**Campus Drive Assignment – Webknot Technologies**

**Name:** NITHESH G

**USN:** ENG22CS0112

**College:** Dayananda Sagar University

**Problem Statement:** Imagine you’re part of a team building a Campus Event Management Platform.

● Admin Portal (Web): Used by college staff to create events (hackathons, workshops, tech talks, fests, etc.).

● Student App (Mobile): Used by students to browse events, register, and check-in on the event day. Your mission is to design and implement a basic event reporting system for this platform.

**1. Assumptions and Decisions**

* Each college has its own set of events, but all stored in one database (with a college\_id field).
* Event IDs will be **globally unique** (auto-increment IDs) but tied to a college\_id.
* Students can register for multiple events.
* Duplicate registrations are prevented by a composite constraint (student\_id + event\_id).
* Attendance is recorded only for registered students.
* Feedback is optional but tied to attendance.

**2. Design**

**a) Data to Track**

* Event creation (title, type, date, college).
* Student registration (event\_id, student\_id).
* Attendance (event\_id, student\_id, status).
* Feedback (event\_id, student\_id, rating 1–5).

**b) Database Schema**

**Tables**:

Colleges(college\_id PK, name)

Students(student\_id PK, name, email, college\_id FK)

Events(event\_id PK, title, type, date, college\_id FK)

Registrations(reg\_id PK, student\_id FK, event\_id FK, UNIQUE(student\_id, event\_id))

Attendance(att\_id PK, student\_id FK, event\_id FK, status)

Feedback(feedback\_id PK, student\_id FK, event\_id FK, rating)

**c) API Design**

| **Endpoint** | **Method** | **Description** |
| --- | --- | --- |
| /events | POST | Create new event |
| /events/:id/register | POST | Register a student |
| /events/:id/attendance | POST | Mark attendance |
| /events/:id/feedback | POST | Submit feedback |
| /reports/popularity | GET | Events sorted by registrations |
| /reports/student/:id | GET | Participation report for a student |
| /reports/top-students | GET | Top 3 active students |

**d) Workflows**

**Registration → Attendance → Reporting**

1. Student registers → stored in Registrations.
2. On event day, admin marks attendance → stored in Attendance.
3. After event, feedback submitted → stored in Feedback.
4. Reports generated from above tables.

**e) Assumptions & Edge Cases**

* Prevent duplicate registration → UNIQUE(student\_id, event\_id).
* If feedback missing → exclude from average calculation.
* Cancelled events → mark in Events table with status="cancelled".
* A student can attend multiple events, and events can have many students (many-to-many).

## 3. Prototype Implementation (Node.js + Express + SQLite)

I have create a **small working prototype** with:

* server.js → Express server.
* db.js → SQLite schema setup.
* routes/ → Event, Registration, Attendance, Reports.
* package.json → Dependencies.

**Error Handling & Validation**

* Duplicate registration prevention
* Invalid rating validation (1-5)
* Missing field validation
* Non-existent student/event checks
* Foreign key constraint enforcement
* Comprehensive error messages

(You can run with: npm install && node server.js)

## 4. Reports

* **Event Popularity**:

SELECT e.title, COUNT(r.reg\_id) AS registrations

FROM Events e

LEFT JOIN Registrations r ON e.event\_id = r.event\_id

GROUP BY e.event\_id

ORDER BY registrations DESC;

* **Student Participation**:

SELECT s.name, COUNT(a.att\_id) AS events\_attended

FROM Students s

JOIN Attendance a ON s.student\_id = a.student\_id

WHERE a.status = 'present'

GROUP BY s.student\_id;

* **Top 3 Most Active Students**:

SELECT s.name, COUNT(a.att\_id) AS events\_attended

FROM Students s

JOIN Attendance a ON s.student\_id = a.student\_id

WHERE a.status = 'present'

GROUP BY s.student\_id

ORDER BY events\_attended DESC

LIMIT 3;

**Reports – sample outputs**

**GET /reports/popularity**

[

{

"event\_id": 6,

"title": "API Test Workshop",

"type": "Workshop",

"date": "2025-03-15",

"college\_name": "Computer Science College",

"registrations": 1

},

{

"event\_id": 1,

"title": "Tech Conference 2025",

"type": "Conference",

"date": "2025-02-15",

"college\_name": "Computer Science College",

"registrations": 0

}

]

**GET /reports/student/1**

{

"student\_id": 1,

"name": "John Doe",

"email": "john.doe@email.com",

"college\_name": "Computer Science College",

"events\_attended": 1,

"events\_present": 1,

"events\_absent": 0,

"total\_registrations": 1

}

**GET /reports/top-students**

[

{

"student\_id": 1,

"name": "John Doe",

"email": "john.doe@email.com",

"college\_name": "Computer Science College",

"events\_attended": 1

}

]

**GET /reports/overview**

{

"totalEvents": 6,

"totalStudents": 7,

"totalRegistrations": 1,

"totalAttendance": 1,

"totalFeedback": 1,

"avgRating": 5

}

**GET /events**

[

{

"event\_id": 6,

"title": "API Test Workshop",

"type": "Workshop",

"date": "2025-03-15",

"college\_id": 1,

"college\_name": "Computer Science College"

},

{

"event\_id": 1,

"title": "Tech Conference 2025",

"type": "Conference",

"date": "2025-02-15",

"college\_id": 1,

"college\_name": "Computer Science College"

}

]

**GET /events/6/registrations**

[

{

"reg\_id": 1,

"student\_id": 1,

"name": "John Doe",

"email": "john.doe@email.com",

"college\_name": "Computer Science College"

}

]

**GET /events/6/attendance**

[

{

"att\_id": 1,

"student\_id": 1,

"name": "John Doe",

"email": "john.doe@email.com",

"status": "present",

"college\_name": "Computer Science College"

}

]

**GET /events/6/feedback**

{

"feedback": [

{

"feedback\_id": 1,

"student\_id": 1,

"name": "John Doe",

"email": "john.doe@email.com",

"rating": 5,

"college\_name": "Computer Science College"

}

],

"average\_rating": 5,

"total\_responses": 1

}