CSCI 485: *Database Design & Collection Architecture*

Student ID: 664 870 797 - **Student name:** Chris Lawrence

Project Title: EventSphere

A. Domain Analysis & Requirements Review

Recap of Domain and Primary Use Case

EventSphere is a MongoDB-backed events platform enabling users to discover, review, and attend in-person or virtual events. Core features include geospatial discovery near a location, full-text search, real-time updates, and analytics on attendance and reviews.

Key Queries to Support

- Nearby events within X km of a location, optionally filtered by category and date window
- Full-text search across title/description/category/tags with relevance sorting
- Upcoming events (date-range filtering and sorting)
- Reviews by event and by venue, with rating aggregation
- Attendance analytics: repeat attendees, peak check-in hours, venue monthly stats
- User attendance history and event check-ins
- Category popularity and trends over time
- **Polymorphic queries**: Events by type (virtual/in-person/hybrid), venues by type (conferenceCenter/park/virtualSpace)
- **Schema versioning queries**: Filter by schema version for migration and compatibility

Data Access Patterns & Performance Priorities

- Most frequent access: event discovery (geo + date + category) and text search
- Secondary: reviews retrieval and basic analytics aggregations
- Write patterns: event CRUD (moderate), check-ins (high volume bursts near event time), reviews (steady)
- Priorities:
  - Availability: browsing and search should remain responsive even under load
  - Consistency: strong consistency for booking/seat updates; eventual consistency acceptable for attendee counts, check-in feeds, and analytics

B. Collection Design Strategy

Collections Overview

- `events`: **Polymorphic** catalog of events with GeoJSON location, scheduling, embedded ticket tiers, and attendee snippets. Supports multiple event types (inPerson, virtual, hybrid, recurring) with type-specific attributes.
- `venues`: **Polymorphic** physical/virtual locations with address, capacity, amenities, and GeoJSON point. Supports multiple venue types (conferenceCenter, park, restaurant, virtualSpace, stadium, theater)

with type-specific details.

- `users`: User profiles and preferences (discovery filters, location, interests)
- `checkins`: Bridge collection for many-to-many user ↔ event attendance with analytics fields
- `reviews`: Feedback on events/venues with ratings and optional tags

Schema Versioning: All collections include `schemaVersion` field (currently "1.0") to support future schema evolution and migration strategies.

Embedding vs Referencing Decisions

- Embedded
  - `events.tickets[]`: small, tightly bound to event; read together for listing/booking
  - `events.attendees[]` (lightweight snippet when used): quick RSVP display
  - `venues.address`: always co-read with venue
- Referenced
  - `events.venueId → venues._id`: venues shared by many events
  - `checkins.eventId`, `checkins.userId`, `checkins.venueId`: analytics-friendly fan-out
  - `reviews.eventId` or `reviews.venueId` and `reviews.userId`

Relationship Mapping & Justification

- 1:1 — venue:address (embedded subdocument for cohesion)
- 1:many — venue:events (reference from `events.venueId`) to avoid venue bloat
- many:many — users:events via `checkins` bridge to support analytics and scale; avoids unbounded arrays in `users` or `events`

C. Schema Design Documentation

Below, each collection includes: purpose/role, document structure, sample, validation highlights, and indexing strategy. Structures are aligned to `DATABASE_DESIGN.md` and generator scripts.

1) `events`

Purpose: Core catalog for discovery and analytics.

Document Structure (key fields):

- `_id: ObjectId`
- `title: String`
- `description: String`
- `category: String`
- `eventType: String` (enum: "inPerson", "virtual", "hybrid", "recurring") - **Polymorphic discriminator**
- `schemaVersion: String` (enum: "1.0") - **Schema versioning**
- `location: { type: "Point", coordinates: [lng:Number, lat:Number] }`
- `venueId: ObjectId | null`
- `venueReference: { name:String, city:String, capacity:Number, venueType:String } | null` - **Extended Reference Pattern**
- `startDate: Date`, `endDate: Date`
- `organizer: String`

- `maxAttendees: Number`, `currentAttendees: Number`
- `price: Number`, `currency: String`, `isFree: Boolean`
- `status: String` (draft|published|cancelled|completed)
- `tickets: [{ tier:String, price:Number, available:Number, sold:Number }]`
- `attendees: [{ userId:ObjectId, checkedIn:Boolean, checkInTime:Date }]`
- `tags: [String]`
- **Polymorphic type-specific fields:**
  - `virtualDetails: { platform:String, meetingUrl:String, recordingAvailable:Boolean, timezone:String }`
  - `recurringDetails: { frequency:String, endRecurrence:Date, exceptions: [Date] }`
  - `hybridDetails: { virtualCapacity:Number, inPersonCapacity:Number, virtualMeetingUrl:String }`
- `metadata: { ageRestriction:String, dressCode:String, accessibilityFeatures: [String] }`
- `createdAt: Date`, `updatedAt: Date`

Sample Document

```
{
  "_id": ObjectId("68ddb640c00b1dff057fbefc"),
  "title": "Entertainment Event",
  "description": "Experience something new and exciting in a welcoming and inclusive atmosphere.",
  "category": "Entertainment",
  "eventType": "hybrid",
  "schemaVersion": "1.0",
  "location": {
    "type": "Point",
    "coordinates": [-123.93446771957665, 49.10036536726016]
  },
  "venueId": null,
  "venueReference": null,
  "startDate": ISODate("2025-10-09T18:37:26.047Z"),
  "endDate": ISODate("2025-10-09T22:37:26.047Z"),
  "organizer": "Art Gallery",
  "maxAttendees": 990,
  "currentAttendees": 25,
  "price": 135,
  "currency": "CAD",
  "isFree": false,
  "status": "cancelled",
  "tags": ["creative", "networking", "adults-only", "outdoor", "educational"],
  "hybridDetails": {
    "virtualCapacity": 877,
    "inPersonCapacity": 50,
    "virtualMeetingUrl": "https://teams.microsoft.com/j/321999401"
  },
  "metadata": {
    "ageRestriction": "21+",
    "dressCode": null,
    "accessibilityFeatures": []
  },
  "tickets": [
    {
      "tier": "General Admission",
      "price": 135,
      "available": 115,
      "sold": 51
    }
  ],
  "createdAt": ISODate("2025-09-22T18:37:26.047Z"),
  "updatedAt": ISODate("2025-09-22T18:37:26.047Z")
}
```

Validation Rules (highlights)

- Required: `title`, `category`, `location`, `startDate`, `createdAt`, `updatedAt`
- GeoJSON `location.type` enum `["Point"]`; `coordinates` length 2, numeric; bounds checks
- String length constraints; numeric minimums; `endDate` after `startDate`

Indexing Strategy

- `location: "2dsphere"` (geo)
- Text index on `title, description, category, tags`
- Single-field: `startDate`, `createdAt`
- Compound: `{category:1,startDate:1}`, `{organizer:1,startDate:1}`, `{location:"2dsphere",startDate:1}`
- Pagination support: `{_id:1,startDate:1}`

2) `venues`

Purpose: Venue catalog for events and geo queries.

Key Fields

- `_id`, `name`, `type`, `description`
- `address { street, city, state, zipCode, country }`
- `location { type:"Point", coordinates:[lng,lat] }`
- `capacity: Number`, `amenities: [String]`, `contact { phone,email,website }`
- `pricing { hourlyRate, dailyRate, currency }`, `availability {...}`
- `rating: Number`, `reviewCount: Number`, `createdAt`, `updatedAt`

Sample Document

```
{
  "_id": ObjectId("68ddb63fc00b1dff057fb398"),
  "name": "Business Center Plaza - Nanaimo",
  "venueType": "conferenceCenter",
  "schemaVersion": "1.0",
  "type": "Conference Center",
  "description": "A conference center located in Nanaimo, perfect for various events and gatherings.",
  "location": {
    "type": "Point",
    "coordinates": [-124.0050633506158, 49.082833644770226]
  },
  "address": {
    "street": "5744 Main St",
    "city": "Nanaimo",
    "state": "BC",
    "zipCode": "V9T 6N3",
    "country": "Canada"
  },
  "capacity": 596,
  "amenities": [
    "Heating", "Storage", "Audio/Visual Equipment", "Catering",
    "Balcony", "Bar", "Dressing Rooms"
  ],
  "contact": {
    "phone": "(906) 231-6324",
    "email": "info@businesscenterplaza.com",
    "website": "https://www.businesscenterplaza.com"
  },
  "pricing": { "hourlyRate": 146, "dailyRate": 342, "currency": "CAD" },
  "availability": {
    "monday": { "open": "09:00", "close": "22:00" },
    "tuesday": { "open": "09:00", "close": "22:00" },
    "wednesday": { "open": "09:00", "close": "22:00" },
    "thursday": { "open": "09:00", "close": "22:00" },
    "friday": { "open": "09:00", "close": "23:00" },
    "saturday": { "open": "10:00", "close": "23:00" },
    "sunday": { "open": "10:00", "close": "20:00" }
  },
  "rating": 4.2,
  "reviewCount": 17,
  "conferenceCenterDetails": {
    "breakoutRooms": 12,
    "avEquipment": ["Video Conferencing", "Projectors", "Whiteboards", "Sound System"],
    "cateringAvailable": false
  },
  "createdAt": ISODate("2024-12-28T23:16:15.999Z"),
  "updatedAt": ISODate("2025-09-20T23:16:15.999Z")
}
```

Validation & Indexes

- Require `name`, `address`, `location`, `venueType`, `schemaVersion`, `createdAt`
- Geo index: `location: "2dsphere"`
- Polymorphic indexes: `{venueType: 1, capacity: 1}`, `{venueType: 1, rating: 1}`

- Support text/filters via fields like `type`, `capacity`

3) `users`

Purpose: User profiles and discovery preferences.

Key Fields

- `_id`, `email`, `profile { firstName, lastName, preferences{ location, radiusKm, categories[] } }`
- Additional app-profile fields for demo data (interests, bio, stats) may exist in generator outputs; core deliverable keeps minimal shape above
- `createdAt`, `lastLogin`

Sample Document

```
{
  "_id": ObjectId("68ddb640c00b1dff057fb511"),
  "email": "barbara.williams@yahoo.com",
  "schemaVersion": "1.0",
  "profile": {
    "firstName": "Barbara",
    "lastName": "Williams",
    "preferences": {
      "categories": ["Fitness", "Politics", "Health & Wellness",
"Meditation"],
      "location": {
        "type": "Point",
        "coordinates": [-80.48434831242973, 43.52343446108092]
      },
      "radiusKm": 38
    }
  },
  "createdAt": ISODate("2025-04-09T23:16:16.008Z"),
  "updatedAt": ISODate("2025-04-09T23:16:16.008Z"),
  "lastLogin": ISODate("2025-04-16T23:16:16.008Z")
}
```

Validation & Indexes

- Require `email`, `profile`, `createdAt`
- Optional Geo for preference location; unique email (optional for demo)

4) `checkins`

Purpose: Bridge collection for attendance (many:many) and analytics.

Key Fields

- `_id`, `eventId`, `userId`, `venueId`, `checkInTime`, `qrCode`, `ticketTier`, `checkInMethod`, `location`, `metadata{deviceInfo, ipAddress, staffVerified}`, `createdAt`

Sample Document

```
{
  "_id": ObjectId("68ddb640c00b1dff05809406"),
  "eventId": ObjectId("68ddb640c00b1dff057fc4e3"),
  "userId": ObjectId("68ddb640c00b1dff057fb507"),
  "venueId": ObjectId("68ddb640c00b1dff05809405"),
  "checkInTime": ISODate("2025-12-17T16:23:09.091Z"),
  "qrCode": "QR-554361",
  "schemaVersion": "1.0",
  "ticketTier": "VIP",
  "checkInMethod": "manual",
  "location": {
    "type": "Point",
    "coordinates": [-121.88338526503668, 37.2974440040109]
  },
  "metadata": {
    "deviceInfo": "iPhone",
    "ipAddress": "112.24.5.114",
    "staffVerified": true
  },
  "createdAt": ISODate("2025-12-17T16:23:09.091Z"),
  "updatedAt": ISODate("2025-12-17T16:23:09.091Z")
}
```

Validation & Indexes

- Require all referenced ids, `checkInTime`, `qrCode`, `createdAt`
- Unique pair index to prevent duplicates: `{eventId:1,userId:1}`
- Indexes for analytics: `eventId`, `userId`, `venueId`, `checkInTime`, `qrCode`, plus compound as needed

5) `reviews`

Purpose: Ratings/comments on events or venues.

Key Fields

- `_id`, `eventId?`, `venueId?`, `userId`, `rating(1–5)`, `comment`, `createdAt`, `updatedAt`

Sample Document

```
{
  "_id": ObjectId("68ddb640c00b1dff057ff76a"),
  "eventId": ObjectId("68ddb640c00b1dff057fbcc5"),
  "userId": ObjectId("68ddb640c00b1dff057fb5a2"),
  "rating": 5,
  "comment": "This was one of the best events I've attended this year.
Highly recommend to anyone interested in this topic.",
  "schemaVersion": "1.0",
  "createdAt": ISODate("2026-03-17T09:04:00.028Z"),
  "updatedAt": null
}
```

Validation & Indexes

- Require `userId`, `rating`, `createdAt`, `updatedAt` and exactly one of `eventId` or `venueId`
- Indexes: `eventId`, `venueId`, `userId`, `rating`, `createdAt`, and compounds for aggregations