Report on Spring Data for NoSQL Databases and Redis

Name: Amo Samuel
Reviewer: Thomas Darko

Configuration and Setup

To begin, I configured Spring Data MongoDB in my Spring Boot project. This involved setting up the necessary dependencies in the pom.xml file and configuring the MongoDB connection properties in the application.properties file.

Entity Mapping

The entities in the hospital system were mapped to MongoDB collections using annotations such as @Document and @MongoId. Here are a few examples:

- **Department**: Represents a department within the hospital. It includes fields like id, code, name, building, and references to a Doctor as the director and a list of Ward entities.
- **Doctor**: Extends an abstract Employee class and includes additional fields like speciality and references to a set of Patient entities and the Department they direct.
- **Patient**: Represents a patient in the hospital, including fields like id, name, surname, address, telephone, diagnosis, and references to the Ward and Doctor associated with the patient.

Repository Interfaces

For data access, I implemented repository interfaces that extend MongoRepository. This allowed me to use built-in methods for CRUD operations and define custom queries where needed. For example, I created a DepartmentRepository to find departments based on the director's ID.

Spring Data Redis Integration

Redis Configuration

To integrate Redis with my Spring Boot application, I used Lettuce as the connection factory. I configured a RedisTemplate bean in the RedisConfig class to manage the serialization of keys and values.

Service and Repository Implementation

I implemented a WardService that interacts with Redis for operations related to the Ward entity. The service provides methods for saving, retrieving, updating, and deleting Ward objects in

Redis. I also implemented custom queries to find wards by the number of beds or a keyword in their name.

RESTful API with Redis

To expose these operations, I developed a WardController with RESTful endpoints. The controller allows for creating, retrieving, updating, and deleting wards, as well as querying wards by certain criteria.

Conclusion

Through this exercise, I gained a deeper understanding of how Spring Data abstracts and simplifies the use of NoSQL databases. I also learned the importance of understanding each NoSQL database’s unique characteristics to effectively use Spring Data's features.