

# Entities and endpoints overview



Campus Drive Assignment – Webkn...  
PDF

give me an idea about what entities and end points i might need

Got it 👍 Based on the **Campus Event Management Platform** scenario in your assignment, here are the **entities** and **API endpoints** you'd likely need:

## Entities (Database Tables / Models)

### 1. College

- `college_id`, `name`, `location`

### 2. Student

- `student_id`, `name`, `email`, `phone`, `college_id` (FK)

### 3. Event

- `event_id`, `title`, `description`, `event_type` (Workshop, Hackathon, Fest, etc.),  
`date`, `location`, `college_id` (FK)

### 4. Registration (link between Student & Event)

- `registration_id`, `student_id` (FK), `event_id` (FK), `registration_time`

### 5. Attendance

- `attendance_id`, `student_id` (FK), `event_id` (FK), `check_in_time`,  
`status`

### 6. Feedback

- `feedback_id`, `student_id` (FK), `event_id` (FK), `rating` (1-5),  
`comments`

## API Endpoints

### ♦ Admin (Web Portal)

#### • Event Management

- `POST /events` → Create an event
- `GET /events?college_id=...` → List all events for a college

- `PUT /events/{event_id}` → Update event
  - `DELETE /events/{event_id}` → Cancel/Delete event
  - **Reports**
    - `GET /reports/event-popularity` → Registrations per event (sorted)
    - `GET /reports/student-participation` → Events attended per student
    - `GET /reports/top-students?limit=3` → Most active students
    - `GET /reports/event-feedback/{event_id}` → Avg feedback per event
- 

## ◆ Student (Mobile App)

- **Browse & Register**
    - `GET /events?college_id=...` → Browse events
    - `POST /events/{event_id}/register` → Register for an event
  - **Attendance**
    - `POST /events/{event_id}/checkin` → Mark attendance
  - **Feedback**
    - `POST /events/{event_id}/feedback` → Submit feedback
- 

## ⚡ Reporting Queries (to back endpoints)

- **Total registrations per event** → `COUNT(registration_id) GROUP BY event_id`
  - **Attendance percentage** → `(attended / total_registrations) * 100`
  - **Average feedback score** → `AVG(rating) GROUP BY event_id`
- 

Would you like me to **draw an ER diagram** (entities + relationships) for this schema so you can directly include it in your design document?

walk me through relationship mappings

Perfect — let's walk through the **relationships between entities** step by step. This will help you build the ER diagram and understand how tables connect.

---

# 🎓 Relationship Mappings

## 1. College ↔ Student

- **One-to-Many**
    - A college can have many students.
    - Each student belongs to one college.
    - **Mapping:** `college_id` in **Student** table (FK).
- 

## 2. College ↔ Event

- **One-to-Many**
    - A college can organize many events.
    - Each event belongs to one college.
    - **Mapping:** `college_id` in **Event** table (FK).
- 

## 3. Student ↔ Event (via Registration)

- **Many-to-Many**
    - A student can register for many events.
    - An event can have many students registered.
    - **Mapping:** `Registration` acts as a junction table with `student_id` and `event_id` as FKs.
    - **Cardinality:** Student (1..) ↔ (..1) Event.
- 

## 4. Student ↔ Event (via Attendance)

- **Many-to-Many (with extra attributes)**
    - Similar to registration, but tracks *check-in*.
    - Each row shows whether a student attended a specific event.
    - **Mapping:** `Attendance` table with `student_id`, `event_id`, and `status` (e.g., Present/Absent).
- 

## 5. Student ↔ Event (via Feedback)

- **Many-to-Many (with feedback data)**
  - A student can give feedback for each event they attend.

- An event can have feedback from many students.
  - **Mapping:** `Feedback` table with `student_id`, `event_id`, `rating`, `comments`.
- 

## Summary of Relationships

- **College** → **Student** = 1:N
  - **College** → **Event** = 1:N
  - **Student** ↔ **Event (Registration)** = M:N
  - **Student** ↔ **Event (Attendance)** = M:N
  - **Student** ↔ **Event (Feedback)** = M:N
- 

👉 In short, **College** is the root entity, and **Student** + **Event** are its direct children. The “interaction” entities (**Registration**, **Attendance**, **Feedback**) serve as **link/junction tables** to capture many-to-many relationships.

---

Would you like me to **draw this as an ER diagram** (with entities, primary keys, and relationships), so you can just drop it into your design doc?