Technical Documentation

Architecture, Infrastructure & Integration Guide

CohrenzAI
Noida, Uttar Pradesh, India
info@cohrenzai.com | +91 8273597975

1. Platform Architecture Overview

CohrenzAI is built on a modern, scalable microservices architecture designed for high performance, reliability, and seamless integration. Our platform combines advanced AI/ML capabilities with robust backend infrastructure to deliver intelligent conversational experiences.

1.1 Core Technology Stack

- Backend Framework: FastAPI (Python) - High-performance async web framework
- AI/ML Engine: Azure OpenAI (GPT o3-mini) for natural language understanding and generation
- Embedding Model: Azure OpenAI text-embedding-3-small for vector representations
- Vector Database: FAISS (Facebook AI Similarity Search) for efficient similarity retrieval
- Primary Database: MySQL for structured data storage (leads, conversations, sessions)
- Document Processing: LangChain framework for document loading, splitting, and chain orchestration
- External Integrations: Google Sheets API for lead export, extensible CRM connectors

1.2 System Architecture Diagram

The CohrenzAI platform follows a layered architecture:

Layer 1 - Client Layer: JavaScript chatbot widget embedded on client websites via a single script tag. Communicates with backend via REST API.

Layer 2 - API Layer: FastAPI server handling chat requests, lead management, intent prediction, and session management with CORS middleware for cross-origin security.

Layer 3 - Intelligence Layer: LangChain-orchestrated pipeline combining FAISS retrieval, prompt engineering, and Azure OpenAI for context-aware response generation.

Layer 4 - Data Layer: MySQL database for persistent storage/conversation history, and FAISS index for vector similarity search.

Layer 5 - Integration Layer: Google Sheets API for lead export, with extensible architecture for CRM, email, and notification integrations.

2. AI & Machine Learning Pipeline

2.1 Document Ingestion & Knowledge Base Construction

Our AI-Ready Knowledge System follows a multi-step pipeline to transform raw business documents into an intelligent, queryable knowledge base:

- Step 1 - Document Loading: Support for PDF files (via PyPDFLoader) and HTML pages (via BSHTMLLoader). Files are loaded from configurable directories.
- Step 2 - Text Chunking: Documents are split into overlapping chunks (1000 characters with 200-character overlap) using RecursiveCharacterTextSplitter for optimal context preservation.
- Step 3 - Embedding Generation: Each chunk is converted into a 1536-dimensional vector using Azure OpenAI's text-embedding-3-small model.
- Step 4 - Vector Indexing: Embeddings are stored in a FAISS index for sub-millisecond similarity search across thousands of document chunks.
- Step 5 - Retrieval: At query time, the system retrieves the top-4 most relevant chunks using cosine similarity search.

2.2 Conversational AI Engine

The conversational engine uses a Retrieval-Augmented Generation (RAG) approach:

- Context Retrieval: User queries are embedded and matched against the FAISS index to find relevant business knowledge
- Conversation Memory: The last 20 chat messages are included as context for conversational continuity
- Prompt Engineering: A carefully crafted prompt template combines retrieved context, conversation history, lead state, and user query
- Response Generation: Azure OpenAI (o3-mini deployment) generates natural, context-aware responses
- Lead-Aware Responses: The prompt adapts based on the current lead capture state (ASK_NAME, ASK_EMAIL, ASK_PHONE, COMPLETED)

2.3 Intent Analysis Engine

Our proprietary intent analysis system examines full conversation histories to produce structured reports:

- User Interest Analysis: Level and nature of interest (exploratory, evaluative, transactional, partnership)
- Segment/Product Identification: Specific products or services the user is interested in
- Actual Requirement Detection: What the user is ultimately trying to achieve
- Gap Analysis: Identifying what the user needs that may not currently be available
- Match Analysis: Identifying existing offerings that align with user needs
- Behavioral Signal Detection: Trust-building, comparison, research phase, credibility checks

3. Lead Capture System

3.1 Multi-Trigger Lead Detection

CohrenzAI uses a sophisticated three-layer trigger system to initiate lead capture at the optimal moment:

- Trigger 1 - Opportunistic: Automatically detects when users share contact information (email/phone) in conversation
- Trigger 2 - Explicit Intent: Activates when users mention signal keywords (pricing, demo, trial, contact, consultation, services, partnership)
- Trigger 3 - Proactive Engagement: After 4+ user messages indicating sustained interest, the system proactively initiates lead capture

3.2 State Machine Lead Flow

Lead capture follows a strict state machine with five states:

- NONE: No lead capture active; normal conversation mode
- ASKED_NAME: System has asked for the visitor's name
- ASKED_EMAIL: System has asked for email address
- ASKED_PHONE: System has asked for phone number
- COMPLETED: All information collected; lead exported to Google Sheets

3.3 Smart Input Processing

Our lead extractor handles various input scenarios intelligently:

- Name extraction from natural phrases ('My name is...', 'I am...', 'I'm...')
- Email validation using industry-standard regex patterns
- Indian phone number validation and normalization (10-digit with country code support)
- Fast-forward capability: if a user provides email AND phone early, the system skips redundant questions
- Casual message detection to avoid misinterpreting greetings as data input

3.4 Intent Summary Generation

Before lead export, the system generates a comprehensive intent summary:

- Topic extraction from conversation history
- Engagement metrics (message count, question count)
- Contact signal detection (opportunistic contact sharing)
- Latest user need summarization
- Maximum 500-character concise summaries for CRM compatibility

4. Integration Guide

4.1 Website Integration (One-Script Deployment)

CohrenzAI can be integrated into any website with a single script tag. The chatbot widget is fully self-contained and creates all necessary DOM elements, styles, and event handlers dynamically.

Integration Steps:

- Step 1: Add a single <script> tag to your HTML with data-api-url and data-public-key attributes
- Step 2: The script automatically creates a floating chat button and expandable chat panel
- Step 3: Session management is handled automatically via localStorage
- Step 4: All communication happens via secure REST API calls

4.2 API Reference

POST /chat - Send a message and receive AI response

- Request: { message: string, session_id: string }
- Response: { answer: string }

GET /chats/{session_id} - Retrieve conversation history

- Parameters: session_id (path), k (query, default=10)
- Response: { chats: [{ message: string, sender: string }] }

GET /predict_intent/{session_id} - Get intent analysis

- Parameters: session_id (path)
- Response: { intent_summary: string }

4.3 Google Sheets Integration

Completed leads are automatically exported to Google Sheets with the following fields: session_id, name, email, phone, intent_summary, and created_at timestamp. This enables real-time lead tracking and CRM workflow integration.

5. Security Architecture

- API Key Authentication: All requests require valid API key and public key headers
- CORS Protection: Cross-Origin Resource Sharing middleware restricts unauthorized domain access
- Input Validation: Pydantic models enforce strict input validation on all API endpoints
- SQL Injection Prevention: Parameterized queries used throughout the database layer
- Environment Variable Management: All secrets stored in .env files, never hardcoded
- Session Isolation: Each conversation is isolated by unique session IDs

- Rate Limiting Ready: Architecture supports Gunicorn + Uvicorn deployment with rate limiting

6. Deployment & Scalability

- Docker-ready architecture for containerized deployments
- Gunicorn + Uvicorn for production-grade async server performance
- Horizontal scaling support for handling thousands of concurrent conversations
- FAISS index supports millions of document chunks with sub-millisecond retrieval
- Database connection pooling for efficient MySQL resource utilization