Kristianstad
University
Sweden

Kristianstad University
SE-291 88 Kristianstad
+46 44-250 30 00
www.hkr.se

Lab 2 Fullstack

DA219B VT25 Full Stack Development

Daniel Jönsson

Content

1 Challenges and Solutions

1.1 Challenges with Referencing Three MongoDB Collections

Initially, I encountered significant difficulties in establishing correct references between three distinct MongoDB collections using their Object IDs. Ensuring data integrity and accurately querying related documents across these collections proved to be a complex task, leading to issues in retrieving and displaying complete and consistent information.

1.2 Solution

To fix this, I reviewed the Mongoose schema definitions for each collection, ensuring that the `ref` option was correctly implemented to point to the appropriate models and Object IDs. I also refined my database query logic, utilizing Mongoose's population features to efficiently retrieve related data across the collections in a single query. This careful schema design and query optimization ensured accurate and efficient data retrieval, resolving the referencing issues. I learned a lot from attending the hands on session that my teacher Deema held to learn how to correctly use the Object IDs and populate.

2.1 Complexities in POST Requests with Two Object IDs

Implementing a POST request that required associating a new document with existing documents from two other collections via their Object IDs presented a unique challenge. I struggled with how to correctly structure the request payload and ensure that the server-side logic could accurately interpret and utilize these two Object IDs to establish the necessary relationships in the database.

2.2 Solution

I resolved this issue by implementing a specific sequence of data fetching operations in my React frontend. First, I posted the necessary IDs from the relevant existing documents. Then, I structured the POST request payload to include both of these Object IDs. On the server side, I implemented a decent validation and data association logic within the POST route handler to correctly utilize these IDs and create the new document with the appropriate references. Furthermore, I strategically introduced a small timeout in the React frontend between the fetching of the IDs and the subsequent POST request. This ensured that the necessary IDs were available before the POST request was initiated, preventing potential race conditions and ensuring the successful creation of the related document.

3 Conclusion

This lab provided experience in managing complex data relationships within a MongoDB database and implementing robust RESTful API endpoints for CRUD operations with React + Vite and .jsx objects.

I learned that the issues of correctly referencing data across three collections and handling POST requests involving multiple Object IDs required a deeper understanding of Mongoose schema design, population techniques, careful frontend data fetching strategies, and server-side data association logic. The successful resolution of these challenges, through understanding schema definition, optimized querying, strategic frontend sequencing with timeouts, and correct header configurations, led to this lab becoming a functional application capable of managing complex data relationships.