**Final Assignment: MongoDB Advanced Concepts**

**Course:** Advanced Database Services  
**Instructor:** Leonardo Moura  
**Due Date:** 8/13/2024

**Final MongoDB Assignment: Student Records Database**

**Overview**

In this assignment, you will be working with a MongoDB database containing student records. The goal is to apply what you've learned about MongoDB by performing various tasks, including indexing, validation, using the MongoDB Data API, and visualizing data with MongoDB Charts.

**Scenario**

You’ve been hired by a fictional university, Seneca Polytechnic, to manage their student records database. Your tasks include optimizing the database, ensuring data integrity, and creating tools to visualize and interact with the data.

**Objectives**

1. **MongoDB Indexing and Search Optimization**
   * *Objective:* Improve the performance of queries on the student records database.
   * **Task:**
     + Create indexes on the student\_records collection to optimize searches by student names, email, and enrollment date.
     + Compare the query performance with and without indexes. Document your findings.
2. **MongoDB Validation**
   * *Objective:* Ensure that all student records adhere to the expected format.
   * **Task:**
     + Implement schema validation for the student\_records collection to enforce data integrity.
     + Ensure that the email field contains a valid email format, age is within a reasonable range (e.g., 18-30), and status is one of the predefined values (e.g., "active", "graduated", "dropped").
     + Test your validation by attempting to insert an invalid document and observe the results.
3. **MongoDB Data API**
   * *Objective:* Create a simple API to interact with the student records database.
   * **Task:**
     + Use MongoDB's Data API (or build your own using Node.js) to create endpoints that allow you to:
       - Retrieve all active students.
       - Add a new student record.
       - Update the status of a student (e.g., from "active" to "graduated").
       - Delete a student record.
     + Document how to use these endpoints.
4. **MongoDB Charts**
   * *Objective:* Visualize the student data to gain insights.
   * **Task:**
     + Use MongoDB Charts to create visual representations of the data.
     + Create at least three charts, such as:
       - A pie chart showing the distribution of students by status ("active", "graduated", "dropped").
       - A bar chart showing the number of students enrolled each year.
       - A line chart tracking the average grade for each course over time.
     + Embed these charts into a report or dashboard for easy access.

**Bonus Task (Optional)**

* **Fun with Data**:
  + Create a small web application that displays a leaderboard of the top 10 students with the highest average grades. The application should fetch data from the MongoDB database and update the leaderboard dynamically.

**Submission**

* Submit your MongoDB scripts, API code, and any additional files (charts, reports) in a GitHub repository.
* Provide a link to your GitHub repository in your submission, along with a brief report (one page) summarizing your findings and reflecting on what you learned.

**Evaluation Criteria**

* Correctness: Does your code work as expected?
* Optimization: Have you successfully improved query performance?
* Data Integrity: Is your schema validation thorough?
* Documentation: Are your API endpoints and charts well-documented?
* Creativity: Have you gone above and beyond to make your submission unique and engaging?

**MongoDB Database Setup**

A sample database named seneca\_students has been created for this assignment. The collection student\_records contains 100 documents representing student data with the following structure:

{

"student\_id": "S12345",

"name": "John Doe",

"email": "john.doe@example.com",

"age": 21,

"courses": [

{

"course\_name": "Database Systems",

"grade": "A"

},

{

"course\_name": "Web Development",

"grade": "B+"

}

],

"enrollment\_date": "2023-09-01T00:00:00Z",

"status": "active"

}

You can download the database https://github.com/mouraleonardo/Student\_Records\_Database\_Sample.

**Grading Criteria**

* **Indexing/Search (25%)**: Effectiveness of indexes and demonstrated performance improvements.
* **Validation (20%)**: Correctness, completeness, and robustness of schema validation.
* **Data API (25%)**: Functionality, security, and documentation of the Data API interactions.
* **Charts (20%)**: Quality, variety, and insights provided by the MongoDB Charts visualizations.
* **Bonus Task (Optional - 10%)**: Creativity and functionality of the optional web application (leaderboard).

**Submission Guidelines**

* Submit all code files, documentation, and any relevant MongoDB scripts through the course’s learning management system by the due date.
* Include a README file in your submission that explains how to run your scripts and applications.