**My approach to the given problem statement**

**Assumptions and Decision:**

Noted down all the key variables and the workflow how things should be procced.

* **Database choice**: I made use of MySQL form simpler design and workflow and as familiar with the structure of the MySQL structure.
* **API Design**: I opted for a **RESTful API** design using Python with the Flask framework. This choice allows for clear, resource-based endpoints ( events, students, registrations), making it easy to integrate with both the admin web portal and the student mobile app. The use of standard HTTP methods (GET, POST, PUT) also simplifies the communication protocol.
* **Data Structure**: I decided to use a **relational database schema** with separate tables for colleges, events, students, registrations, attendance, and feedback. This approach ensures data normalization and integrity, preventing issues like data duplication. For instance, the events table includes a college\_id foreign key to handle the scale assumption of multiple colleges.
* **Report Generation**: For generating reports, I created dedicated API endpoints (reports, event popularity) instead of simple database queries. This decision encapsulates the business logic on the server side, ensuring that the reports are consistent and can be easily consumed by different front-end applications. It also allows for potential future enhancements, such as filtering or caching, without changing the API contract.

**Use of LLM:**

I have Used Gemini to brainstorm the idea given in the problem statement. With the minimal idea I got the clear outlook of the API endpoints and also the structure to be used as to get the desired regional output.

The conversation With the Gemini API is Attached in link below:

<https://g.co/gemini/share/af5554b3ea9d> - Initial Idea Discussion

<https://g.co/gemini/share/6a1aabae3f36> - For the MySQL database

**Deviated from AI suggestions**:

As the Gemini suggested for the AI's schema was a good starting point, but it was too simple for the given problem. It didn't account for the **multi-college scale assumption**. I added a colleges table and a college\_id foreign key to the Events and Students tables. This critical change ensures that event IDs are unique within a college and allows for separate data management per college as required by the assignment. The AI's initial suggestion also did not include a way to handle different event types, so I added an event\_type column to the Events table to accommodate the bonus requirement of filtering by type (e.g., Workshop/Fest/Seminar).