Faculties of Engineering & Technology

Dumka Engineering College

Software Requirements Specification (SRS)

DEC Chatbot — Voice & Text Assistant for Dumka Engineering College

Submitted by

Neelesh Ranjan

Under the guidance of

Mrs. Sunidhi Priyadarshini
(Asst. Prof., Department of Computer Science & Engineering)

Polytechnic Compound Road, Dumka, Jharkhand 814101

SRS Report submitted to Dumka Engineering College in partial fulfilment for the award of the degree of
BACHELOR OF TECHNOLOGY
In the department of Computer Science & Engineering

2025

Jharkhand University of Technology, Ranchi

TABLE OF CONTENTS

Introduction

1.1. Purpose

This document provides detailed information about the Software Requirements Specification of the DEC Chatbot. The SRS follows IEEE-style organization and will guide supervisors, developers, testers, QA and maintenance personnel.

1.1.1. Scope

The DEC Chatbot is a web-based and voice-enabled assistant for Dumka Engineering College. Users can ask questions about admissions, academics, exams, timetables, faculty contacts, events, and campus services. The system supports both text and voice I/O and can escalate to a human when confidence is low.

Overall Description

2.1. Product Perspective

The chatbot integrates into the DEC website as an embeddable widget and exposes Flask-based REST APIs. It uses Python libraries for NLU and information retrieval from a curated Knowledge Base and official college resources.

2.2. Product Function

1. Accept user queries (text and voice).
2. Perform speech-to-text (ASR) and text-to-speech (TTS).
3. Detect intents and extract entities using NLU.
4. Retrieve answers from the Knowledge Base, calendars, and directories.
5. Provide suggested follow-ups and links to official pages.
6. Escalate to staff on low confidence or on request.
7. Log conversations for analytics and improvement.

2.3. User Characteristics

Administrator, Faculty/Staff, and Student/Visitor users with general web proficiency.

2.4. Assumptions and Dependencies

Stable internet access; access to official college data sources; periodic KB updates by admins; browser support for microphone access.

2.5. Tools and Technologies

Python 3.x, Flask, Rasa or Transformers+spaCy, SentenceTransformers/FAISS, PostgreSQL, SQLAlchemy, Web Speech API, pyttsx3/gTTS, Docker, Gunicorn+Nginx.

General Requirements

3.1. Specific Requirements

Requirement 1: The system must support both text and voice conversations.

Requirement 2: The chatbot must return accurate answers from an Admin-maintained Knowledge Base.

Requirement 3: Intent confidence threshold (e.g., 0.75) shall be used to decide clarification or escalation.

Requirement 4: Conversation logs must be stored with PII masking as per policy.

Requirement 5: Admins can manage KB entries, thresholds, and escalation recipients via a secure dashboard.

3.2. Tools and Technologies

Backend: Python (Flask), NLU: Rasa/Transformers, Database: PostgreSQL, Vector Search: FAISS, Frontend: HTML/CSS/JS, Deployment: Docker.

External Interfaces

4.1. User Interfaces

Three modules: Client (web chat), Admin (dashboard), and optional mobile webview. Login/role-based access for administrators.

4.2. Hardware Interfaces

Client devices with microphone/speaker; server with sufficient CPU/RAM (optional GPU).

Functional Requirements

1. Description: The system shall allow users to chat without registration.

• Input: Message or voice.

• Output: Bot reply and suggestions.

2. Description: The system shall allow admins to update Knowledge Base entries.

• Input: Authenticated admin edits.

• Output: Updated KB and index refreshed.

3. Description: The system shall convert speech to text and back to audio.

• Input: Microphone audio.

• Output: Transcribed text and synthesized speech.

4. Description: The system shall retrieve answers from KB and official sources.

• Input: Detected intent + entities.

• Output: Answer with confidence score.

5. Description: The system shall escalate to staff on low confidence.

• Input: Low score or user request.

• Output: Email/portal notification to staff and acknowledgement to user.

5.1. Constraints

5.1.1. Operational and Environmental

Usable on modern browsers; microphone permission needed; secure access for admin endpoints.

5.1.2. Legal Constraints

Display privacy policy and obtain consent for audio recording; comply with college data policies.

5.1.3. Design Constraint

Lightweight widget; containerized backend; compatibility with hosting environment.

Non-Functional Requirements

6.1. Software Application Attributes

6.1.1. Reliability – >95% response success for KB-backed queries.

6.1.2. Maintainability – Modular design; documented admin operations.

6.1.3. Availability – Target 99% during working hours.

6.1.4. Security – HTTPS; RBAC; encryption at rest where applicable.

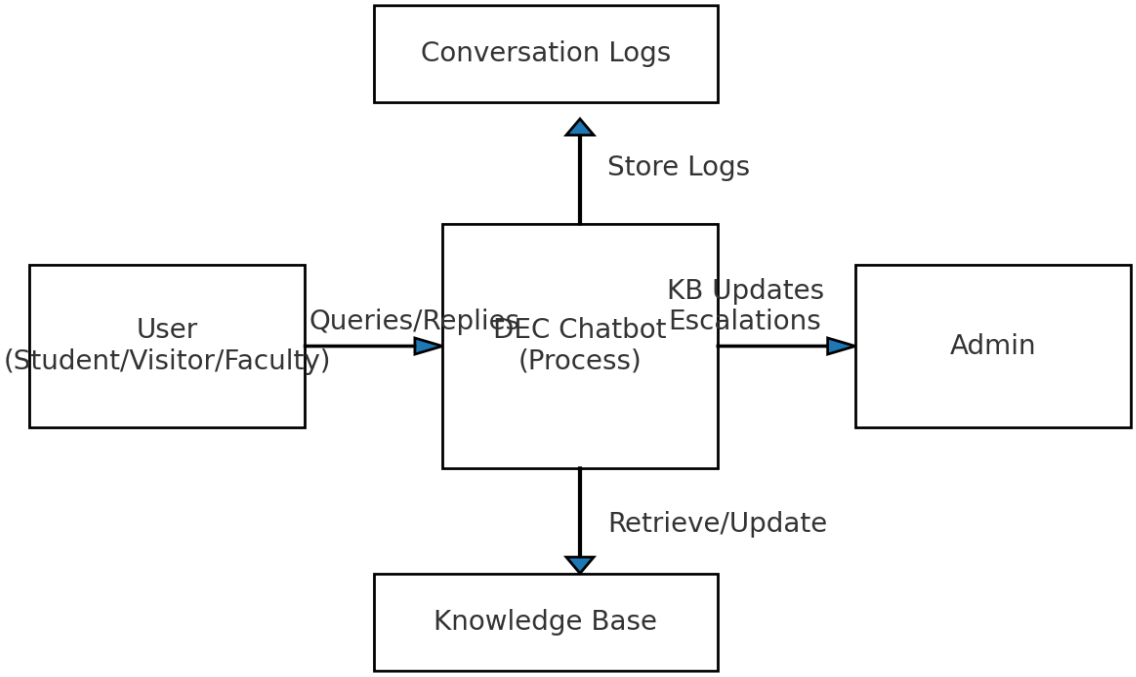
6.1.5. Portability – Works across major browsers; containerized backend.

6.1.6. Usability – Clear prompts; quick replies; accessible controls.

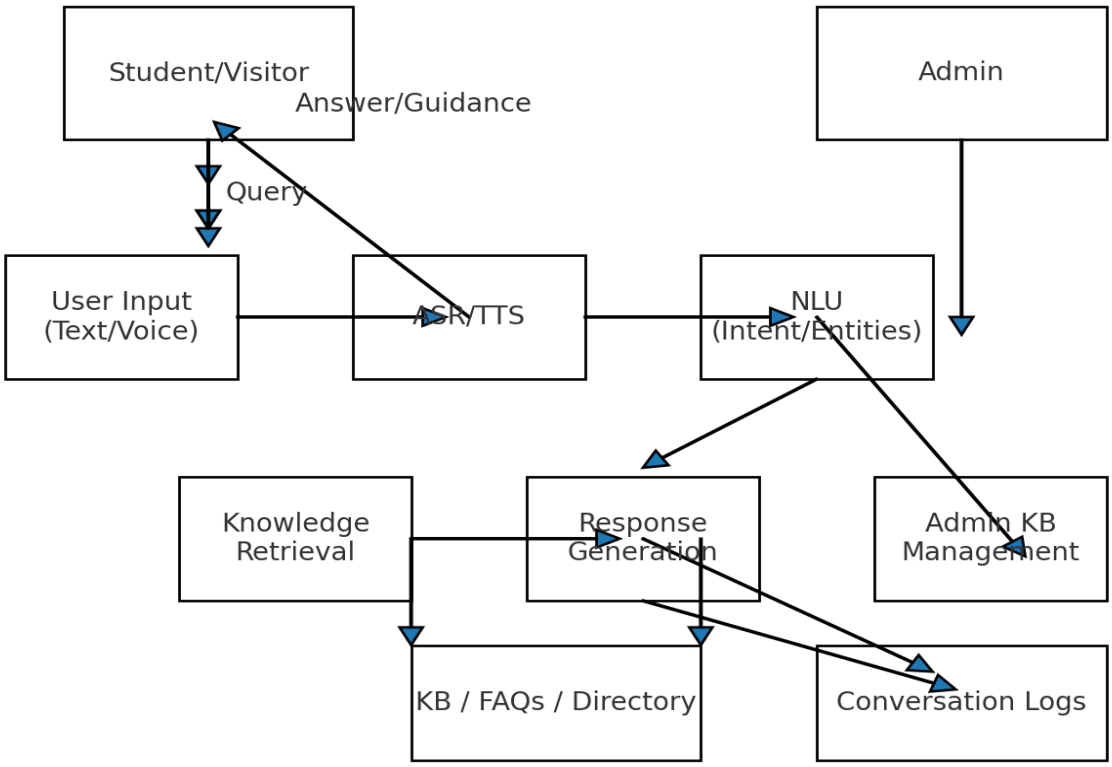
6.2. Look and Feel – College branding; consistent fonts; minimal color palette.

Data Flow Diagrams

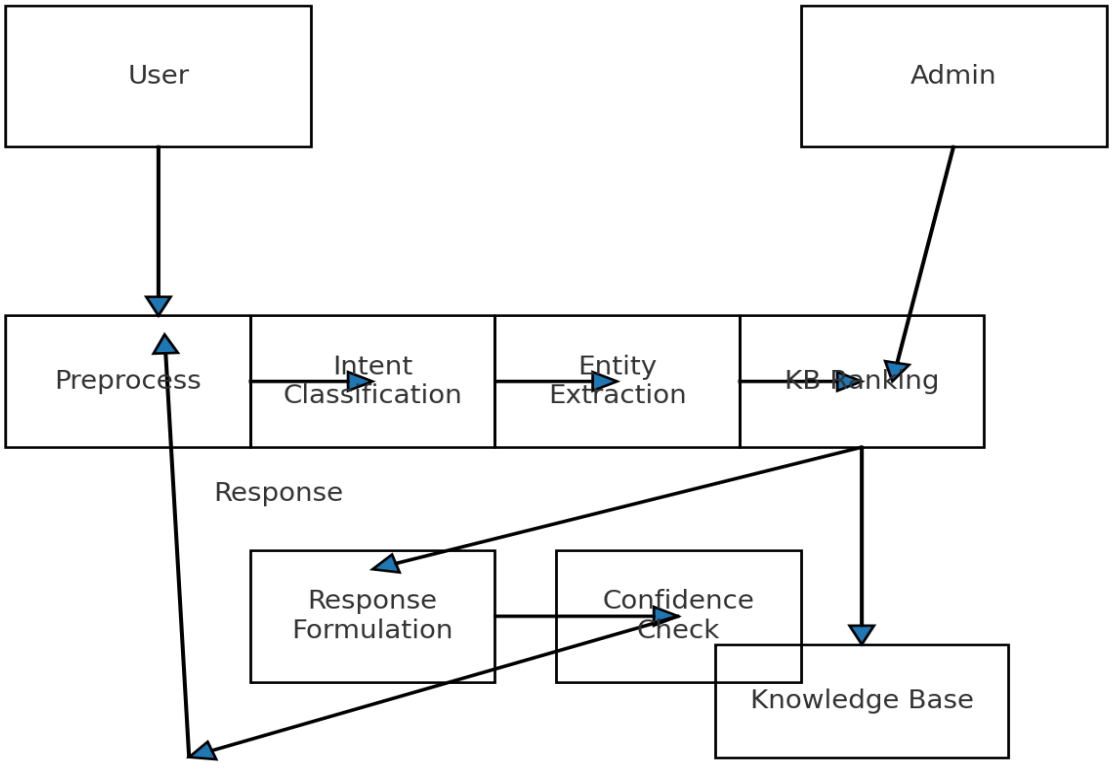
7.1. Data Flow Diagram (DFD Level-0)



7.2. Data Flow Diagram (DFD Level-1)



7.3. Data Flow Diagram (DFD Level-2)



References

[1] Python Documentation – https://www.python.org/

[2] Flask Documentation – https://flask.palletsprojects.com/

[3] Rasa – https://rasa.com/

[4] spaCy – https://spacy.io/

[5] SentenceTransformers – https://www.sbert.net/

[6] Web Speech API – MDN Docs