Vels University

MTECH – CSE

3rd Semester (2025- 2026)

24880107

Thanigaivel G

Project Guide :

Dr. Thirumal M.E, PHD

Dr. Kumar M.E, PHD

A Low-Code Framework for Conversational AI Assistants Leveraging Retrieval-Augmented Generation and LLMs

Abstract:

Conversational AI has evolved rapidly with the advent of Large Language Models (LLMs), enabling intelligent and context-aware interactions between users and systems. However, integrating these models into enterprise applications typically requires extensive coding and AI expertise. This project presents a low-code framework to build a Conversational AI Assistant using **Retrieval-Augmented Generation (RAG)** and **LLMs**, integrated with platforms like **Oracle APEX**. The assistant enables natural, domain-specific communication while retrieving accurate responses from enterprise knowledge sources. This approach reduces development time, lowers technical barriers, and makes AI accessible in business workflows like ERP or CRM systems.

1. Introduction:

- Natural language interfaces have become essential in modern applications due to their ability to improve usability and accessibility. Traditional chatbot systems, however, are often rule-based, lacking contextual awareness or dynamic knowledge access. The emergence of LLMs such as OpenAI's GPT and Google's T5 has enabled significant advancements in generative responses and semantic understanding.

- Combining these capabilities with **Retrieval-Augmented Generation (RAG** which augments LLMs with external document or database knowledge has opened up a new path for building smart assistants. Still, technical challenges hinder widespread adoption. By leveraging **low-code platforms** like Oracle APEX, this project proposes a solution that allows even non-developers to configure and deploy intelligent conversational systems within enterprise settings.

2. Problem Statement:

Despite the capabilities of LLMs and RAG, enterprise adoption remains slow due to:
- High complexity in integrating AI with existing systems.
- Lack of context-awareness in traditional chatbots.
- Time-consuming and skill-intensive development cycles.
- Difficulty accessing proprietary or internal knowledge sources through AI.
- No-code/low-code environments lacking AI-native components.

Thus, there is a need for a **low-code, intelligent, and enterprise-ready conversational AI assistant** that leverages **RAG and LLMs** for smarter, real-time, and domain-specific interactions.

3. Objectives:

To design a low-code architecture for a conversational AI assistant. To integrate LLMs with enterprise data using RAG. To allow contextual question-answering from internal sources (e.g., documents, databases). To deploy the assistant via Oracle APEX or similar low-code tools. To reduce AI integration complexity in enterprise applications.

4. Literature Survey:

Several studies highlight the potential of conversational agents in enterprise settings. Research in NLP has led to the development of powerful LLMs capable of understanding and generating human-like text. Tools like Oracle APEX and Microsoft Power Platform provide low-code solutions for building enterprise applications. However, limited work has been done on combining these technologies to create ERP-specific assistants.

5. System Architecture:

Components: -

- **User Interface:** Web or mobile-based chat interface built using low-code tools.

- **Middleware Layer:** Handles prompt formatting, user session management, and logging.

- **ERP Connector:** Integrates with ERP backend using APIs or direct SQL access.

Data Flow:

- User inputs query in natural language.
- Middleware formats input and sends it to the LLM.
- LLM processes input and returns structured commands.
- Middleware translates command into ERP-compatible actions.
- Results are fetched from ERP and displayed to the user.
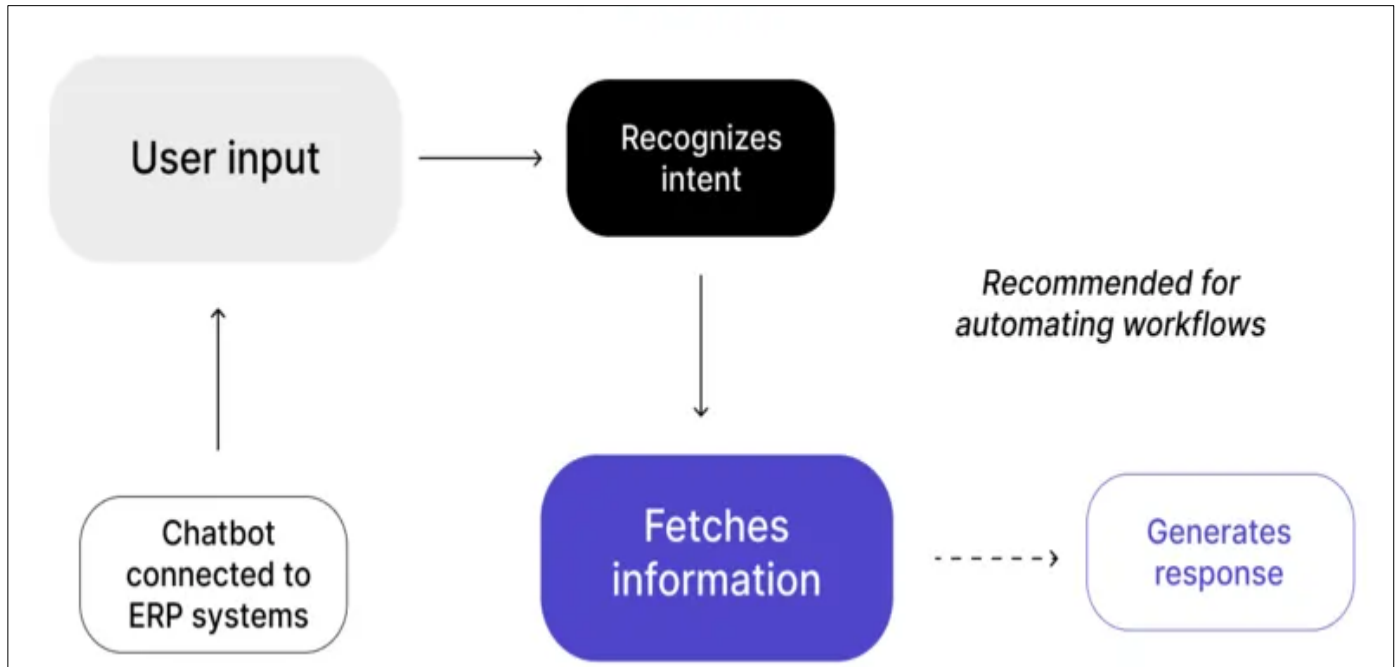
6. Methodology:

- **User Query Handling:** Accept user input via a frontend chat interface (Oracle APEX page or web component).

- **Embedding Generation:** Convert user queries and documents into vector representations using models like OpenAI embeddings or Sentence-BERT.

- **Knowledge Retrieval:** Search internal databases or document stores using similarity search (e.g., FAISS or pgvector).

- **Augmented Prompting:** Combine retrieved documents with the original query in a prompt.

- **LLM Response Generation:** Use LLMs (like GPT-4) to generate context-aware responses.

- **Display & Feedback:** Show results in the frontend. Log feedback for improvements or training.

7. Features:

Conversational AI Interface (chat-style). Natural language understanding and query resolution. Integration with enterprise databases/documents. Retrieval-augmented context injection. Low-code deployment via Oracle APEX. Scalable vector-based document search. Modular and reusable architecture.

8. Tools and Technologies:

- Oracle APEX (Low Code Platform)
- GPT-4 API
- FAISS Vector
- LangChain
- Python Rest API
- Oracle DB

10. Expected Outcomes:

A working conversational assistant with RAG integration. Seamless integration with structured databases and unstructured documents. Reduction in development time for AI-based features. Improved user satisfaction through context-rich responses. Demonstration of low-code AI orchestration in an enterprise use case.

11. Conclusion:

This project bridges the gap between AI capabilities and business usability by implementing a low-code conversational AI assistant. Through Retrieval-Augmented Generation, the assistant is able to provide accurate and domain-relevant responses, outperforming traditional chatbots. The low-code approach ensures rapid development, ease of customization, and accessibility for enterprises lacking deep AI expertise. This solution serves as a template for future enterprise-grade AI implementations.

12. Future Work: -

Integrate voice input and output for hands-free interaction. Expand to multi-agent systems for task automation (e.g., scheduling, data updates). Support real-time multilingual conversations. Use reinforcement learning from user feedback to improve response accuracy. Implement auto-summarization and document parsing for knowledge expansion. Deploy on cloud platforms for scalability and high availability.

References:

1. Siar Sarferaz (2025). Implementing Generative AI into ERP Software.

2. Yue Yin, Stefan Decker (2025). An LLM-Driven Chatbot in Higher Education for Databases and Information Systems

3. Oracle APEX Documentation. https://docs.oracle.com/en/database/oracle/apex/

4. LlamaIndex: https://llamaindex.ai

5. LangChain Docs: https://docs.langchain.com

6. OpenAI Embeddings API. https://platform.openai.com/docs/guides/embeddings