Conversational Intelligence Assistant – Voice Input, Text Output

Executive Summary

Our Conversational Intelligence Assistant is built to humanize machine interactions, leveraging voice recognition and audio processing that is resilient to noise and developing responses that adapt to the user. The assistant hears a user's voice query in the world, understands natural speech, and responds with clear and contextual responses, just like a skilled sales rep or support executive.

Our innovations include:

- **Voice to text intelligence** via Whisper (small/medium/large) to accurately transcribe
- **Noise reduction preprocessing** (RNNoise, WebRTC VAD) for voice clarity in a noisy environment
- **Context-driven learning** from audio, FAQ, and domain data using RAG models
- **Ambiguity resolution and fallback** using real-time confidence scoring, so no questions are left unasked.

This system is built for scale, easily adaptable to retail, enterprise support, and customer engagement platforms. It will enable quicker product discovery, smoother support experience, and learn intelligently from human conversations over time.

Problem Understanding and Scope

Conversational interaction is instinctive to users but very difficult for machines to replicate--and especially difficult in unpredictable environments like retail stores or customer service centers. It is challenging for machines to:

- Decode unclear voice inputs or noisy environments
- Understand vague, multi-intent, or ambiguous requests
- Maintain the same high quality, consistency, and user contextual awareness without any human intervention.

Target user include customers seeking product information, sales agents who need quick back end support, or website users with accessibility needs.

Scope:

- Build a real-time voice-to-text process that minimizes latency
- Ensure relevance of response and clarity of response through retrieval and context learning
- Minimize user frustration by providing backoffs in a graceful manner
- Allow machines to continuously self-learn based on past interactions as well as updates in product/FAQ information

Knowledge Strategy

To allow intelligible and domain aware output and response, our assistant will learn and adapt through:

A. Audio Learning and Preprocessing

- **Noise Suppression:** Remove noise from the raw voice input (the audio) using RNNoise, and WebRTC Voice Activity Detection (VAD).
- **Whisper Model:** Convert the cleaned speech to text using Whisper model (small, medium, large), depending on storage and speed requirements.

B. Post-Transcription Correction

- **DistilBERT** fixes domain-related transcription errors (for example, converting "foam" to "phone" in the context of an electronics store).
- **Contraction Expansion and Grammar Fixing** included for informal inputs.

C. Data Sources

- **FAQs:** Structured question-answer pairs in a FAQ, indexed using Elastic Search capabilities.
- **Web Knowledge:** Product data, descriptions, and reviews scraped, and indexed and encoded semantically.
- **Historical Chat Logs:** Used in reinforcement learning so future responses can be fine-tuned.

D. Reinforcement Learning and Safety

- **KL-Divergence based optimization** will ensure the assistant is evolving to allow the agent better align with the human preferred output over time.
- **Proximal Policy Optimization (PPO)** will be used to ensure any new policy increments involving learning, occur in a safe manner and are relative incremental. This will reduce the likelihood of reward hacking i.e., when the assistant might find a useful shortcut, to exploit the reward associated with that signal, rather than providing user value.

Conversation Design

A good voice assistant must not only be able hear — it must be able to understand.

A. Voice Input Handling

- Whisper transforms speech to text after performing preprocessing
- Domain-specific vocabulary is adjusted based on fine-tuning
- Low-quality input is cleaned and filtered to remove noises

B. Understanding Ambiguous Queries

- Ambiguity Detection via **word-overlap scoring** (i.e., if words matched are <20%, then it is likely out of context)
- Rasa NLU tracks **intent confidence** (< 0.7 will trigger a fallback)

C. Response Generation

T5 transformer generates responsive, human-like replies by producing directly from:

- Chat history context
- FAQ content retrieval
- Knowledge embeddings

D. Fallback Triggers

When the assistant is lacking clarity, one out of two fallback trigger is taken:

- Make the user clarify the information being requested ("Could you specify which product you meant?").
- Escalate to a human operator (optional phase).

E. Human Alignment - The HHH Framework

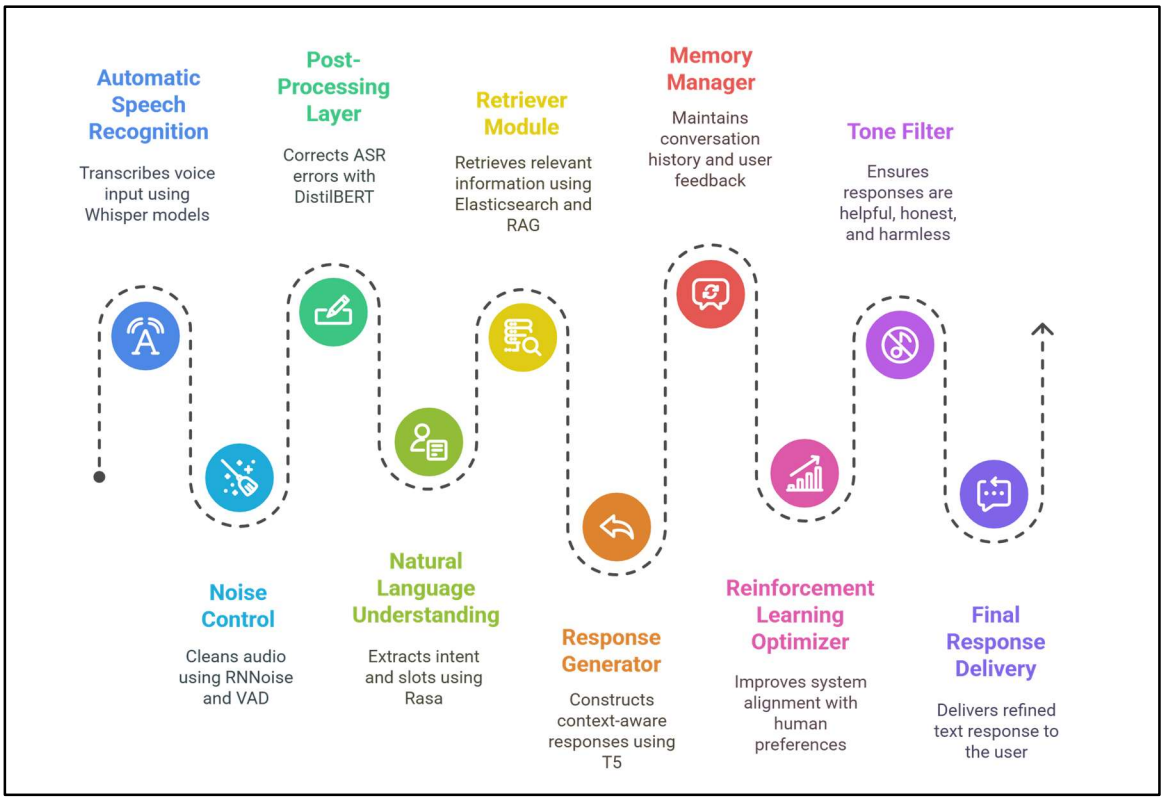
All conversational outputs will be filtered using the **Helpful, Honest, Harmless (HHH)** framework:

- **Helpful:** This input will need to assist the user make constructive progress toward their goal.
- **Honest:** To ensure that hallucination was prevented, and no misleading or factually incorrect responses were provided.

- **Harmless:** avoid producing biased, unsafe or inappropriate reponses.

Strategizing human alignment explicitly advances trust, equity and responsible AI use with any mode of deployment.

Technical Architecture



- Our assistant utilizes a modular voice-to-text pipeline for practical use. It starts with **Noise Control** which uses **RNNoise** and **WebRTC VAD** to clean up the audio input. Then Whisper ASR converts voice into text.

- The **Post-Processing layer** uses **DistilBERT** post-processor which fixes known ASR errors, and expands contractions. Next, **Rasa NLU (Natural Language Understanding)** determines the user's primary intention and extracts the necessary information. If user confidence is low a fallback is triggered.

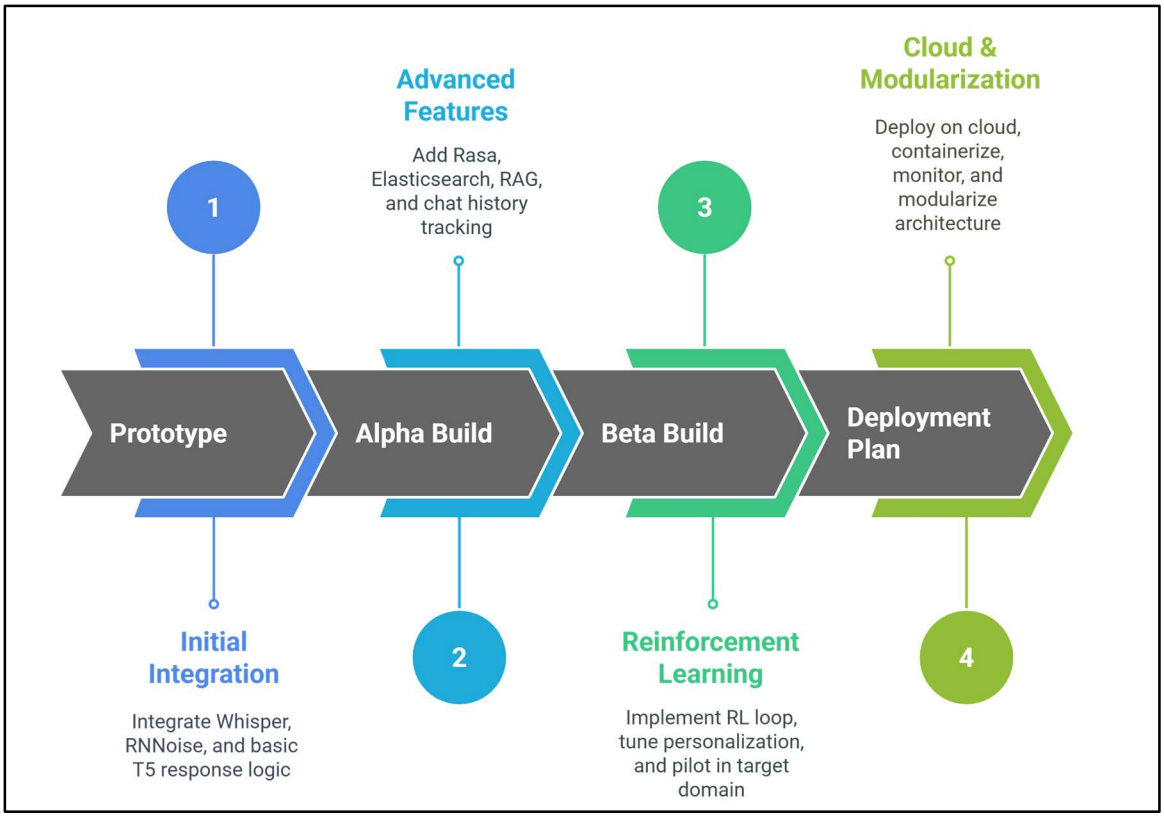
- The **Retriever Module** uses **Elasticsearch + RAG** to retrieve all relevant information from the FAQs or the web. Then the **T5-based Response Generator**, which produces fluent, coherent and contextual responses to what the user asked.

- A **Memory Manager** stores their previous conversation to allow for personalized / context-based continuity. The **Reinforcement Learning Optimizer** (using KL-Divergence

& PPO) keeps the tone, domain goals, and requirements in line, preventing any reward hacking in subsequent conversation.

- Prior to delivery, the response has passed through a **Tone Filter** which applies the **HHH framework: Helpful**, **Honest**, and **Harmless**. The response to the user is then delivered as a clear, polished response.

Implementation Plan



A. Phase 1: Prototype

- **Integrate Whisper + RNNoise pipeline**
  Set up the core voice input system by integrating noise suppression (RNNoise) with Whisper ASR for accurate transcription.

- **Develop T5 response logic over static FAQ database**
  Use a T5 model to generate responses from the preloaded FAQ dataset to simulate early conversation capability.

- **Deploy on local or testing cloud platform.**
  Deploy the prototype locally or on a test cloud environment to test the pipeline end-to-end.

B. Phase 2: Alpha Build

- **Integrate Rasa for fallback and multi-intent detection**
  Enable intent recognition and slot recognition along with fallback prompts for unclear or low-confidence queries.

- **Connect Elasticsearch + RAG for dynamic answer fetching**
  Use semantic search and retrieval-augmented generation to dynamically fetch and respond using live content.
- **Add chat history tracking**
  Store past user interactions to enable memory, personalization, and multi-turn conversation.

C. Phase 3: Beta Build

- **Add reinforcement learning loop with KL-Divergence**
  Refine sample responses using user feedback through reinforcement learning, while avoiding drift using KL regularization.

- **Tune fallback logic and personalization capabilities**
  Enhance fallback prompts and enhance the responses based on the previous interactions or user behavior.

- **Begin real world pilot in intended domain**
  Assess performance of the assistant with real users in intended environment (e.g., in retail, support) and collect feedback regarding usage performance.

D. Phase 4: Deployment Plan

- **Cloud Hosting: Vercel or AWS Lambda**
  Implement cost-effective and globally available on-demand serverless infrastructure that can scale with usage.

- **Containers: Dockerized microservices**
  Package each module (ASR, NLP, RAG) into independent containers for modularity and maintainability.

- **Monitoring: Log + feedback loop for improvement**
  Implement logging, analytics, and error tracking to assess system performance, and feedback the learning loops.

- **Modularity: Plug-and-play architecture for ASR/NLP/RAG endpoints**
  Build the system in such a way that allows for any of the core components to be swapped out or upgraded without disturbing the pipeline.

Innovation Highlights

Team Information

- **Meet Jethwa -** *Project Manager & NLP Lead*
  Supervises project execution, coordinates across all modules and leads ASR-NLP integration. Jethwa is also primarily responsible for the Whisper fine-tuning, T5 response logic, and matching the conversational model to the domain needs.

- **Shubham Vishwakarma -** *Technical Architect & Backend Lead*
  Designs and builds the back end of the microservices architecture including, Reinforcement learning, Elasticsearch integration, and RAG pipelines. He is also the primary responsible person to the deployment set up with Docker, Vercel/AWS.

- **Dhiraj Nair - Knowledge & Retrieval Engineer**
  Responsible for data pipelines to ingest and index FAQs - product databases and web content; configuration of Elasticsearch and the embedding-optimal query strategies that would allow for accurate retrieval of knowledge.

- **Jeet Nakrani - Frontend Developer & UX Engineer**
  Responsible for the voice-enabled user interface and display for the conversation, and ensuring real-time interactions and accessibility. He is also partially responsible for integrating the ASR outputs and fallback prompts into the frontend flow.