Campus Event Reporting (Approach)

Name: Akshata
Date: 07-09-2025
Project: Webknot Technologies — Campus Event Reporting (Prototype)

1. Short summary

I have built a small event reporting prototype: a backend (FastAPI + SQLite) with APIs for event creation, student registration, attendance check-in, and feedback collection. I have also provided endpoint reports for event popularity, attendance percentage, and student participation. AI tools were used for brainstorming only, final design and README is self written.

2. Key assumptions

- student_id and event_id are integer autoincrement keys.
- One student registering only once for an event.
- Attendance allowed only for registered students.
- Feedback is one rating (1–5) per (student, event).
- Multi-college support via `college_id` column.
- Prototype uses SQLite.

3. Decisions made

- Database: SQLite for ease of setup and reproducibility.
- Backend: FastAPI for quick APIs and built-in docs.
- UI: Lovable.
- IDs: Integer autoincrement for simplicity.

4. Where I used AI

I used an LLM to:

- Propose schema & API list.
- Generate SQL report queries.
- List edge cases and validation rules.

AI chat screenshots are included in the AI log folder.

5. Followed vs Deviated from AI

Followed: Table structure - students, events, registrations, attendance, feedback.
Deviated:

- AI suggested UUIDs(Universally Unique Identifiers), I used integers for SQLite simplicity.
- AI suggested allowing anonymous feedback, I made it as only registered students for feedback.

6. Edge cases & handling

- Duplicate registration → rejected (unique constraint).
- Check-in without registration → rejected.
- Event cancelled → disallow register/check-in.
- Missing feedback → allowed (treated as NULL in averages).

7. Security Considerations (Prototype Scope)

- All inputs validated through Pydantic models.
- SQL injection prevented by using parameterized queries.
- Feedback rating restricted to 1–5.