# **Real-Time AI Sales Intelligence and Sentiment-Driven Deal Negotiation Assistant**

# Project Documentation

# Project Overview

The Real-Time AI Sales Intelligence Tool is designed to assist sales teams during live negotiations by analyzing buyer behavior and providing real-time insights. It enhances decision-making through sentiment analysis, intent detection, dynamic deal recommendations, and post-call insights.

# Table of Contents

1. Introduction
2. Software and Hardware Requirements
3. SDLC Steps
4. Implementation Details
5. API Integrations
6. Testing and Evaluation
7. Future Enhancements
8. References

# **Introduction**

# This project is designed to assist sellers and salespersons in enhancing their interactions with customers during calls. By analysing the context of the conversation, the system helps sellers understand the buyer’s sentiment and intent, recommend products tailored to their needs, and generate effective negotiation strategies to close deals. The project integrates advanced AI capabilities to ensure sales representatives can respond dynamically and effectively, ultimately improving customer satisfaction and sales outcomes.

# Software and Hardware Requirements

## Software

1. **Operating System:** Windows 10/11, macOS, or Linux
2. **Python Version:** Python 3.9+
3. **Libraries:**
   * pyaudio
   * google-cloud-speech
   * transformers (Hugging Face)
   * cohere
   * gspread
   * oauth2client
4. **APIs:**
   * Google Speech-to-Text API
   * Google Sheets API
   * Cohere API

## Hardware

1. **Processor:** Intel i5/i7 or equivalent
2. **Memory:** Minimum 8 GB RAM
3. **Storage:** 100 GB free disk space
4. **Microphone:** For recording audio during calls

# SDLC Steps

**1. Planning**

* Define project scope: Real-time analysis of buyer-seller interactions.
* Identify tools: Python, Jupyter Notebook, PyAudio, APIs.
* Set milestones: Recording audio, transcribing, sentiment/intent analysis, recommendation ,negotiation terms generation.

**2. Analysis**

* Identify user requirements for real-time call analysis.
* Evaluate APIs for transcription, sentiment detection, intent analysis, recommendations, negotiation coach and data storage.

**3. Design**

* Modular design for components:
  + Audio recording module.
  + Transcription module.
  + Analysis modules (sentiment and intent).
  + Data storage module.

**4. Development**

* Implement functionality in Python.
* Ensure smooth integration of APIs and libraries.

**5. Testing**

* Unit testing for each module.
* End-to-end testing for the complete system.

**6. Deployment**

* Deploy on local systems or cloud.

**7. Maintenance:** Periodic updates for API changes and bug fixes.

# API Integrations

* **Google APIs:** Enable Drive, Sheets, and Speech-to-Text.
* **Hugging Face:** Use pre-trained sentiment and classification models.
* **Cohere API:** Generate answer to the input, recommendations and negotiation coach dynamically.
* **GROQ API:** Summarizes the conversation.

# Testing and Evaluation

1. **Unit Testing:** Verify individual components.
2. **Integration Testing:** Ensure seamless interaction among modules.
3. **Performance Testing:** Test for latency during real-time calls.

# Future Enhancements

1. Deploy as a cloud-based solution for scalability.
2. Enhanced analytics dashboard for visualizing insights.
3. Build a recommendation system based on previous chat summaries, if available, to recommend products. This involves creating a dataset of summarized chats using a summarization model and finding a laptop sales dataset with attributes like models, prices, discounts, and other specifications. The recommendation model will analyse both datasets to generate suggestions tailored to the buyer's preferences.

# References

* PyAudio Documentation
* Google Cloud Speech-to-Text API
* Hugging Face Transformers
* Cohere API Documentation
* Google Sheets API