PER SCHOLAS

CP 325.7:
Creating a MongoDB Database

Version 1.0, 08/14/23
Click here to open in a separate window.

Introduction

This document will challenge you with taking the next steps in your Capstone Project journey.

Assignment Objective

- Create a MongoDB database for your application.

Submission

This portion of the Capstone Project will not be submitted on its own; only the final project will be submitted. You are encouraged to seek feedback from instructors and peers along the way.

Instructions

You finally have full control over your data! Take a moment to celebrate, and then create a MongoDB cluster for your project.

Once your cluster has been created, begin modeling the data for your application. Define schemas, create indexes, and integrate the database into your API routes. You have the freedom to use either the native MongoDB driver or Mongoose to interface with your database.

If you need to make changes to the front end or back end of your application to accommodate your own data, now is the time to do so.

Once everything is connected and working, be sure to populate your database with some test data and play with your application a bit. Now is a wonderful time to discover bugs, and brainstorm feature ideas because the next step is a big one: *using React as the front-end framework for your application*, which will give you an opportunity to refactor code where necessary.

Partial Requirements

Let’s look at which Capstone Project requirements are related to this stage of development. Requirements not related to this stage of development have been omitted. For a full list of requirements, see CP 323.1 - Planning a Project, or CP 323.10 - Capstone Completion.

(20%) Project Structure, Standardization, and Convention	**Weight**
Project is organized into appropriate files and directories, following best practices.	**2%**
Project contains an appropriate level of comments.	**2%**
Project is pushed to GitHub, and contains a README file that documents the project, including an overall description of the project.	**5%**
Standard naming conventions are used throughout the project.	**2%**
Ensure that the program runs without errors (comment out things that do not work, and explain your blockers - you can still receive partial credit).	**4%**
Level of effort displayed in creativity, presentation, and user experience.	**5%**

(12%) Core JavaScript	**Weight**
Demonstrate proper usage of ES6 syntax and tools.	**2%**
Use functions and classes to adhere to the DRY principle.	**2%**
Use Promises and async/await, where appropriate.	**2%**
Use Axios or fetch to retrieve data from an API.	**2%**
Use sound programming logic throughout the application.	**2%**
Use appropriate exception handling.	**2%**

(9%) Database	Weight
Use MongoDB to create a database for your application.	5%
Apply appropriate indexes to your database collections.	2%
Create reasonable schemas for your data by following data modeling best practices.	2%

(19%) Server	Weight
Create a RESTful API using Node and Express. * For the purposes of this project, you may forgo the HATEOAS aspect of REST APIs.	7%
Include API routes for all four CRUD operations.	5%
Utilize the native MongoDB driver or Mongoose to interface with your database.	5%
Include at least one form of user authentication/authorization within the application.	2%

(35%) Front-End Development	Weight
Use CSS to style the application.	5%
Create at least four different views or pages for the application.	5%
Interface directly with the server and API that you created.	5%

The following section is NOT included in the requirements for this project. Completing this section is NOT required. This section will NOT negatively impact your grade if left unfinished.

This section is intended for learners looking to go the extra mile by showcasing additional skills such as project management, and optional technologies like TypeScript.

You must complete ALL other requirements to receive credit for this section; however, this extra credit will not be included if you have already received the maximum 100% grade. The extra credit can only offset points lost elsewhere.

(5%) Extra Credit	**Weight**
Adhere to Agile principles and the Scrum framework. Perform stand-up sessions (with an instructor) when possible.	**1%**
Successfully track your project using a software similar to Jira.	**1%**
Build your application primarily with TypeScript.	**3%**