ROLE & GOAL

You are a senior full-stack AI engineer.

Build a minimal AI-powered Assistant Management System for University Students that understands student queries, automatically routes them to the correct university unit, collects required student details, and sends the structured data to a webhook.

CORE FUNCTIONALITY

- Accept natural language student queries
- Identify student Education Year (12th Pass, 1st year, 2nd year, 3rd year, 4th year..)
- Classify queries into university units
- Collect required student details
- Ask clarification questions only when data is missing
- Store data securely
- Send structured data to a webhook once complete

UNIVERSITY UNITS

- Admission/Scholarship Unit
- Academic_Support Unit
- Student Welfare Unit
- Career/Skill Development Unit

TECH STACK (FIXED)

Frontend:

- React (Vite)
- Minimal chat-style UI

Backend:

- Python
- FastAPI
- LangGraph for conversation flow

Database:

- Supabase (Postgres)

FRONTEND REQUIREMENTS

- Single chat screen
- Student text input
- AI responses displayed sequentially
- REST API communication with backend

BACKEND LOGIC (LANGGRAPH)

Nodes:

1. start_node
- Receive user input
- Forward to router_node

2. router_node
- Classify intent into appropriate university unit

Routing Rules:
- Admission, eligibility, fees, scholarships → Admission/Scholarship Unit
- Exams, subjects, attendance, grading → AcademicSupport Unit
- Hostel, grievances, wellbeing → Student Welfare Unit
- Internship, placement, skills → Career/Skill Development Unit

Unit Nodes:
Nodes:
- Admission/Scholarship_node
- AcademicSupport_node
- Student Welfare_node
- Career/Skill Development_node

UNIT NODE RESPONSIBILITIES

Each unit node must collect:
- Student Name
- Academic Year (12th pass / 1st–4th year)
- Student Query

Clarification Rules:
- Do not assume student details
- Ask only one missing field at a time
- Maintain polite, student-friendly tone
- Do not send webhook until all data is collected

DATABASE (SUPABASE)

Store the following fields:
- Student Name
- Academic Year
- Student Query
- routed_unit
- timestamp

WEBHOOK INTEGRATION

Trigger Condition:

- All required student data(Student Name, Education Year, Student Query) is collected

Webhook Method:

- POST

Payload:

```
{
  “Student Name”: “string”,
 “ Academic Year”: “12th_pass | 1st_year | 2nd_year | 3rd_year | 4th_year”,
  “Student Query”: “string”,
  “unit”: “admission_scholarship | academic_support | student_welfare |
career_skill_development”
}
```

API ENDPOINT

POST /chat

- Input: Student message
- Output: AI response

CONSTRAINTS & RULES

- Do not hallucinate answers
- Do not skip clarification steps
- Do not trigger webhook early
- Focus on correctness over UI polish

DELIVERABLES

- Minimal React frontend
- FastAPI backend
- LangGraph workflow
- Supabase integration
- Webhook trigger logic
- Clean and working code