CSCI485 MongoDB Project - Deliverable 2

Database Design & Collection Architecture

Due Date: [October 07, 2025]
Points: 50 points
Submission Format: PDF document

Overview

Building on your approved project proposal, this deliverable focuses on designing the core database architecture for your MongoDB application. You will create a comprehensive database design that demonstrates proper NoSQL modeling principles, defines collection schemas with validation rules, and establishes clear relationships between your data entities.

Deliverable Components

1. Database Design Document (30 points)

Submit a well-structured PDF document (5-8 pages) containing the following sections:

A. Domain Analysis & Requirements Review (5 points)

- Brief recap of your chosen domain and primary use case
- List of 5-8 specific queries your application will need to support
- Identification of data access patterns and performance requirements (which is accessed most frequently)

B. Collection Design Strategy (10 points)

- **Collection Overview**: Description of each collection (minimum 4 collections)
- **Embedding vs. Referencing Decisions**: Detailed rationale for when you chose to embed documents vs. use references
- **Relationship Mapping**: Clear documentation of all relationships (1:1, 1:many, many:many) with justifications

C. Schema Design Documentation (15 points)

For each collection, provide:

- **Purpose and Role**: What this collection represents in your domain
- **Document Structure**: Detailed field breakdown with data types
- **Sample Document**: Example document showing realistic data

- **Validation Rules**: Schema validation constraints you'll implement
- **Indexing Strategy**: Proposed indexes with performance justification (we can add later)

2. MongoDB Schema Implementation Files (20 points)

Submit working MongoDB scripts demonstrating:

A. Collection Creation Scripts (create_collections.js) 10 points

```
// Example structure expected:
db.createCollection("collectionName", {
  validator: {
    $jsonSchema: {
      // Your validation schema
    }
  }
})
```

B. Sample Data Insertion (sample_data.js) 5points

- Minimum 10 sample documents across all collections
- Demonstrate all relationship types
- Realistic, domain-appropriate data

C. Initial Index Creation (create_indexes.js) (we will add later)5 points

- Minimum 5 strategic indexes
- Include compound indexes where appropriate
- Comment explaining each index's purpose

Technical Requirements

Schema Design Standards

- Use descriptive field names in camelCase
- Implement appropriate data types (String, Number, Date, Array, Object, etc.)
- Include required field constraints

Relationship Implementation

- **One-to-One**: Demonstrate with embedded documents or references
- **One-to-Many**: Show both embedding and referencing
- **Many-to-Many**: Implement using arrays of references or intermediate collections as per your requirements

Data Quality Requirements

- All sample data must be realistic and consistent
- Proper data distribution for meaningful query testing
- Include edge cases (empty arrays, optional fields, etc.)

Common Design Mistakes to Avoid

1. **Over-Normalization**: Don't replicate relational database design patterns
2. **Under-Indexing**: Missing indexes on frequently queried fields
3. **Inappropriate Embedding**: Embedding large or frequently changing subdocuments
4. **Poor Validation**: Weak or missing schema validation rules
5. **Unrealistic Data**: Sample data that doesn't reflect real-world usage patterns

Getting Started Tips

1. **Start with Use Cases**: List your most important queries first, then design collections to support them efficiently
2. **Consider Growth**: Think about how your data might scale - will embedded arrays grow unbounded?
3. **Iterate**: Don't be afraid to refine your design based on query performance

Remember: Good database design is crucial for your final project success. Invest time in this deliverable to avoid major redesign work later!