PER SCHOLAS

# SBA 319:

# MongoDB Database Application

Version 1.0, 08/01/23

Click here to open in a separate window.

## Introduction

This assessment measures your understanding of MongoDB and your capability to implement its features in a practical manner. You have creative freedom in the topic, material, and purpose of the web application you will be developing, so have fun with it! However, remember to plan the scope of your project to the timeline you have been given.

This assessment has a total duration of **three (3) days**. This is a **take-home assessment**.

You have **three total days** (including weekends and holidays) to work on this assessment. This assessment will be due at **5:00pm** on the third day after it is assigned. Your instructor will provide you with at least four hours of class time to work on the assessment, during which time you may discuss details of the project with them, including topic, scope, and implementation.

## Objectives

- Create a server application with Node, Express, and MongoDB.
- Create a CRUD API using Express and MongoDB.
- Create MongoDB indexes.
- Use MongoDB indexing to make efficient queries.
- Create MongoDB validation rules.
- Use MongoDB validation to ensure data consistency.

## Submission

Submit the link to your completed assessment using the **Start Assignment** button on the Assignment page in Canvas.

Your submission should include:

- A link to the GitHub repository for your project.

## Instructions

You will create a small Node, Express, and MongoDB server application. The topic and content of this application is entirely up to you; be creative!

Your work will be graded according to the technical requirements listed in the following section. Creativity and effort always work in your favor, so feel free to go beyond the scope of the listed requirements if you have the time.

Keep things simple. Like most projects you will encounter, you should finish the absolute minimum requirements *first*, and then add additional features and complexity if you have the time to do so. This will also help you understand what you can get done in a specific allotment of time if you were to be asked to do something similar in the future.

Once you have an idea in mind, briefly discuss it with your instructors to determine if it is appropriate for the amount of time you have been given.

Since topic and content are secondary to functionality for this assessment, we have included some resources below for free content that you can use to populate your application. Once you have gotten your functionality in place, you can return and fill in the content with something interesting.

**Resources for free content:**

- **Text:** Lipsum, a Lorem Ipsum text generator.
- **Images:** Pexels, a resource for stock photos (and other media).
- **GIFs:** Motion Elements, a resource for GIFs (and other media).

## Requirements

The requirements listed here are **absolute minimums**. Ensure that your application meets these requirements before attempting to further expand your features.

Create your application locally, and initialize a local git repo. Make frequent commits to the repo. When your application is complete, **push your repo to GitHub and submit the link to the GitHub page** using the submission instructions at the top of this document.

| Requirement | Weight |
| --- | --- |
| Use at least three different data collections within the database (such as users, posts, or comments). | **5%** |
| Utilize reasonable data modeling practices. | **10%** |
| Create GET routes for all data that should be exposed to the client, using appropriate query commands to retrieve the data from the database. | **10%** |
| Create POST routes for data, as appropriate, using appropriate insertion commands to add data to the database. At least one data collection should allow for client creation via a POST request. | **10%** |
| Create PATCH or PUT routes for data, as appropriate, using appropriate update commands to change data in the database. At least one data collection should allow for client manipulation via a PATCH or PUT request. | **10%** |
| Create DELETE routes for data, as appropriate, using appropriate delete commands to remove data from the database. At least one data collection should allow for client deletion via a DELETE request. | **10%** |
| Include sensible indexes for any and all fields that are queried frequently. For fields that may have a high write-to-read ratio, you may forgo indexes for performance considerations. Make comments of this where applicable. | **5%** |
| Include sensible MongoDB data validation rules for at least one data collection.<br><br>Note: this may be accomplished in a number of ways. If you choose to perform this task outside of your application's code, you must include a way to test the validation within the application's routes. This can be as simple as providing a POST route that attempts to create an invalid document and displays the resulting error. | **5%** |
| Populate your application's collections with sample data illustrating the use case of the collections. You must include at least five sample documents per collection.<br><br>Note: **Double-check this requirement before submission**. Testing your delete routes may leave you under the requirement. To be safe, populate your collections with sample data well above the requirement (we recommend 10-20 documents). | **5%** |
| Utilize reasonable code organization practices. | **5%** |
| Ensure that the program runs without errors (comment out things that do not work, and explain your blockers - you can still receive partial credit). | **10%** |

| | |
|---|---|
| Commit frequently to the git repository. | **5%** |
| Include a README file that contains a description of your application.<br><br>This README must include a description of your API's available routes and their corresponding CRUD operations for reference. | **5%** |
| Level of effort displayed in creativity and user experience. | **5%** |

## Bonus Objectives

The objectives listed here are **not required**. Ensure that your application meets the requirements above before attempting to further expand your features.

These bonus objectives *cannot* increase your overall score above 100%. Successful completion of these objectives can; however, make up for lost points above. **Ensure your application works as outlined by the requirements above before attempting these objectives, time permitting.**

| | | |
|---|---|---|
| Use Mongoose to implement your application.<br><br>Note: The validation requirements above **must still be implemented database-side**, but should *also* be implemented application-side within your Mongoose schema(s). | **+1%** | |

## Reflection (Optional)

Once you have completed your project, answer the following questions to help solidify your understanding of the process and its outcomes, as well as improve your ability to handle similar tasks in the future.

- *What could you have done differently during the planning stages of your project to make the execution easier?*

| |
|---|
| |

- *Were there any requirements that were difficult to implement? What do you think would make them easier to implement in future projects?*

- *What would you add to or change about your application if given more time?*

- *Use this space to make notes for your future self about anything that you think is important to remember about this process, or that may aid you when attempting something similar again:*