**CRUD Task 3**

**3. Event Management System**

A system to manage events like workshops, seminars, or meetups.

**Schema:**

* **Event**: title, date, location, description
* **Participant**: name, email, eventId

**Tasks:**

* **Create an Event** (POST /events)
* **Register a Participant** (POST /participants)
* **Fetch All Events** (GET /events)
* **Update Event Details** (PATCH /events/:id)
* **Delete an Event** (DELETE /events/:id)

**✅ Event Management System - API Endpoints & URLs**

|  |  |  |
| --- | --- | --- |
| Operation | Method | URL Example |
| Create an Event | POST | http://localhost:5000/events?title=Workshop&date=2025-03-10&location=NYC&description=AI Workshop |
| Register a Participant | POST | http://localhost:5000/participants?name=John Doe&email=john@mail.com&eventId=event123 |
| Fetch All Events | GET | http://localhost:5000/events |
| Update Event Details | PATCH | http://localhost:5000/updateevent?id=<id>&title=New Workshop Title |
| Delete an Event | DELETE | <http://localhost:5000/deleteevent?id=event123> |

**Code**

const express = require("express");

const mongoose = require("mongoose");

const app = express();

app.use(express.json());

mongoose.connect("mongodb://127.0.0.1:27017/eventDB", { useNewUrlParser: true, useUnifiedTopology: true })

  .then(() => console.log("MongoDB Connected"));

const Event = mongoose.model("Event", new mongoose.Schema({

    title: String, date: String, location: String, description: String

}));

const Participant = mongoose.model("Participant", new mongoose.Schema({

    name: String, email: String, eventId: String

}));

// ✅ Create an Event

app.post("/events", async (req, res) => res.send(await new Event(req.query).save()));

// ✅ Register a Participant

app.post("/participants", async (req, res) => res.send(await new Participant(req.query).save()));

// ✅ Fetch All Events

app.get("/events", async (req, res) => res.send(await Event.find()));

// ✅ Update Event Details

app.patch("/updateevent", async (req, res) => {

    const { id, ...updateData } = req.query;

    if (!id) return res.status(400).send({ message: "Event ID is required" });

    res.send(await Event.findByIdAndUpdate(id, updateData, { new: true }));

});

// ✅ Delete an Event

app.delete("/deleteevent", async (req, res) => {

    if (!req.query.id) return res.status(400).send({ message: "Event ID is required" });

    res.send(await Event.findByIdAndDelete(req.query.id));

});

app.listen(5000, () => console.log("Server running on port 5000"));

### ****1. Find All Events Happening on a Specific Date****

(Replace *"2025-03-20"* with an actual date)

**db.events.find({ date: "2025-03-20" });**

### ****2. Find All Participants for a Specific Event****

(Replace *"event123"* with an actual *eventId*)

**db.participants.find({ eventId: "event123" });**

### ****3. Count the Number of Participants for Each Event****

**db.participants.aggregate([**

**{ $group: { \_id: "$eventId", totalParticipants: { $sum: 1 } } }]);**

### ****4. Find Events Happening in a Specific Location****

**db.events.find({ location: "New York" });**

### ****5. Update Event Location****

(Replace *{event\_id}* with an actual event’s ID)

**db.events.updateOne({ \_id: ObjectId("{event\_id}") }, { $set: { location: "Los Angeles" } });**

### ****6. Delete Participants of a Specific Event****

(Replace *"event123"* with an actual *eventId*)

**db.participants.deleteMany({ eventId: "event123" });**

### ****7. Find the Most Recent Event (Upcoming Event)****

**db.events.find().sort({ date: 1 }).limit(1);**

### ****8. Find All Participants Who Registered with a Gmail Account****

**db.participants.find({ email: /@gmail\.com$/i });**

### ****9. Retrieve Events Sorted by Title in Ascending Order****

**db.events.find().sort({ title: 1 });**

### ****10. Find Events Where the Description Contains "Tech"****

**db.events.find({ description: /Tech/i });**

## **5 API Queries** (For Testing the Express API)

### ****1. Add a New Event****

**POST /events?title=Tech Conference&date=2025-04-10&location=California&description=Annual Tech Meetup**

### ****2. Register a Participant for an Event****

(Replace *"event123"* with an actual *eventId*)

**POST /participants?name=John Doe&email=john.doe@example.com&eventId=event123**

### ****3. Get All Events****

GET /events

### ****4. Update an Event's Date****

(Replace *{event\_id}* with an actual event’s ID)

**PATCH /updateevent?id={event\_id}&date=2025-05-15**

### ****5. Delete a Specific Event****

(Replace *{event\_id}* with an actual event’s ID)

**DELETE /deleteevent?id={event\_id}**