Solution Document for GenAI VoiceBot

Introduction

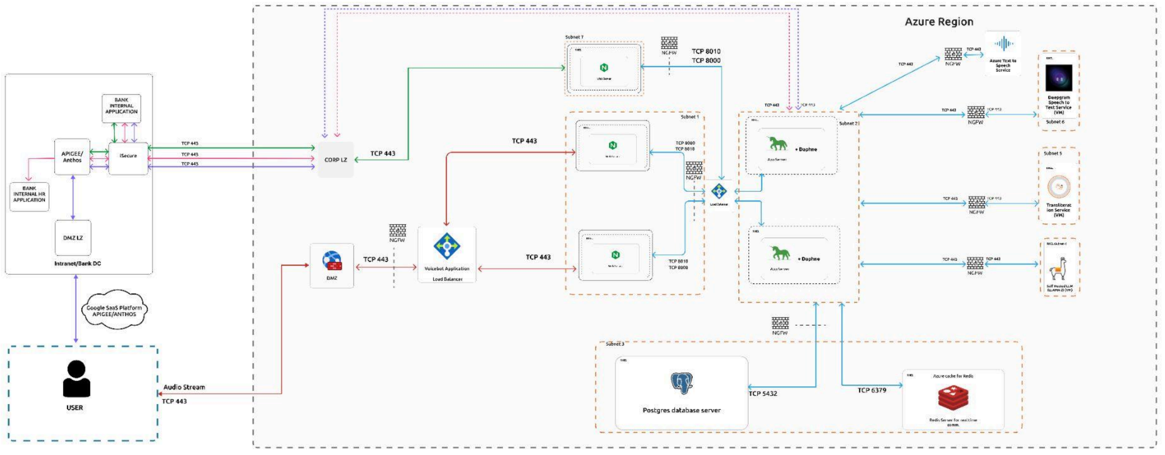
This document is intended to describe Exotel's voice bot architecture, deployment model, and data flow comprehensively. The document is structured as follows:

- The deployment architecture section: Covers deployment aspects.
- The data flow section - Illustrates the flow of data in the context of the voice bot application.
- The software and tech-stack section - Describes the tech-stack requirements of deploying the voice bot.

Deployment Architecture

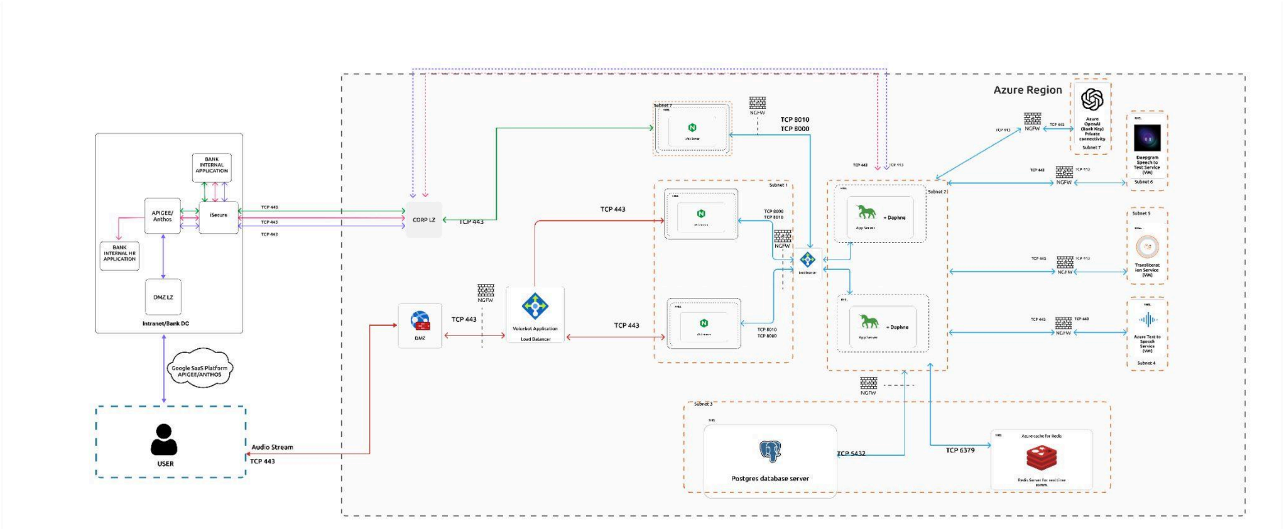
1. Deployed on Azure Cloud using Azure LLM

The following diagram covers the deployment view of the voicebot application deployed in the private cloud environment.



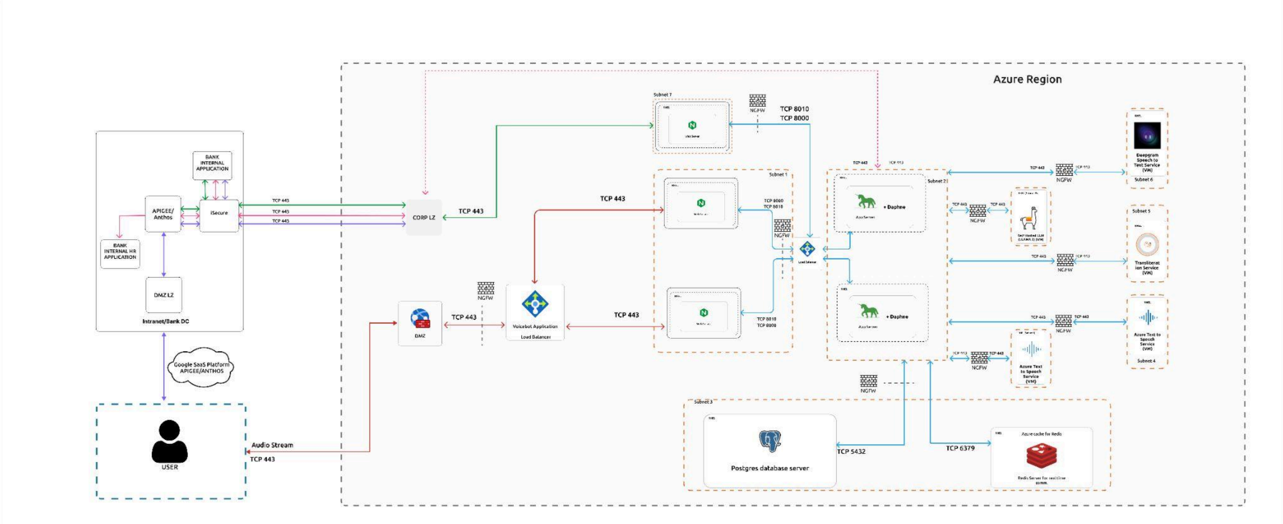
Voicebot Architecture Diagram with Self Hosted LLM (LLAMA 2) and Azure TTS as PAAS

2. Deployed on Azure Cloud using Hosted LLM



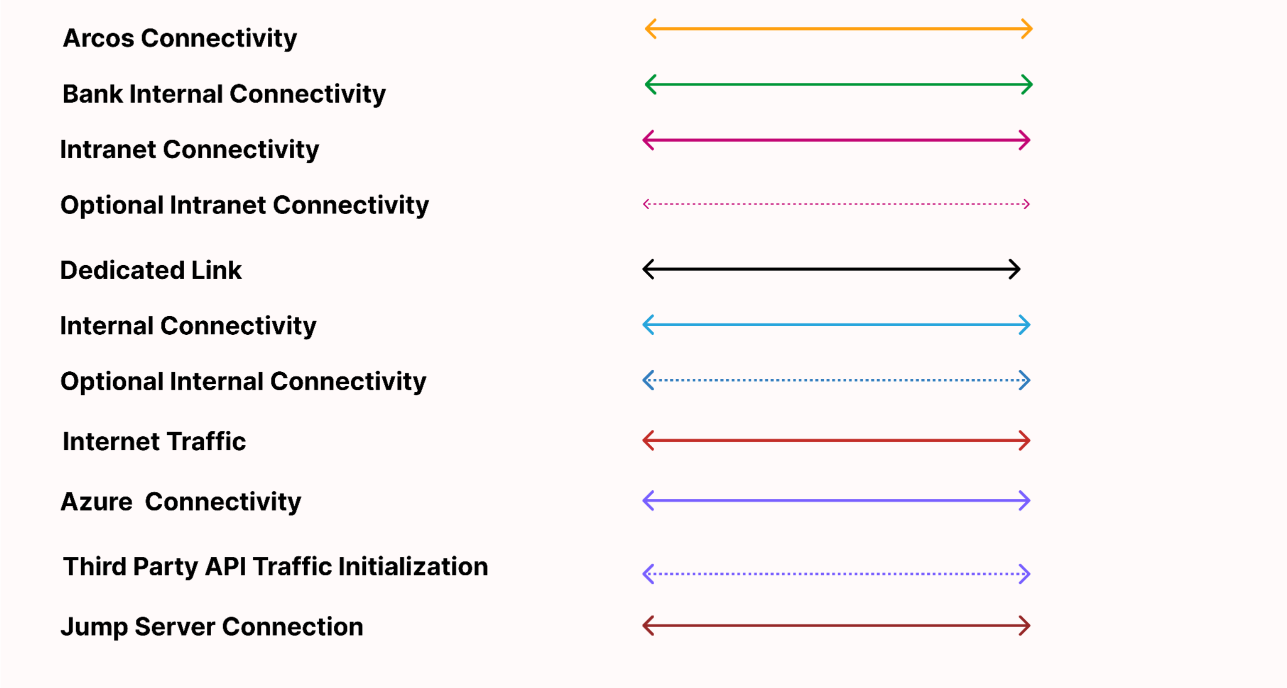
Voicebot Architecture Diagram OpenAI and On-Prem Azure TTS

3. Deployed on Azure Cloud using Hosted LLM and On-Prem Azure TTS



Voicebot Architecture Diagram with Self Hosted LLM (LLAMA 2) and On-Prem Azure TTS

Architecture Legend Diagram



Data flow details

Figure 2 depicted below provides a high-level view of the data flow for the voice bot application.

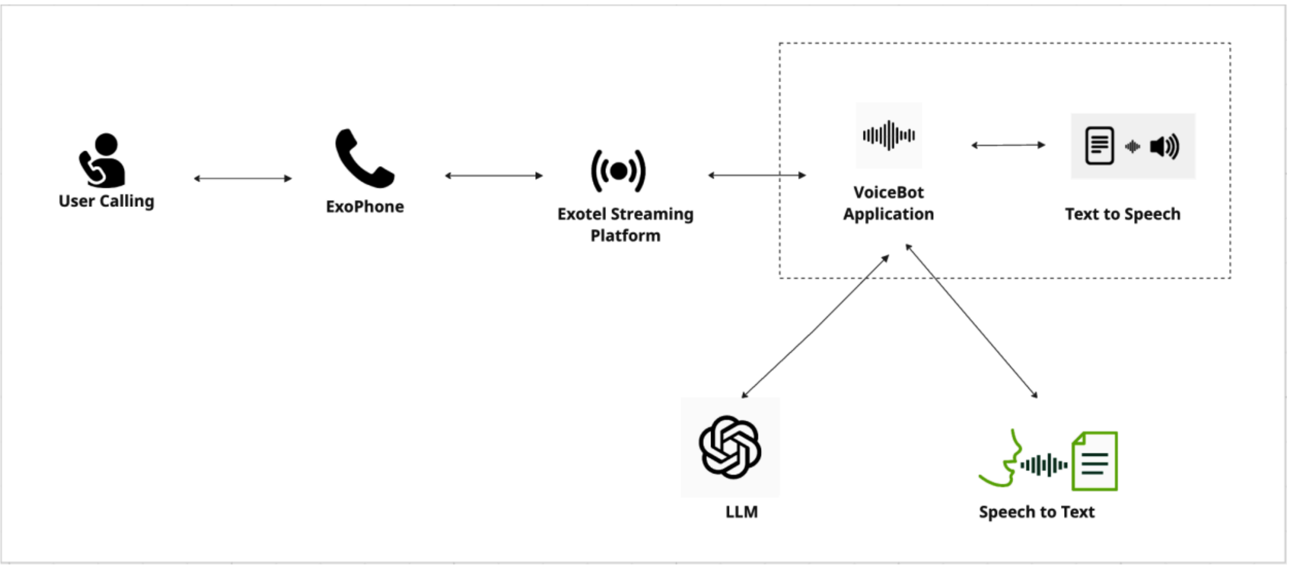


Figure 2: High-level data flow diagram

Sequence Flow

This section provides the end-to-end sequence flow involved in the voicebot.

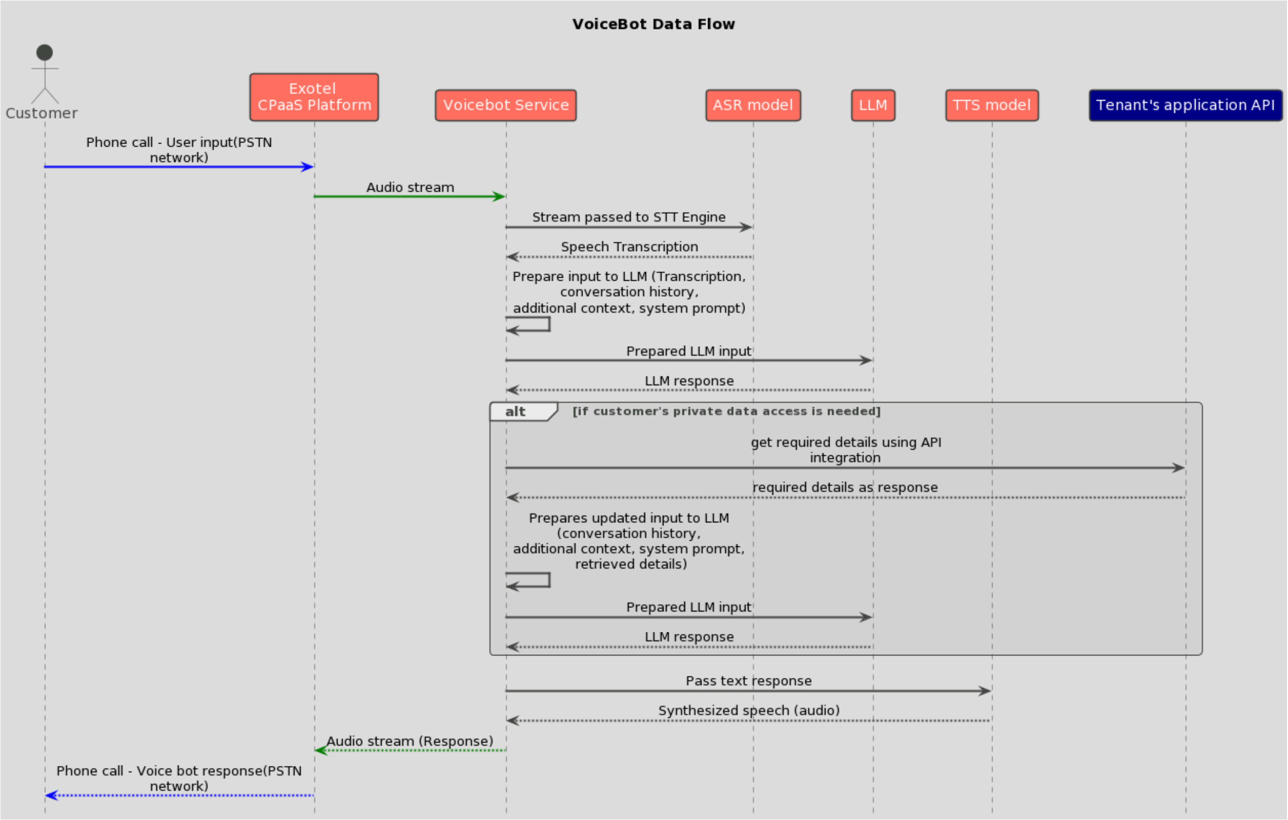

VoiceBot Data Flow
Customer
Exotel CPaaS Platform
Voicebot Service
ASR model
LLM
TTS model
Tenant's application API
Phone call - User input(PSTN network)
Audio stream
Stream passed to STT Engine
Speech Transcription
Prepare input to LLM (Transcription, conversation history, additional context, system prompt)
Prepared LLM input
LLM response
alt
[if customer's private data access is needed]
get required details using API integration
required details as response
Prepares updated input to LLM (conversation history, additional context, system prompt, retrieved details)
Prepared LLM input
LLM response
Pass text response
Synthesized speech (audio)
Audio stream (Response)
Phone call - Voice bot response(PSTN network)

Figure 3: Sequence flow diagram

1. **Call Initiation:**
    a. In inbound calls, the customer dials into the Exophone number.
    b. In outbound calls, the Exophone initiates a call to the customer's number based on a trigger from the Voicebot application.
2. **Call Handling by ExoPhone:**
    a. The Exophone connects the call and initiates a bidirectional audio stream with the Exotel Streaming Platform.
3. **Bidirectional Stream Establishment:**
    a. Exotel Streaming Platform establishes a bidirectional stream with the Voicebot application.
4. **Voicebot Application Processing:**

   a. The Voicebot application receives the audio stream.
   b. The audio is continuously passed from the Voicebot application to the ASR engine in real-time.

5. **Speech-to-Text Transcription:**
   a. The ASR engine transcribes the speech into text.
   b. The ASR engine outputs the transcribed text back to the Voicebot application.
      i. If transcription fails, the voicebot application requests clarification or to repeat.
6. **Query Processing by LLM:**
   a. The Voicebot application passes the transcribed text to the Large Language Model (LLM) along with the required prompt and conversation history required
   b. The LLM processes the query and formulates a response.
      i. There are cases where the voicebot application requires details from the Exotel tenant's application to respond to the customer. In these cases, the voice bot application fetches necessary details by calling the required API of the tenant's application
7. **Converting Text Response to Speech:**
   a. The Voicebot application receives the text response from LLM
   b. The text response is passed to the TTS engine.
   c. The TTS engine converts the text-to-speech audio
      i. Optionally, transliteration is performed to support Indic languages
8. **Streaming the Response to the Customer:**
   a. The speech audio stream is passed back to the Exotel Streaming Platform
   b. Exotel Streaming Platform sends the audio stream to the Exophone
   c. The customer hears the response via the Exophone
9. **Call Disposition:**
   a. The interaction continues until the query is resolved or the call is terminated.
   b. On-call completion, the Voicebot application logs the call detail records and, if necessary, schedules follow-up tasks.

Infra and Hosting

The architecture outlined above constitutes the framework for our project, while the technical stack detailed below encompasses the infrastructure and hosting solutions employed to meet the necessary capacity requirements.

1. **Exophone:**
    a. Virtual Numbers from the Exotel CPaaS platform.
2. **Exotel Streaming Platform:**
    a. CPAAS Platform: https://support.exotel.com/support/solutions/articles/3000108630-working-with-the-stream-and-voicebot-applet
3. **Voicebot Application:**
    a. Web Server: Nginx 1.18.0: Nginx (pronounced "engine-x") is a powerful and popular open-source web server software. It is known for its high performance, scalability, and efficiency in handling concurrent connections. Nginx is often used as a reverse proxy server, load balancer, and HTTP cache, making it versatile for various web hosting needs. Nginx 1.18.0 is installed on a Linux operating system
    b. App Server: Gunicorn 20.0.4: Gunicorn, short for "Green Unicorn," is a popular open-source WSGI (Web Server Gateway Interface) HTTP server for running Python web applications. It is designed to be a lightweight yet powerful server, providing a reliable way to serve web applications. Gunicorn is often used in conjunction with web frameworks like Django or Flask. With its ability to handle multiple concurrent requests and simplicity in deployment, Gunicorn is a preferred choice for hosting Python web applications. Gunicorn 20.0.4 is installed on a Linux operating system to enhance the performance and reliability of web applications.
    c. Socket App Server: Daphne 3.0.0: Daphne is an open-source, ASGI (Asynchronous Server Gateway Interface) server for deploying and serving WebSocket applications in Python. It is designed to work seamlessly with ASGI-compatible frameworks like Django Channels. Daphne provides support for handling asynchronous communication, making it well-suited for real-time applications and chat systems. With version 3.0.0, Daphne introduces improvements and features to enhance the performance and reliability of WebSocket applications. Daphne 3.0.0 is installed on a Linux operating system to manage WebSocket connections and deliver responsive real-time experiences efficiently.
    d. Redis Server: Redis 5.0
    e. Database Server: Postgres 12.4

   f. Programming Language: Python 3.10 (Django Framework): Python 3.10 is the programming language version, and Django is a high-level web framework that simplifies web development in Python, providing tools for building web applications efficiently. Django Framework is dependent on Python as its core programming language, leveraging its features and functionalities to facilitate rapid and maintainable web development.
   g. Transliteration Flask Server: A Flask-based in-house module developed to transliterate Roman Hindi to Devanagari Hindi to achieve human-like voice modulation from TTS service. The module uses Python programming language and Flask framework which is a low-level framework based on Python for small applications.

4. **ASR:**
   a. Programming Language: Python 3.10 (Flask Framework)
5. **LLM:**
   a. Azure GPT (Open AI GPT 3.5 Turbo):
   b. LLama2
6. **Text To Speech:**
   a. Azure TTS
7. **Transliteration Service:**
   a. Programming Language: Python 3.10 (Flask Framework)

Features Available

1. Multilingual bot.
2. Interruption handling of the bot.
3. Pause the bot mid-conversation.
4. Intent Extraction: Perform Actions mid-voicebot journey based on user data fetched via API.
5. Push data via API mid-conversation.
6. Interact with Exotel tenant’s Applications to authenticate/validate users and authorise user-specific actions (Example: Integrating with HRMS to provide information to only authorized users).
7. Feedback collection via required other channels such as WhatsApp messages.
8. Flexibility to update Prompts on the fly.
9. Perform actions based on transcription like summarisation (Can be customized).
10. Ability to transfer to a real agent as a fallback or feature.

Hardening Process:

1. We will need root access on the OS to install the following:
   a. Nginx
   b. Gunicorn
   c. Daphne
   d. Python
   e. Redis
   f. Postgresql
   g. Access to read/write code
2. Internet connectivity to install OS packages and Python packages and libraries.
3. Internal Connectivity between App Server/Transliteration Server/DB server for necessary ports according to architecture.