

Multilingual Conversation AI for Restaurant Automation

Nischit D S¹, Aishwarya C S², Nayana A², Varsha K G², Yashika S K²

¹Guide-Assistant Professor, Department of Information Science and Engineering

²Student Scholars, Department of Information Science and Engineering

^{1,2}GM Institute of Technology, Davanagere

Abstract : Restaurants are required to respond to menu inquiries, take orders, clarify any customizations, and provide customers with information about the status of their orders all while overseeing the preparation of food and the delivery of in-person service. In times of high demand, the need for multiple customer interactions can overwhelm the staff, lead to longer waiting times, and result in errors when order details are transferred manually. Although digital menus and mobile ordering systems standardise the ordering process, they can force customers to go through fixed menus and forms. This paper outlines the design of a Multilingual Conversational AI for Restaurant Automation system that incorporates both voice and text interaction, multilingual natural language processing (NLP), response generation based on a large language model (LLM), and structured order summarisation. The system takes customer requests via an application or a messaging-oriented interface, converts speech into text where necessary, detects the language and the intent, extracts the food items and the quantities, and uses the restaurant's menu data to produce appropriate contextual responses. A query-summary component converts the conversation into structured order data, containing the names of the items, the quantities, the prices, and the billing information. The architecture is based on a client-server model, with the user interface dealing with the interactions and the backend services handling the AI processing and database operations. The implementation context includes the use of Flutter/Dart, Firebase Authentication, Firebase Firestore, Google Gemini AI, speech-to-text, text-to-speech, and an NLP REST API. The paper covers the motivation, related work, architecture, workflow, the various implementation modules, and an evaluation plan. Since the project materials provided do not include controlled benchmark measurements, no

claims are made regarding unsupported accuracy, latency, or cost reduction figures.

Keywords: Conversational AI; Restaurant Automation; Multilingual NLP; Large Language Models; Voice-to-Text; Text-to-Voice;

I. INTRODUCTION

Restaurants regularly field menu questions, take orders, handle customization requests, offer suggestions, and answer questions about order status. When these interactions occur at the same time, employees may have difficulty responding quickly without sacrificing service quality. While taking orders manually, there can be errors in item names, quantities, add-ons and billing too. Digital menus and ordering applications have made the process easier, but often rely on structured navigation and can be difficult for customers who prefer to speak or describe a request naturally.

Advances in NLP, speech recognition, speech synthesis and LLMs make conversational interfaces increasingly feasible. Unlike the static menu form, a conversational system can interpret a request, ask follow-up questions, remember previous turns and generate an order summary. Restaurant ordering is a useful, but challenging application, since it combines transactional requirements with natural-language dialog. A system needs to understand what a customer means but also ensure that prices, availability, quantities and final order details are consistent with restaurant records.

Another challenge is language accessibility in multilingual countries like India. Customers may prefer messages in regional languages, in variants of English or in a mix of languages. Local food names, accents and code-switching can be misinterpreted by a system that assumes one language or formal phrasing. Thus the project focuses on multilingual interaction by voice and text, including a channel oriented to WhatsApp as described in the project report.

The proposed Multilingual Conversational AI for Restaurant Automation system is presented in this paper. The main contribution is the combination of multilingual interaction, LLM-based restaurant responses, voice input and output, menu-data retrieval, and conversation-to-order summarization in one workflow. It aims to reduce repetitive manual work, but keeps customer participation in clarification/confirmation steps. The paper is structured as follows: II. Related Work is reviewed in section Section III describes the architecture and methodology; Section IV describes the implementation, Section V discusses the evaluation and limitations, and Section VI concludes.

II. RELATED WORK

A. Conversational Ordering Systems : Raux et al. introduced PizzaPal, a voice-only agent for pizza ordering, which supports flexible interaction by using conversational language understanding, dialogue-state tracking, and task decomposition [1]. This work demonstrates that ordering is not just question answering: the system has to keep track of an order state and handle changes over multiple turns. Peng et al. [2] studied an approach to extract structured information from restaurant conversations, by transforming unstructured dialog into a form that can be processed by kitchen and receipts. These concepts have inspired the proposed system in the form of the query-summary and order-extraction module.

B. Restaurant Automation Based on LLM : Tayebi et al. presented an integrated approach to the development of a restaurant customer service application that incorporates voice-to-text conversion, response generator based on an LLM, database of the menu, text-to-voice transformation, and query summary generator [3]. This research can be considered the main basis of architecture of the current project. Context of conversation related to restaurants and evaluation are highlighted in the above research.

Qin and Zhou present Eva, an LLM-based multilingual voice agent network for restaurant management [4]. This system combines speech recognition, task agents, and speech synthesis modules. Yang et al. design HYLR-FO, which is a language model and rule-based hybrid architecture designed to be more flexible in terms of language processing but ensure order fulfillment at the same time [5].

C. Multilingual and Context-Aware Conversation: Multi-language conversational systems have to determine the language spoken, maintain context in the conversation, and respond accordingly when users switch languages. In a context-aware trilingual conversational service robot designed for restaurant applications, Pirabaharan et al.

discuss a robot that supports English, Sinhala, and Tamil languages [6]. This shows the significance of the number of languages supported and context-aware order placement. For our current project, determining the language and normalizing it is an explicit step in the process.

D. Research Gap: The existing literature provides evidence of advancements made in voice ordering, restaurant LLMs, structured extraction, and multilingual agent domains. But all of these features need to be integrated properly in order to serve any practical purpose within the context of a restaurant. Understanding of the language needs to relate to the menu, price information needs to be extracted through structured databases, dialogue state needs to indicate modifications in orders, and lastly a structured summary needs to be generated before finalizing the order.

III. SYSTEM ARCHITECTURE AND METHODOLOGY

A. Design Overview : The architectural design follows a client-server approach. The client receives voice or text inputs and presents conversations, clarification queries, and order summaries. Backend services process language, AI responses, menu lookups, and order preparations. The system consists of five logical components: (1) user interaction and voice/text input, (2) speech and multilingual text processing, (3) AI response generator with domain knowledge of restaurants, (4) query summarizer and order extractor, and (5) database-based order processor and response generator. The frontend may be developed using Flutter and Dart, whereas a text channel based on WhatsApp may be used for communication in case of required integration. Text input is processed by the multilingual text processor component directly. Voice input will be passed through the Voice-to-Text (VTT) service first. The generated text will then be processed to determine language, purpose of the message, food items requested, their quantities, and the context of the conversation. AI response generator will create a reply to a request, recommend some items, or pose a clarification query depending on the analysis results and restaurant domain knowledge. The query summarizer component processes the conversation to extract order fields.

B. METHODOLOGY WORKFLOW:

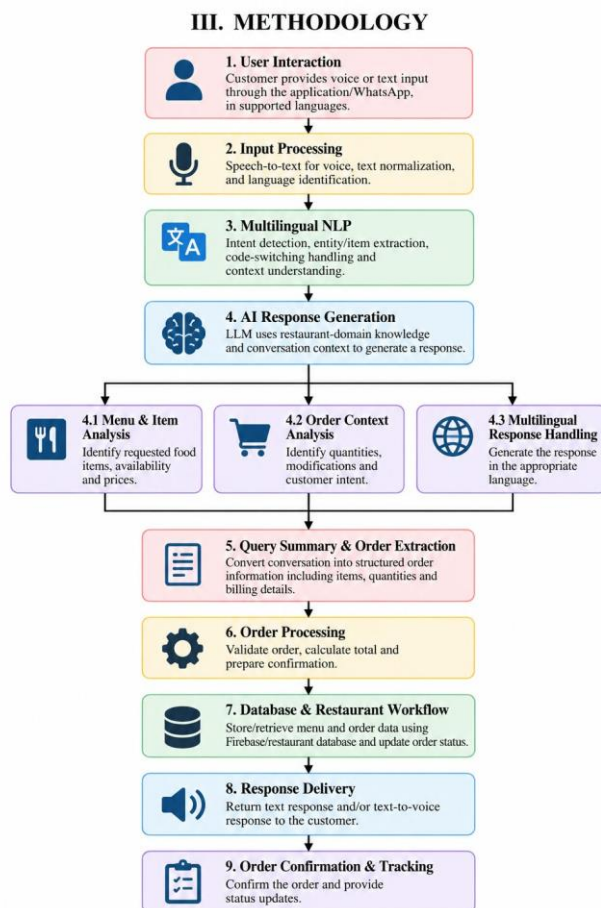


Fig. 1. Multilingual Conversational AI for Restaurant Automation Workflow

Step 1: Customer interaction: The customer initiates the conversation using either text or voice and uses one of the languages that the interface supports. The interface needs to offer controls for initiating, continuing, and ending the conversation.

Step 2: Input processing: voice inputs get translated into text form through VTT while the textual inputs get processed straightaway. The text normalization step may bring about uniformity in the formatting without losing food names and their quantity. The language identification process will identify the nature of the request/response to be made.

Step 3: Multilingual NLP – In this step, the system recognizes the intention of the request, which could be related to making a menu inquiry, selecting an item, customizing it, changing the quantity, making recommendations, billing inquiry, or order status inquiry. Extraction of entities helps to recognize the food items, quantities, sizes, additional items, and other necessary details.

Step 4: LLM Response Generation: The interpreted query is presented to the language model in the form of a request that includes context information about the

restaurant, such as the names of the menu items, descriptions, prices, availability, and prior conversation history. The language model generates a response, asks for more information, or suggests what to do next.

Step 5: Order Summary and Extraction of Order: The conversational input is translated into a structured order that includes item name, quantity, customization, price, and billing details, where possible. In case of missing or contradictory information, the system should seek clarification before placing an order.

Step 6: Validation and order processing: Extracted items are checked against the menu. Current prices are retrieved, totals are calculated, and an itemized summary is displayed. The customer can confirm or modify the order. An ambiguous statement must not be treated as a confirmed transaction.

Step 7: Operation on Database: Order details are stored or modified within the database set up for the application. Firebase Authentication will assist in authenticating users if needed, and Firebase Firestore will be used to store the menu and orders.

C. Data and Context Understanding: Restaurant data must contain item IDs, names, types, descriptions, prices, and availability. Context from conversation helps understand such utterances as “add one more,” “take away that,” and “make it spicy.” If any training data related to a specific domain exists, it can be used to enhance restaurant domain terminology and behavior in responding to messages. The report of the project mentions domain-specific fine-tuning as a goal, but neither a validated training dataset size nor fine-tuned parameters are provided. Thus, this paper views fine-tuning as a project goal.

D. Methodology Flow:

The diagram above demonstrates the working procedure of the Multilingual Conversational AI for Restaurant Automation model, starting from the receipt of customer input to generating the order confirmation message.

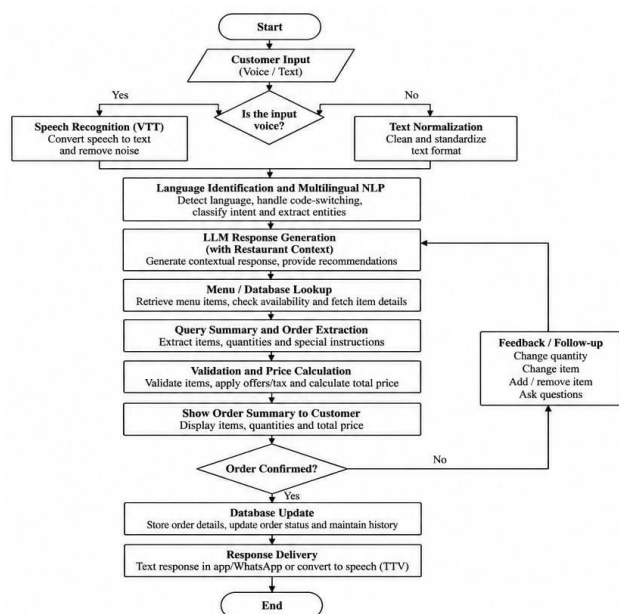


Fig. 2. Flowchart of the Proposed System.

1. *Customer Input*: The customer gives an instruction using voice or text messages in any supported language.

2. *Input Processing*: The system determines the type of input received and translates the voice input to text through Voice to Text (VTT). The text input is then cleaned and normalized.

3. *Language Recognition and Multilingual Natural Language Processing*: The system performs language recognition, code s4. Generation of Responses by the LLM: The Large Language Model (LLM), based on the contextual data related to the restaurant, comprehends the request and produces responses or recommendations.

4. *Generation of Responses by the LLM*: The Large Language Model (LLM), based on the contextual data related to the restaurant, comprehends the request and produces responses or recommendations.

5. *Retrieving Data about the Menu from the Restaurant Database*: The system accesses the menu and its related prices, offers, and availability from the restaurant database.

6. *Forming the Order Summary from the Query*: The query of the customer is transformed into an order summary.

7. *Validation and Pricing*: The system validates the items chosen, quantity validated, discounts and/or taxes applied to the order, and computes the total price of the order.

8. *Customer Confirmation*: The order is summarized and presented to the customer. The customer confirms his/her order or requests for modifications.

9. *Looping*: If there are any changes or modifications made by the customer, such as change of item and quantity, or follow up questions, the system takes action accordingly and repeats the process until the order is confirmed.

10. *Database Update*: Once confirmation is made, the order information will be stored in the database, along with the updated order status and order history, on Firebase/Firestore.

11. *Response Delivery*: Lastly, the system sends the confirmation message in text form or turns the response into speech using text to voice (TTV).

IV. IMPLEMENTATION

A. *User Interface and Communication Layer* :The user interface takes input from the user, shows the conversation, and shows the order summary and control buttons. Flutter/Dart is recognized within the technology context of the project. Useful features of the user interface are text input, voice input, visual transcript, message log, and order confirmation. Implementation of a WhatsApp channel can help to avoid the necessity of creating an app if implemented correctly. Error messages and prompts must be clear and must not give any technical information to the user.

B. *Speech processing*:VTT is used to convert customer speech to text as the input to language detection and NLP. Quality of transcription becomes essential for names of foods, amounts and customization. TTV is used to convert generated text answers to speech for those customers that want to listen. Speech components must be tested with various accents, speaking speed, language switching, and realistic background noise. It is stated in the project materials that VTT and TTV are capabilities; however, no speech engine is applicable to all supported languages.

C. *Multilingual NLP & AI Response Generation*:NLP engine will identify the language, understand the intent, extract entities and keep the conversation in context. It has to differentiate between a question and an instruction to modify an order. For instance, “What is the price of dosa?” is a menu-related question and “Add two dosas” is a request to modify the order. In case of mixed language request, it should treat the request as one request rather than two. If it’s not sure about the entity or quantity, it can ask for confirmation.

The Google Gemini AI is named as the LLM service for conversational responses in the technology stack of the project. The response generator makes use of the customer request with specific details about the restaurant.

The prompts must limit the responses in terms of only being able to use the available data from the menu. The database will be the final authority on transaction facts, with validation checks beforehand.

D.Database and Query Summary: Generator Firebase Firestore is chosen as the database to store structured data. The conceptual menu record can have item ID, name, category, description, price, and availability. The order record may have a session/customer ID, selected items, their quantities, customization, total amount, status, time stamp, and confirmation status. The exact structure depends on the final application. Firebase Authentication can be used to handle accounts, if necessary. Guest session requires its own secured session management approach. Query Summary Generator combines flexible dialogues with the structured data that is required for billing and order processing. A user's query like "two masala dosas and one coffee" can be mapped into items and quantities. If later the customer says "make it one dosa," then the query summary should show the correct quantity instead of keeping the conflicting one. Price should be taken from the database, and total should be calculated explicitly. Prior to order submission, the user should know the order and total.

*D. Security and Reliability :*The software must collect minimal personal information and limit the availability of order details. Access control and database configuration must ensure that one user cannot read or edit the details of any other customer. The API keys and the credentials of services must not be embedded in client applications. The error handling must handle transcription errors, unsupported languages, non-available items, network outages, double submissions, and ambiguous requests.

V.EVALUATION PLAN AND DISCUSSION

*A. Functional Testing Scenarios :*The project report provided does not have any controlled experimental results regarding accuracy, latencies, and satisfaction levels of the user. This means that the evaluation should be based on a set test plan and not arbitrary numbers. Simple tests should involve checking text order, voice order, menu questions, recommendations, quantity modifications, cancellations, and billing queries. Each test should note if the intent is recognized and if the answer is appropriate. Multi-lingual tests should involve all the languages to be used in deployment, code-switched queries, informal speech and common regional food items. Language recognition, item and quantity extraction and use of the correct language should be noted in the response. Speech tests should include different accents and speaking.

Tests of order extraction should involve comparing the order structure resulting at the end with a manually confirmed order structure. This test should include item name, quantity, customization, price, total, and any change that occurs during the dialogue turn. Tests on the database should ensure that only the confirmed orders are kept while multiple submissions do not result in duplicate entries.

B. Evaluation Measures: Measures to be used in evaluation are: (1) classification accuracy; (2) precision and recall for food item and quantity extraction; (3) exact order match rate; (4) speech transcription error rate; (5) end-to-end response time; (6) clarification rate for ambiguous requests; and (7) usability ratings provided by users as well as clarity of responses. The results need to include the number of cases evaluated, language breakdown, environment of testing, and examples of errors made.

*C. Expected Benefits :*The suggested solution may lead to savings of time and effort spent on repeatedly responding to the same menu prompts and recording the order. Natural language commands may prove more convenient to use in comparison with multiple navigation through the menu screens, especially for customers preferring voice and regional language communication. Summaries may be helpful in verifying customer orders and may help to avoid errors due to manual recording. Database-based menu queries can provide uniformity of pricing and descriptions.

D. Limitations: System effectiveness is reliant on speech recognition, language coverage, menu data quality, network availability, and whether the system can respect domain rules. Code-switching and food names of certain regions may be problematic in cases when few training examples exist. Responses from the LLM may be inaccurate in case the transactional facts are not checked with the database. Network latency and charges related to cloud services may be possible. The described scope of the project is limited to ordering and communication with customers only; it does not include managing inventory or supply chain.

VI. CONCLUSION AND FUTURE WORK

The current research provided the methodology and system design for Multilingual Conversational AI for Restaurant Automation. The system will provide voice and text interaction, multilingual NLP, response generation based on LLM, access to the menu data, order summarization in structured form, and order processing with the help of database technology. The main purpose of the system will be the support of natural customer

interaction using regional languages and code switching while providing order verification through the conversion of the dialogue into a validated order structure. The system uses the client-server approach and has the following implementation technologies as Flutter/Dart, Firebase Authentication, Firebase Firestore, Google Gemini AI, speech-to-text and text-to-speech, and NLP REST API.

Future directions involve implementing the pipeline end-to-end with an officially reported multilingual test corpus and analyzing results of transcriptions, order extraction, comparing times across different hardware and network conditions, and performing usability studies with actual customers and restaurant workers. Some additional improvements that can be pursued in the future are better menu-based retrieval capability, supporting even more regional languages, noise-robustness, accent tolerance, and even POS system integration when possible. All quantitative claims must only be asserted with reproducibility in mind.

REFERENCES

- [1] Raux, A., Ma, Y., Yang, P., & Wong, F. "PizzaPal: Conversational Pizza Ordering Using a High-Density Conversational AI Platform." Proceedings of EMNLP: System Demonstrations, pp. 151–156, 2018. DOI: 10.18653/v1/D18-2026
- [2] B. Peng, M. Seltzer, Y.C. Ju, G. Zweig, and K.F. Wong, "May I Take Your Order? A Neural Model for Extracting Structured Information from Conversations," Proceedings of EACL, vol. 1, pp. 450-459, 2017.
- [3] P. Tayebi, Y. Sun, and Y. Guo, "From Menus to Management: A Comprehensive Framework Utilizing Large Language Models for Automated Restaurant Ordering System," 2025 IEEE/ACIS SERA. doi: 10.1109/SERA65747.2025.11154611.
- [4] Z. (T.) Qin and J. Zhou, "Eva: An LLM-Based Multilingual Voice-Agent Network for Restaurant Operations," Proceedings of AAMAS 2025, Demonstration Track.
- [5] S. Yang, D. Kim, and S. Lee, "HYLR-FO: Hybrid Method Using Language Models and Rule-Based Methods for On-Device Food Ordering," Electronics, vol. 14, no. 4, p. 775, 2025. doi: 10.3390/electronics14040775.
- [6] K. Pirabakaran et al., "Context-aware trilingual conversational service robot for restaurant," IEEE, 2025.
- [7] A. Wang, and A. Wang, "Artificial Intelligence in Restaurant Business: Changing Experiences, Processes and Finances," 2026. doi: 10.4236/ojbm.2026.141025.
- [8] R. Hallikar and A. Porwal, "Multilingual and Accent Aware Transformation of Read Speech to Conversational Speech," 2026. doi: 10.1109/ACCESS.2026.3653970.
- [9] A. Rahman et al., "Multilingual Sentiment Analysis in Restaurant Reviews through Aspect Focused Learning," Scientific Reports, 2025. doi: 10.1038/s41598-025-12464-y.
- [10] M. F. Shorbaji, A. A. Alalwan, and R. Algharabat, "AI-Enabled Mobile Food-Ordering Applications and Customer Experience: A Systematic Literature Review and Future Research Directions," Journal of Theoretical and Applied Electronic Commerce Research, 2025. doi: 10.3390/jtaer20030156.