retrieves and sub-ranks the top five related past interactions using cosine similarity, enabling highly relevant historical answers to support current queries.

For domain-specific questions, such as HR policy-related issues, the system extracts keywords and tagged categories from user input to query a Neo4j knowledge graph. Connected policy nodes are retrieved and incorporated into system prompts to produce policy-aware responses. Responses are also cached in Redis to reduce latency, and user feedback is leveraged to adjust the weighting of episodic memory retrieval, promoting higher-quality responses in future interactions.

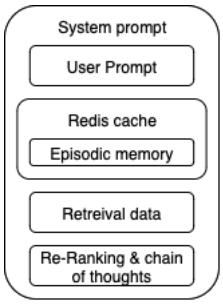

Fig 3 : System prompt structure

As shown in Fig. 3, the system prompt organizes the components of the RAG-based IT support assistant to generate personalized and accurate answers. Retrieved document chunks are first re-ranked based on semantic similarity, and relevant past interactions from episodic memory are incorporated. The prompt also uses a chain-of-thought approach within the LLM to reason through the query step by step.

Finally, the knowledge base is dynamically updated by incorporating user-validated answers, enabling continuous learning and adaptive knowledge management.

D. Data Source and Collection

The knowledgebase was built using Kaggle datasets and IT manuals, providing structured and unstructured IT support content for the RAG system [23]. For form-filling automation, the primary dataset was the Customer Support Tickets dataset from Hugging Face. Using Google Collab, only English-language tickets were retained, non-essential columns were removed, and ticket subjects were extracted to train the support request intent model [24]. Additional data for request type and category prediction was collected from publicly available IT support resources, structured in YAML, and converted to a JSON metadata repository containing issue title, content, predicted category, and issue type [25]. Priority detection used IT support guides to map keywords and descriptions to critical, high, medium, and low levels [26]. The TTS module was trained using the LJSpeech dataset (13,100 audio clips from a single female speaker) [27], while sentiment analysis employed the IMDB movie review dataset from Kaggle (50,000 labeled reviews) [28]

E. Bias Mitigation

Class imbalance in the dataset was addressed using stratified sampling and careful selection of examples across request types and categories. Limiting the dataset to English tickets avoided inconsistencies introduced by multilingual data. Priority detection incorporates both rules and semantic analysis to prevent underestimation of critical issues.

IV. Results & Discussion

A. Knowledgebase RAG generation Accuracy

The RAG-based knowledgebase framework was tested to evaluate how well it generates accurate IT support responses compared to other RAG systems. The goal was to check if the answers matched user queries and remained relevant. For measuring accuracy and reliability in knowledge retrieval and response generation, RAGAS, a standard evaluation method for RAG tasks, was used.

Table I shows the knowledge base evaluation using RAGAS metrics. For IT-related queries like “Can’t connect to Wi-Fi,” the system scored 1.00 in context recall and faithfulness, and 0.92 in answer relevancy. “My computer won’t turn on” also performed well with scores of 0.92, 0.97, and 1.00, respectively. Irrelevant prompts such as “My dog is barking” scored 0 across all metrics. These results show strong domain-specific performance with appropriate filtering of unrelated queries.

TABLE 1. EVALUATION RESULT USING RAGAS METRICS

Prompts	Criteria		
	Context recall	Answer relevancy	Faithfulness
Can’t connect to wifi	1	0.92	1
My computer won’t turn on	0.92	0.97	1
My dog is barking	0	0	0
Email is not not syncing on my phone	0	0.89	0.79

B. Accuracy of Emotion Recognition and Voice Integration Modules

The system evaluation demonstrated high performance across modules. The STT component achieved an accuracy of 95%, ensuring reliable transcription of user inputs. The Text-to-Speech TTS module obtained a Mean Opinion Score of 4.3/5, indicating natural and clear speech output. Emotion detection based on facial modality reached 80% accuracy, effectively capturing user expressions. Additionally, sentiment analysis achieved 85% accuracy, enhancing the system’s ability to interpret and respond to emotional context.

TABLE 2. MODULE PERFORMANCE EVALUATION RESULTS

Module	Metric	Result
STT	Accuracy	95%
TTS	MOS	4.3/5
Emotion Detection	Facial modality	80%
Sentiment Analysis	Accuracy	85%

CONCLUSION

This paper presented a secure and intelligent IT support chatbot system designed for enterprise environments. By integrating multi-agentic architecture, RAG-based knowledge retrieval, episodic memory, form automation, and a 3D conversational avatar, the framework provides accurate, context-aware, and personalized support while reducing latency and costs. Experimental evaluation using RAGAS metrics demonstrated strong domain-specific performance, with precise retrieval and effective filtering of irrelevant queries. Future work will focus on extending the system with multilingual support and integration with ITSM platforms.

V. REFERENCES

[1] Saberion, "OASYS," LOLC Technologies, [Online]. Available: https://lolc.oasys.lk. [Accessed january 2025].

[2] H. Yang, S. Yue and Y. He, "Auto-GPT for Online Decision Making: Benchmarks and Additional Opinions," arXiv, 2023.

[3] . A. van den Oord, "WaveNet: A generative model for raw audio," arXiv, September 2016. [Online]. Available: https://arxiv.org/abs/1609.03499. [Accessed 21 June 2025].

[4] "Amazon Polly – Text-to-Speech Service," Amazon Polly, [Online]. Available: https://aws.amazon.com/polly/. [Accessed 21 June 2025].

[5] OpenAI, "Whisper: Robust speech recognition via large-scale weak supervision," OpenAI, 2022. [Online]. Available: https://openai.com/research/whisper. [Accessed 21 June 2025].

[6] M. Barakat, "Deep learning-based expressive speech synthesis: A systematic review," *EURASIP Journal on Audio, Speech, and Music Processing,* 2024.

[7] "The Future of Speech is Synthetic," Number Analytics, 2025. [Online]. Available: https://www.numberanalytics.com/. [Accessed 21 June 2025].

[8] S. Devlin, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Association for Computational Linguistics*, 2019.

[9] A. Mollahosseini, "AffectNet: A database for facial expression, valence, and arousal computing in the wild," *IEEE Transactions on Affective Computing,* vol. 10, no. 1, p. 18–31, 2017.

[10] F. Ringeval, "AVEC 2017: Real-time emotion and sentiment recognition," in *Proceedings of the 7th International Workshop on Audio/Visual Emotion Challenge (AVEC)*, 2017.

[11] Unity Technologies, "Unity real-time development platform," Unity Technologies, 2023. [Online]. Available: https://unity.com/. [Accessed 20 June 2025].

[12] E. Deng, "EMOCA: Emotion driven monocular face capture and animation," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[13] S. Meshram, N. Naik and M. Vr, "Conversational AI: chatbots," in *2021 International Conference on Intelligent Technologies (CONIT)*, Hubli, India, 2021.

[14] M. Bucur, "Exploring large language models and retrieval augmented generation for automated form filling," University of Twente, Enschede, 2023.

[15] A. R. Hegde, ""Automated government form filling for aged and monolingual people using interactive tool"," *Disability and Rehabilitation: Assistive Technology,* Vols. vol. 19, no. 61, pp. 1-11, 2023.

[16] D. Patil, "Bot form filler," *International Journal of Research Publication and Reviews,* vol. 4, no. 10, pp. 1-3, 2023.

[17] S. Wang, Y. Zou, I. Keivanloo, B. Upahyaya and J. Ng, "An intelligent framework for auto-filling web forms from different web applications," *International Journal of Business Process Integration and Management,* vol. 8, no. 1, p. 16, 2017.

[18] W. Metellus , S. Balireddi, M. Maelzer, M. Ganaba and E. Shiroma, "Artificial Intelligence Agent for Contextual Guidance in Form Filling," 20 2024 2024. [Online]. Available: https://www.tdcommons.org/dpubs_series/7575. [Accessed january 2025].

[19] S. Jo and J. Seo, "ProxyLLM: LLM-Driven Framework for Customer Support Through Text-Style Transfer," arXiv, December 2024. [Online]. Available: https://arxiv.org/abs/2412.09916. [Accessed 21 June 2025].

[20] M. Pink, Q. Wu, V. A. Vo, J. S. Turek, A. Huth and M. Toneva, "Position: Episodic Memory is the Missing Piece for Long-Term LLM Agents," arXiv, 2025.

[21] Y. Yuan, "A hybrid RAG system with comprehensive enhancement on complex reasoning," arXiv, August 2024. [Online]. Available: https://arxiv.org/abs/2408.05141. [Accessed 21 June 2025].

[22] G. Agrawal, S. Gummuluri and C. Spera, "Beyond-RAG: Question identification and answer generation in real-time conversations," arXiv, October 2024. [Online]. Available: https://arxiv.org/abs/2410.10136. [Accessed 21 June 2025].

[23] T. Bueck, "Multilingual Customer Support Tickets," 2025. [Online]. Available: https://www.kaggle.com/datasets/tobiasbueck/multilingual-customer-support-tickets/. [Accessed 28 August 2025].

[24] T. Bueck, "customer-support-tickets," [Online]. Available: https://huggingface.co/datasets/Tobi-Bueck/customer-support-tickets. [Accessed 26 August 2025].

[25] A. Nash, "50 Most Common IT Support Problems and Their Solutions," 2024. [Online]. Available: https://itadon.com/blog/50-most-common-it-support-problems-and-their-solutions/. [Accessed 26 Augest 2025].

[26] AorBorC Technologies, "Support Ticket Priority Levels: 5-Step Guide," 2025. [Online]. Available: https://www.aorborc.com/support-ticket-priority-levels-5-step-guide/. [Accessed 26 August 2025].

[27] K. Ito and L. Johnson, "The LJSpeech Dataset," 2017. [Online]. Available: https://keithito.com/LJ-Speech-Dataset/. [Accessed 28 August 2025].

[28] Kaggle, "IMDB Dataset of 50K Movie Reviews," [Online]. Available: https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews. [Accessed 28 August 2025].

[29] Z. Xu, " Retrieval-augmented generation with knowledge graphs for customer service question answering," in *47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2024.