iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** VII **Month of publication:** July 2025

DOI: https://doi.org/10.22214/ijraset.2025.73171

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VII July 2025- Available at www.ijraset.com

CampusBuddy: A Multimodal NLP-Powered Chatbot for Malayalam Voice and Text-Based College Queries

Rejimoan R[1], Gnanapriya B[2], Jayasudha J S[3]
[1]*Department of Computer Science and Engineering, Annamalai University, Chidambaram, India*
[2]*Department of Computer Science and Engineering, Annamalai University, Chidambaram, India*
[3]*Department of Computer Science, Central University of Kerala, India*

Abstract: ***CampusBuddy is a chatbot powered by AI that helps Malayalam speakers talk and write to each other. The system uses Natural Language Processing technologies like OpenAI's GPT-4o Mini for conversational intelligence, Whisper for recognizing Malayalam speech accurately, and Silero TTS for making speech sound real. The chatbot makes it easy and natural for students and staff to ask academic and administrative questions, and it is designed especially for Malayalam-speaking users in schools. The system operates using the following order: the system receives Malayalam speech input, utilises Whisper to convert it into English text, relies on Retrieval-Augmented Generation (RAG) with OpenAI's model and ChromaDB to ensure the responses are relevant to the context, and finally is translated back into Malayalam and output as speech or text.CampusBuddy is built on Streamlit for scalability and supports multimodal interactions, giving users both voice and text outputs to make it easier to use.***
Keywords: AI-powered chatbot, Malayalam language processing, Voice-based interaction, Natural Language Processing

I. INTRODUCTION

The advancement of Natural Language Processing (NLP) and Artificial Intelligence (AI) has signif- icantly influenced how humans interact with machines, making communication more intuitive and user-friendly. Voice-based chatbots have become an essential part of this evolution, providing users with the ability to interact with systems through speech, which is more natural than typing or other traditional input methods [1]. However, most voice-based AI systems are designed for widely spo- ken languages like English, leaving regional languages like Malayalam underserved. This gap in language support limits the accessibility of these advanced technologies for native speakers, espe- cially in contexts such as education and customer service, where tailored communication is crucial.

Existing voice-based chatbot systems, such as Alexis or those based on Rasa X, mainly focus on major languages and struggle with regional languages due to a lack of dedicated resources and models [2]. Traditional Automatic Speech Recognition (ASR) systems [2] often rely on methods like Hidden Markov Models (HMMs), which have limited accuracy when dealing with regional accents and linguistic variations [3]. Moreover, these systems typically offer text-based interactions, missing the opportunity to provide a more engaging user experience through voice communication. Additionally, scalability remains a challenge when deploying these systems in environments with a large number of users, such as educational institutions [4, 3, 5]. These limitations hinder the broader adoption of voice-based systems in regions with diverse language needs, creating a demand for solutions that specifically address these gaps. To address these challenges, this project focuses on developing a voice-based chatbot for the Malayalam language, leveraging advanced NLP and machine learning models. The proposed solu- tion integrates OpenAI's GPT-4o mini as the main conversational model, using Whisper for ASR to accurately recognize Malayalam speech, and Silero models for Text-to-Speech (TTS) synthesis, enabling the bot to respond in spoken Malayalam.

The proposed project has successfully established a modular architecture that promotes main- tainability and scalability. The core system utilizes OpenAI's embedding model (text-embedding- 3-large) to vectorize college-specific data, which is then stored in a Chroma vector database for efficient retrieval. This approach enables context-aware responses through Retrieval-Augmented Generation (RAG), significantly enhancing the accuracy and relevance of the chatbot's answers to domain-specific queries. The implementation leverages Streamlit for creating an intuitive user inter- face, allowing seamless voice interactions while maintaining a clean, responsive design that displays conversation history with both text and audio outputs for maximum accessibility.

©IJRASET: All Rights are Reserved

A key technical achievement in the implementation phase has been the seamless integration of cross-lingual capabilities. The system now effectively handles the full pipeline of processing Malay- alam voice input, transcribing it to text using Whisper, translating to English for knowledge retrieval, generating contextually relevant responses, translating back to Malayalam, and finally converting to natural-sounding Malayalam speech using the Silero TTS model. This implementation addresses a significant gap in regional language support by creating a bridge between English-dominant AI technologies and Malayalam users. The modular code structure separates UI components from back- end utilities, facilitating future enhancements while maintaining system integrity across the complex processing chain of voice recognition, language translation, knowledge retrieval, and speech syn- thesis. With the outlined methodology and successful implementation, the system demonstrates the viability of multilingual voice assistants that can serve educational institutions in Kerala. The imple- mentation not only achieves the technical goals but also creates an accessible interface that requires minimal technical knowledge to use, making advanced AI technology available to Malayalam speak- ers regardless of their technical expertise. This solution sets a foundation for regional language AI systems, expanding their usability in educational settings and other domains that require tailored, localized communication.

The paper is organized as Section 2 provides a literature review, analyzing existing research and identifying gaps. Section 3 introduces the proposed methodology, explaining the system's design and workflow. Section 4 describes the implementation process and test cases, highlighting practical execution. Section 5 presents the results and discusses their implications, supported by comparative metrics. Finally, Section 6 concludes the paper with key findings and suggests directions for future research.

II. LITERATURE REVIEW

The literature review explores advancements in chatbot development across natural language pro- cessing, text-to-speech synthesis, and conversational agent benchmarking. Key studies cover the Transformer model's impact on NLP for efficient data processing, accented TTS frameworks for im- proved accessibility, and evaluations of NLU platforms like IBM Watson for intent recognition. Ad- ditionally, frameworks for college inquiry and healthcare chatbots highlight the practical applications of voice and text interfaces in academic and healthcare settings, emphasizing efficient query handling and service automation. These insights provide a foundation for selecting NLP architectures, opti- mizing language adaptability, and enhancing real-time user interactions in chatbot applications.

A. Attention is All you Need

Vaswani et al. [6] first proposed the Transformer model that replaced RNNs and LSTMs with a self-attention-based architecture. It allows parallel processing of data, which significantly improves training efficiency and scalability. The multi- head self-attention mechanism enables the model to capture intricate relationships across the entire input sequence, allowing it to understand the lan- guage pattern in a holistic manner. This, coupled with the encoder-decoder structure, makes the Transformer particularly effective for sequence-to-sequence tasks like machine translation and text summarization. While the Transformer has transformed NLP, it has its limitation. Its computa- tional complexity is O, which leads to it being very intensive in terms of resources required for longer sequences. More importantly, there is an absence of inductive bias, meaning it requires much larger datasets for optimal results, making it less applicable in low-resource scenarios. Variants like Longformer and Reformer have optimized the self-attention mechanism to mitigate such challenges. Despite all these, the Transformer model undoubtedly influenced NLP. Many of the state-of-the-art models like BERT, GPT, and T5, have recently dominated results on many NLP tasks, paving the way for using its core concepts in developing innovative and effective NLP applications.

B. Accented Text-to-Speech Synthesis with Limited Data

The study by Xue Hao Zhou et al. [7] explores the possibility of using TTS technology to en- hance usability and user experience. TTS technology translates written text into natural speech, a feature that allows visually disabled people and people with non-native language skills to take in information more easily and quickly. Recent breakthroughs in deep learning have further improved TTS systems into highly expressive and human-like speech, thanks in large part to models, such as Tacotron and WaveNet, and then the Transformer-based approaches. In summary, the proposed Ac- cented TTS framework addresses the challenges of creating speech in a specific accent. It applies a multistage processing pipeline starting from grapheme-to-phoneme conversion, where it provides an accurate mapping of the text to phonemes via the phonetic rules inherent in the target accent; then the Phoneme-to-Mel spectrogram prediction will capture essential prosodic features such as rhythm and pitch, contributing to more natural audio. Due to using accent-specific data in restricted capacities, the framework would prove suitable for low-resource languages. To test the effectiveness of the framework, experiments are carried out to compare synthesized speech with authentic samples.

The quality of accent representation and the overall speech quality is assessed through metrics such as MOS and subjective listening tests. It is essential for validation so that the framework can be used practically in real-world applications such as voice assistants and audiobook narration.

C. *Benchmarking Natural Language Understanding Services for Building Conversational Agents*

Xingkun Liu et al. [8] performed a detailed comparison of four significant NLU platforms: IBM Watson, Google Dialogflow, Microsoft LUIS, and Rasa. The experiment covered 25,000 user utter- ances on different domains with focus on intent classification and entity recognition. While all of the platforms performed equally when it comes to overall performance, IBM Watson shone brightest in intent classification. It has a clear ability to correctly interpret even very complex queries from users. However, it is somewhat wanting in the area of entity recognition due to false positives, especially where proper entities should be detected. It may, therefore, degrade user experience in some contexts such as customer service. The only limitation of the study was the exclusion of IBM Watson's latest 'Contextual Entity' annotation tool, which is designed to improve Watson's entity recognition capa- bilities. Future research should investigate how this tool impacts Watson's performance and overall NLU capabilities. Table 1 shows that all four NLU platforms—IBM Watson, Google Dialogflow,

Table 1: Comparison of NLU Services[8]

NLU services	Intent			Entity		
	Precision	Recall	F1	Precision	Recall	F1
Rasa	0.863	0.863	0.863	0.859	0.694	0.768
Dialogflow	0.870	0.859	0.864	0.782	0.709	0.743
LUIS	0.855	0.855	0.855	0.837	0.725	0.777
Watson	0.884	0.881	0.882	0.354	0.787	0.488

Microsoft LUIS, and Rasa perform similarly in intent classification and entity recognition across domains. IBM Watson excels in intent classification but has a higher false positive rate in entity recognition, which may affect user experience in applications like customer service.

D. *AI and Web-Based Interactive College Enquiry Chatbot*

Akshada Phalle et al. [9] have a research study that presents a framework for a chatbot with the use of Rasa X to answer any form of college-related inquiries. The chatbot provides an interface for both voice and text interactions, so the user can choose to either speak or type. This will utilize Rasa NLU for intent classification and entity extraction and Rasa Core for dialogue management. For voice interaction, the following libraries are used: Google Speech Recognition and pyttsx3. The chatbot was found to be efficient in answering a wide range of queries about colleges, including admissions, courses, and facilities. It can respond with timely and accurate information, thereby relieving the administrative staff and making it more efficient overall. At present, it is only limited to the English language, which may not be accessible to diverse educational institutions. Deployment of Rasa X in production environments also requires resource-intensive computation, which has to be considered carefully. Future research might include extending the language capabilities of the chatbot, while exploring how to incorporate the other college systems in real-time data access and develop better dialogue management techniques. This would really make the chatbot a much more valuable asset for these institutions of learning. Future research might include extending the language capabilities of the chatbot, while exploring how to incorporate the other college systems in real-time data access and develop better dialogue management techniques. This would really make the chatbot a much more valuable asset for these institutions of learning.

E. *A smart chatbot architecture based NLP and machine learning for health care assistance*

According to Ayanouz et al. [10], in their research, the healthcare system assistant smart chatbot offers a structure for access in effective communication between the healthcare systems and the user. The module User Interaction provides both voice and text interfaces to handle the different prefer- ences of users. NLP Processing module helps in the classification of intent and extraction of entities from the query performed by the user for understanding the intent behind the query. The module Dialogue Management ensures smooth conversations that would be contextually relevant. Similarly, the Response Generation module uses a question-answering system and Node server to generate responses. To enhance efficiency, the system is integrated with ServiceNow and the

Intelligent Au- tomation Engine (IAE). ServiceNow assists in generating tickets and fetching knowledge, while the IAE helps in automating the routing of queries and streamlines workflows. This makes the chatbot more timely and accurate in response, thus enhancing the healthcare experience.

F. Alexis : A Voice based Chatbot using Natural Language Processing

Meet Popat et al. [2] who discuss the study about Alexis, a chatbot that gives instant and precise answers to university-related questions. Alexis is a multichannel interaction tool: it supports text and voice interactions. The chatbot uses NLP techniques for interpreting user queries and provid- ing appropriate responses using TF-IDF vectorization and cosine similarity. NLTK is applied for tokenization, while HMMs are applied for speech recognition. However, the report indicates prob- lems, mainly in terms of speech recognition by noise, or natural language understanding that seems to understand difficult-to- represent conversations. Therefore, the only course left in this aspect is for continual research and development towards greater improvement. The overall components of Alexis architecture involve: input component; a chatbot core with core knowledge and functions; in- put conversion into natural speech or responses; the chatbot engine with input/output response from the input or questions it receives.

The current chatbot systems have glaring shortcomings in the adaptability to regional languages, multimodal interaction, scalability within educational environments, speech recognition accuracy of regional accents, and customizing to the specific needs of the users. Systems such as Alexis and Rasa X mainly support major languages and exclude regional languages like Malayalam, which makes the service inaccessible to various linguistic groups. Further, the majority of developed chatbots are text-dependent in terms of interaction but are far from being multicomponents, meaning not support- ing interactive voice communications easily that will improve the regionally enhanced experience. Scalability problems appear to be inherent as regards educational systems having user supports for thousands; such scenarios become difficult to gain more adaptability. In addition, where a HMM- based method has been used in implementing a chatbot like Alexis for translating speech into texts, in most cases with respect to regional accents there has a problem in determining accents to get an incorrect rate. Lastly, the existing frameworks are hardly flexible for domain- specific customization and therefore have been unable to fill some unique needs, like college-specific queries. All these gaps point to the need for a more versatile, scalable, and regionally adaptable chatbot framework.

Many Malayalam-speaking users face significant challenges in accessing crucial college infor- mation due to language barriers. This lack of clear communication restricts their ability to obtain necessary information about admissions, courses, and campus facilities, hindering their engagement and decision-making processes.

III. PROPOSED METHODOLOGY

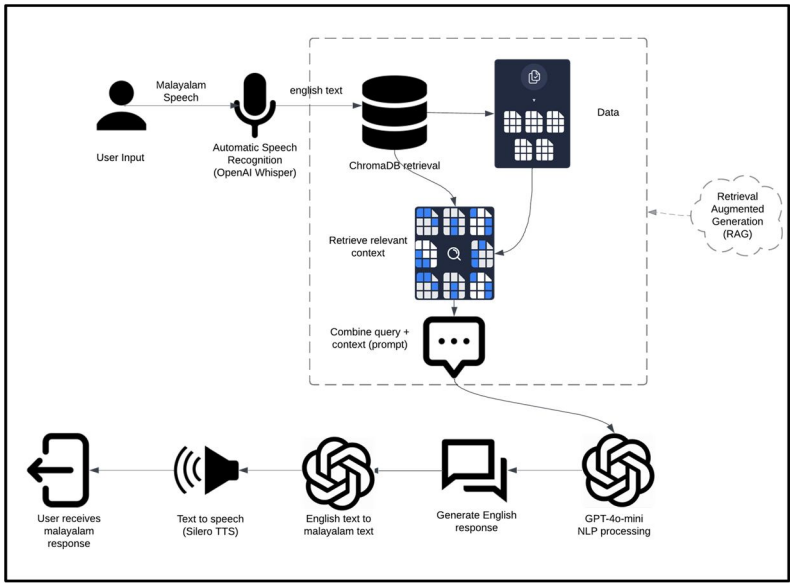


Figure 1: Design for CampusBuddy voicebot

Figure 1 shows overview of the system architecture highlighting the key components and their interactions to ensure seamless execution of the application.

A. User Input

The chatbot interaction starts when a user speaks in Malayalam. This spoken query is the primary input, triggering the system to process and generate a response. By enabling voice-based communi- cation, the chatbot ensures a seamless and natural interaction experience for users.

B. Automatic Speech Recognition (ASR)

The spoken Malayalam input is transcribed into English text using OpenAI's Whisper ASR model. This conversion plays a critical role in accurately capturing the user's query while preserving lin- guistic nuances. The generated English text is then passed on for further processing.

C. ChromaDB Retrieval (RAG-Based Context Extraction)

To enhance response accuracy, the system utilizes a Retrieval-Augmented Generation (RAG) ap- proach:

- Data Embedding: College-related information is stored in a pdf which is preprocessed and stored in ChromaDB as vector embeddings.
- Context Retrieval: When a query is received, the system searches the database for relevant information.
- Query Augmentation: The retrieved context is combined with the user's query, forming a more detailed and context-rich prompt for the language model.

D. NLP Processing (Response Generation)

The augmented query is processed by GPT-4o Mini, which generates an intelligent, context-aware response. This step ensures that the chatbot provides relevant and meaningful answers tailored to the user's query while incorporating necessary college-related information.

E. English to Malayalam Translation

Instead of relying on external translation tools, GPT-4o Mini directly translates the generated English response into Malayalam. This approach improves translation accuracy while ensuring that responses remain coherent and contextually appropriate.

F. Text-to-Speech (TTS) Conversion

The translated Malayalam text is then converted into speech using Silero TTS. This step allows the chatbot to deliver responses in a natural-sounding Malayalam voice, ensuring a smooth and engaging conversational experience for users.

G. Output Delivery

The final output consists of both Malayalam text and Malayalam voice. Users can read the chat- bot's response in text format or listen to it as synthesized speech. This dual-mode output enhances accessibility, providing users with multiple ways to consume the chatbot's response based on their preference.

IV. IMPLEMENTATION AND TEST CASES

A. Implementation

This section provides a detailed discussion of the algorithms and implementation strategies adopted for the project, ensuring a systematic and effective development process.

1) Vector Store Initialization and Document Processing Algorithm

The Vector Store Initialization and Document Processing Algorithm (Algorithm 1) enables efficient text retrieval by leveraging embeddings and vector storage, integrating tools like Langchain, OpenAI API, ChromaDB, PyPDFLoader, dotenv, and uuid. The workflow begins with configuring environ- ment variables and initializing a vector store using OpenAI's text embeddings. PDF documents are loaded, split into chunks via RecursiveCharacterTextSplitter, and assigned unique UUIDs to preserve integrity; these chunks are then embedded with OpenAI's text-embedding-3-large model and stored in ChromaDB for rapid retrieval. The subsequent Retrieval-Augmented Generation (RAG) pipeline fetches the top five relevant results for user queries from the vector database, integrating them into structured prompts to ensure contextually accurate responses from the language model.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

2) Multimodal Conversational AI Processing Algorithm

The Multimodal Conversational AI Processing Algorithm (Algorithm 2) enables seamless Malay- alam voice interactions by combining speech recognition, translation, knowledge retrieval, and speech synthesis technologies. The system begins by initializing all required components including Ope- nAI's embedding model, GPT-4o Mini for response generation, ChromaDB for knowledge retrieval, and Silero TTS for speech synthesis.

When receiving voice input, it first converts speech to text using Whisper, then translates the text to Malayalam while preserving the original context. The translated query triggers a knowledge retrieval process from ChromaDB, with the results fed into a RAG-enhanced prompt to generate accurate responses. These responses are converted back to Malayalam and transformed into natural-sounding speech through Silero TTS, with Aksharamukha handling script conversion for optimal pronunciation. The system further enhances user experience through customized UI styling, creating a complete solution for regional language voice interactions.

Algorithm 1 Processing and Storing PDF Data Using Langchain and ChromaDB

Require: Langchain, OpenAI API, ChromaDB, PyPDFLoader, dotenv, uuid
function LOAD_CONFIGURATION
Load environment variables from .env file
end function
function INITIALIZE_VECTOR_STORE
Set configuration parameters:
DATA_PATH ← "data"
CHROMA_PATH ← "chroma_db"
Initialize the OpenAI embeddings model with text-embedding-3-large
Initialize the Chroma vector store with:
collection_name = "data" embedding_function = embeddings_model persist_directory = CHROMA_PATH
return vector_store
end function
function LOAD_DOCUMENTS(data_path)
Load PDF documents from data_path using PyPDFDirectoryLoader
return raw_documents
end function
function SPLIT_DOCUMENTS(raw_documents) Define text splitting parameters:
chunk_size = 300
chunk_overlap = 100 length_function = len is_separator_regex = False
Initialize RecursiveCharacterTextSplitter Split raw_documents into chunks
return chunks
end function
function GENERATE_UNIQUE_IDS(chunks) Generate a unique uuid4() for each chunk **return** uuids
end function
function STORE_DOCUMENTS(vector_store, chunks, uuids)
Add chunks to vector_store using corresponding uuids
end function **function** MAIN
Call load_configuration()
Initialize vector_store using initialize_vector_store()
Load raw_documents using load_documents(DATA_PATH)
Split raw_documents into chunks using split_documents(raw_documents) Generate unique IDs using generate_unique_ids(chunks)
Store chunks in vector_store using store_documents(vector_store, chunks, uuids)
end function
if __name__ == "__main__" **then**
Call main()
end if

Algorithm 2 Processing Audio and Generating Malayalam Responses Using LLM and TTS

Require: PyTorch, OpenAI Whisper, LangChain, ChromaDB, Silero TTS, Aksharamukha
function INITIALIZE_MODELS
Load OpenAI embeddings model (text-embedding-3-large) Initialize LLM (gpt-4o-mini) with temperature 0.5
Create ChromaDB vector store (data) with embeddings and persist directory Load Silero TTS model (silero_tts) for Indic languages
Configure retriever with top-5 search results
return dictionary containing: embeddings_model, llm, vector_store, tts_model, retriever
end function
function TRANSCRIBE_AUDIO_WITH_WHISPER(audio_bytes) Initialize OpenAI Whisper client
Save audio_bytes to temporary WAV file
Open file and transcribe using OpenAI Whisper (whisper-1)
Delete temporary file
return transcribed text
end function
function TRANSLATE_TO_MALAYALAM(english_text, llm)
Construct translation prompt: "Translate the following text from English to Malayalam"
Invoke LLM with prompt
return translated Malayalam text
end function
function CONVERT_TEXT_TO_MALAYALAM_AUDIO(malayalam_text, tts_model) Convert Malayalam text to Roman script using Aksharamukha
Generate speech using Silero TTS (malayalam_female voice)
Save generated audio to temporary WAV file
return file path
end function
function PROCESS_AUDIO_AND_RESPOND(audio_bytes, models) Retrieve LLM, TTS model, and retriever from models
Transcribe Audio: Convert audio_bytes to English text using Whisper Translate Question: Convert English text to Malayalam
Retrieve Knowledge: Search relevant documents in ChromaDB Construct RAG prompt:
"You are a helpful assistant... Question: [English text]. Knowledge: [Retrieved content]" Generate response using LLM
Translate Response: Convert response to Malayalam
Generate Speech: Convert Malayalam text to speech using Silero TTS
return malayalam_question, malayalam_response, audio_path
end function
function GET_CUSTOM_CSS
Define CSS styles for UI elements
return CSS as formatted string
end function

3) *Streamlit-Based Voice Assistant Interface Algorithm*

Algorithm 3: Streamlit Voice Assistant Interface implements a responsive web interface for the Malayalam voice assistant using Streamlit's framework. The system initializes with customized UI configurations including page settings, visual styling, and instructional headers to guide user interaction. It maintains conversation continuity through session-state managed chat history that dynamically displays each query-response pair in real-time. The interface integrates audio capture functionality, processes inputs through the NLP pipeline, and renders outputs in both textual and spoken formats, while continuously updating the interactive elements to deliver a fluid conversational experience. This comprehensive solution combines web technologies with voice processing to create an accessible Malayalam-language assistant with persistent dialog context and intuitive controls.

Algorithm 3 Streamlit Voice Assistant Interface

Require: Streamlit, audio_recorder_streamlit, and utility functions
function GET_MODELS
return initialize_models()
end function **function** MAIN
st.set_page_config(page_title="CampusBuddy", layout="wide") st.markdown(get_custom_css(), unsafe_allow_html=True)
models ← get_models() ▷ Load and cache models
▷ Display header section
st.markdown(' ', unsafe_allow_html=True)

```
st.markdown('<h1 class="main-title">CampusBuddy: Malayalam Voice Assistant</h1>', unsafe_allow_html=True) st.markdown('<p class="subtitle">Ask your questions in Malayalam about our college!</p>', unsafe_allow_html=True) st.markdown('</div>', unsafe_allow_html=True)
if 'chat_history' ∈/ st.session_state then
    st.session_state.chat_history ← []                                    ▷ Initialize chat history
end if

st.markdown('<h2 class="section-header"> Record Your Question</h2>', unsafe_allow_html=True) audio_bytes ← audio_recorder(text="", recording_color="#e84545", neutral_color="#6aa84f", icon_size="2x") if audio_bytes then
    try:
    with st.spinner("Transcribing your question..."):
    with st.spinner("Generating response..."):
    malayalam_question, malayalam_response, audio_path ← process_audio_and_respond(audio_bytes, models)
    st.session_state.chat_history.append({ "question": malayalam_question, "response": malayalam_response, "audio_path": audio_path
                                                                          ▷ Audio Recording Section
                                                                          ▷ Update chat history
    })
    if st.session_state.chat_history then
        st.markdown('<h2 class="section-header"> Conversation History</h2>', unsafe_allow_html=True)
        for entry ∈ reversed(st.session_state.chat_history) do with st.container():
            st.markdown('<div class="chat-message">', unsafe_allow_html=True) st.markdown("Your Question:")
            st.write(entry["question"]) st.markdown("Assistant's Response:") st.write(entry["response"]) st.audio(entry["audio_path"], format='audio/mp3')
            st.markdown('</div>', unsafe_allow_html=True)
        end for
    end if end if
end function
if __name__ == "_main_" then

▷ Display Chat History
    main()                                                                ▷ Entry point of the script
end if
```

B. Test Cases

1) Voice Processing (Speech-to-Text)

Objective: Evaluate the accuracy of OpenAI Whisper in converting Malayalam speech to text under different conditions. The accuracy measurements and test cases are detailed in Table 2.

Method: Speech inputs are provided in both controlled and noisy environments to test transcription accuracy.

Expected Result: High accuracy in recognizing and converting spoken Malayalam into text, as observed in the results presented in Table 2.

Table 2: Speech-to-Text Testing

Test Case ID	Objective	Prerequisite	Actual Result	Accuracy(%)	
				Expected	Result
TVP01	Convert Malayalam speech to text	Clear voice input	Accurate transcription	98	95
TVP02	Process noisy speech input	Background noise	Recognizable transcription	90	90

2) Response Generation (NLP)

Objective: Validate the ability of GPT-4o Mini to generate accurate and context- aware responses for user queries. The response accuracy is summarized in Table 3.

Method: Queries of varying complexity are submitted to evaluate relevance and coherence of re- sponses.

Expected Result: High-quality, relevant responses that accurately address user queries.

Table 3: Response Generation Testing

Test Case ID	Objective	Prerequisite	Actual Result	Accuracy(%)	
				Expected	Result
TRG01	Generate accurate college-related answers	User query	Relevant response	98	95
TRG02	Handle complex queries	Multi-part question	Context-aware response	90	90

3) Text-to-Speech (TTS)

Objective: Assess the effectiveness of Silero TTS in converting generated text into clear and natural Malayalam speech. The accuracy is mentioned in the Table 4.

Method: Text responses of varying lengths are converted into speech and analyzed for clarity and fluency.

Expected Result: High-quality, natural-sounding speech output with proper pronunciation.

Table 4: Text-to-speech Testing

Test Case ID	Objective	Prerequisite	Actual Result	Accuracy(%)	
				Expected	Result
TTS01	Convert text response to speech	Generated text	Clear Malayalam speech	98	95
TTS02	Handle long responses	Paragraph-length text	Natural and fluent speech	95	91

4) English to Malayalam Text Conversion

Objective: Test the accuracy and efficiency of GPT-4o Mini in translating English text responses into Malayalam. The accuracy is summarized in the Table 5

Method: Various English responses are translated into Malayalam and compared against human translations for accuracy.

Expected Result: High-fidelity Malayalam translations preserving original context and meaning.

Table 5: English to Malayalam response Testing

Test Case ID	Objective	Prerequisite	Actual Result	Accuracy(%)	
				Expected	Result
TEM01	Convert English text to Malayalam	English text input	Correct Malayalam text	97	97

5) UI and System Testing

Objective: Evaluate the responsiveness and usability of the chatbot's user interface, ensuring correct data display and minimal response delays.

Method: Simulated user interactions are conducted to measure response time and UI accuracy.

Expected Result: Smooth user experience with fast responses and accurate data representation.

Table 6: UI Testing

Test Case ID	Objective	Prerequisite	Actual Result	Expected Result	Accuracy(%)
TUI01	Test UI response time	User input	Response in 10 secs	Immediate response	80
TUI02	Test accuracy of displayed data	Query submission	Correct data retrieval	90%	85

The system's user interface was tested based on response time and accuracy of displayed data. Test cases involved measuring the time taken for responses to be displayed after user input and verifying the correctness of retrieved data. The results, summarized in Table 6 - UI Testing, demonstrate the chatbot's performance in terms of real-time interaction efficiency and data accuracy.

6) Post Integration Testing

Objective: Verify the end-to-end functionality and stability of CampusBuddy after integrating all components. The final results are shown in the Table 7.

Method: The chatbot is tested under real-world conditions to evaluate overall system performance and reliability.

Expected Result: A fully operational chatbot with smooth functionality across all modules.

Table 7: Post Integration Testing

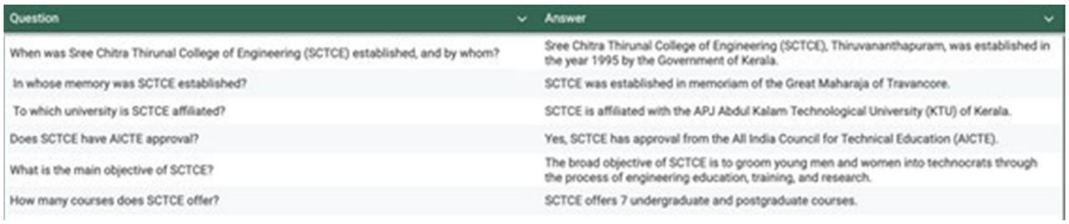
Test Case ID	Objective	Prerequisite	Actual Result	Expected Result	Accuracy(%)
TPI01	Overall system evaluation	Fully integrated system	Pass with minor issues	Fully operational	90

V. RESULTS AND DISCUSSION

This section evaluates CampusBuddy, a Malayalam voice-based chatbot, focusing on its accuracy, response relevance, and overall system performance. The evaluation considers multiple aspects, including Automatic Speech Recognition (ASR) accuracy, response retrieval performance, Text-to- Speech (TTS) quality, and system responsiveness. The impact of different input conditions on system accuracy is also analyzed.

A. Dataset and Vectorized Data Representation

The chatbot's knowledge base consists of structured question-answer pairs extracted from the college website and other relevant sources. The data is preprocessed and stored as vectorized embeddings for efficient retrieval. The sample data has been shown in figure 2. The Vectorized Data Representation



Question	Answer
When was Sree Chitra Thirunal College of Engineering (SCTCE) established, and by whom?	Sree Chitra Thirunal College of Engineering (SCTCE), Thiruvananthapuram, was established in the year 1995 by the Government of Kerala.
In whose memory was SCTCE established?	SCTCE was established in memoriam of the Great Maharaja of Travancore.
To which university is SCTCE affiliated?	SCTCE is affiliated with the APJ Abdul Kalam Technological University (KTU) of Kerala.
Does SCTCE have AICTE approval?	Yes, SCTCE has approval from the All India Council for Technical Education (AICTE).
What is the main objective of SCTCE?	The broad objective of SCTCE is to groom young men and women into technocrats through the process of engineering education, training, and research.
How many courses does SCTCE offer?	SCTCE offers 7 undergraduate and postgraduate courses.

Figure 2: sample data

is as

- Questions and answers are embedded using OpenAI's text-embedding-3-large model.
- The embeddings are stored in ChromaDB, allowing semantic similarity-based retrieval during chatbot interactions.

B. Response Retrieval and Accuracy

The effectiveness of ChromaDB's vector search was measured using Top-1 Accuracy and Mean Average Precision (MAP@5) over a benchmark dataset of 500 queries. The result is shown on the table 8.

Table 8: Response Retrieval Accuracy

Evaluation Metric	Score(%)
Top-1 Accuracy	94
MAP@5	92
MRR	94

$$\text{Top-1 Accuracy} = \frac{\text{Number of times the correct response is ranked 1st}}{\text{Total Number of queries}} \quad (1)$$

$$P(k) = \frac{\text{Number of relevant responses in top-}k}{k} \quad (2)$$

$$AP = \frac{\sum_{k=1}^{5} P(k) \times \text{Relevance}(k)}{\text{Total relevant responses}} \quad (3)$$

$$MAP@5 = \frac{\sum_{i=1}^{N} AP_i}{N} \quad (4)$$

- Equation (1) measures how often the most relevant response appears as the first retrieved result. where

$P(k)$ - precision at k

AP - Average precision N - Number of queries

$MAP@5$ - Mean Average Precision

- Equation (2-4) together measures the quality of the top-5 retrieved results by computing the precision at each relevant response's position and averaging over all queries.
- The chatbot retrieved the most relevant response in the first result 93.3% of the time.
- The high $MAP@5$ score indicates that relevant responses consistently appeared within the top five retrieved results.

C. Language Model Response Quality

GPT-4o Mini's response quality was evaluated using BLEU Score (for text accuracy) and Human Evaluation (for contextual correctness). The result is shown on the Table **??**.

Table 9: Language Model Accuracy

Evaluation Metric	Score
BLEU Score	88%
Human Evaluation	92%

where

$$BLEU = BP \times \exp\left(\sum_{n=1}^{N} \omega_n \log p_n\right) \quad (5)$$

BP is the Brevity Penalty, given by

$$BP = \begin{cases} 1 & c > r \\ e^{\frac{1-r}{c}} & c \le r \end{cases} \quad (6)$$

- Equation (5) measures how closely a model-generated response matches a human reference translation.
- For our result, the BLEU score is 88% which indicates that the chatbot's responses have a high overlap with the reference responses in terms of n-gram matching.

D. System Efficiency and Response Time

The chatbot's response time was analyzed for different input conditions as shown in Table 10.

- The system maintained an average response time of 11 seconds, making interactions nearly real-time.
- Noisy inputs led to increased processing time due to ASR corrections.

Table 10: Response Time

Query Type	Avg. Processing Time (s)
Simple factual query	8
Complex multi-sentence query	11
Noisy speech input	13

E. Challenges and Improvements

1) Speech Recognition in Noisy Environments: While Whisper ASR performed well in ideal con- ditions, background noise occasionally led to minor misinterpretations in Malayalam phonemes. Future improvements will involve noise suppression techniques to enhance ASR performance in real-world settings.
2) Handling Ambiguous Queries: The chatbot sometimes misinterpreted contextually similar queries, requiring further optimization in vector embeddings and response ranking to improve query disam- biguation.
3) Voice Modulation and Naturalness: Although Silero TTS provides intelligible and fluent Malay- alam speech, improvements in intonation, pauses, and prosody modeling would further enhance the naturalness of responses.
4) Scalability and Expansion: The current knowledge base is limited to scraped college-related data. Expanding the dataset and integrating real-time data sources would improve the chatbot's ability to handle diverse and dynamic queries.

VI. CONCLUSION AND FUTURE WORK

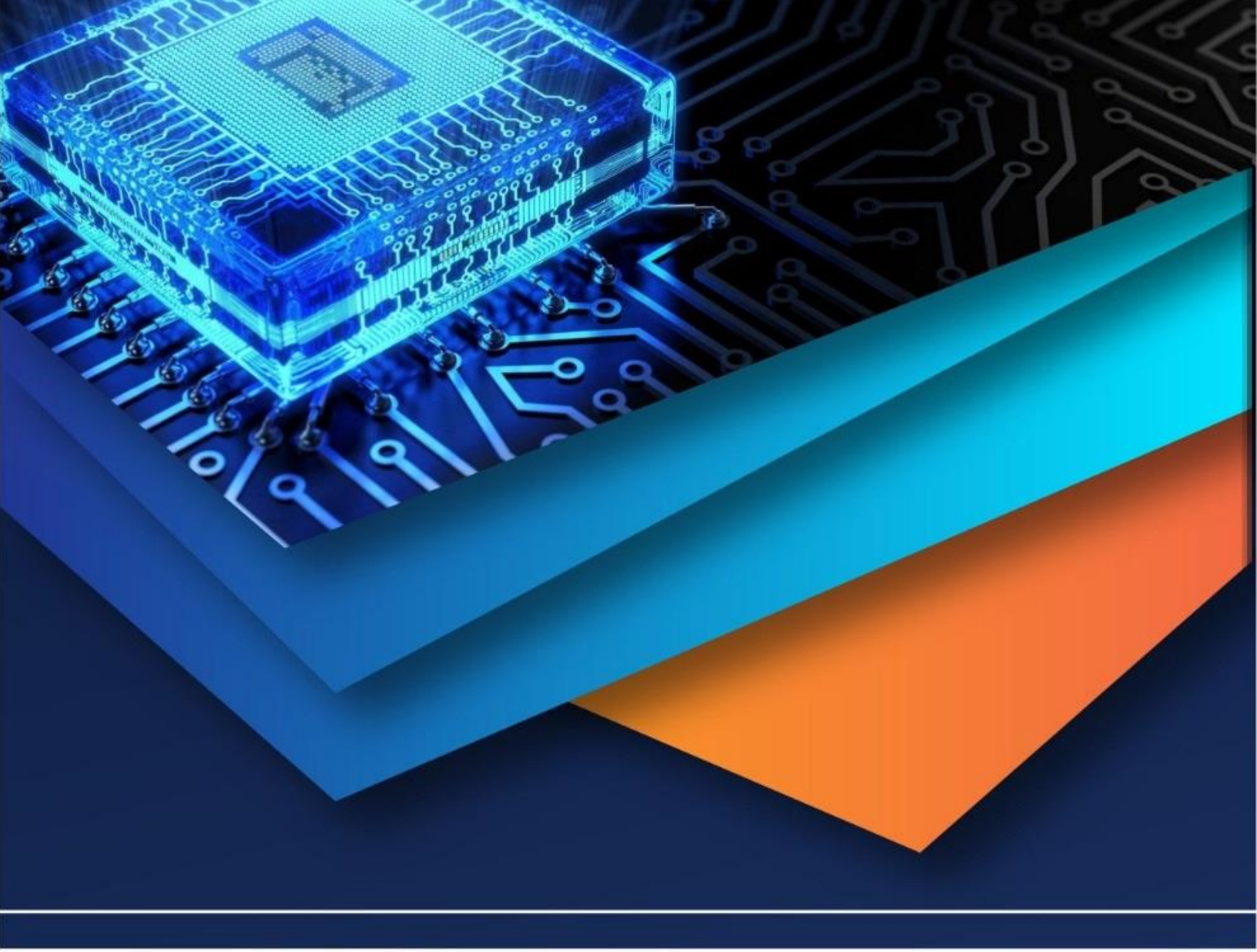
CampusBuddy shows how AI-powered chatbots can help Malayalam speakers by using speech recognition, natural language processing, and text-to-speech technologies together. The system can understand voice queries in Malayalam, get the right information from a structured knowledge base, and give answers in both spoken and written forms. Its modular design makes it easy to add new features in the future, and its focus on making it easy to use in different languages sets the stage for AI applications in education that are open to everyone. Going forward, possible improvements could include better handling of background noise, support for regional dialects, and the ability to connect to real-time data sources. Using the chatbot for more than just education, such as in healthcare or government services, could have an even bigger effect on society. Also, using lighter versions for mobile devices would make it easier to get to in areas with poor internet access. These improvements would make AI help more useful and available to people who don't speak English, making it easier for people to use technology.

Conflict of Interest
The authors declare that they have no conflict of interest.

Funding
This work is not funded.

REFERENCES

[1] Sangeeta Kumari, Zaid Naikwadi, Akshay Akole, and Purushottam Darshankar. Enhancing college chat bot assistant with the help of richer human computer interaction and speech recog- nition. In Proceedings of the International Conference on Electronics and Sustainable Com- munication Systems (ICESC 2020). IEEE Xplore, 2020.
[2] Meet Popat, Aayush Doshi, Yashraj Rai, Deep Vakharia, and Grinal Tuscano. Alexis: A voice- based chatbot using natural language processing. International Journal of Engineering Re- search & Technology (IJERT), 11(04), April 2022.
[3] Nguyen TT, Le AD, Hoang HT, and Nguyen T. Neu-chatbot: Chatbot for admission of national economics university. Computers and Education: Artificial Intelligence, 2:100036, 2021.
[4] G. S. Ramesh, G. Nagaraju, Vemula Harish, and P. Kumaraswamy. Chatbot for college web- site. In International Conference on Advances in Computer Engineering and Communication Systems, Learning and Analytics in Intelligent Systems. Springer Nature, 2021.
[5] Praveen Prasannan, Stephy Joseph, and Rajeev R R. A chatbot in malayalam using hybrid ap- proach. In Proceedings of the 17th International Conference on Natural Language Processing (ICON): System Demonstrations, December 2020.
[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. Advances in Neural Information Processing Systems, 30, 2017.
[7] Xue Hao Zhou, Ming Yang Zhang, Zhizheng Wu, Yi Zhou, and Haizhou Li. Accented text-to- speech synthesis with limited data. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 32, 2024.
[8] Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. Benchmarking natural language understanding services for building conversational agents. In Increasing Naturalness and Flexibility in Spoken Dialogue Interaction. Springer Nature, 2021.
[9] Akshada Phalle, Saniya Kadam, Sakshi Sonphule, and Ila Savant. Ai and web-based interac- tive college enquiry chatbot. International Research Journal of Engineering and Technology (IRJET), 8, November 2021.
[10] Soufyane Ayanouz, Boudhir Anouar Abdelhakim, and Mohammed Benhmed. A smart chatbot architecture based on nlp and machine learning for health care assistance. In Proceedings of the 3rd International Conference on Networking, Information Systems & Security, 2020.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)