📘 A.I Final Hackathon Project

Project: AI Campus Admin Agent
Duration: 9:00am – 9:00pm
Frameworks: LangGraph / OpenAI Agent SDK + FastAPI (Postman testing required) + (optional) Next.js frontend

---

🎯 Project Goal

Build an **AI-powered Campus Admin Agent** that:

- Manages student records (CRUD operations).
- Provides real-time **streaming responses** to API clients.
- Generates **analytics/statistics** about campus data.
- Maintains conversation context using **memory**.
- Uses **tool calls** to interact with the campus database & utilities.
- Exposes everything as a **FastAPI endpoint** (with streaming).
- (Optional) Provides a **Next.js dashboard** for admins to visualize analytics.
- Use database **MongoDB** / **Postgress**

🛠️ Features to Implement

1. Core Agent

- Built with **LangGraph** or **OpenAI Agent SDK**.
- Must support **memory** for contextual multi-turn conversations.

- Must support **streaming** responses:
  - FastAPI `/chat/stream` endpoint that yields tokens as Server-Sent Events (SSE) or WebSocket.

---

2. Tools (Functions)

📌 Student Management

- `add_student(name, id, department, email)`
- `get_student(id)`
- `update_student(id, field, new_value)`
- `delete_student(id)`
- `list_students()`

📌 Campus Analytics

- `get_total_students()`
- `get_students_by_department()`
- `get_recent_onboarded_students(limit=5)`
- `get_active_students_last_7_days()` (based on activity logs, mock if needed)

📌 Campus FAQ (optional but nice to have)

- `get_cafeteria_timings()`
- `get_library_hours()`

- `get_event_schedule()`

📌 **Notifications**

- `send_email(student_id, message)` → Mock logging/emailing.

---

3. FastAPI Endpoints

- `/chat` → Normal chat (sync).
- `/chat/stream` → **Streaming chat responses** (SSE or WebSocket).
- `/students` → REST-style endpoints for CRUD ops (optional but useful for Postman tests).
- `/analytics` → Returns JSON with statistics (counts, charts data).

---

4. Analytics Dashboard (Optional Stretch)

Admins should be able to see:

- Total students.
- Students by department (pie chart).
- Recent onboardings (list).
- Activity heatmap.

📌 Build in **Next.js**:

- Page `/dashboard` fetches from `/analytics`.
- Use simple charts (e.g., Recharts).

📂 Suggested Project Structure

```
campus-admin-agent/
│── backend/
│   ├── main.py          # FastAPI app (endpoints)
│   ├── agent.py         # LangGraph/OpenAI agent setup
│   ├── tools.py         # Tools: DB ops, analytics, notifications
│   ├── db.py            # SQLite database helper
│── frontend/ (optional)
│   ├── pages/index.js    # Next.js chat
│   ├── pages/dashboard.js # Analytics dashboard
│── README.md             # Setup instructions
```

✅ Deliverables

1. **FastAPI backend** with:
    - `/chat` (normal chat)
    - `/chat/stream` (streaming chat)
    - `/analytics` (JSON stats)
    - `/students` (CRUD endpoints)
2. **Agent with memory & tool calls**.
3. **Postman collection** testing all endpoints.
4. **Analytics features** (total students, dept breakdown, etc.).
5. (Optional) **Next.js dashboard**.

Submit: https://forms.gle/Q9z8vuwdZF5erPSq5