Documentation detailing the Arhitecture of a Voice-Only Omani Arabic Mental Health Chatbot

Muhammad Hussain

[1] National University of Computer and Emerging Sciences, Islamabad

Abstract :

This report documents the development of "Sakina," a voice-only conversational AI designed to provide mental health support exclusively in the Omani Arabic dialect. The system integrates real-time speech-to-text, a dual-model response generation core using GPT-4o and Claude Opus 4 for cultural and therapeutic accuracy, and natural-sounding Omani Arabic voice synthesis. The architecture focuses on providing an accessible, empathetic, and authentic user experience.

1 Architecture Design

This document outlines the system architecture for "Sakina," a voice-only Omani Arabic mental health chatbot. The system is designed to be modular, integrating multiple AI services to handle different parts of the conversation pipeline, from voice input to voice output. The architecture prioritizes response quality and cultural appropriateness through a unique dual-model approach while managing performance to ensure a smooth user interaction.

1.1 System Design and Components

The core components are:

1. User Interface (UI): A web interface created with Gradio, featuring a microphone input for the user and an audio output for the bot's response. It also displays the text conversation for clarity.
2. Speech Processing: Microsoft Azure Cognitive Services is used for both Speech-to-Text (STT) and Text-to-Speech (TTS), specifically configured for the Omani Arabic dialect (ar-OM).
3. Response Generation: A primary Large Language Model (GPT-4o) generates the initial bot response.
4. Response Validation & Fallback: A secondary LLM (Claude Opus 4) is used to validate and enhance the primary model's response for cultural and therapeutic accuracy. It also serves as a fallback if the primary model fails.

1.2 Data Flow

1. Voice Capture: The user records their voice using the Gradio microphone component in the browser.
2. Speech-to-Text (STT): The user audio is sent to Azure which processes the audio and returns the transcribed Arabic text.
3. Primary Response Generation:

- The transcribed user text is appended to the conversation history.
- This updated history is sent to OpenAI's GPT-4o alongside the user input text.
- GPT-4o generates a response based on the system prompt and conversation context.

4. Parallel Validation and Speech Synthesis: This is a critical step for latency optimization. As soon as the GPT-4o response is received, two tasks start in parallel.

- The initial GPT-4o response text is sent to Azure TTS to begin generating the voice audio.
- The user's query and GPT-4o's response is sent to Claude Opus 4. Claude checks it against the validation prompt and returns either "good" or a short, enhancing follow-up sentence which flows naturally with the gpt response to further enhance it.

5. Audio Combination and Output:

- If the validation result from Claude is an enhancement, a second call is made to Azure to generate audio for the follow-up sentence.
- The audio from the initial response and the enhancement (if any) are combined into a single audio byte stream. This ensures seamless playback with no pause.
- The final combined audio data is returned to the Gradio audio output component and plays automatically in the user's browser.

6. History Management: Every 6 conversation turns, the system uses GPT-4o to summarize the oldest parts of the dialogue, keeping only the 4 most recent turns in full. This prevents the context from becoming too large, which would slow down future responses.

1.3 Model Integration

- Azure Cognitive Services: It is used for its high-quality Omani Arabic dialect support in both STT and TTS (ar-OM-AyshaNeural voice).
- OpenAI GPT-4o: It serves as the primary "brain" of the chatbot, responsible for generating quick and contextually-aware responses based on the detailed SYSTEM PROMPT.
- Anthropic Claude Opus 4: It plays two key roles:
  - Quality Validator: It acts as a "supervisor," reviewing GPT-4o's response in parallel to ensure it meets cultural and therapeutic standards. This allows for real-time quality control without adding to the initial response latency.
  - In the rare case that the GPT-4o API call fails, the system gracefully degrades by calling Claude to generate a response, ensuring the conversation doesn't stop.

2 Comparative Analysis of the Dual-Model Approach

This section presents a comparative analysis of the dual-model architecture implemented in the chatbot. The objective is to evaluate the effectiveness of this design against simpler, single-model

alternatives. The evaluation focuses on key metrics including dialect authenticity, cultural appropriateness, therapeutic quality, and system latency.

2.1 Evaluation Methodology

To provide a fair comparison, three distinct system architectures were evaluated using a consistent set of criteria.

2.2 Models Compared

- GPT-4o Only: A baseline system where OpenAI's GPT-4o handles all response generation. This represents a standard, high-performance single-model approach.
- Claude Opus 4 Only: A second baseline where Anthropic's Claude Opus 4 handles all response generation. This model is known for its deep contextual understanding and cautiousness.
- Dual-Model - Implemented System: The final production architecture. GPT-4o generates the initial response, which is immediately sent for text-to-speech synthesis. In parallel, Claude Opus 4 validates this response, providing a corrective or enhancing follow-up if necessary.

2.3 Evaluation Criteria

The evaluation was based on a combination of qualitative and quantitative metrics:

2.3.1 Qualitative Criteria:

- Dialect Authenticity: How closely the response matched the natural Omani Arabic dialect.
- Cultural Appropriateness: The ability to navigate sensitive topics like family, religion, and mental health stigma respectfully.
- Therapeutic Quality: The empathetic and supportive nature of the response.

2.3.2 Quantitative Criteria:

- Average End-to-End Latency: Measured in seconds from the end of user speech to the start of the bot's spoken response.

2.4 Overall Assessment

2.4.1 Qualitative Comparison of Model Responses

- GPT-4o Only: Fast and functionally correct, but often lacked the specific cultural depth required. Prone to generic advice.
- Claude Opus 4 Only: Culturally and therapeutically superior with a deep understanding of nuance. However, initial response generation was slower compared to Gpt-4o.
- Dual-Model: Achieved the highest overall quality by leveraging GPT-4o for speed and Claude for depth, creating consistently appropriate and supportive responses.

2.4.2 Quantitative Performance Benchmarks (Average Latency)

- GPT-4o Only: 4 – 6 seconds
- Claude Opus 4 Only: 5-8 seconds
- Dual Model: 6-10 seconds

2.5 Discussion

GPT-4o Only offered the lowest latency, however its tendency toward generic advice would likely fail to build rapport with users facing specific cultural challenges.
Claude Opus 4 Only produced significantly higher-quality responses but at the cost of increased and less consistent latency compared to GPT.
Dual-Model successfully mitigates the weaknesses of the single-model approaches. It harnesses the speed of GPT-4o for the initial part of the response while using Claude's deeper reasoning as a parallel validation step.
The dual-model architecture, despite its higher average latency, is the superior design. The additional 2-4 seconds per turn is a necessary and justified investment for the dramatic improvement in response quality, cultural authenticity, and overall user safety. The system is not just a technical implementation but a carefully considered solution designed to meet the unique needs of its target audience.

3 Cultural Adaptation Implementation Guide:

The primary challenge in developing the chatbot was not just technical implementation, but ensuring deep and authentic cultural adaptation. Standard AI models, primarily trained on Western data and formal Arabic, are ill-equipped to handle the specific linguistic and cultural nuances of the Omani context.
The approach used to transform a general-purpose AI into a culturally-aware Omani mental health companion is centered on a detailed System Prompt to the model that defines the AI's persona, boundaries, and communication style. This prompt strictly instructs the models to communicate exclusively in a natural, empathetic Omani Arabic dialect, keeping responses short and conversational. It embeds key cultural values by directing the AI to offer its support around family, religion, and community norms, and to suggest locally-resonant coping strategies like prayer and consulting with elders. Crucially, it establishes clear safety boundaries, forbidding clinical advice. This prompt-based framework is reinforced by the dual-model architecture, where Claude 4 Opus validates GPT-4o's adherence to these cultural rules in real-time, ensuring a consistently authentic and appropriate user experience.

4 Crisis Intervention and Escalation Procedures

User safety is the highest priority for the "Sakina" chatbot. A dedicated protocol has been implemented to manage crisis situations, such as when a user expresses thoughts of self-harm or is in immediate distress. The system's design philosophy is to leverage the advanced reasoning capabilities of the primary language model (GPT-4o) to handle crisis detection and response as an integrated part of the conversation, thereby avoiding additional latency from separate API calls for user query sentiment classification.

4.1 Crisis Detection Mechanism

The system relies on the LLM's inherent ability to understand context and sentiment. The LLM is instructed explicitly to watch for key indicators of a crisis, including but not limited to:

- Mentions of suicide or self-harm.
- Expressions of extreme hopelessness or a desire to die.
- Any indication of imminent danger to the user or others.

4.2 Response and Escalation Procedure

Upon detecting any of these indicators, the SYSTEM PROMPT commands the AI to immediately abandon its standard conversational style and execute a strict, multi-step safety protocol:

- Immediate Empathetic Acknowledgment: The first priority is to show deep empathy and validate the user's feelings, reassuring them that their life is valuable.
- Urge for Human Connection: The bot is instructed to immediately ask the user to speak with a trusted individual, reinforcing the need for human support.
- Provide Actionable Resources: The model is required to provide a list of pre-defined, local Omani emergency and support contact numbers. This includes the national psychological helpline (+968 24 607 555), emergency services (9999), and a major hospital contact (Sultan Qaboos Hospital: 24144625).
- Promote a Safe Environment: The prompt guides the AI to advise the user to remain in a safe location, preferably with trusted people.
- Confirm Safety: The AI is instructed to not end the conversation until it has made a reasonable attempt to confirm the user is taking steps toward safety.

5 Performance Benchmarks

This section details the final performance metrics of the chatbot, focusing on latency, accuracy, and scalability. The benchmarks were established through a series of 12 live test conversations conducted using the optimized application code and Gradio interface.

5.1 Latency

End-to-end latency is the most critical performance metric for a real-time conversational AI significantly affecting user experience.

Table 1
Average Latency Breakdown per Conversational Turn

Step #	Process	Average Latency (s)	Notes
1	**Speech-to-Text (Azure STT)**	3.86	Consistent performance for transcribing user's voice.
2	**Primary Response (GPT-4o)**	2.15	Fast and efficient "thinking" time for the initial response.
3	**Parallel Validation & TTS**	5.15	Limited by the Claude validation call, now the slowest task in this block.
3a.	Azure TTS (Initial)	(2.59)	*(Component of Step 3)*
3b.	Claude Validation	(5.15)	*(Component of Step 3) - The primary driver of this block's latency.*
4	**Enhancement TTS (Conditional)**	2.33	Occurs frequently (67% of test turns), adding a small delay for quality.
5	**History Summarization (Infrequent)**	4.42	Occurs every 6 turns, adding a one-time delay to the end of the turn.
-	**Total End-to-End Latency**	**13.21**	The average user-perceived delay.

5.2 Accuracy

Accuracy in this context is measured qualitatively, focusing on the system's ability to meet the user's intent and adhere to its defined cultural and therapeutic persona.
Transcription Accuracy (STT): The Azure ar-OM model demonstrated high accuracy, correctly transcribing Omani dialect, including common slang and code-switched (Arabic-English) phrases.
Response Accuracy (LLM): This refers to the quality and appropriateness of the response. Thanks to the dual-model validation system, the rate of culturally tone-deaf or therapeutically unhelpful responses was observed to be very low.

5.3 Scalibility

Dependency on Cloud Services: The system's scalability is fundamentally tied to the scalability of the cloud services it relies on (OpenAI, Anthropic, and Microsoft Azure). These are commercial, auto-scaling services designed to handle massive global traffic.

6 Production Deployment and Maintenance Guide

This section provides instructions for deploying the AI Mental Health Companion into a production environment. The application is a Python-based Gradio web application that integrates with three critical external cloud services:
OpenAI API (GPT-4o): For primary chat responses.
Anthropic API (Claude Opus 4): For response validation, enhancement, and as a fallback model.
Azure Cognitive Services (Speech-to-Text & Text-to-Speech): For all audio interaction.

6.1 Current Status

This project has been developed and tested locally using Gradio's development interface. The following guide outlines the requirements and steps necessary for production deployment.

6.2 Prerequisites

6.2.1 Infrastructure

- A dedicated server or Virtual Machine (VM).
- Root or sudo access to the server.
- A domain name (e.g., sakina.yourdomain.com) pointing to your server's public IP address.

6.2.2 Software

- Python 3.9+ and pip.
- Nginx (or another reverse proxy) for secure web traffic (HTTPS).
- Docker and Docker Compose (Recommended) for containerization, which simplifies dependency management and deployment.

6.2.3 API Keys and Credentials You must have active accounts and API keys for the following services:

- OpenAI: OPENAI_API_KEY
- Anthropic: ANTHROPIC_API_KEY
- Azure:
  - AZURE_SPEECH_KEY
  - AZURE_SPEECH_REGION

6.3 Production Setup Instructions

This setup uses Docker for containerization and Nginx as a reverse proxy. This is a robust and standard pattern for deploying web applications.

- Create a Dockerfile
- Create a docker-compose.yml file
- Build and Run the Container using command docker-compose up -d –build
- Configure Nginx for your Domain using sudo nano /etc/nginx/sites-available/sakina
  - This step will route traffic from your public domain (e.g., https://sakina.yourdomain.com) to the Gradio application running inside the Docker container on port 7860.
- Obtain SSL Certificate with Certbot using sudo certbot –nginx -d sakina.yourdomain.com
  - application will be live and accessible at https://sakina.yourdomain.com.

6.4 Maintenance and Monitoring

- Viewing Container Logs: Use the docker logs command to see the latency metrics printed in the code.
- API Usage: Regularly check your usage dashboards for OpenAI, Anthropic and Azure.

7 Future Roadmap

This roadmap outlines key areas for development to evolve "Sakina" from a functional prototype into a robust, scalable, and more intelligent mental health companion.

7.1 Implement True User Session Management

Problem:
INITIAL_HISTORY is a global variable, meaning all simultaneous users would share the same conversation. This is the single biggest barrier to scaling.
Recommendation:
Refactor the code to ensure that every user's conversation history, summary, and state are completely isolated.

7.2 Containerize for Production Deployment

Problem:
Running a single python script is not stable for production. It doesn't handle crashes or high traffic well.
Recommendation:
Use Docker to package the application and its dependencies into a container. This ensures consistency and simplifies deployment.

7.3 Introduce User Personalization & Memory

Problem:
The bot has no memory of users between sessions.
Recommendation:
Implement a simple database to store user profiles. Assign a unique ID to each user, allowing "Sakina" to remember past conversations and provide more continuous, personalized support.

7.4 Implement Response Streaming for Reduced Latency

Problem:
The user waits for the entire response to be generated and synthesized, creating noticeable lag.
Recommendation:
Start playing the audio for the first sentence using streaming while the next sentence is still being synthesized. This dramatically reduces perceived wait time.
Update the API calls to stream the text response from the LLM, allowing the TTS process to begin even before the full response is received.

8 Conclusion

In conclusion, "Sakina" demonstrates a successful integration of advanced AI technologies—including large language models and speech services—to create a culturally-aware mental health companion specifically tailored for the Omani dialect. The current architecture, featuring a real-time validation and enhancement loop, ensures high-quality, empathetic, and safe interactions. The project not only provides a functional prototype but also establishes a clear and ambitious roadmap for future development. This includes scaling for multi-user support and introducing deeper personalization that can offer even more effective and accessible mental health support over time.