Spring Boot URL Shortener Performance Evaluation Assignment

Objective

The objective of this assignment is to evaluate the candidate's ability to implement a Spring Boot application and conduct performance testing using JMeter. The candidate is required to create a single-page URL shortener application, interface with a SQL-compatible database using Hibernate, and use JMeter to benchmark and improve the performance of the application.

Assignment Description

Create a Spring Boot Application:

1. Develop a single-page Spring Boot URL shortener application. The UI should be simple and without any CSS (design will not be evaluated). The application should allow users to input a long URL and generate a shortened version of it. It should also convert the shortened URL back to the original long URL when accessed.

Integrate a SQL-Compatible Database:

2. Choose a SQL-compatible database of your choice (For example - Postgres, MySQL, MariaDB) and integrate it with the Spring Boot application. The database should store the long URLs and their corresponding shortened versions.

Use Hibernate to Interface with the Database:

3. Implement Hibernate for database interaction in your Spring Boot application. This should handle the persistence of URLs and their shortened counterparts, as well as the retrieval of the original long URLs when required.

Performance Testing with JMeter:

4. Using Apache JMeter, create a test plan to benchmark the performance of the URL shortening and conversion processes. Measure the time taken to shorten a URL and the time taken to convert a shortened URL back to the original long URL. Record the results for analysis and comparison.

Performance Improvement:

5. Analyze the performance test results and identify any bottlenecks or areas of improvement. Implement necessary changes or optimizations to the application to improve its performance, and re-run the JMeter tests to validate the improvements. Some areas of improvement includes caching and multithreading. You are free to use any optimization strategies, but note that all optimization should be within the application. This means you're not allowed to add other components like a external caching tool Redis, a external queuing tool like Kafka etc.

Submission Guidelines

Source Code:

1. Submit the complete source code of the Spring Boot application, along with the necessary configuration files and build scripts.

Database Schema:

2. Include the SQL scripts used to create the database schema and any required indexes or constraints.

JMeter Test Plan:

3. Submit the JMeter test plan file (.jmx) used for performance testing, along with a brief description of the test plan components and their purpose.

Performance Test Results and Analysis:

4. Provide a report detailing the initial performance test results, the optimizations made, and the subsequent test results after implementing the improvements. Include a discussion of the changes made and their impact on the application's performance.

README:

5. Include a README file containing instructions on how to set up, run, and test the application, as well as any other relevant information.

Evaluation Criteria

Functionality:

1. The Spring Boot application should work as described, generating shortened URLs and converting them back to the original long URLs.

Code Quality:

2. The submitted code should be well-structured, follow best practices, and include appropriate documentation.

Database Design and Integration:

3. The chosen SQL-compatible database should be well-implemented and effectively integrated with Hibernate and the Spring Boot application.

Performance Testing and Improvement:

4. The candidate should demonstrate a thorough understanding of performance testing and the ability to identify and implement improvements to the application's performance.

Documentation and Presentation:

5. The submitted documents should be clear, concise, and well-organized, effectively conveying the necessary information for setup, testing, and evaluation.