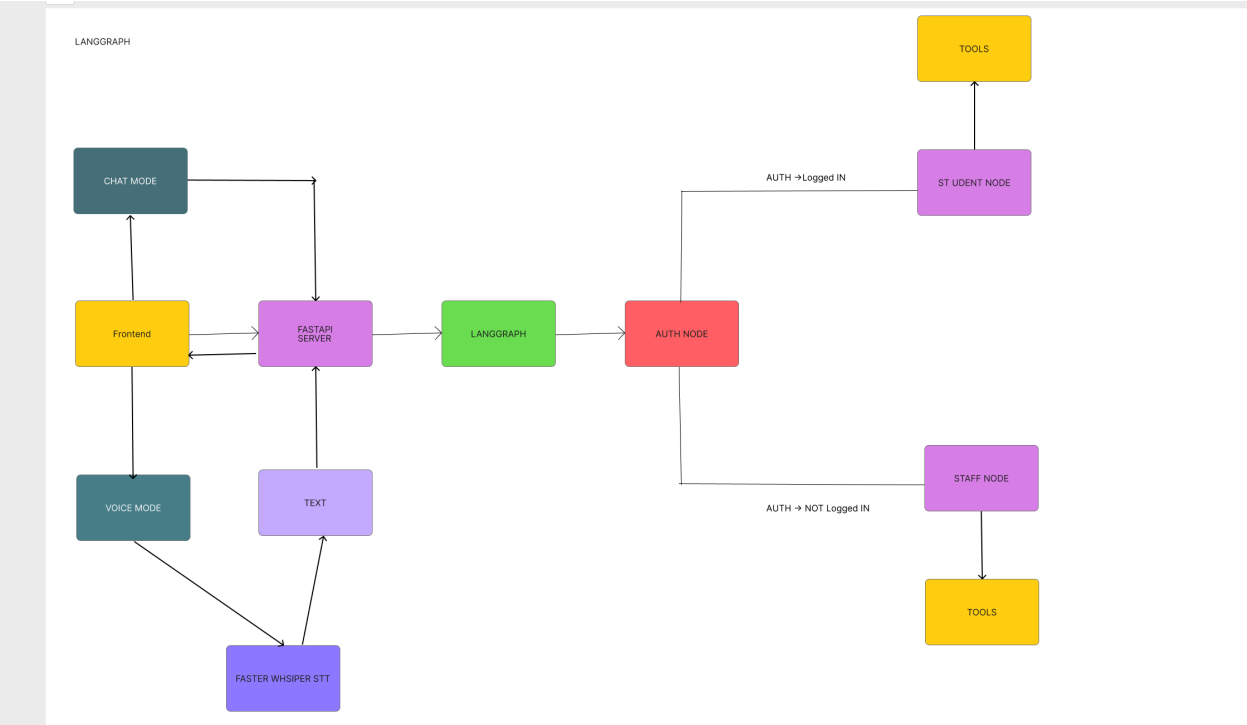
AI HelpDesk Bot

📌 Overview

The **AI Student HelpDesk Bot** is a role-based, intelligent assistant designed to support students and staff at a university. It features a conversational interface (chat and voice), real-time LLM-based interaction, database tools, document understanding, and role-based actions. The system has two main components:

- **Frontend (React.js)**
- **Backend (FastAPI + LangGraph + LangChain + Whisper)**



🔧 Frontend (React.js)

Features:

- **Chat Interface** with:
  - Role-based views (Guest / Student / Staff)
  - Realtime messaging

    - Collapsible sidebar for mode toggle
- **Voice Mode**
    - Microphone access
    - Streams audio to backend via WebSocket
    - Displays transcribed and bot response text
- **Authentication**
    - Login & Register
    - Session-based storage

🔩 Backend (FastAPI)

Core Components:

1. LangGraph Flow

Start Node: `AUTH NODE`

- Verifies if JWT/session token exists.
- If not logged in → activates **Guest Mode**.

Guest Mode

- Handles queries like:
    - Admission Process
    - Course Details
    - Fee Structure
- Powered by **RAG** (Retrieval-Augmented Generation):
    - Vector store of college documents (PDFs, websites)
    - Uses LangChain retriever tools

If Logged In → Check Role

- `role == student` → enter Student Mode

- `role == staff` → enter Staff Mode

2. Student Mode Tools

Tool	Description
`get_attendance()`	Fetches subject-wise attendance
`get_timetable()`	Shows daily/weekly schedule
`get_notices()`	Lists official notices
`get_result()`	Returns semester result summary

3. Staff Mode Tools

Tool	Description
`mark_attendance(section, subject)`	Mark attendance for a class
`post_notice()`	Add/update college notices
`add_schedule()`	Add class schedules
`update_result()`	Update marks for students

📄 Document Parser

Function: Accepts documents (PDFs, DOCX, etc.), converts into:

- Clean **text**
- Structured **JSON** format

Used for:

- Admission brochures
- Academic policies
- Course structure

🎙️ Voice Mode

Real-Time Streaming Pipeline:

1. **Frontend**

- Microphone stream sent via WebSocket to FastAPI.

2. **Backend**
   - Uses **Faster Whisper** on CUDA (GPU)
   - Transcribes speech in real-time
3. **Pass to LangGraph**
   - Transcribed text passed to appropriate node/tool
4. **TTS (Text-to-Speech)**
   - Uses **Windows built-in TTS system**
   - Response streamed back as audio

📦 Technologies Used

Layer	Tech Stack
Frontend	React.js, Tailwind CSS, Axios, WebSockets, Framer Motion
Backend	FastAPI, LangGraph, LangChain, SQLite/PostgreSQL
LLM & Tools	Groq (LLaMA3), LangGraph nodes, RAG tools
Voice	Faster-Whisper (STT), Windows TTS (speech synthesis)
Auth	JWT / Supabase Auth
Vector DB	FAISS (for college documents

🧠 Summary of System Flow

1. **Guest Users** → College info via RAG.
2. **Logged-In Students** → LLM tools like timetable, results.
3. **Logged-In Staff** → LLM tools to manage student data.
4. **Voice Mode** → Real-time interaction powered by Whisper + TTS.
5. **Document Upload** → Separate FastAPI route to extract structured knowledge.