Rodriques Perry

CS-499

Milestone 3 Narrative

24 November 2024

**The artifact used for Algorithms and Data Structures enhancement is from CS-340, completed on 23 June 2024. The goal of this project is an interactive database for rescue animal centers. The project is written in Python/Dash and is designed to interact with MongoDB, through the use of PyMongo, to work as a backend and frontend structure to Create, Read, Update, and Delete (CRUD) animals from an animal shelter.**

This artifact was selected because it clearly demonstrates a significant translation of a Python-based project with MongoDB and Dash into a more complex, modern full-stack web application using **Spring Boot** for the backend and **React** for the frontend and MongoDB for the database. This transformation shows a strong understanding of software development principles and technologies, as well as the ability to improve the project by refactoring it into a more scalable and maintainable solution.

By adding indexes to fields like breed, ageUponOutcome, and animalType I demonstrated an understanding of data structures (B-trees in MongoDB). These indexes reduce the complexity of search operations from O(n) to O(log n). I also introduced pagination. By introducing pagination in the repository and service layers, the application avoids fetching large datasets all at once which improves performance and memory utilization, showcasing knowledge of breaking down data into manageable chunks. Lastly, the introduction of Redis caching stores frequently queried results, reducing repeated database lookups. This leverages hashing and memory-based storage to optimize read operations.

Improvements made to the Artifact include:

1. Scalability:
   1. Added pagination and filtering ensures the application can handle large datasets without performance degradation.
2. Performance:
   1. Implementing MongoDB indexes and Redis caching significantly improves query performance and reduces latency.
   2. Pagination minimizes data transfer over the network, optimizing the application for real-time interactions.

The enhancements successfully meet the planned course outcomes planned in Module One. By implementing caching and indexing, the enhancement demonstrates proficiency in algorithmic design and optimization, effectively reducing query complexity and improving response times. These methods, combined with paginated and filtered data access, ensure the system can manage large data volumes efficiently while balancing memory usage, processing speed, and scalability—key aspects of managing trade-offs in design. The use of modern tools like Redis for caching and MongoDB for indexing and pagination showcases expertise in leveraging industry-standard technologies to streamline data retrieval and improve performance. Additionally, asynchronous processing with Spring Boot supports scalability by enabling the system to handle high request volumes effectively. Furthermore, the enhancements incorporate a strong security mindset, as seen in its focus on secure data handling practices. Strategies like time-limited caching prevent stale data issues, and the restriction of cache storage to non-sensitive data ensures data privacy and reduces the risk of leaks. By architecting secure data access controls, the enhancements uphold data security principles, addressing potential vulnerabilities proactively. Overall, the enhancement showcases a well-rounded application of algorithmic principles, innovative tools, and secure design practices, meeting and exceeding the planned course outcomes.

A major challenge was integrating Redis caching into the application. Ensuring that caching was implemented effectively to reduce database load, while also handling the complexities of cache invalidation and maintaining data consistency, required a lot of attention to detail. I had to carefully ensure that sensitive data was not stored in the cache and that caching strategies such as time-limited storage were applied correctly. Additionally, debugging and fine-tuning the pagination and filtering functionality involved working with both Spring Boot and MongoDB, which required understanding how to best configure the database and repository layers to meet specific use cases. Working through these I developed stronger problem-solving skills and learned how to approach complex systems from a design and optimization perspective.