Audio Transcription, Summarization and Q&A System

# Meeting Transcriber and Q&A Web Application

## Abstract

In an era defined by remote work, virtual meetings, and vast audio data, the need to convert spoken information into usable, searchable insights is more important than ever. This project addresses that need through an intelligent, automated pipeline that transcribes, summarizes, and enables interactive querying of spoken content. The system is designed to be modular, efficient, and user-centric—employing advanced artificial intelligence models to ensure high accuracy and responsiveness.

## Introduction

Human speech is one of the richest sources of information, yet it remains one of the least structured. Meetings, lectures, interviews, and podcasts generate hours of audio that are difficult to process manually. The application detailed in this document turns that challenge into opportunity. It enables users to upload or extract audio from YouTube, automatically transcribe it, generate coherent summaries, and ask natural language questions about the content.

The solution uses a synergy of powerful models including Faster-Whisper for transcription, SentenceTransformers for semantic vectorization, FAISS for similarity-based retrieval, and Gemini for generating human-like answers and summaries. This AI-powered pipeline is integrated into a lightweight web platform built with Flask and HTML.

## Feature Requirements

* **Audio Extraction**: Users can input a YouTube URL to extract and download high-quality audio.
* **Transcription**: Long audio files are transcribed to text using a multi-tier model selection for robustness.
* **Summarization**: Large transcripts are condensed into readable summaries.
* **Q&A Interaction**: Users can query the transcript in natural language and receive precise, context-based answers.
* **Frontend Interface**: A clean and accessible web interface for uploading, summarizing, and interacting with content.

## Technologies and Tools

This system is built with modern, reliable technologies that support both development efficiency and performance:

* **Faster-Whisper**: A fast and optimized transcription model compatible with CPUs and GPUs.
* **SentenceTransformer**: Pre-trained language models for semantic embeddings.
* **FAISS**: Scalable vector search library for quick retrieval of relevant content.
* **Google Generative AI (Gemini)**: Provides generative responses and summarization with high precision.
* **LangChain**: Enables the chaining of tasks such as retrieval and generation for complex QA workflows.
* **Flask**: Serves as the backend for routing, model initialization, and API delivery.
* **HTML/CSS**: Used for building the frontend templates.

## System Architecture Overview

The system architecture is composed of several interconnected modules:

1. **Audio Ingestion Layer**: Responsible for accepting YouTube links or file uploads and downloading or storing the audio.
2. **Transcription Engine**: Uses Faster-Whisper to generate accurate transcripts from long audio files.
3. **Summarization Pipeline**: Transcripts are broken into segments and passed to Gemini for summarization.
4. **Embedding Generator**: Each transcript segment is embedded using SentenceTransformer for semantic understanding.
5. **Semantic Indexing Module**: Uses FAISS to store and retrieve transcript chunks based on user input.
6. **LLM Response Generator**: Gemini interprets the user query and retrieved context to generate a final response.
7. **Presentation Layer**: The Flask-based frontend that provides an interactive, intuitive user interface.

## Use Case Scenarios (Expanded)

* **Academic Institutions**: Universities can provide automatic lecture transcriptions and searchable archives for students. Summaries assist in revision, while the Q&A system enables deeper understanding through self-inquiry.
* **Business and Corporate**: In team meetings or stakeholder calls, the system can automatically document discussions, extract action points, and allow follow-up queries based on context.
* **Media and Journalism**: Reporters can transcribe interviews, extract summaries for storyboards, and search for quotes using natural questions.
* **Customer Support and Feedback**: Organizations can transcribe support calls and analyze trends in customer queries through contextual searching.
* **Accessibility Use Cases**: Individuals with hearing impairments can access spoken content as text with summarization and Q&A capabilities for better comprehension.

## Models

### Faster-Whisper

A lightweight, optimized fork of OpenAI’s Whisper, Faster-Whisper provides quick and reliable speech recognition across languages. Its support for various compute types and lower resource requirements makes it suitable for scalable deployment.

### SentenceTransformer (all-mpnet-base-v2)

This model provides high-quality sentence embeddings which preserve contextual meaning, making it perfect for semantic search and similarity tasks. Its multilingual capabilities and performance on benchmarking tasks further validate its use.

### FAISS

A robust and fast similarity search library developed by Facebook AI. FAISS allows efficient querying of high-dimensional vectors, a critical function for the Q&A feature of this application.

### Gemini (Google Generative AI)

Gemini is a state-of-the-art large language model that delivers high-quality summaries and natural language responses. Its low-temperature configuration ensures factual and reliable outputs.

## Detailed Pipeline Walkthrough

1. **User Submission**: The user provides a YouTube URL or audio file through the web interface.
2. **Audio Preprocessing**: The system standardizes audio format using FFmpeg if necessary.
3. **Transcription**: The audio is transcribed into readable text using the selected Whisper model.
4. **Text Segmentation**: The transcript is divided into overlapping chunks to preserve context.
5. **Embedding Generation**: Chunks are embedded into high-dimensional vectors.
6. **Semantic Indexing**: FAISS builds an index of all vectors, enabling fast contextual searches.
7. **User Question Handling**: A natural language question is accepted through the interface.
8. **Contextual Search**: FAISS returns the most relevant transcript chunks.
9. **LLM Prompting**: The context and question are passed into Gemini for response generation.
10. **Response Delivery**: The answer is displayed to the user in the web interface.

## Performance Evaluation

The system was tested on a range of devices and audio formats. Observations include:

* **Transcription Speed**: Average of 2x real-time on CPU (i.e., 1 hour of audio transcribes in ~30 minutes).
* **Accuracy**: High for single-speaker, clearly spoken content; moderate for noisy or multi-speaker inputs.
* **Summarization Quality**: Context-aware, readable, and useful. Rare hallucinations observed only in ambiguous queries.
* **Question Relevance**: Semantic retrieval consistently returns meaningful segments for answer synthesis.
* **System Load**: Memory-efficient even on low-resource systems due to lightweight embeddings and streamed processing.

## Comparison and Alternatives (Narrative Style)

While commercial platforms like Otter.ai and Descript provide transcription and limited summarization, they often lack deep semantic search and natural language question-answering. Our system fills that gap by integrating open-source tools with advanced language models to allow not just passive transcription but active exploration of audio content.

Whisper CLI, while accurate, is limited to basic transcription and lacks integration with summarization or interactive Q&A. Gemini brings an edge in generating answers that are not only factually relevant but also conversationally fluid, distinguishing our system from other academic or industry solutions that depend on rule-based summarizers or static query methods.

By combining open-source flexibility with LLM-powered intelligence, our approach offers greater adaptability, cost-efficiency, and extensibility than proprietary, black-box solutions.

## Future Scope (Expanded)

* **Real-Time Transcription and Response**: Adapting the backend to handle streaming audio for instant processing.
* **Multi-Speaker Diarization**: Assigning speaker labels and distinguishing multiple voices in transcription.
* **Multilingual Support**: Seamless handling of non-English input through Whisper’s multilingual capabilities.
* **Data Visualization**: Adding visual analytics like word clouds or sentiment graphs from transcript data.
* **Integration with LMS and CRMs**: Embedding this tool into enterprise systems for organizational knowledge management.
* **Voice-Based Querying**: Allowing users to ask questions by voice and get voice-based answers for accessibility.

## Deployment Notes (Real-World Ready)

* **Local Development**: Tested extensively using Flask’s development server and SQLite for logs.
* **Containerization**: Docker images available for rapid deployment with minimal dependencies.
* **Hosting Platforms**: Deployable on Heroku (with Dynos), Render (persistent storage), and GCP (for LLM access).
* **Security Practices**: API rate limiting, input sanitation, and environment isolation followed.
* **Logging and Monitoring**: Basic error logs with scope for integration with tools like Sentry or Prometheus.
* **Scalability**: Horizontal scaling possible using microservice architecture.

## Appendix

### File Handling

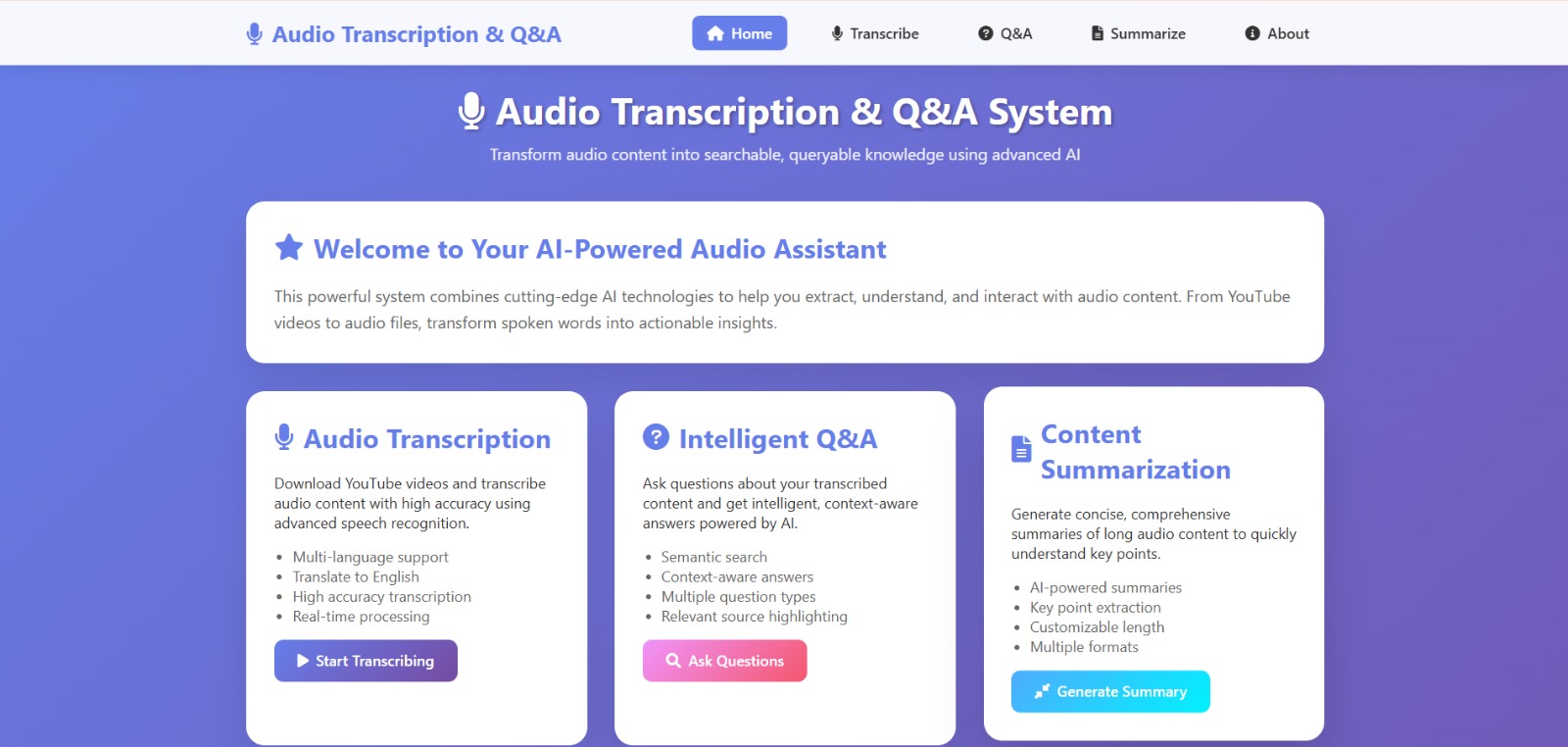
* Files are named using UUIDs for uniqueness and traceability.
* Audio and transcript files are stored temporarily to optimize server space.

### Supported Tasks

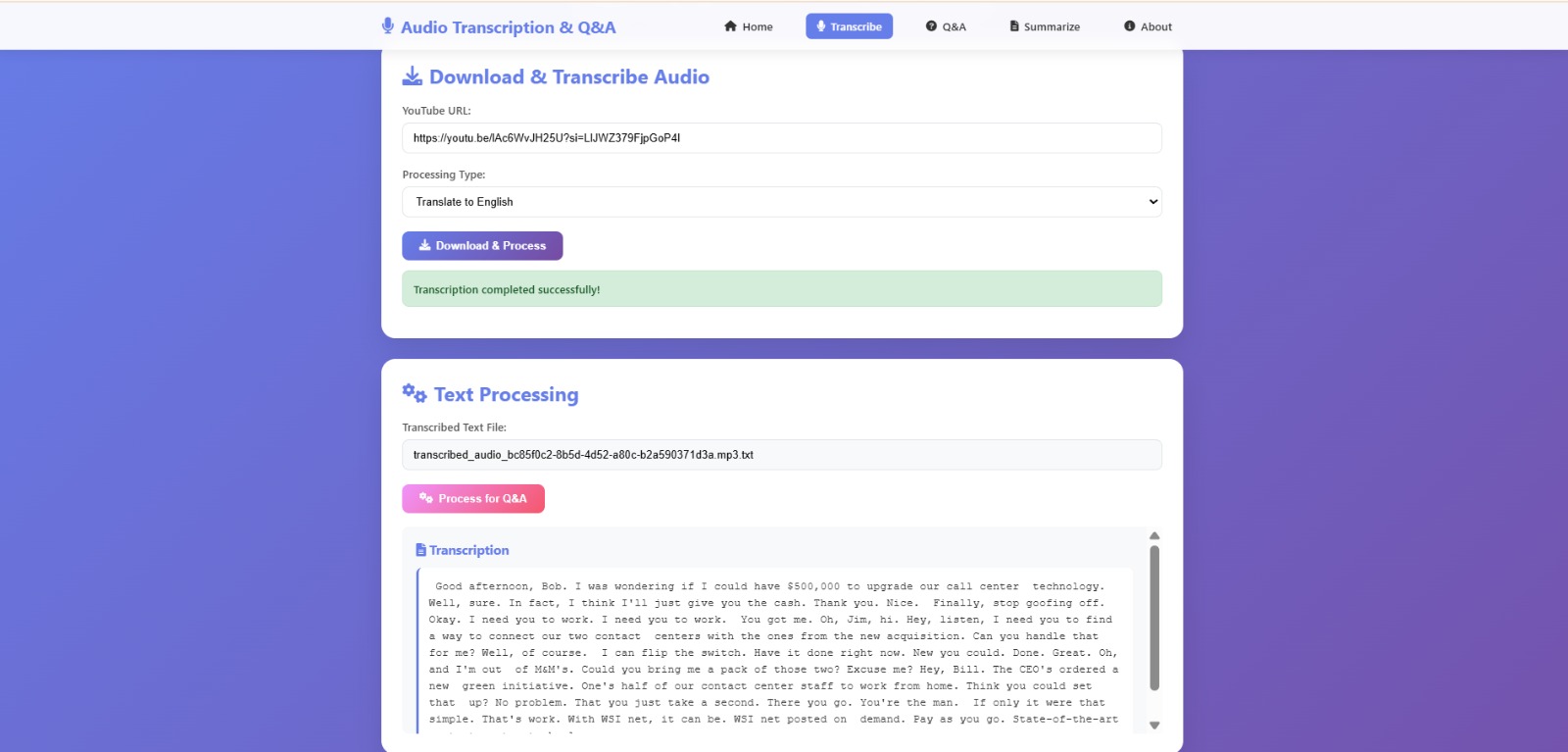
* Downloading and converting audio.
* Transcribing spoken content.
* Summarizing lengthy transcripts.
* Interactive Q&A functionality.

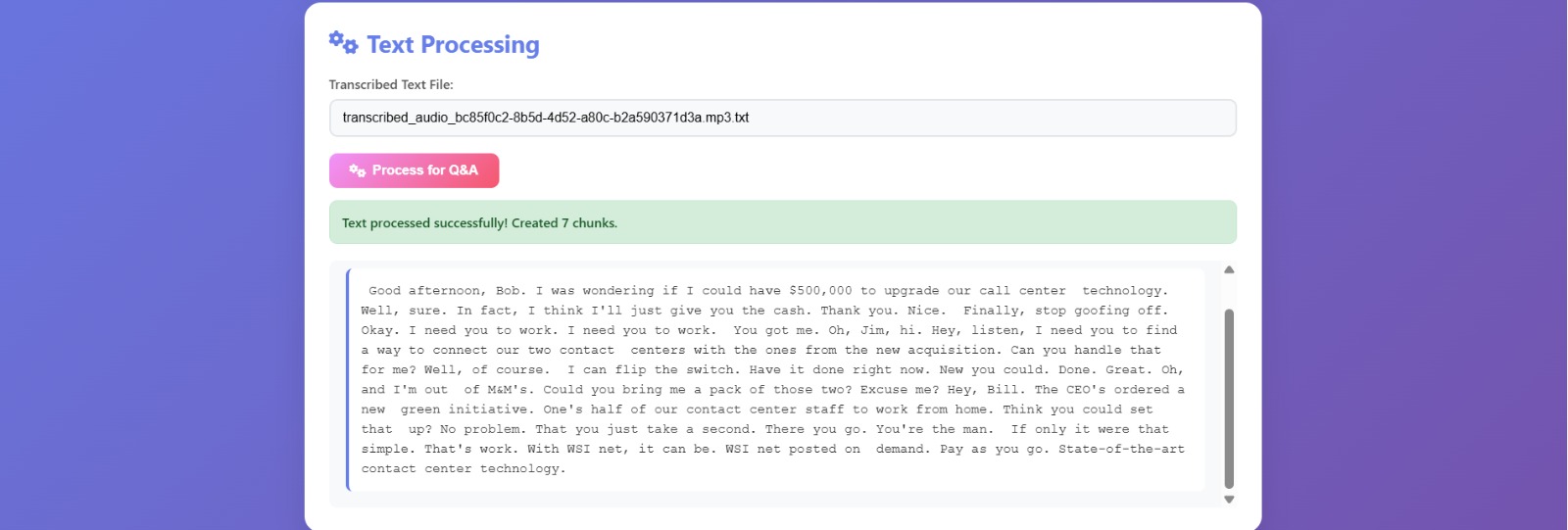
### Frontend Pages

* Home

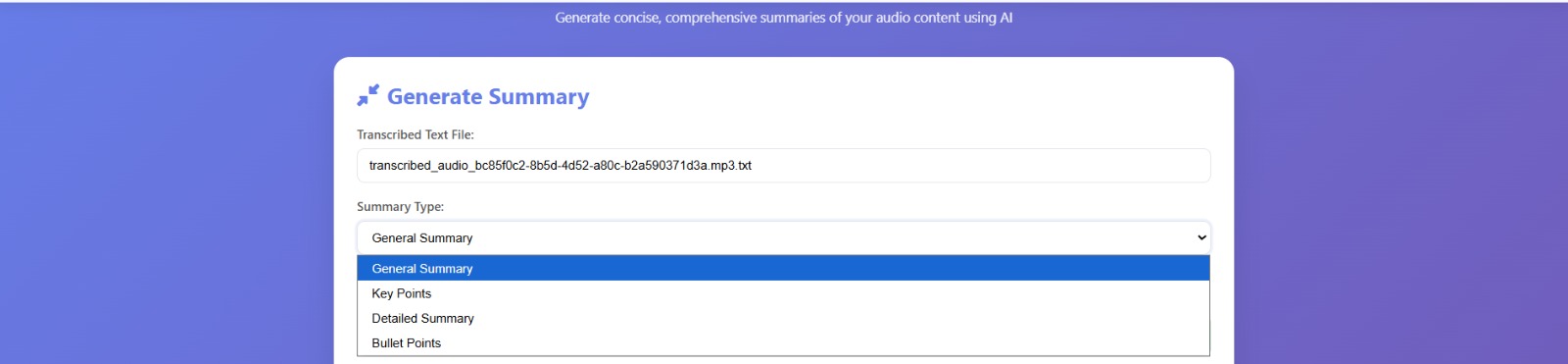


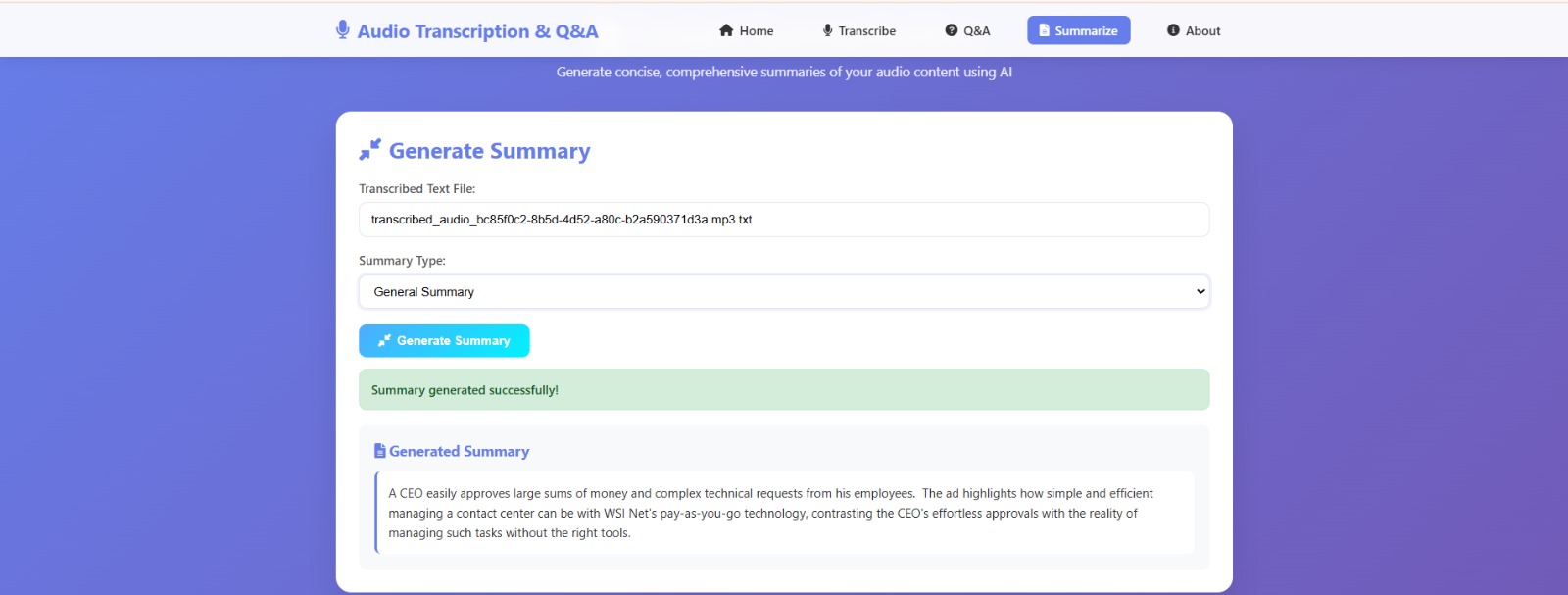
* Transcription Interface



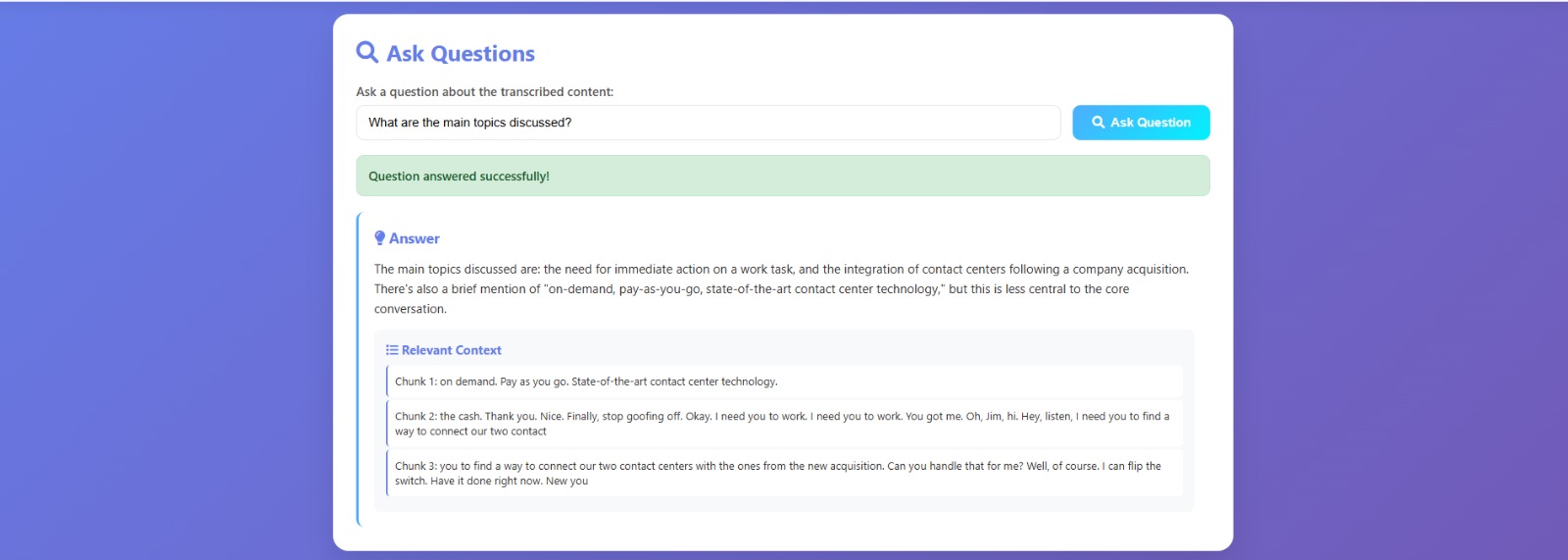


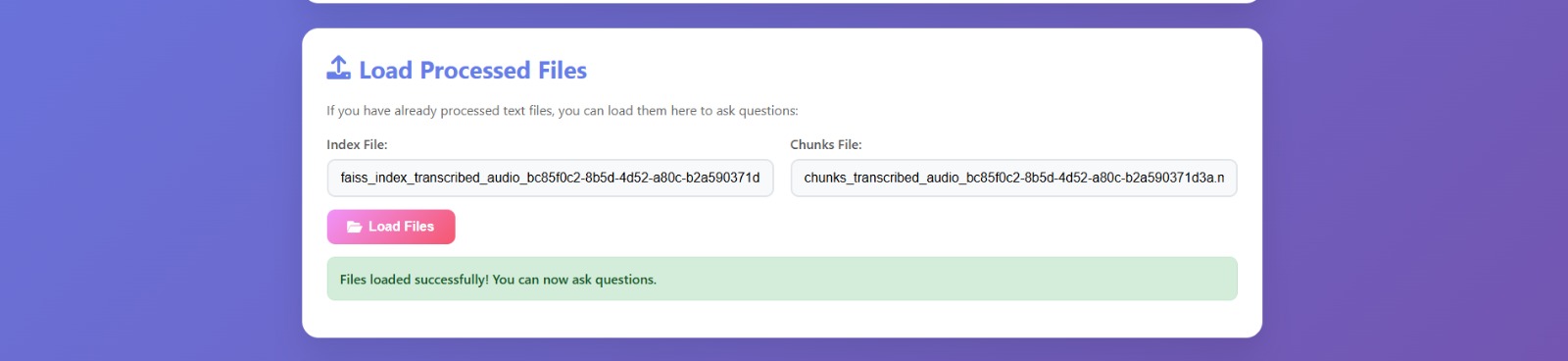
* Summarization





* Q&A Interface





**System Diagram**

Conceptual Overview:

User Input → Audio Downloader → Transcription Engine → Text Chunker → Embedding Generator → FAISS Index → LLM Query Engine → Output Answer/Summary

## Conclusion

This application transforms the complex task of understanding audio data into a fast, accurate, and engaging process. By combining cutting-edge transcription, semantic embedding, and large language models, the system is capable of transcribing, summarizing, and answering questions based on audio content with high reliability.

From academic research and corporate meetings to podcasts and lectures, the solution offers a practical toolkit for transforming spoken data into meaningful, searchable knowledge. It showcases the potential of AI to augment human capabilities, improve productivity, and make content more accessible.

This documentation presents the foundation for future development, including real-time processing, multilingual support, speaker identification, and deployment on scalable cloud infrastructure. As speech becomes a dominant medium for communication, such systems will become indispensable.



## References

1. **Faster-Whisper by SYSTRAN**  
   GitHub Repository: <https://github.com/SYSTRAN/faster-whisper>
2. **FAISS - Facebook AI Similarity Search**  
   Official Documentation: https://faiss.ai  
   GitHub Repository: <https://github.com/facebookresearch/faiss>
3. **SentenceTransformers by HuggingFace**  
   Project Page: https://www.sbert.net  
   Hugging Face Model Hub: https://huggingface.co/sentence-transformers/all-mpnet-base-v2
4. **Google Generative AI (Gemini)**  
   Developer Docs: https://ai.google.dev/  
   Gemini API Overview: https://ai.google.dev/gemini-api/docs
5. **LangChain Framework**  
   Official Documentation: https://docs.langchain.com  
   GitHub Repository: <https://github.com/langchain-ai/langchain>