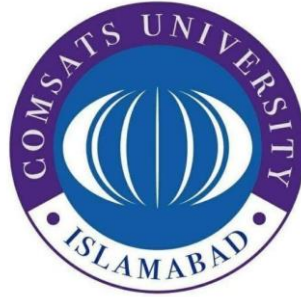


## **Design Notes**



Submitted by:

**ABDUL MATEEN (FA23-BCS-014)**

**MUHAMMAD ZEESHAN TARIQ (FA23-BCS-143)**

**B**

Submitted to:

**Mr. Talha Irfan**

## Database Schema (MongoDB)

We are using MongoDB because the data is mostly read-heavy, large, and semi-structured.

### Main Collection:

**books**

**Fields:**

- `_id` (ObjectId)
- `book_id` (Integer)
- `title` (String)
- `authors` (Array of strings)
- `average_rating` (Float)
- `isbn` (String)
- `isbn13` (String)
- `language_code` (String)
- `num_pages` (Integer)
- `ratings_count` (Integer)
- `text_reviews_count` (Integer)
- `publication_date` (String)
- `publisher` (String)

No joins are needed because data is stored in one document.

### Indexes Used:

Indexes improve reading speed.

### Created Indexes:

1. `{title: 1}`  
Used for searching by title.
2. `{authors: 1 }`  
Used for author filtering.
3. `{ average_rating: -1 }`  
Used to quickly get books with highest ratings.
4. `{ ratings_count: -1 }`  
Used for sorting by popularity.

These indexes reduce query time but increase storage a little.

### Why these Indexes?

- Title search becomes fast
- Author queries return faster
- High-rating sorting becomes efficient
- Better response performance for REST API

Without indexes, MongoDB would scan the whole database every time.

## **Trade-offs / Limitations:**

### **Pros:**

- Fast read performance
- Easy scaling
- Flexible schema
- Good for analytical queries

### **Cons:**

- Indexes take extra storage
- Write performance becomes slightly slower
- Data not normalized