```
   "logMessage": "Success!"
} )
```

MongoDB will automatically delete documents from the `log_events` collection when the documents' `expireAt` value is older than the number of seconds specified in `expireAfterSeconds`, i.e. `0` seconds older in this case. As such, the data expires at the specified `expireAt` value.

5.1.3 Optimization Strategies for MongoDB

There are many factors that can affect database performance and responsiveness including index use, query structure, data models and application design, as well as operational factors such as architecture and system configuration.

This section describes techniques for optimizing application performance with MongoDB.

***Analyzing MongoDB Performance* (page 213)** Discusses some of the factors that can influence MongoDB's performance.

***Evaluate Performance of Current Operations* (page 216)** MongoDB provides introspection tools that describe the query execution process, to allow users to test queries and build more efficient queries.

***Optimize Query Performance* (page 216)** Introduces the use of *projections* (page 62) to reduce the amount of data MongoDB sends to clients.

***Design Notes* (page 218)** A collection of notes related to the architecture, design, and administration of MongoDB-based applications.

Analyzing MongoDB Performance

As you develop and operate applications with MongoDB, you may need to analyze the performance of the application and its database. When you encounter degraded performance, it is often a function of database access strategies, hardware availability, and the number of open database connections.

Some users may experience performance limitations as a result of inadequate or inappropriate indexing strategies, or as a consequence of poor schema design patterns. *Locking Performance* (page 213) discusses how these can impact MongoDB's internal locking.

Performance issues may indicate that the database is operating at capacity and that it is time to add additional capacity to the database. In particular, the application's *working set* should fit in the available physical memory. See *Memory and the MMAPv1 Storage Engine* (page 214) for more information on the working set.

In some cases performance issues may be temporary and related to abnormal traffic load. As discussed in *Number of Connections* (page 214), scaling can help relax excessive traffic.

Database Profiling (page 215) can help you to understand what operations are causing degradation.

Locking Performance

MongoDB uses a locking system to ensure data set consistency. If certain operations are long-running or a queue forms, performance will degrade as requests and operations wait for the lock.

Lock-related slowdowns can be intermittent. To see if the lock has been affecting your performance, refer to the *server-status-locks* section and the *globalLock* section of the `serverStatus` output.

Dividing `locks.timeAcquiringMicros` by `locks.acquireWaitCount` can give an approximate average wait time for a particular lock mode.

`locks.deadlockCount` provide the number of times the lock acquisitions encountered deadlocks.