

# MongoDB Assignment – Event Management & Ticketing System

**Student Name:** Arpan Harnal

**Project:** Event Management & Ticketing System using MongoDB

This document explains the MongoDB backend code written for the Event Management and Ticketing System. The aim of this assignment is to demonstrate understanding of MongoDB concepts such as database creation, collections, CRUD operations, aggregation pipelines, and indexing. The project is implemented at a beginner level using simple logic and a clear structure.

## Database Used

The database used for this project is **eventManagementDB**. MongoDB automatically creates the database when data is inserted into collections.

## Schema Design (schemas.js)

MongoDB is a schema-less database; however, collections are created to logically organize data. The schemas.js file is used to create the required collections.

```
db.createCollection("users")
db.createCollection("categories")
db.createCollection("events")
db.createCollection("tickets")
```

These commands create collections for storing users, event categories, events, and ticket bookings.

## Sample Data Insertion (sampleData.js)

Sample data is inserted to test the functionality of the database and to demonstrate queries and reports. Simple and realistic values are used to keep the data easy to understand.

```
// Users
db.users.insertMany([
  {
    userId: "U001",
    name: "Amit Kumar",
    email: "amit.kumar001@gmail.com",
    phone: "9876543201",
    role: "organizer",
    createdAt: new Date("2024-01-05")
  },
  {
    userId: "U002",
    name: "Priya Sharma",
    email: "priya.sharma002@gmail.com",
    phone: "9876543202",
    role: "organizer",
    createdAt: new Date("2024-01-10")
  },
  {
    userId: "U003",
    name: "Rahul Verma",
    email: "rahul.verma003@gmail.com",
    phone: "9876543203",
    role: "attendee",
    createdAt: new Date("2024-02-01")
  },
  {
    userId: "U004",
    name: "Sneha Singh",
    email: "sneha.singh004@gmail.com",
    phone: "9876543204",
    role: "attendee",
    createdAt: new Date("2024-02-05")
  }
])
```

The users collection stores both organizers and attendees.

```
// Categories
db.categories.insertMany([
  { categoryId: "C001", name: "Music", description: "Music concerts and live performances" },
  { categoryId: "C002", name: "Technology", description: "Technical workshops and seminars" },
  { categoryId: "C003", name: "Sports", description: "Sports events and tournaments" },
  { categoryId: "C004", name: "Arts", description: "Art exhibitions and cultural programs" }
])
```

Categories help in grouping similar types of events.

## Events and Tickets Data

Events are created by organizers and tickets are booked by users. Each ticket booking is linked to an event and a user.

```
// Events
db.events.insertMany([
  {
    eventId: "E001",
    title: "Live Music Night",
    description: "Enjoy live band performances",
    category: "Music",
    dateTime: new Date("2024-10-15"),
    venue: "Delhi Auditorium",
    organizerId: "U001",
    price: 1000,
    totalTickets: 200,
    availableTickets: 150,
    status: "upcoming"
  },
  {
    eventId: "E002",
    title: "AI Basics Workshop",
    description: "Introduction to Artificial Intelligence",
    category: "Technology",
    dateTime: new Date("2024-11-20"),
    venue: "Bangalore Tech Hall",
    organizerId: "U002",
    price: 800,
    totalTickets: 150,
    availableTickets: 90,
    status: "upcoming"
  }
])

// Tickets
db.tickets.insertMany([
  {
    ticketId: "T001",
    eventId: "E001",
    userId: "U003",
    bookingDate: new Date("2024-09-20"),
    quantity: 2,
    totalAmount: 2000,
    status: "booked"
  },
  {
    ticketId: "T002",
    eventId: "E002",
    userId: "U004",
    bookingDate: new Date("2024-10-01"),
    quantity: 1,
    totalAmount: 800,
    status: "booked"
  }
])
```

This data represents ticket bookings for events by users.

## CRUD Operations (queries.js)

CRUD operations are used to manage data in MongoDB, including creating, reading, updating, and deleting records.

```
// CREATE
db.events.insertOne({
  eventId: "E003",
  title: "Beginner Coding Workshop",
  description: "Basic coding workshop for beginners",
  category: "Technology",
  dateTime: new Date("2024-12-05"),
  venue: "Online",
  organizerId: "U001",
  price: 500,
  totalTickets: 100,
  availableTickets: 100,
  status: "upcoming"
})

// READ
db.events.find({ category: "Music" })

// UPDATE
db.events.updateOne(
  { eventId: "E001" },
  { $set: { status: "completed" } }
)

// DELETE
db.events.deleteOne({ eventId: "E003" })
```

These queries demonstrate how event data can be created, fetched, updated, and deleted.

## Aggregation Queries

Aggregation pipelines are used to generate analytical reports from the data.

```
// Top 5 events by ticket sales
db.tickets.aggregate([
  { $group: { _id: "$eventId", totalTicketsSold: { $sum: "$quantity" } } },
  { $sort: { totalTicketsSold: -1 } },
  { $limit: 5 }
])

// Total revenue earned by each organizer
db.events.aggregate([
  {
    $lookup: {
      from: "tickets",
      localField: "eventId",
      foreignField: "eventId",
      as: "ticketDetails"
    }
  },
  { $unwind: "$ticketDetails" },
  {
    $group: {
      _id: "$organizerId",
      totalRevenue: { $sum: "$ticketDetails.totalAmount" }
    }
  }
])

// Number of attendees per event
db.tickets.aggregate([
  {
    $group: {
      _id: "$eventId",
      totalAttendees: { $sum: "$quantity" }
    }
  }
])

// Events grouped by category and status
db.events.aggregate([
  {
    $group: {
      _id: {
        category: "$category",
        status: "$status"
      },
      totalEvents: { $sum: 1 }
    }
  }
])
```

## Indexes

Indexes are created to improve query performance and reduce search time.

```
db.events.createIndex({ category: 1 })
db.events.createIndex({ dateTime: 1 })
db.tickets.createIndex({ eventId: 1 })
```

Indexes help MongoDB quickly locate documents when searching by category, date, or event.

## Conclusion

This assignment provided hands-on experience with MongoDB. It helped in understanding schema design, data insertion, CRUD operations, aggregation pipelines, and indexing. The project demonstrates a simple backend structure for an Event Management and Ticketing System and is suitable for beginner-level learning.