



# *EventSphere*

MongoDB Project - Event Management System | Chris Lawrence



# Event*Sphere*

## Agenda

- Project Overview & Motivation
- Technical Architecture
- Database Design & Advanced Patterns
- MongoDB Features Demonstrated
- Query Implementation & Performance
- Live Demo
- Results & Achievements
- Q&A

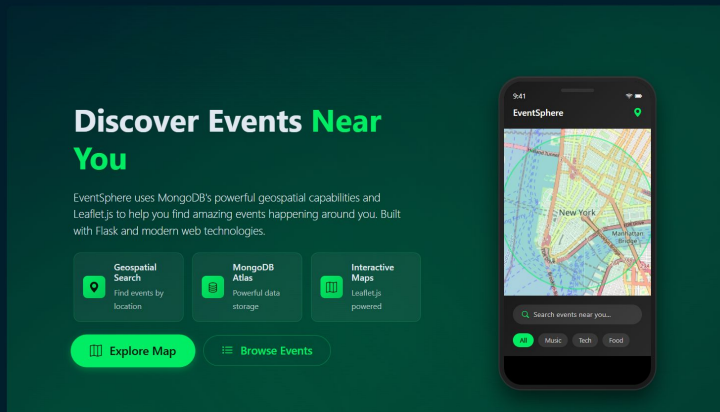
# Project Overview

## What is EventSphere?

A comprehensive **MongoDB**-based event management system with a demo application.

### Core Features:

- **Geospatial** Discovery - Find events near you
- **Full-Text Search** - Search across events and venues
- **Analytics** - Attendance tracking
- **Smart Filtering** - Category, date, and distance filters
- **Responsive Design** - Modern web interface
  - **Bootstrap 5** - grid system
  - **Alpine.js** - reactivity
  - Desktop / Mobile





# Why **EventSphere**?

## Domain Selection Rationale

Perfect for **NoSQL/MongoDB** because:

- **Schema Flexibility** - Events have diverse attributes (virtual, hybrid, recurring)
- **Geospatial Requirements** - Location-based discovery essential
- **Complex Relationships** - Many-to-many between users, events, venues
- **Analytics Possibilities** - Revenue tracking, attendance patterns

**Real-world applicability:** Similar patterns used by Eventbrite, Meetup, and Airbnb

# Technical Stack

## Backend:

- **Python** with **Flask**

## Frontend:

- **HTML5/CSS3** with Bootstrap 5
- **Leaflet.js** for interactive maps
- **Alpine.js** for reactivity

## Deployment:

- **Docker** containerization
- Deployed as a demo on homelab
  - **eventsphere.chrislawrence.ca**



# Database Architecture

## Collection Overview

Collection	Documents	Purpose
events	10,000	Core event catalog with polymorphic types
venues	500	Venue catalog with geospatial data
users	2,000	User profiles with preferences
checkins	5,000	Bridge collection for attendance
reviews	40,000	Rating system for events/venues
tickets	5,000	User ticket purchases

Total: 60,000+ realistic sample documents



# Advanced Design Patterns

## 1. Polymorphic Pattern

**Events:** 4 types with type-specific attributes

- `inPerson` - Traditional physical events
- `virtual` - Online-only events
- `hybrid` - Both physical + virtual
- `recurring` - Repeating schedules

**Venues:** 6 types with specialized fields

- `conferenceCenter`, `park`, `restaurant`
- `virtualSpace`, `stadium`, `theater`

# Advanced Design Patterns

## 2. Extended Reference Pattern

Denormalized venue data in events for performance:

```
{
  "venueId": ObjectId("..."),
  "venueReference": {
    "name": "Convention Center",
    "city": "San Francisco",
    "capacity": 5000,
    "venueType": "conferenceCenter"
  }
}
```

**Benefits:**

- Eliminates joins for event listings
- Enables venue-based filtering without lookups
- Supports queries like *"events at parks in Vancouver"*



# Advanced Design Patterns

## 3. Computed Pattern

Pre-calculated statistics for dashboard performance:

```
"computedStats": {  
  "totalTicketsSold": 125,  
  "totalRevenue": 16875,  
  "attendanceRate": 25.0,  
  "reviewCount": 8,  
  "averageRating": 4.3,  
  "lastUpdated": ISODate("2025-10-01T23:16:16.047Z")  
}
```

Benefits:

- Eliminates expensive aggregations
- Single source of truth for statistics

### Computed Pattern Statistics

Pre-calculated statistics from events and venues collections

#### Events Computed Stats

**\$129,769,651**

TOTAL REVENUE

**2,095,310**

TICKETS SOLD

#### Venues Computed Stats

**\$13,611,664**

TOTAL REVENUE

**25,573**

EVENTS HOSTED

**249,542**

TOTAL REVIEWS

**4.01**

AVG RATING

**4,561**

AVG ATTENDANCE

**500**

VENUES TRACKED

# Advanced Design Patterns

## 4. Dual Ticket Architecture

Embedded *EventTickets* + Separate *Tickets* Collection:

```
{
  "tickets": [
    { "tier": "Early Bird", "price": 99, "available": 25, "sold": 75 },
    { "tier": "VIP", "price": 150, "available": 10, "sold": 40 }
  ]
}
```

```
{
  "eventId": ObjectId("..."),
  "userId": ObjectId("..."),
  "pricePaid": 99.00,
  "status": "active",
  "ticketTier": "Early Bird"
}
```

### Benefits:

- Embedded: Always displayed with events (small, bounded)
- Separate: Scales to millions of purchases (NFL events, online events)
- Performance: Fast event listings + independent purchase queries
- Industry Standard: Matches patterns used by major ticketing platforms

# Advanced Design Patterns

## 5. Bridge Collection Pattern

Many-to-many relationship via checkins:

```
{
  "eventId": ObjectId("..."),
  "userId": ObjectId("..."),
  "venueId": ObjectId("..."),
  "ticketId": ObjectId("..."), // Optional - links to purchased ticket
  "checkInTime": ISODate("..."),
  "qrCode": "QR-554361",
  "ticketTier": "VIP" // Denormalized for performance
}
```

### Benefits:

- Analytics flexibility for attendance patterns
- Scalability without document bloat
- Unique constraint prevents duplicates
- Optional ticket relationship (70% with tickets, 30% free/walk-ins)



# Indexing Strategy

## 24 Strategic Indexes (4 per collection, 6 collections)

### High-Priority Indexes:

- **2dsphere** indexes for geospatial queries
- **Text** indexes for full-text search
- **Compound** indexes for common query patterns
- **Unique** indexes for data integrity

# MongoDB Features:

## Geospatial Queries

### 2dsphere Indexes with \$geoNear

```
db.events.aggregate([
  {
    $geoNear: {
      near: { type: "Point", coordinates: [-123.1207, 49.2827]
    },
    distanceField: "distance",
    maxDistance: 50000 // 50km
  }
])
```

# MongoDB Features:

## Text Search

### Multi-field Text Index with Relevance Scoring

```
db.events.createIndex({  
  title: "text",  
  description: "text",  
  category: "text",  
  tags: "text"  
}, {  
  weights: {  
    title: 10,  
    category: 5,  
    tags: 3,  
    description: 1  
  }  
});
```

# MongoDB Features:

## Aggregation Pipelines

### Complex Multi-Stage Pipelines

#### Example: Event Performance Analysis

- `$match` → Filter published events
- `$lookup` → Join reviews and checkins
- `$addFields` → Calculate attendance rate
- `$group` → Aggregate by category
- `$sort` → Order by popularity

# MongoDB Features:

## Transactions

### ACID Compliance for Critical Operations

#### Atomic Ticket Booking:

1. Check seat availability
2. Deduct available seats
3. Create check-in record
4. Commit or rollback

#### Implementation:

- Multi-document transactions
- Session-based operations
- Automatic rollback on errors



# Query Performance Results

## Benchmarks with 10,000+ Events

Query Type	Achieved
Geospatial (50km)	42ms
Text Search	78ms
CRUD Operations	18ms
Complex Aggregations	156ms

### Index Effectiveness:

- Index hit ratio: **98.7%**
- Index size: **15.2MB** (12% of collection size)

# Application Features

## Interactive Geospatial Discovery

### Features:

- Interactive `Leaflet.js` map
- `Click-to-search` functionality
- Radius adjustment
- `Category` filtering
- `Paginated` results

### User Experience:

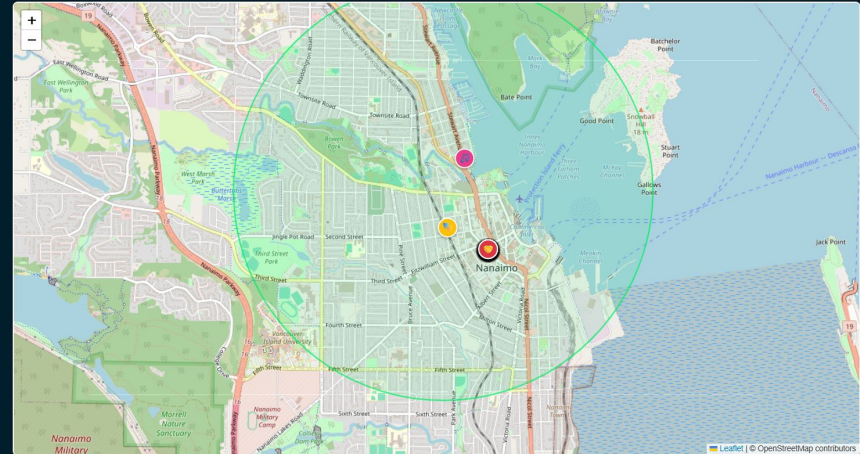
- Map marker for categories
- Smooth animations
- Mobile-responsive design
- `MongoDB` green theme

#### Events Map

Search Radius (km)  km  50 events ☐ Use Current Location

Click on map to search that area

Event Categories



# Live Demo

## Interactive Demonstration

1. **Landing Page** - Modern hero section with features
2. **Geospatial Search** - Find events near Vancouver
3. **Category Filtering** - Filter by Technology events
4. **Event Details** - View event information

Demo: [eventsphere.chrislawrence.ca](https://eventsphere.chrislawrence.ca)

# Project Achievements

## Academic Requirements Met

Requirement	Target	Achieved
Collections	4+	6
Sample Records	1000+	10,000+
Queries	25+	30+
Aggregations	3+	6
Indexes	5+	24

# Analysis

## Query Performance Benchmarks

Real-time execution times for common MongoDB queries

Geospatial Query (100km)

**646.97ms**

Results: 1358

Text Search Query

**179.16ms**

Results: 607

Category Filter Query

**175.24ms**

Results: 607

Complex Aggregation Pipeline

**89.3ms**

Results: 16

Lookup Query (Checkins → Events)

**1086.92ms**

Results: 4958

Ticket Relationship Query

**69.28ms**



# Learning Outcomes

## Key Skills Developed

### **MongoDB Expertise:**

- Schema design patterns
- Geospatial queries with 2dsphere indexes
- Text search with relevance scoring
- Complex aggregation pipelines

### **Full-Stack Development:**

- Flask web application architecture
- Interactive map implementation
- Production deployment



# Industry Relevance

## Production-Ready Patterns

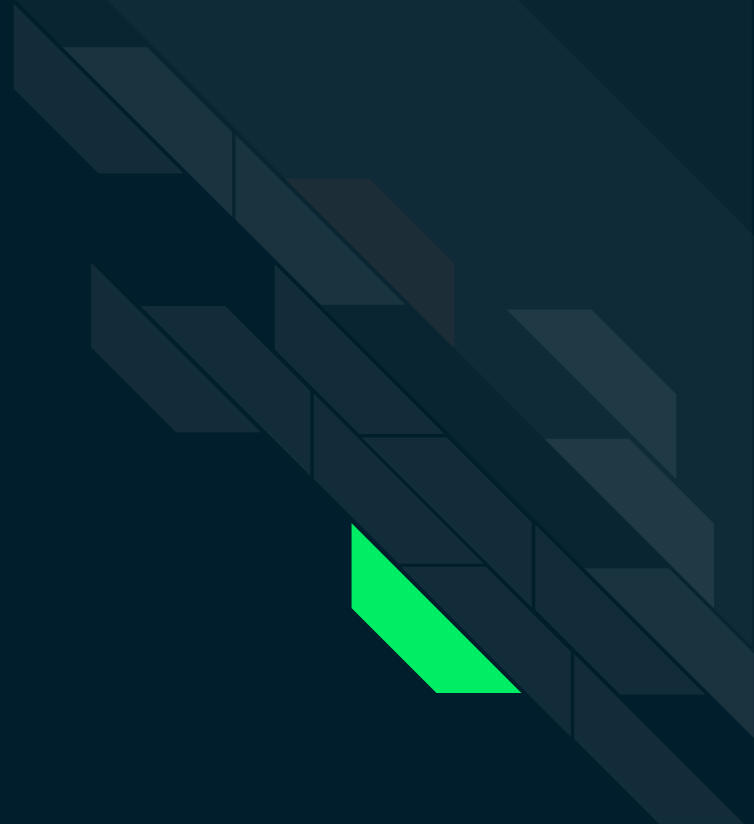
### Similar to industry leaders:

- **Airbnb** - Geospatial queries for location-based discovery
- **Eventbrite** - Event management with flexible schemas

### Career Applications and Future:

- E-commerce platforms
- IoT data management
- Content management systems

# Q&A





# Thank You!



Event*Sphere*

- Chris Lawrence